Appendix: Self-Supervised Learning of Object Parts for Semantic Segmentation

Adrian Ziegler
Technical University of Munich
adrian.ziegler@tum.de

Yuki M. Asano
QUVA Lab
University of Amsterdam
y.m.asano@uva.nl

A. Appendix

A.1. Implementation details

Model training Our model is implemented in Torch [13] and PyTorch Lightning [6]. We use faiss [11] for K-Means clustering and the MapEquation software package [3] for community detection.

We chose to train a ViT-Small as the amount of parameters is roughly equivalent to a ResNet-50's (21M vs. 23M). Further, we use a student-teacher setup and the teacher weights are updated by the exponential moving average of the student weights following [2, 7]. The exponential moving average for updating the teacher weights is adapted with a cosine schedule starting at 0.9995 and going up to 1 i.e. a hard copy. We train the ViT-Small with a cosine learning rate schedule going down to 0 over 50 training epochs. The initial projection head learning rate is $1e{-}4$ and the backbone's learning rate is $1e{-}5$. The projection head consists out of three linear layers with hidden dimensionality of 2048 and Gaussian error linear units as activation function [9]. The output dimensionality is 256 and the resulting tensors are then passed through a l2-bottleneck and the prototype matrix C to produce cluster assignment predictions. As discussed, we use a queue for sinkhorn-knopp clustering with a length of 8192. We set the temperature to 0.1 and use Adam as an optimizer with a cosine weight decay schedule. The alignment happens to a fixed output size of 7x7 during training. This makes sure that the local and global crop feature maps have the same spatial resolution. The augmentations used are random color jitter, Gaussian blur, grayscale and multi-crop augmentations. The global crop's resolution is 224x224 and the local crop's resolution is 96x96. We generate global and local crops with the constrain that they intersect at least by 1% of the original image size to make sure that there is a non-negligible intersection where we can apply our clustering loss to. In Figure 1 we show the generation process for two ImageNet pictures.

Figure 1. **Bounding box generation example for cluster assignment alginment**. The left column shows the global crops, the right column the local crops. Each global crop has $N-1$ bounding boxes as it produces prediction targets for all remaining $N-1$ crops. Each local crop has N_{gc}, the number of global crops, bounding boxes as it is used to predict the prediction targets of each global crop.

Fully unsupervised semantic segmentation For CBFE and CD we take PVOC12 *train* to find good hyperparameter configurations i.e. clustering granularities K, the precision threshold for CBFE as well as Markov time and the co-occurrence probability threshold for CD. We use a segmentation of our embedding space to 200 clusters as we found this granularity to work best on PVOC for CBFE. Before doing foreground-focused clustering using the cluster mask, we bilinearly interpolate the embeddings to the desired mask size. For CBFE we report the precision thresholds used in Table 1. To evaluate the unsupervised saliency estimator baseline method, we use the saliency head provided by the MaskContrast authors [14]. For the computation of the Jaccard distance we assign unlabelled objects to foreground. These objects have a separate class in the

PVOC dataset.

We cluster the embedding space to 100 clusters as we found this granularity to work best on PVOC for CD. To construct the co-occurrence network, we calculate the conditional co-occurrence probability on each image and then average over all images the cluster appeared. The MapEquation software package can be instructed to constrain the number of found communities. We use this setting to find exactly as many communities as there are object categories in the given dataset (for PVOC it is 20). All clusters that are not in communities are assigned to background, which are just 4 out of 100 for our network, as we already focus clustering on foreground. We set the co-occurrence probability threshold to 5%. All edges below this threshold are ignored by Infomap. Further as stopping criterion we set the Markov time to 1.5. The other parameters are left at default value. We report results averaged over 5 seeds.

Evaluation details Since we evaluate the pre-GAP layer4 features or the spatial tokens, their output resolution does not match the mask resolution. To fix that, we do bilinear interpolation before applying the linear or FCN head or interpolate the clustering results by nearest neighbor. For a fair comparison between ResNets and ViTs, we use dilated convolution in the last bottleneck layer of the ResNet such that the spatial resolution of both network architectures match (28x28 for 448x448 input images). All overclustering results were computed using downsampled 100x100 masks to speed up the Hungarian matching as we found that the results do not differ from using full resolution masks.

