Does Robustness on ImageNet Transfer to Downstream Tasks?

Supplementary Material

A. Training Details

Object Detection on MS-COCO 2017. For Swin Transformer, we employ the setting used in [23]. For ResNet50-based models, we train Mask-RCNN using the default model and hyperparameter configurations from mmdetection [3], except for the learning rate of SGD, which we set to 0.04 based on the grid-search over 0.01, 0.04, and 0.08. We replace random initialization with pretrained weights from DeepAug+AugMix, ANT, or adversarially-trained models.

Semantic Segmentation on ADE20K. For Swin Transformer, we employ the setting used in [23]. For ResNet-50-based models, we train UperNet using the default model and hyperparameter configurations from mmsegmentation [4], except for the backbone type, which we set to ResNet50. We replace random initialization with pretrained weights from DeepAug+AugMix and ANT.

Image Classification on CIFAR10. We train all models using the same configuration. Following the recommendations made by [30], we use batch size of 64, momentum of 0.9, weight decay of 5e-4 and the learning rate of 0.01, which we reduce by a factor of 10 every 50 epochs.

B. Additional Experiments

B.1. Degree of fine-tuning

In this section, we study how the degree of fine-tuning affects the performance of robust transfer for the full-network transfer learning, where we use pretrained, robustified ImageNet weights as our initialization. It is reasonable to expect that when we use a small learning rate for fine-tuning, we should be able to retain some of the robustness properties that the robustified ImageNet models have. On the other hand, small learning rates slow down training processes and can fail to reach convergence under limited computational resources. To investigate this trade-off, we train PRIME-pretrained ImageNet models with varying learning rates under the same computational budget. The results are shown in Fig. 5. As expected, the smaller the learning rate, the lower the clean performance becomes, and consequently the raw performance on the corrupted data also decreases at least for the COCO object detection task. However, the percentage of the performance drop becomes less significant for the small learning rate cases. This partially confirms our intuition. We leave as future work maintaining robustness while improving performance on downstream tasks for transfer learning.

B.2. Data augmentation during fine-tuning

While we focus on studying how much robustness transfer learning can retrain from robust pretrained ImageNet models to downstream tasks, it is also reasonable to simply use robustification techniques during transfer learning since our ultimate goal is to have a robust model on downstream tasks. We use PRIME [26] as a data augmentation technique that is intended to robustify image classification models. We chose this method because it does not require to modify the loss function unlike ANT and AugMix, and is much simpler than DeepAug. The results are shown in Fig. 6. We use three ImageNet pretrained weights to initialize ResNet50, where Regular is a standard ImageNet pretrained weights, and DeepAug+ and PRIME indicate that the corresponding ImageNet training incorporates either DeepAug+ or PRIME data augmentation. Swin-T is pretrained on ImageNet, but without any robustification technique during pre-training. For transfer learning from Swin-T, we use PRIME.

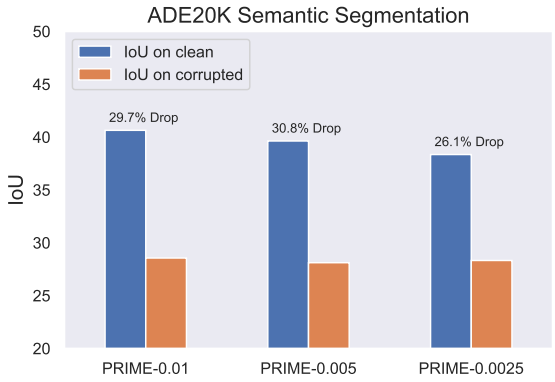
In Table 4 and 5, we compare the effect of the PRIME data augmentation during transfer learning. While applying PRIME during transfer learning mitigates the performance drop from the clean data to corrupted data, we can also see that it significantly hurts the clean performance on COCO. Interestingly, for ADE20K semantic segmentation, applying PRIME during fine-tuning slightly improves the clean performance.

In Figure 6, we compare the effect of the PRIME data augmentation during transfer learning across models. Here again we can observe that PRIME hurts the clean performance for COCO but does not significantly affect ADE20K. Even in the PRIME fine-tuning setting, we can see that Swin-T performs the best among the models we evaluated.

We note that for PRIME during transfer learning, we omit the geometric transformation module from PRIME so that the PRIME data augmentation does not distort geometric information that is tied with object bounding boxes and

[3] https://github.com/open-mmlab/mmdetection/blob/master/configs/_base_/models/mask_rcnn_r50_fpn.py, https://github.com/open-mmlab/mmdetection/blob/master/configs/_base_/datasets/coco_instance.py

[4] https://github.com/open-mmlab/mmsegmentation/blob/master/configs/upernet/upernet_r50_512x512_80k_ade20k.py, https://github.com/open-mmlab/mmsegmentation/blob/master/configs/_base_/models/upernet_r50.py, https://github.com/open-mmlab/mmsegmentation/blob/master/configs/_base_/datasets/ade20k.py, https://github.com/open-mmlab/mmsegmentation/blob/master/configs/_base_/schedules/schedule_80k.py



