A Further Method Details and Overview

Recall in Section 3.2, to leverage information across a semantic spectrum, we extract features from throughout ResNet-50, instead of using only features from the final convolutional map as in \[22\]. Following the notation in Section 3.1 and Section 3.2, we provide a formal description of this procedure here.

Concretely, given a layer \(l\), we define \(y[l] = f[l](v) \in \mathbb{R}^{D_F \times H_F \times W_F}\) as the feature map of height \(H_F\), width \(W_F\) and feature resolution \(D_F\) obtained at the \(l\)th layer of encoding via \(f\). Then given a predefined set of layers \(L\), we obtain a final feature map \(F \in \mathbb{R}^{D'_F \times H'_F \times W'_F}\) via simple layer fusion:

\[
F = [S_1(f^{[1]}(v)); S_2(f^{[2]}(v)); \ldots; S_l(f^{[l]}(v))], \forall l \in L
\]

(5)

where \(S_l\) is a bilinear upsampling or downsampling layer which projects the feature map at layer \(l\) to a desired, common spatial dimensionality \(H'_F \times W'_F\) for the concatenation operator \([\cdot; \cdot; \cdot]\). In our work, we let \(L = \{\text{res2, res3, res4}\}\).

We also provide pseudo-code to outline our framework. In Algorithm 1, we detail our KMeans bootstrapping process (c.f. Section 3.3). In Algorithm 2, we describe the von Mises-Fisher (vMF) clustering consistency loss (c.f. Section 3.4). Finally, in Algorithm 3, we show how these two components fit into the larger framework. Input parameters and hyperparameters are described in the corresponding sections of the main text.

**Algorithm 1** Bootstrap segmentation masks (for a single batch of images \(\{I\}\)).

function BootstrapMasks(\(f_{\theta}, \{I\}, \ell, K_{\min}, K_{\max}\))

Extract feature map \(y = f[\ell](\{I\})\)

Sample \(K \sim \text{Unif}(K_{\min}, K_{\max})\) \(\triangleright\) Scale-dynamic sampling

KMeans clustering on \(y\) with \(K\) clusters to obtain feature prototypes \(P\)

\(\{M\} \leftarrow \text{Label features in } y \text{ via Euclidean-closest prototype in } P\)

Interpolate \(\{M\}\) to spatial dimensions of \(\{I\}\)

return Masks \(\{M\}\)
end function

**Algorithm 2** Clustering consistency loss (for a single batch of images \(\{I\}\)).

function ClustConsistency(\(f_{\theta}, f_{\xi}, \ell, \{I\}, K, \lambda_{\text{loss}}\))

Augmentations \(t \sim T, t' \sim T'\)

\(v = t(\{I\}), v' = t'(\{I\})\)

Extract feature maps \(y = f[\ell](v), y' = f[\ell](v')\)

KMeans clustering on \(y, y'\) with \(K\) clusters to obtain feature prototypes \(P, P'\)

\(L = \lambda_{\text{loss}}L_{\text{clus}}\) \(\triangleright\) (3)

Update \(f_{\theta}\) to minimize \(L, f_{\xi}\) as EMA of \(f_{\theta}\)
end function
Algorithm 3 CYBORGSTM Training Flow (for a single batch of images \( \{I\} \)).

\[
\text{for } i = 1 : W \text{ do} \\
\quad \text{ClustConsistency}(f_\theta, f_\xi, \ell, \{I\}, K, 1) \quad \triangleright \text{vMF warmup period} \\
\text{end for} \\
\]

Masks \( \{M\} = \text{BootstrapMasks}(f_\theta, \{I\}, \ell, K_{\text{min}}, K_{\text{max}}) \quad \triangleright \text{Section 3.3} \\
\]

\[
\text{for } t = W + 1, W + 2, \ldots \text{ do:} \\
\quad \text{if } t \% N = 0 \text{ then} \\
\quad \quad \text{Masks } \{M\} = \text{BootstrapMasks}(f_\theta, \{I\}, \ell, K_{\text{min}}, K_{\text{max}}) \\
\quad \text{end if} \\
\quad \text{Augmentations } t = \mathcal{T}, t' \sim \mathcal{T}' \\
\quad \text{v} = t(\{I\}), \text{v}' = t'(\{I\}) \\
\quad \text{Extract object-level representations } h_m, h'_m \text{ using } \{M\} \quad \triangleright 1 \\
\quad \mathcal{L} = \mathcal{L}_{\text{mask}} \quad \triangleright 2 \\
\quad \text{Update } f_\theta \text{ to minimize } \mathcal{L}, f_\xi \text{ as EMA of } f_\theta \\
\quad \text{if } t \% M = 0 \text{ then} \\
\quad \quad \text{ClustConsistency}(f_\theta, f_\xi, \ell, \{I\}, K, \lambda_{\text{vMF}}) \\
\quad \text{end if} \\
\text{end for}
\]

B Experimental Details

Linear transfer. We perform linear classification on PASCAL VOC07 and semi-supervised linear transfer on ImageNet-1k, akin to [52]. For VOC07, we extract a frozen feature map from the penultimate layer of our ResNet-50 pretrained backbone, before downsampling to a 2×2 spatial grid via adaptive average pooling. We flatten and \( \ell^2 \)-normalize the features, yielding \( \mathbb{R}^{8192} \) feature vectors. We then train per-class SVMs on the trainval split, performing a 3-fold cross validation on costs \( C \in \{0.01, 0.1, 1.0, 10.0\} \), before evaluating mAP on the test split.

