A. Appendix

A.1. SemCL Pseudocode

The forward-backward propagation of SemCL is given in Algorithm 1.

Algorithm 1 SemCL: PyTorch-style Pseudocode

```
# Employ MoCo v3 framework
# f_b: base encoder := backbone + proj mlp + pred mlp
# f_m: momentum encoder := backbone + proj mlp
# m: momentum coefficient
for x_a, x_na in dataloader: # load a minibatch x_a, x_na with batchsize N
    # x_a: anchor images, x_na: ¬anchor images
    # anchor images' forward process
    x_a0, x_a1, x_a2 = aug_0(x_a), aug_1(x_a), aug_2(x_a) # augmentation: [N, 3, H, W] each
    q_a0, q_a1, q_a2 = f_b(x_a0), f_b(x_a1), f_b(x_a2) # queries: [N, C] each
    k_a0, k_a1, k_a2 = f_m(x_a0), f_m(x_a1), f_m(x_a2) # keys: [N, C] each
    # ¬anchor images' forward process
    x_na0, x_na1, x_na2 = aug_0(x_na), aug_1(x_na), aug_2(x_na)
    q_na0, q_na1, q_na2 = f_b(x_na0), f_b(x_na1), f_b(x_na2)
    k_na0, k_na1, k_na2 = f_m(x_na0), f_m(x_na1), f_m(x_na2)

    # stack keys
    pos_kstk=stack([k_a0,k_a1,k_a2],dim=1) # [N, 3, C]
    neg_kstk=stack([k_na0,k_na1,k_na2],dim=1) # [N, 3, C]

    loss=(paried_infonce(q_a0,k_a1,neg_kstk)
          +paried_infonce(q_a1,k_a2,neg_kstk)
          +paried_infonce(q_a2,k_a0,neg_kstk)
          # symmetrized
          +paried_infonce(q_na0,k_na1,pos_kstk)
          +paried_infonce(q_na1,k_na2,pos_kstk)
          +paried_infonce(q_na2,k_na0,pos_kstk))
    loss.backward()

    update(f_b) # optimizer update: f_b
    f_m = m*f_m + (1-m)*f_b # momentum update: f_m
```

A.2. Implementation of Paired InfoNCE Loss

A pseudocode is listed in Algorithm 2. The difference between unpaired and paired InfoNCE losses lies in the calculation of negative logits. N and D denote the batch size and the embedding dimension respectively. For a query of the shape $[N, D]$ having M paired negative samples, negative logits are the matrix multiplication (@) between a new dimension inserted query at $dim = 1$ and negative keys of the shape $[N, M, D]$. After removing the inserted dimension from the negative logits, it is concatenated with the positive logits as $(M + 1)$ dimension logits, where the first element $[:, 0]$ is the positive logit. Finally, the InfoNCE loss is also implemented by the cross-entropy loss [17], where labels are set to the indices of the positive logits. $\hookrightarrow$ is the newline symbol.

Algorithm 2 Paired InfoNCE Loss

```
# query: [N, D] Tensor with query samples
# pos_key: [N, D] Tensor with positive samples
# neg_keys: [N, M, D] Tensor with negative samples
# τ: temperature
def paried_infonce(query, pos_key, neg_keys, temperature=τ):
    # Logits of positive pairs
    pos_logit = sum(query * pos_key, dim=1) # pos_logit:[N, 1]
    query = query.unsqueeze(1) # query: [N, 1, D]
    neg_logits = query @ transpose(neg_keys)# [N, 1, M]
    neg_logits = neg_logits.squeeze(1)# [N, M]

    logits = concat([pos_logit, neg_logits], dim=1)# [N, 1+M]
    # Ground truth class indices
    labels = zeros(len(logits))# [N]

    return cross_entropy(logits / temperature, labels)

def transpose(x):
    return x.transpose(-2, -1)
```