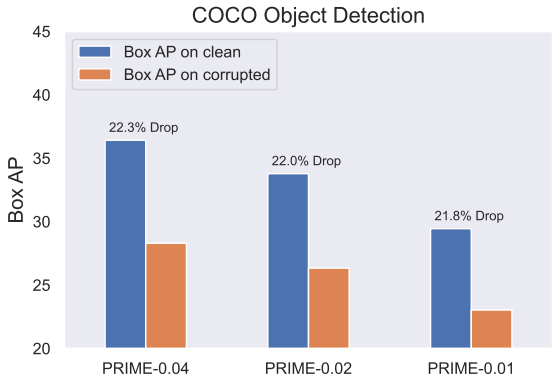


Figure 5. The effect of the degree of fine-tuning robust ImageNet backbones for downstream tasks. We use the data augmentation technique called PRIME , which corrupts images during fine-tuning. We measure the degree of fine-tuning by the value of the learning rate we use for fine-tuning. We evaluate three different settings with varying learning rates. We can see that for both ADE20K semantic segmentation and COCO object detection, the smaller the learning rate is, the lower the clean performance becomes. However, the percentage of performance drop becomes less significant for the small learning rate, which confirms our intuition that small learning rates better retain the robustness property of the original ImageNet backbones.

COCO	PRIME	PR-PRIME
Box AP clean	36.42	31.47
Box AP corrupted	28.30	24.80
BoX AP Drop	22.29 %	21.21 %

Table 4. The effect of PRIME data augmentation during transfer learning from ImageNet models to COCO object detection. Both PRIME and PR-PRIME are initialized with weights that are pretrained using PRIME on ImageNet. For PRIME, we use a standard training for transfer learning. For PR-PRIME, we apply the PRIME data augmentation during transfer learning.

semantic labels in ADE20K and COCO. Furthermore, we use the default hyperparameters of PRIME that are designed for ImageNet, and did not incorporate the JSD consistency loss in the PRIME module. More careful tuning of PRIME and incorporation of the JSD consistency loss into object detection / semantic segmentation systems might lead to better robust transfer results, but we leave this as future work.

C. Acknowledgements

We thank the anonymous reviewers for their helpful feedback on this work. This research was in part funded by the Masason Foundation to YY.

ADE20K	PRIME	PR-PRIME
IoU clean	40.64	40.91
IoU corrupted	28.56	30.56
IoU Drop	29.73 %	25.31 %

Table 5. The effect of PRIME data augmentation during transfer learning from ImageNet models to ADE20K semantic segmentation. Both PRIME and PR-PRIME are initialized with weights that are pretrained using PRIME on ImageNet. For PRIME, we use a standard training for transfer learning. For PR-PRIME, we apply the PRIME data augmentation during transfer learning.

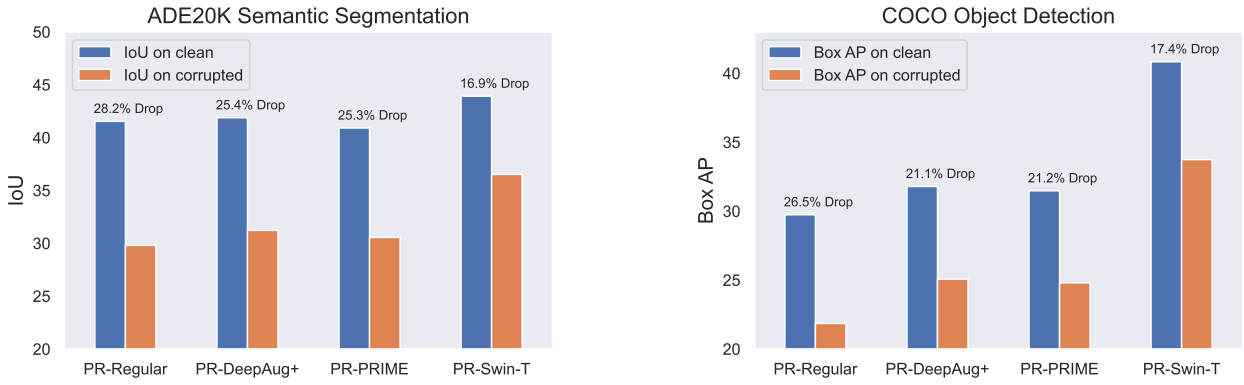


Figure 6. The effect of data augmentation during transfer learning from ImageNet pretrained weights to downstream tasks. The 'PR' stands for PRIME, the robustification data augmentation technique we used during fine-tuning. We compare three ImageNet-pretrained ResNet50 models (Regular, DeepAug+, and PRIME) with Swin-T, which is pretrained on ImageNet and also use PRIME during transfer learning. During ImageNet training, DeepAug+ and PRIME use the DeepAug+AugMix and PRIME data augmentation, respectively. Interestingly, it seems that the PRIME fine-tuning does not affect the clean performance on the semantic segmentation task (ADE20K), although it hurts the clean performance on the COCO object detection. (See Figure 2 for comparison.) Regarding the difference in architecture, we can see that Swin-T still performs the best in terms of both raw performance metrics on clean/corrupted data and the percentage of the performance drop.