We fine-tune the linear head for 25 epochs with a learning rate drop after 20 epochs and a batch size of 120. For most checkpoints we found a learning rate of 0.01 to work well except for MaskContrast [14] and MoCo-v2 [8] where we use a learning rate of 0.1. All heads were trained on downsampled 100x100 masks to increase training speed. For evaluation, we stick to 448x448 masks as it does not require Hungarian matching and is thus fast.

The FCN head is fine-tuned for 30 epochs equaling the 20k iterations used in [15]. Again we use a learning rate of 0.01 with a drop to 0.001 after 15 epochs and a batch size of 64. The design of our fully convolutional head follows [15]: We use two convolutional layers with ReLU non-linearites. Frozen features and convolved head features are then concatenated and sent through another convolutional layer. The resulting feature maps ϕ are then transformed to the desired output classes by a 1x1 convolution. During training we apply 2D-dropout on ϕ.

Method	Precision Threshold
Leopart IN	40%
Leopart IN+CC	30%

Table 1. **Precision values used for classifying clusters as foreground.**

	K=500		
arch	PVOC12	COCO-Thing	COCO-Stuff
ViT-S/16	53.5	55.9	43.6
ViT-B/8	**59.7**	**56.8**	**45.9**

(a) Overclustering results on PVOC, COCO-Thing and COCO-Stuff. The results are comparable to Tab. 3 in the paper.

	ViT-S/16	ViT-B/8
DINO	4.6	**5.3**
+ Leopart	18.9	**21.2**
+ CBFE	36.2	**40.4**
+ CD	38.4	**40.3**

(b) Fully unsupervised semantic segmentation results on PVOC.

Method	arch	Jacc. (%)
Leopart IN CBFE	ViT-S/16	58.6
Leopart CC CBFE	ViT-S/16	58.2
Leopart CC CBFE	ViT-B/8	**63.0**

(c) Foreground extraction results on PVOC. The results are comparable to Table 7 in the paper.

Table 2. **Comparison of ViT-S/16 and ViT-B/8 performances.** We further improve state-of-the-art on all experiments by training a larger model with our loss and running CBFE and CD.

A.2. Additional Experiments

Fine-Tuning a larger backbone To push the boundaries of state-of-the-art even further, we fine-tune a ViT-Base with patch size 8 (ViT-B/8) for 100 epochs with Leopart. We start again from a DINO initialization. The results are reported in Tab. 6 in the paper and Tab. 2. Training a larger backbone boosts transfer learning performance by up to 6% on PVOC as shown in Tab. 2a. The gains on COCO-Thing and COCO-Stuff are around 1% and 2% respectively.
For fully unsupervised semantic segmentation, training a larger backbone even shows more relative gain than training a ViT-Small. This is apparent from the 35% relative gain for a ViT-B/8 and the 33.8% relative gain for a ViT-S/16 over their respective DINO intializations, as can be deduced from Tab. 2b. Overall, we are able to improve state-of-the-art by additional 2% just by taking a larger model. Interestingly, CD cannot improve over CBFE indicating that the choice of hyperparameters for community detection might not be optimal. We leave this to future work.
A larger model also improves foreground extraction using our CBFE method by more than 4% as shown in Tab. 2c.

Leopart with different initializations To show the generality and robustness of our approach, we fine-tune with Leopart starting from a Moco-v3, MAE and supervised initialization. The results are shown in Table 3. Leopart is good at fine-tuning even more recent SSL methods and larger pretrained backbones like MAE (where our method adds +28% in K=500 performance). Our method is even

Figure 2. **More cluster masks for PVOC12 *val* obtained by our CBFE method.** Overall, the masks capture the object shape well but at times they include too much background. Also small objects are not detected at times as can be seen in the first picture from the right in the first row, which is a limitation discussed.