A.3. Downstream Tasks Setups

A.3.1 Semantic Segmentation

To maintain consistency with previous studies, we adopt the Deeplabv3+ [3] structure for ResNets and UPerNet [19] for Swin Transformers [11] in the semantic segmentation task. All-layer fine-tune is conducted end-to-end on training sets of VOC2012, Cityscapes, ADE20K and COCO 2017 respectively. The data augmentation strategy is as follows: rescale each sample by its short edge with a random ratio from the range $[0.5, 2.0]$, then random crop to 769×769 for Cityscapes and 513×513 for the others, and random horizontal flip. We adopt AdamW [12] with $wd = 1.0 \times 10^{-4}$, and base $lr = 5 \times 10^{-5}$. The learning rate schedule is step (multiple base lr by 0.1 at 75% and 90% of training) for Cityscapes and OneCycle [15] for the other datasets. All models are tuned for $80k$ iterations on corresponding benchmark with batch size 16.

A.3.2 Object Detection and Instance Segmentation

We use `mmdetection` [2] as our codebase for object detection and instance segmentation tasks. SemCL backbones of ResNet50 and Swin-T, pretrained on corresponding SemCL

sub-datasets, are fine-tuned on training sets of VOC07+12, Cityscapes and COCO 2017, respectively. The model is Faster R-CNN [13] for VOC and Cascade Mask R-CNN [1, 8] for Cityscapes and COCO. For VOC and COCO, the image scale (short edge) is $[480, 800]$ during training and 800 during inference. For Cityscapes, the image scale is $[704, 1024]$ during training and 1024 during inference. The evaluation metric of object detection is AP_{50} for VOC and AP^{box} for Cityscapes and COCO. For instance segmentation, the evaluation metric is AP^{mask}. We adopt AdamW [12] with $wd = 0.05$. The training schedule is $3\times$(36 epochs for VOC and COCO, 192 epochs for Cityscapes) with a step learning rate schedule (multiple base lr by 0.1 at 75% and 90% of training) with base $lr = 1 \times 10^{-4}$ for Cityscapes and $lr = 6 \times 10^{-5}$ for VOC and COCO. The batch size is 16.

A.3.3 Depth Estimation

We use the `Monocular Depth Estimation Toolbox` [9] as our code base for depth estimation tasks. Considering that Binsformer [10] achieves previous state-of-the-art results, we adopt it as the model for SemCL pretrained ResNet50, Swin-T and Swin-L. SemCL backbones pretrained on SemCL-City are fine-tuned and evaluated on test sets of Cityscapes and KITTI [16, 6], and those pretrained on SemCL-COCO are fine-tuned and evaluated on test set of NYUv2 [14]. The crop size (short edge) for Cityscapes and KITTI is $[352, 704]$, and $[416, 544]$ for NYUv2. We use AdamW with $wd = 0.01$ and OneCycle [15] learning rate schedule with base $lr = 5 \times 10^{-5}$ for all benchmarks. The evaluation metrics are Absolute Relative Error (AbsRel) and Root Mean Square Error (RMSE). The batch size is 16.

A.4. Ablation Study

In this part, we ablate the following aspects of SemCL: the performance difference between paired and unpaired (as in MoCo v3 [5]) InfoNCE loss under the SemCL framework (see Section A.4.1), pretraining batch size (see Section A.4.2) since paired InfoNCE loss intrinsically decouples the number of negative samples from batch size, and training length and dataset scale (see Sections A.4.3 and A.4.4) to investigate model saturation under the SemCL framework.

Setup. As Cityscapes supports all downstream tasks, Swin-T is pretrained on the SemCL-City dataset for 100 epochs with batch size 128 (for faster pretraining) as baseline settings, unless otherwise specified. Other settings remain the same as in the pretext task (see Section 3.3).

