A. Appendix

A.1. Implementation: Object detection backbones

The R50-dilated-C5 and R50-C4 backbones are similar to those available in `Detectron2` [60]: (i) *R50-dilated-C5*: the backbone includes the ResNet conv_5 stage with a dilation of 2 and stride 1, followed by a 3×3 convolution (with BN) that reduces dimension to 512. The box prediction head consists of two hidden fully-connected layers. (ii) *R50-C4*: the backbone ends with the conv_4 stage, and the box prediction head consists of the conv_5 stage (including global pooling) followed by a BN layer.

A.2. Implementation: COCO keypoint detection

We use Mask R-CNN (keypoint version) with R50-FPN, implemented in [60], fine-tuned on COCO `train2017` and evaluated on `val2017`. The schedule is $2\times$.

A.3. Implementation: COCO dense pose estimation

We use DensePose R-CNN [1] with R50-FPN, implemented in [60], fine-tuned on COCO `train2017` and evaluated on `val2017`. The schedule is "s1$\times$".

A.4. Implementation: LVIS instance segmentation

We use Mask R-CNN with R50-FPN, fine-tuned in LVIS [27] `train_v0.5` and evaluated in `val_v0.5`. We follow the baseline in [27] (arXiv v3 Appendix B).

LVIS is a new dataset and model designs on it are to be explored. The following table includes the relevant ablations (all are averages of 5 trials):

pre-train	BN	1× schedule AP^{mk}	1× schedule $\text{AP}^{\text{mk}}_{50}$	1× schedule $\text{AP}^{\text{mk}}_{75}$	2× schedule AP^{mk}	2× schedule $\text{AP}^{\text{mk}}_{50}$	2× schedule $\text{AP}^{\text{mk}}_{75}$
super. IN-1M	frozen	24.1	37.3	25.4	24.4	37.8	25.8
super. IN-1M	tuned	23.5	36.6	24.8	23.2	36.0	24.4
MoCo IN-1M	tuned	23.2	36.0	24.7	24.1	37.4	25.5
MoCo IG-1B	tuned	24.3	37.4	25.9	**24.9**	**38.2**	**26.4**

A supervised pre-training baseline, end-to-end tuned but with BN frozen, has 24.4 AP^{mk}. But tuning BN in this baseline leads to worse results and overfitting (this is unlike on COCO/VOC where tuning BN gives better or comparable accuracy). MoCo has 24.1 AP^{mk} with IN-1M and 24.9 AP^{mk} with IG-1B, both outperforming the supervised pre-training counterpart under the same tunable BN setting. Under the best individual settings, MoCo can still outperform the supervised pre-training case (24.9 *vs*. 24.4, as reported in Table 6 in Sec 4.2).

A.5. Implementation: Semantic segmentation

We use an FCN-based [43] structure. The backbone consists of the convolutional layers in R50, and the 3×3 convolutions in conv_5 blocks have dilation 2 and stride 1. This is followed by two extra 3×3 convolutions of 256 channels, with BN and ReLU, and then a 1×1 convolution for per-pixel classification. The total stride is 16 (FCN-16s [43]). We set dilation = 6 in the two extra 3×3 convolutions, following the large field-of-view design in [6].

Training is with random scaling (by a ratio in [0.5, 2.0]), cropping, and horizontal flipping. The crop size is 513 on VOC and 769 on Cityscapes [6]. Inference is performed on the original image size. We train with mini-batch size 16 and weight decay 0.0001. Learning rate is 0.003 on VOC and is 0.01 on Cityscapes (multiplied by 0.1 at 70-th and 90-th percentile of training). For VOC, we train on the `train_aug2012` set (augmented by [30], 10582 images) for 30k iterations, and evaluate on `val2012`. For Cityscapes, we train on the `train_fine` set (2975 images) for 90k iterations, and evaluate on the `val` set. Results are reported as averages over 5 trials.

A.6. iNaturalist fine-grained classification

In addition to the detection/segmentation experiments in the main paper, we study fine-grained classification on the iNaturalist 2018 dataset [57]. We fine-tune the pre-trained models end-to-end on the `train` set (~437k images, 8142 classes) and evaluate on the `val` set. Training follows the typical ResNet implementation in PyTorch with 100 epochs. Fine-tuning has a learning rate of 0.025 (*vs*. 0.1 from scratch) decreased by 10 at the 70-th and 90-th percentile of training. The following is the R50 result:

pre-train	rand init.	$\text{super.}_{\text{IN-1M}}$	$\textbf{MoCo}_{\text{IN-1M}}$	$\textbf{MoCo}_{\text{IG-1B}}$
accuracy (%)	61.8	**66.1**	65.6	65.8

MoCo is ~4% better than training from random initialization, and is closely comparable with its ImageNet supervised counterpart. This again shows that MoCo unsupervised pre-training is competitive.