For ImageNet-1k, we follow settings from [52] and finetune our backbone on the 1% and 10% training subset splits as released in [4]. We report the top-1 and top-5 accuracy on the official val split.

PASCAL VOC object detection. For object detection, we initialize a ResNet-50-C4 backbone with pretrained weights from CYBORGSTM, before inserting into a Faster RCNN architecture. We then finetune with the trainval07+12 split from PASCAL VOC, before reporting all metrics on the test2007 split via the Detectron2 API [49]. We use all default training settings from [42] for standardized comparison.

COCO instance segmentation. For segmentation, we initialize a ResNet-50-FPN backbone with pretrained weights from CYBORGSTM, before inserting into a Mask RCNN architecture. We then finetune with the train2017 split from COCO, before reporting all metrics on the val2017 split via the Detectron2 API [49]. Again, we use all default training settings from [42] for standardized comparison, except we follow a 1x training schedule with learning rate decay scheduled accordingly.
### Table C.1: Transfer Learning on CityScapes, ResNet-50 backbones.

<table>
<thead>
<tr>
<th>Method</th>
<th>Pretraining Data</th>
<th>mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) RANDOM INIT.</td>
<td>–</td>
<td>63.5</td>
</tr>
<tr>
<td>2) SUPERVISED</td>
<td>Supervised, ImageNet</td>
<td>73.7</td>
</tr>
<tr>
<td>3) SIMCLR [9]</td>
<td>ImageNet</td>
<td>73.1</td>
</tr>
<tr>
<td>4) BYOL [17]</td>
<td>ImageNet</td>
<td>71.6</td>
</tr>
<tr>
<td>5) MOCO-V2 [11]</td>
<td>ImageNet</td>
<td>74.5</td>
</tr>
<tr>
<td>6) DENSECL [46]</td>
<td>ImageNet</td>
<td>75.7</td>
</tr>
<tr>
<td>7) DETCO [51]</td>
<td>ImageNet</td>
<td>76.5</td>
</tr>
<tr>
<td>8) MOCO-V2 [11]</td>
<td>COCO</td>
<td>73.8</td>
</tr>
<tr>
<td>9) DENSECL [46]</td>
<td>COCO</td>
<td>75.6</td>
</tr>
<tr>
<td>10) BYOL [17]</td>
<td>COCO</td>
<td>72.2</td>
</tr>
<tr>
<td>11) ORL [52]</td>
<td>COCO</td>
<td>72.7</td>
</tr>
<tr>
<td>12) CYBORGs(ours)</td>
<td>COCO</td>
<td>75.9</td>
</tr>
</tbody>
</table>

### Table C.2: Transfer Learning on LVIS, ResNet-50 backbones.

<table>
<thead>
<tr>
<th>Method</th>
<th>Pretraining Data</th>
<th>LVIS, 1x schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>AP^{bb}</td>
</tr>
<tr>
<td>1) SUPERVISED</td>
<td>–</td>
<td>20.4</td>
</tr>
<tr>
<td>2) SOCO [47]</td>
<td>ImageNet</td>
<td>20.0</td>
</tr>
<tr>
<td>3) DENSECL [46]</td>
<td>COCO</td>
<td>19.8</td>
</tr>
<tr>
<td>4) BYOL [17]</td>
<td>COCO</td>
<td>20.5</td>
</tr>
<tr>
<td>5) ORL [52]</td>
<td>COCO</td>
<td>23.9</td>
</tr>
<tr>
<td>6) CYBORGs(ours)</td>
<td>COCO</td>
<td>26.3</td>
</tr>
</tbody>
</table>

### C Further Semantic Segmentation Results

#### CityScapes semantic segmentation
We further evaluate the segmentation prowess of CYBORGs representations by transferring to out-of-distribution data. We choose the real-world urban driving scenes CityScapes dataset, and follow the finetuning protocol of [46], training a FCN model on the train_fine split (2975 images) for 40k iterations before testing on the val split. We demonstrate strong results even when compared to state-of-the-art ImageNet-pretrained SSL methods, suggesting that our bootstrapping of segmentation masks can lead to representations which sensibly adapt to similar segmentation tasks under different data distributions. In particular, we outperform (+0.3 mIoU) DENSECL [46], which is a contrastive learning method specifically designed for strong downstream performance on dense prediction tasks, including object detection and semantic segmentation.

#### LVIS long-tail instance segmentation
An emerging property of self-supervised representations is their ability to generalize to the semantics of unseen, diverse object distributions [15, 43, 48]. By assessing transfer performance of CYBORGs on the LVIS long-tail instance segmentation benchmark, with 1203 object categories across ~164k images [18], we hope to verify a similar property in our representations.

We follow the standard finetuning protocol in [42], adjusting parameters for a 1x schedule. In comparison to other SSL methods pretrained on COCO, we offer a significant improvement of +3.4 AP^{bb} and +3.3 AP^{mk}.