A.4.1 Paired/unpaired InfoNCE loss

For instance discrimination [18] task, ImageNet is a prevalent choice since IN samples are characteristic by centered main subject and less surroundings information. As for the SemCL framework, the goal of the pretext task is to endow the model with the ability to distinguish an object from its surroundings, for which we adopt datasets whose samples contain more surrounding info with semantic labels. By exploiting the small-scale semantic information, object-environment pairs are compared at the sub-scene level. In this way, paired InfoNCE loss is applied to push the subject away from its surroundings in the embedding space.

The difference between paired and unpaired InfoNCE losses in the SemCL framework is illustrated in Table 1. Compared to the models pretrained with paired InfoNCE loss, the models pretrained with unpaired InfoNCE loss performs worse in all downstream tasks. The unpaired pretrained model is even not a patch on the ImageNet supervised pretrained counterparts in the depth estimation task. Although the unpaired model is on par with the paired one in instance segmentation, its AP^{box} is 0.5 points lower than the paired model. As for the semantic segmentation task, the gain of the unpaired model is small compared to the IN pretrained baseline, while the paired model achieves an improvement of 0.28 points.

A.4.2 Batch Size

In the MoCo series [7, 4, 5], the pretext task is instance discrimination [18]. In each batch of N+1 samples, each sample, together with its augmentations, is compared with

Table 1: Paired/unpaired InfoNCE loss vs. semantic segmentation, object detection and depth estimation (lower is better) on Cityscapes using Swin-T.

(a) Semantic segmentation, object detection and instance segmentation.

InfoNCE	Semantic seg.	Object det.	
	mIoU	AP^{box}	AP^{mask}
IN-22k	78.67	45.9	39.8
Unpaired	78.60(-0.07)	45.7(−0.2)	40.2(+0.4)
Paired	79.17(+0.50)	46.2(+0.3)	40.2(+0.4)

(b) Depth estimation.

InfoNCE	Dep. estimation	
	Abs Rel	RMSE
IN-22k	0.134	4.643
Unpaired	0.138(+0.004)	4.660(+0.017)
Paired	0.133(−0.001)	4.575(−0.068)

Table 2: Training length vs. semantic segmentation, object detection and depth estimation (lower is better) on Cityscapes using Swin-T.

Training length(ep)	Semantic seg. mIoU	Instance seg. AP^{mask}	Dep. estimation Abs Rel	RMSE
IN-22k	78.67	39.8	0.134	4.643
100	79.17(+0.50)	40.2(+0.4)	0.133(−0.001)	4.575(−0.068)
300	79.53(+0.86)	40.4(+0.6)	0.132(−0.002)	4.563(−0.080)

the remaining N negative samples. And the results of the ImageNet linear classification protocol have a positive correlation with the number of negative samples (batch size) [7]. But things are different for SemCL, the mechanism of contrasting at the sub-scene level decouples the number of negative samples from the batch size.

The results of the downstream tasks vs. pretraining batch size are shown in Figure 1. In Figures 1a to 1c the results of the SemCL models are insensitive to the pretraining batch size. The slight deterioration of the RMSE in Figure 1b and AP^{mask} in Figure 1c from $batchsize = 256$ may be related to the combination of SyncBN and gradient accumulation during pretraining. SemCL is friendly to those who suffer from insufficient GPU memory.

A.4.3 Training Length

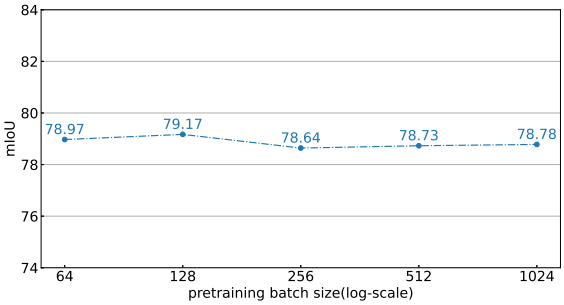
In Table 2 we report the performance on downstream tasks vs. training length. Swin-T benefits from longer pretraining in both the instance segmentation and depth estimation tasks.