A.7. Fine-tuning in ImageNet

Linear classification on frozen features (Sec. 4.1) is a common protocol of evaluating unsupervised pre-training methods. However, in practice, it is more common to fine-tune the features end-to-end in a downstream task. For completeness, the following table reports end-to-end fine-tuning results for the 1000-class ImageNet classification, compared with training from scratch (fine-tuning uses an initial learning rate of 0.03, *vs*. 0.1 from scratch):

pre-train	random init.	$\textbf{MoCo}_{\text{IG-1B}}$
accuracy (%)	76.5	**77.3**

As here ImageNet is the downstream task, the case of MoCo pre-trained on IN-1M does not represent a real scenario (for reference, we report that its accuracy is 77.0% after fine-tuning). But unsupervised pre-training in the *separate*, unlabeled dataset of IG-1B represents a typical scenario: in this case, MoCo improves by 0.8%.

pre-train	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{mk}	AP^{mk}_{50}	AP^{mk}_{75}
random init.	36.7	56.7	40.0	33.7	53.8	35.9
super. IN-1M	40.6	61.3	44.4	36.8	58.1	39.5
MoCo IN-1M	40.8 (+0.2)	61.6 (+0.3)	44.7 (+0.3)	36.9 (+0.1)	58.4 (+0.3)	39.7 (+0.2)
MoCo IG-1B	41.1 (+0.5)	61.8 (+0.5)	45.1 (+0.7)	37.4 (+0.6)	59.1 (+1.0)	40.2 (+0.7)

(a) Mask R-CNN, R50-FPN, **2**× schedule

pre-train	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{mk}	AP^{mk}_{50}	AP^{mk}_{75}
random init.	41.4	61.9	45.1	37.6	59.1	40.3
super. IN-1M	41.9	62.5	45.6	38.0	59.6	40.8
MoCo IN-1M	42.3 (+0.4)	62.7 (+0.2)	46.2 (+0.6)	38.3 (+0.3)	60.1 (+0.5)	41.2 (+0.4)
MoCo IG-1B	42.8 (+0.9)	63.2 (+0.7)	47.0 (+1.4)	38.7 (+0.7)	60.5 (+0.9)	41.3 (+0.5)

(b) Mask R-CNN, R50-FPN, **6**× schedule

Table A.1. Object detection and instance segmentation fine-tuned on COCO: **2× *vs.* 6× schedule**. In the brackets are the gaps to the ImageNet supervised pre-training counterpart. In green are the gaps of at least **+0.5** point.

A.8. COCO longer fine-tuning

In Table 5 we reported results of the 1× (~12 epochs) and 2× schedules on COCO. These schedules were inherited from the original Mask R-CNN paper [32], which could be suboptimal given later advance in the field. In Table A.1, we supplement the results of a 6× schedule (~72 epochs) [31] and compare with those of the 2× schedule.

We observe: (i) fine-tuning with ImageNet-supervised pre-training still has improvements (41.9 AP^{bb}); (ii) training from scratch largely catches up (41.4 AP^{bb}); (iii) the MoCo counterparts improve further (*e.g.*, to 42.8 AP^{bb}) and have larger gaps (*e.g.*, +0.9 AP^{bb} with 6×, *vs.* +0.5 AP^{bb} with 2×). Table A.1 and Table 5 suggest that the MoCo pre-trained features can have *larger* advantages than the ImageNet-supervised features when fine-tuning *longer*.

A.9. Ablation on Shuffling BN

Figure A.1 provides the training curves of MoCo with or without shuffling BN: removing shuffling BN shows obvious overfitting to the pretext task: training accuracy of the pretext task (dash curve) quickly increases to >99.9%, and the kNN-based validation classification accuracy (solid curve) drops soon. This is observed for both the MoCo and end-to-end variants; the memory bank variant implicitly has different statistics for q and k, so avoids this issue.

These experiments suggest that without shuffling BN, the sub-batch statistics can serve as a "signature" to tell which sub-batch the positive key is in. Shuffling BN can remove this signature and avoid such cheating.

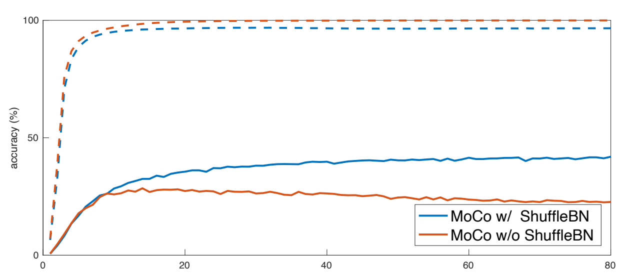

Figure A.1. **Ablation of Shuffling BN**. *Dash*: training curve of the pretext task, plotted as the accuracy of $(K{+}1)$-way dictionary lookup. *Solid*: validation curve of a kNN-based monitor [61] (not a linear classifier) on ImageNet classification accuracy. This plot shows the first 80 epochs of training: training longer without shuffling BN overfits more.