Init	Arch	At init. LC	At init. K=500	After Leopart LC	After Leopart K=500
Superv.	ViT-S/16	68.1	55.1	72.5	61.6
MoCo-v3 [5]	ViT-S/16	13.4	5.8	42.0	31.2
MAE [4]	ViT-B/16	47.5	10.0	68.9	38.4

Table 3. **Transfer learning results starting from various initializations**. Leopart consistently improves upon the initialization (init.) and thus shows the generality of our method. Comparable to Tab. 3 in the paper.

able to boost the performance of a ViT pretrained with supervision showing the wide applicability of our dense loss.

DenseCL with DINO init For further comparison to our closest comptetitor in transfer learning, DenseCL [15], we trained a ViT with DINO initalization using their loss and following the setting of Tab. 3 for PVOC12. We find a performance of 54% and 17.1% for LC and K=500 evaluation respectively, *i.e*. fine-tuning with Leopart still outperforms by >15% for LC and >40% for K=500. These results indicate that DenseCL (perhaps due to its global-pooled loss term) does not seem apt for fine-tuning as it barely improves the DINO init (+3.4% for LC and −0.3% for K=500).

Queue usage ablation. We show that the usage of a queue improves our results as shown in Table 4, comparable to the experiments of Table 1 in the main paper. This means that enough diversity for equi-partitioned clustering can be achieved with this simple mechanism.

queue	LC	Num. clusters 100	Num. clusters 300	Num. clusters 500
✗	67.2	35.0	45.7	48.1
✓	**67.8**	**38.2**	**47.2**	**50.7**

Table 4. **Queue Ablation**. A clustering queue improves performance.

Method	ADE20k-Street K=500	ADE20k-Street K=1000
Random ViT	1.5	2.0
Sup. ViT	5.4	7.2
DINO [2]	5.7	7.0
Leopart IN	<u>6.9</u>	<u>9.3</u>
Leopart CC	**7.6**	**10.0**

Table 5. **Ade20k overclustering results.** Evaluated on 111 parts classes taken from ADE20k street scenes.

Predicting ADE20k parts To quantitatively support our claim that we learn object parts, we run experiments on Ade20K [16] street scenes that feature annotations for 111 different part classes and 1983 images. We pretrain on COCO and report overclustering results given ground-truth parts annotations. As shown in Table 5, Leopart improves DINO's parts mIoU by 1.9% and 3% with a clustering granularity of $K = 500$ and $K = 1000$ respectively. This shows that our method increases object part correspondence. Interestingly, our gain improves with higher clustering granularity. Also, while the supervised ViT outperformed DINO in transfer learning it is not superior when it comes to discovering object parts.

A.3. Additional visualizations

We provide further cluster masks in Figure 2 and segmentation map visualizations on PVOC12. Next to community detection results shown in Figure 4, we also show unmerged foreground clustering results with K=100 in Figure 3 to give the reader an impression of the segmentation granularities of each object. In Figure 5, we also show segmentation maps obtained from classic overclustering results by grouping clusters to objects using label information.

A.4. Datasets Details

A.4.1 PASCAL

For fine-tuning linear heads as well as the FCN head, we use the *trainaug* split featuring 10582 images and their annotations. We evaluate on PVOC12 *val* that has 1449 images. During evaluation we ignore unlabelled objects as well and the boundary class following [14]. For hyperparameter tuning of our fully unsupervised segmentation method, we use the PVOC12 *train* split with 1464 images.

A.5. COCO

We use the COCO benchmark in two ways to further isolate different object definitions. For instance, COCO-thing has one class for vehicles whereas PVOC distinguishes between boats, busses and cars. Also, things have a fundamentally different object definition as stuff. First, we extract stuff annotations i.e. object w/o a clear boundary, often in the background. For that, we use the COCO-Stuff annotations [1]. We further merge the 91 fine labels to 15 coarse labels, as in [10]. We also assign the coarse label "other" to non-stuff object as the label does not carry any semantic meaning. The resulting labels are:

```
['water', 'structural', 'ceiling',
'sky', 'building', 'furniture-stuff',
'solid', 'wall', 'raw-material',
'plant', 'textile', 'floor',
'food-stuff', 'ground', 'window']
```

Non-Stuff objects are ignored during training and evaluation.

