A Appendix

A.1 Additional Ablation Results

Table A.1 is the ViT-B counterpart of Table 2 on backbone adaptation. The observations are similar to that of ViT-L: comparing with the baseline using no propagation (“none”), various propagation strategies show good gains.

Table A.2 presents Table 5 with additional details about FLOPs, parameters, and inference time, plotted in Figure 3.

Table A.3 is the ablation on pre-training strategies for LVIS. Similar to Table 4, MAE pre-training has large gains over supervised pre-training.

Figure A.1 is the LVIS counterpart of Figure 3. The trends are similar to those in COCO, while the gain of IN-21K supervised pre-training is larger because it significantly improves rare category detection in LVIS.

Figure A.2 is the RetinaNet [14] counterpart of Figure 3, showing the trade-off between accuracy and model size. Here, we evaluate ViTDet with a one-stage RetinaNet [14] detector head and compare it to using Swin and MViTv2 as hierarchical backbones, all without hyper-parameter tuning. Compared to using Mask R-CNN and Cascade R-CNN (Table 5 and Figure 3), we observe similar trends with RetinaNet. In particular, our plain-backbone detector presents better scaling behavior (e.g. ViT-H gains +3.4 AP\text{box} over MViTv2-H). These results suggest that the proposed training recipe transfers well to different detectors and that our proposed plain backbone adaptations are general and can likely work with even more detection architectures.

A.2 Implementation Details

Architectures. We build a simple feature pyramid of scales \( \{ \frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4} \} \) (see Sec. 3). The \( \frac{1}{32} \) scale is built by stride-2 \( 2 \times 2 \) max pooling (average pooling or convolution works similarly). The \( \frac{1}{16} \) scale simply uses the ViT’s final feature map. Scale \( \frac{1}{8} \) (or \( \frac{1}{4} \)) is built by one (or two) \( 2 \times 2 \) deconvolution layer(s) with stride=2. In the \( \frac{1}{4} \) scale case, the first deconvolution is followed by LayerNorm (LN) [1] and GeLU [12]. Then for each pyramid level, we apply a \( 1 \times 1 \) convolution with LN to reduce dimension to 256 and then a \( 3 \times 3 \) convolution also with LN, similar to the per-level processing of FPN [13].

We study three detection frameworks: Mask R-CNN [11], Cascade Mask R-CNN [3] and RetinaNet [14]. For Mask R-CNN and Cascade Mask R-CNN, we incorporate some common best practices developed since they [11,3] were presented years ago. We use 2 hidden convolution layers for the RPN and 4 hidden convolution layers for the RoI heads as per [16]. These hidden convolution layers are followed by LN. For all three detection frameworks, We use the same detection implementation for both plain and hierarchical backbones.

We use a patch size of 16 for all ViT backbones. As ViT-H in [6] by default has a patch size of 14, after pre-training we interpolate the patch embedding filters from \( 14 \times 14 \times 3 \) to \( 16 \times 16 \times 3 \).
prop. strategy & AP$^{\text{box}}$ & AP$^{\text{mask}}$ & prop. strategy & AP$^{\text{box}}$ & AP$^{\text{mask}}$
\hline
none & 48.9 & 43.9 & none & 48.9 & 43.9 \\
4 global blocks & 51.2 (+2.3) & 45.5 (+1.6) & naïve & 50.6 (+1.7) & 45.2 (+1.3) \\
4 conv blocks & 51.0 (+2.1) & 45.3 (+1.4) & basic & 50.7 (+1.8) & 45.2 (+1.3) \\
shifted win. & 50.1 (+1.2) & 44.8 (+0.9) & bottleneck & 51.0 (+2.1) & 45.3 (+1.4) \\
\hline

(a) Window attention with various cross-window propagation strategies.

prop. locations & AP$^{\text{box}}$ & AP$^{\text{mask}}$ & prop. blks & AP$^{\text{box}}$ & AP$^{\text{mask}}$
\hline
none & 48.9 & 43.9 & none & 48.9 & 43.9 \\
first 4 blocks & 49.1 (+0.2) & 44.1 (+0.2) & 2 & 50.7 (+1.8) & 45.2 (+1.3) \\
last 4 blocks & 50.9 (+2.0) & 45.4 (+1.5) & 4 & 51.2 (+2.3) & 45.5 (+1.6) \\
evenly 4 blocks & 51.2 (+2.3) & 45.5 (+1.6) & 12 & 50.4 (+1.5) & 45.1 (+1.2) \\
\hline

(b) Convolutional propagation with different residual block types (4 blocks).

(c) Locations of cross-window global propagation blocks.

(d) Number of global propagation blocks.

Table A.1: The ViT-B counterpart of Table 2 (backbone adaptation).

**Hyper-parameters for COCO.** Our default training recipe is as follows (unless noted in context for ablation). The input size is $1024 \times 1024$, augmented during training by large-scale jitter [7] with a scale range of $[0.1, 2.0]$. We use AdamW [15] ($\beta_1, \beta_2=0.9, 0.999$) with step-wise learning rate decay. We use linear learning rate warm-up [8] for 250 iterations. The batch size is 64, distributed across 64 GPUs (1 image per GPU).