A.4.4 Dataset Scale

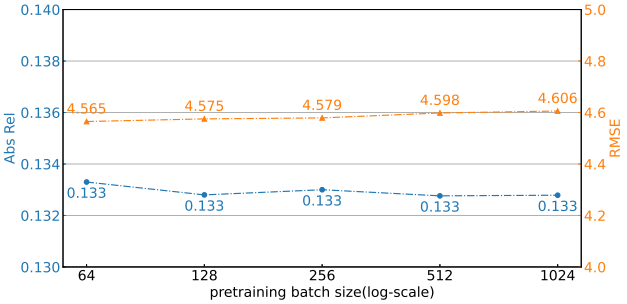
In Section 4.2, the performance of SemCL pretrained ResNet50 and Swin-T lags behind their IN pretrained counterparts, which is considered to be due to insufficient pretraining samples in the SemCL-VOC dataset. In Table 3 we perform the comparison between the pretraining datasets SemCL-VOC (14,203 pairs) and SemCL-Stuff (a mixture of SemCL-VOC, SemCL-ADE and SemCL-COCO, 996,633 pairs) on the VOC2012 semantic segmentation and VOC2007 object detection tasks. Swin-T is pretrained on the two datasets for 30k iterations (273 epochs for SemCL-VOC and 4 epochs for SemCL-Stuff) respectively. The model pretrained on SemCL-Stuff significantly outperforms the model pretrained on SemCL-VOC. Swin-T enjoys the benefit of pretraining on a larger datasets.

References

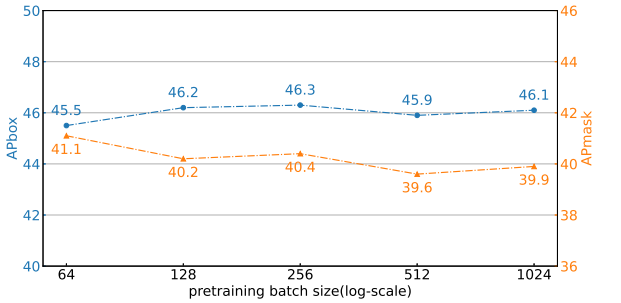
[1] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018. 2



(a) Semantic segmentation



(b) Depth estimation



(c) Object detection&instance segmentation

Figure 1: Comparison of pretraining batch sizes in downstream tasks. Due to hardware limitations, pretraining batch sizes of 256, 512 and 1024 are achieved through gradient accumulation.

Table 3: Dataset scale vs. semantic segmentation (Pascal VOC2012 val) and object detection (Pascal VOC2007 test) using Swin-T.

Pretraining dataset	Semantic Seg. mIoU	Object det. AP_{50}
IN-22k	80.90	86.58
SemCL-VOC	81.57(+0.67)	86.30(−0.28)
SemCL-Stuff	81.70(+0.80)	86.69(+0.11)

[2] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 1

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018. 1

[4] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 2

[5] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9640–9649, 2021. 2

[6] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 2

[7] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 2, 3

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2

[9] Zhenyu Li. Monocular depth estimation toolbox. https://github.com/zhyever/Monocular-Depth-Estimation-Toolbox, 2022. 2

[10] Zhenyu Li, Xuyang Wang, Xianming Liu, and Junjun Jiang. Binsformer: Revisiting adaptive bins for monocular depth estimation. *arXiv preprint arXiv:2204.00987*, 2022. 2

[11] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 1

[12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 1, 2

[13] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 2

[14] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. *ECCV (5)*, 7576:746–760, 2012. 2

[15] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE, 2019. 1, 2

[16] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *International Conference on 3D Vision (3DV)*, 2017. 2

[17] Aaron Van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv e-prints*, pages arXiv–1807, 2018. 1

[18] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018. 2

[19] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 1