Second, we extract foreground annotations by using the panoptic labels provided by [12]. We merge the instance-level annotations to an object category with a script the authors provided. Further, we merge the 80 fine categories to coarse categories obtaining 12 unique object classes:

```
['electronic', 'kitchen', 'appliance',
'sports', 'vehicle', 'animal',
'food', 'furniture', 'person',
'accessory', 'indoor', 'outdoor']
```

The background class is ignored during training and evaluation.

We fine-tune the linear and FCN head on a subset of 10% of the data i.e. 11829 images. We evaluate on the full 5000 validation images.

A.5.1 ADE20k

Overall, ADE20k features 3687 different objects that can act as parts. We constrain our evaluation to street scenes that contain parts annotations. This reduces our data to 1983 images and to the following 111 object parts:

```
['arcades', 'arch', 'arm',
'back', 'balcony', 'balustrade',
'bars', 'base', 'basket',
'bell', 'bicycle path', 'blind',
'blinds', 'branch', 'bumper',
'chimney', 'cloud', 'clouds',
'column', 'columns', 'cornice',
'crosswalk', 'dome', 'door',
'door frame', 'doorbell', 'dormer',
'double door', 'drain pipe', 'eaves',
'entrance', 'entrance parking',
'exhaust pipe', 'face', 'fence',
'fender', 'fire bell', 'fire escape',
'garage door', 'garage doors',
'gas cap', 'gate', 'grille',
'ground', 'gutter', 'handle', 'head',
'headboard', 'headlight', 'hip tiles',
'hood', 'house number', 'housing',
'housing lamp', 'lamp',
'lamp housing', 'lattice', 'left arm',
'left foot', 'left hand', 'left leg',
'license plate', 'logo',
'metal shutter', 'metal shutters',
'mirror', 'pipe', 'pipe drain',
'pole', 'porch', 'post', 'railing',
'rain pipe', 'revolving door',
'right arm', 'right foot',
'right hand', 'right leg', 'rim',
'road', 'roof', 'roof rack',
'rose window', 'saddle',
'shop window', 'shutter', 'shutters',
'sidewalk', 'sign', 'skylight',
'staircase', 'steering wheel',
'step', 'steps', 'taillight',
'terrace', 'torso', 'tower', 'tree',
'trunk', 'vent', 'wall', 'wheel',
'window', 'window scarf', 'windows',
'windshield', 'wiper', 'car',
'buildings', 'building']
```

Figure 3. **K=100 overclustering visualization without merging clusters to objects.** Note that the cluster colors are not unique as we have 100 different clusters: Same cluster means same color but not the other way around. Interestingly, Leopart learns a different segmentation granularity depending on the object category. For instance, cars and humans are segmented into various parts, but birds are usually kept whole.

