A. Appendix

A.1. Training details

While CutLER is agnostic to the underlying detector, we use popular Mask R-CNN [27] and Cascade Mask R-CNN [4] for all experiments, and use Cascade Mask R-CNN by default, unless otherwise noted. We train the detector on ImageNet with initial masks and bounding boxes for 160K iterations with a batch size of 16. When training the detectors with a ResNet-50 backbone [28], we initialize the model with the weights of a self-supervised pretrained DINO [7] model. We explored other pre-trained models, including MoCo-v2 [9], SwAV [6], and CLD [46], and found that they give similar detection performance. Therefore, we initialize model weights with DINO by default.

We also leverage the copy-paste augmentation [16, 19] during the model training process. Rather than using the vanilla copy-paste augmentation to improve the model’s ability to segment small objects, we randomly downsample the mask with a scalar uniformly sampled between 0.3 and 1.0. We then optimize the detector for 160K iterations using SGD with a learning rate of 0.005, which is decreased by 5 after 80K iterations and a batch size of 16. We apply a weight decay of $5 \times 10^{-5}$ and a momentum of 0.9.

For the multi-round of self-training, in each stage, we initialize the detection model using the weights from the previous stage. We optimize the detector using SGD with a learning rate of 0.01 for 80K iterations and a batch size of 16. We apply a momentum of 0.9.

A.2. Datasets used for zero-shot evaluation

**COCO and COCO20K** [32] is a large-scale object detection and instance segmentation dataset, containing about 115K and 5K images in the training and validation split, respectively. Additionally, COCO has an unannotated split of 123K images. We test our model in a class-agnostic manner on COCO val2017 and COCO 20K, without fine-tuning on any images in COCO. COCO 20K is a subset of the COCO trainval2014 [32], containing 19817 randomly sampled images, used as a benchmark in [38,43,50]. We report class-agnostic COCO style averaged precision and averaged recall for object detection and segmentation tasks.

**Pascal VOC** [17] is another popular benchmark for object detection. We evaluate our model on its trainval07 split in COCO style evaluation matrices.

**UVO** [45]. Unidentified Video Objects (UVO) is an exhaustively annotated dataset for video object detection and instance segmentation. We evaluate our model on UVO val by frame-by-frame inference and report results in COCO style evaluation matrices.

**LVIS** [24] collected 2.2 million high-quality instance segmentation masks for over 1000 entry-level object categories, which naturally constitutes the long-tailed data distribution. We report class-agnostic object detection and instance segmentation results on LVIS val split, containing about 5K images.

**CrossDomain** [29] contains three subsets of watercolor, clipart, and comics, in which objects are depicted in watercolor, sketch and painting styles, respectively. We evaluate our model on all annotated images from these three datasets, i.e., train test.

**Objects365 V2** [36] presents a supervised object detection benchmark with a focus on diverse objects in the wild. We evaluate CutLER on the 80K images from its val split.

**OpenImages V6** [31] unifies image classification, object detection, and instance segmentation, visual relationship detection, etc. in one dataset. We evaluate CutLER on its 42K images from the val split.

**KITTI** [18] presents a dataset captured from cameras mounted on mobile vehicles used for autonomous driving research. We evaluate CutLER on 7521 images from KITTI’s trainval split.

We provide the summary of these datasets used for zero-shot evaluation in Table 12.

A.3. Additional results for zero-shot detection & segmentation

In this section, we use official COCO API and provide more results with standard COCO metrics, including AP across various IoU thresholds - AP (averaged over IoU thresholds from 0.5 to 0.95 with a step size of 0.05), AP$_{50}$ (IoU@0.5) and AP$_{75}$ (IoU@0.75), and AP across scales - AP$_{S}$ (small objects), AP$_{M}$ (medium objects) and AP$_{L}$ (large objects). We provide detailed results on all these benchmarks listed in Table 12 and report these results in Table 13. We report the performance of object detection for all datasets. In addition, for those datasets that provide annotations for instance segmentation, we also present the performance of the instance segmentation task. It is worth noting that on these datasets without segmentation labels, CutLER can still predict instance segmentation masks, but since we do not have ground truth masks to be compared, we cannot evaluate the results.

A.4. CutLER vs. Selective Search

Selective Search [41] is a popular unsupervised object discovery method, used in many early state-of-the-art detectors such as R-CNN [22] and Fast R-CNN [21]. However, generating possible object locations with sliding windows greatly reduces inference speed (please refer to [41] for more details on selective search). We compare CutLER’s performance to selective search in Fig. 7 and observe that CutLER provides a significant improvement in both precision and recall, which indicates that CutLER is a...
datasets | domain | testing data | # images | instance segmentation label
--- | --- | --- | --- | ---
COCO [32] | natural images | val2017 split | 5,000 | ✓
COCO20K [32] | natural images | a subset of COCO | 20,000 | ✓
UVO [45] | video frames | val split | 7,356 | ✓
LVIS [24] | natural images | val split | 19,809 | ✓
KITTI [18] | traffic images | trainval split | 7,521 | ✓
Pascal VOC [17] | natural images | trainval07 split | 9,963 | ✓
Clipart [29] | clip arts | train/test split | 1,000 | ✓
Watercolor [29] | paintings | train/test split | 2,000 | ✓
Comic [29] | sketches | train/test split | 2,000 | ✓
Objects365-V2 [36] | natural images | val split | 80,000 | ✓
OpenImages-V6 [31] | natural images | val split | 41,620 | ✓