We search for the learning rate ($lr$), weight decay ($wd$), drop path rate ($dp$), and epochs, for each model size (B, L, H) and for each model type (ViT, Swin, MViTv2). The hyper-parameters used are in Table A.4. We also use a layer-wise $lr$ decay [4][2] of 0.7/0.8/0.9 for ViT-B/L/H with MAE pre-training, which has a small gain of up to 0.3 AP; we have not seen this gain for hierarchical backbones or ViT with supervised pre-training.

**Hyper-parameters for LVIS.** Our LVIS experiments in Table 7 follow the COCO settings in Table 5. For LVIS, we set $lr = 2e^{-4}/1e^{-4}$ (ViT-L/H), $wd = 0.1$, and $dp = 0.4$. We fine-tune for 100 epochs. We use a test score threshold of 0.02 (smaller values did not help) and repeat factor sampling ($t = 0.001$) [9]. We output $\leq 300$ detections per image following [9] (vs. COCO’s default 100).

**MAE for hierarchical backbones.** We implement a naïve extension of MAE pre-training [10] for the hierarchical backbone ablation (Sec. 4.2). MAE enjoys the efficiency benefit from plain ViT by skipping the encoder mask token [10]. Extending this strategy to hierarchical backbones is beyond the scope of this paper. Instead, we adopt a straightforward solution in which we do not skip the encoder mask token (similar to [5]), at the cost of slower training. We use normalized pixels as the MAE reconstruction target [10] and set the decoder depth as 2.
Table A.2: Detailed measurements of Table 5 and Figure 3.

<table>
<thead>
<tr>
<th>backbone</th>
<th>pre-train</th>
<th>Mask R-CNN</th>
<th>Cascade Mask R-CNN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AP\textsuperscript{box}</td>
<td>AP\textsuperscript{mask}</td>
<td>FLOPs</td>
</tr>
<tr>
<td>hierarchical-backbone detectors:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Swin-B</td>
<td>1K, sup</td>
<td>50.1</td>
<td>44.5</td>
</tr>
<tr>
<td>Swin-B</td>
<td>21K, sup</td>
<td>51.4</td>
<td>45.4</td>
</tr>
<tr>
<td>Swin-L</td>
<td>21K, sup</td>
<td>52.4</td>
<td>46.2</td>
</tr>
<tr>
<td>MViTv2-B</td>
<td>1K, sup</td>
<td>52.4</td>
<td>46.7</td>
</tr>
<tr>
<td>MViTv2-L</td>
<td>1K, sup</td>
<td>53.2</td>
<td>47.1</td>
</tr>
<tr>
<td>MViTv2-B</td>
<td>21K, sup</td>
<td>53.1</td>
<td>47.4</td>
</tr>
<tr>
<td>MViTv2-L</td>
<td>21K, sup</td>
<td>53.6</td>
<td>47.5</td>
</tr>
<tr>
<td>MViTv2-H</td>
<td>21K, sup</td>
<td>54.1</td>
<td>47.7</td>
</tr>
<tr>
<td>our plain-backbone detectors:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ViT-B</td>
<td>1K, MAE</td>
<td>51.6</td>
<td>45.9</td>
</tr>
<tr>
<td>ViT-L</td>
<td>1K, MAE</td>
<td>55.6</td>
<td>49.2</td>
</tr>
<tr>
<td>ViT-H</td>
<td>1K, MAE</td>
<td>56.7</td>
<td>50.1</td>
</tr>
</tbody>
</table>

Table A.3: The LVIS counterpart of Table 4 (COCO pre-training ablation). The observations are similar to Table 4: MAE pre-training has large gains over supervised pre-training. Here we also report rare category results. We observe that both IN-21K supervised and IN-1K MAE pre-training significantly improve AP\textsuperscript{mask}\textsubscript{rare}, especially for ViT-L. (Mask R-CNN, 1024 resolution, no soft-nms.)
Table A.4: Hyper-parameters for COCO. Multiple values in a cell are for different model sizes. The epochs are chosen such that training longer starts to overfit.

<table>
<thead>
<tr>
<th>backbone</th>
<th>pre-train</th>
<th>lr</th>
<th>wd</th>
<th>dp</th>
<th>epochs</th>
</tr>
</thead>
<tbody>
<tr>
<td>ViT-B/L</td>
<td>none</td>
<td>$1.6 \times 10^{-4}$</td>
<td>0.2</td>
<td>0.1/0.4</td>
<td>300/200</td>
</tr>
<tr>
<td>ViT-B/L</td>
<td>supervised</td>
<td>$8 \times 10^{-5}$</td>
<td>0.1</td>
<td>0.1/0.4</td>
<td>50</td>
</tr>
<tr>
<td>ViT-B/L/H</td>
<td>MAE</td>
<td>$1 \times 10^{-4}$</td>
<td>0.1</td>
<td>0.1/0.4/0.5</td>
<td>100/100/75</td>
</tr>
<tr>
<td>Swin-B/L</td>
<td>supervised</td>
<td>$10^{-7}$/$8 \times 10^{-9}$</td>
<td>0.05</td>
<td>0.3</td>
<td>50</td>
</tr>
<tr>
<td>MViTv2-B/L/H</td>
<td>supervised</td>
<td>$8 \times 10^{-5}$</td>
<td>0.1</td>
<td>0.4/0.5/0.6</td>
<td>100/50/36</td>
</tr>
</tbody>
</table>

Acknowledgement. We would like to acknowledge Xinlei Chen, Saining Xie, Piotr Dollár, and Christoph Feichtenhofer for discussions and support.

References

10. Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv:2111.06377*, 2021.