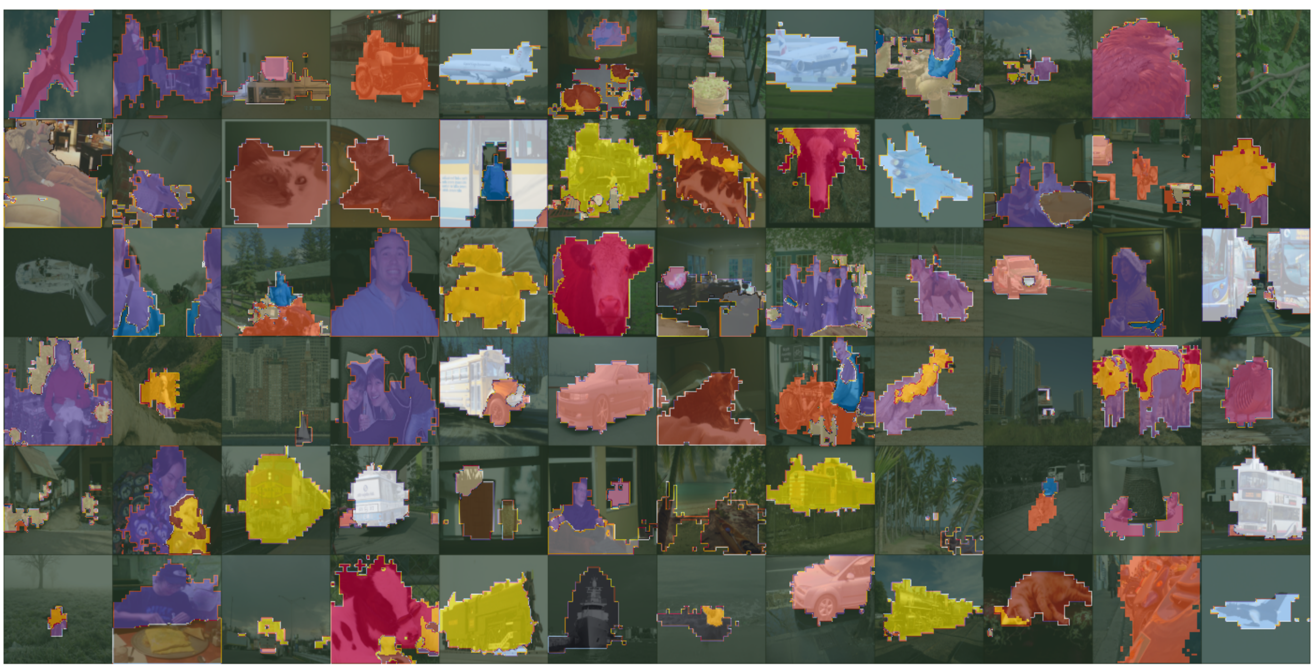

Figure 4. **More fully unsupervised segmentation results obtained through our community detection method.** Our method, manages to merge the object parts clusters from Figure 3 to objects in most of the cases. However, as our method does a hard cluster to community assignment, each cluster can only be used for one object. This limitation can be seen for the car wheel class in the 4th row and 5th and 6th pictures from the right. The bus' wheel is mistakenly assigned to the car category. Also, objects that share many parts such as bicycles and motorcycles are mistakenly merged to one category.

Figure 5. **Overclustering results by merging 500 clusters using ground-truth labels.** The resulting segmentation maps are more crisp than their fully unsupervised counterparts in Figure 4. Further, similar object categories are not merged together. However at times, the object is not fully segmented but just parts of it. This is likely due to the fact that some clusters segmenting an object also have a significant background overlap and thus our precision-based cluster matching matches them to the background class.

Figure 6 shows some exemplary street scene images and their corresponding parts masks. During evaluation of our feature space clustering we ignore non-part pixels shown in dark green.

References

[1] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1209–1218, 2018. 4

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 1, 3

[3] A. Eriksson D. Edler and M. Rosvall. The mapequation software package. *GitHub. available online at http://www.mapequation.org and https://github.com/mapequation/infomap*. 1

[4] He et al. Masked autoencoders are scalable vision learners. *arXiv:2111.06377*, 2021. 3

[5] Xinlei Chen et al. An empirical study of training self-supervised vision transformers. *ICCV*, 2021. 3

[6] et al. Falcon, WA. Pytorch lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3, 2019. 1

[7] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *NeurIPS*, 2020. 1

[8] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning, 2020. 2

(a) Ade20k street scene images

(b) Ade20k street scene parts masks

Figure 6. **ADE20k street scene images and masks data visualized as used for overclustering results reported in Table 5.**

[9] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 1

[10] Xu Ji, João F. Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *ICCV*, 2019. 4

[11] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017. 1

[12] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *CVPR*, pages 9404–9413, 2019. 4

[13] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. 2019. 1

[14] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. Unsupervised semantic segmentation by contrasting object mask proposals. In *ICCV*, 2021. 1, 2, 4

[15] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *CVPR*, pages 3024–3033, 2021. 2, 3

[16] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, pages 633–641, 2017. 3