Table 12. Summary of datasets used for zero-shot evaluation.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>AP_{box} _50</th>
<th>AP_{box} _75</th>
<th>AP_{box} _S</th>
<th>AP_{box} _M</th>
<th>AP_{box} _L</th>
<th>AR_{box} _1</th>
<th>AR_{box} _10</th>
<th>AR_{box} _100</th>
<th>AP_{mask} _S</th>
<th>AP_{mask} _M</th>
<th>AP_{mask} _L</th>
<th>AR_{mask} _1</th>
<th>AR_{mask} _10</th>
<th>AR_{mask} _100</th>
</tr>
</thead>
<tbody>
<tr>
<td>COCO</td>
<td>21.9</td>
<td>11.8</td>
<td>12.3</td>
<td>3.7</td>
<td>12.7</td>
<td>29.6</td>
<td>6.8</td>
<td>19.6</td>
<td>32.8</td>
<td>18.9</td>
<td>9.2</td>
<td>9.7</td>
<td>2.4</td>
<td>8.8</td>
</tr>
<tr>
<td>COCO20K</td>
<td>22.4</td>
<td>11.9</td>
<td>12.5</td>
<td>4.1</td>
<td>12.7</td>
<td>29.5</td>
<td>6.8</td>
<td>19.7</td>
<td>33.1</td>
<td>19.6</td>
<td>9.2</td>
<td>10.0</td>
<td>2.8</td>
<td>8.9</td>
</tr>
<tr>
<td>UVO</td>
<td>31.7</td>
<td>14.1</td>
<td>16.1</td>
<td>3.7</td>
<td>11.3</td>
<td>25.3</td>
<td>6.8</td>
<td>24.5</td>
<td>42.5</td>
<td>31.6</td>
<td>14.1</td>
<td>16.1</td>
<td>3.7</td>
<td>11.3</td>
</tr>
<tr>
<td>LVIS</td>
<td>8.4</td>
<td>3.9</td>
<td>4.5</td>
<td>2.7</td>
<td>9.1</td>
<td>15.1</td>
<td>2.4</td>
<td>9.2</td>
<td>21.8</td>
<td>6.7</td>
<td>3.2</td>
<td>3.5</td>
<td>1.9</td>
<td>6.1</td>
</tr>
<tr>
<td>KITTI</td>
<td>18.4</td>
<td>6.7</td>
<td>8.5</td>
<td>0.5</td>
<td>5.6</td>
<td>19.2</td>
<td>6.2</td>
<td>16.6</td>
<td>27.8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Pascal VOC</td>
<td>36.9</td>
<td>19.2</td>
<td>20.2</td>
<td>1.3</td>
<td>6.5</td>
<td>32.2</td>
<td>16.5</td>
<td>32.8</td>
<td>44.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Clipart</td>
<td>21.1</td>
<td>6.0</td>
<td>8.7</td>
<td>1.1</td>
<td>5.8</td>
<td>11.6</td>
<td>6.6</td>
<td>27.0</td>
<td>40.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Watercolor</td>
<td>37.5</td>
<td>10.9</td>
<td>15.7</td>
<td>0.1</td>
<td>1.1</td>
<td>20.0</td>
<td>19.4</td>
<td>37.8</td>
<td>44.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Comic</td>
<td>30.4</td>
<td>7.7</td>
<td>12.2</td>
<td>0.0</td>
<td>1.3</td>
<td>16.0</td>
<td>8.5</td>
<td>28.2</td>
<td>38.4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Objects365-V2</td>
<td>21.6</td>
<td>10.3</td>
<td>11.4</td>
<td>3.0</td>
<td>10.4</td>
<td>20.4</td>
<td>3.0</td>
<td>15.4</td>
<td>34.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OpenImages-V6</td>
<td>17.3</td>
<td>9.5</td>
<td>9.7</td>
<td>0.4</td>
<td>2.3</td>
<td>14.9</td>
<td>6.5</td>
<td>17.6</td>
<td>29.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 13. Detailed zero-shot evaluation results on all benchmarks used in this work.

Figure 7. Precision-recall curve for comparing selective search and CutLER on VOC07 trainval.

A.5. Training details for label-efficient and fully-supervised learning

We train the detector on the COCO [32] dataset using the bounding box, and instance mask labels. To evaluate label efficiency, we subsample the training set to create subsets with varying proportions of labeled image. We train the detector, initialized with CutLER, on each of these subsets.

As a baseline, we follow the settings from MoCo-v2 [9] and train the same detection architecture initialized with a MoCo-v2 ResNet50 model, given its strong performance on object detection tasks. MoCo-v2 and our models use the same training pipeline and hyper-parameters and are trained for the 1x schedule using Detectron2 [51], except for extremely low-shot settings with 1% or 2% labels. Following previous works [47], when training with 1% or 2% labels, we train both MoCo-v2 and our model for 3,600 iterations with a batch size of 16.

Our detector weights are initialized with ImageNet-1K pre-trained CutLER, except for the weights of the final bounding box prediction layer and the last layer of the mask prediction head, which are randomly initialized with values taken from a normal distribution. For experiments on COCO with labeling ratios below 50%, during model training, we use a batch size of 16, and learning rates of 0.04 and 0.08 for model weights loaded from the pre-trained CutLER and randomly initialized, respectively. For experiments on COCO with labeling ratios between 50% and 100%, the learning rates of all layers decay by a factor of 2.

For a fair comparison, baselines and CutLER use the
same hyper-parameters and settings.

A.6. More visualizations

We provide more qualitative visualizations of CutLER’s zero-shot predictions in Fig. 9.
Figure 9. More visualizations of CutLER’s predictions.