A1. Detailed Architectures

The detailed architecture specifications are shown in Table 1, where an input image size of 224×224 is assumed for all architectures. “Concat n × n” indicates a concatenation of n × n neighboring features in a patch. This operation results in a downsampling of the feature map by a rate of n. “96-d” denotes a linear layer with an output dimension of 96. “win. sz. 7 × 7” indicates a multi-head self-attention module with window size of 7 × 7.

A2. Detailed Experimental Settings

A2.1. Image classification on ImageNet-1K

The image classification is performed by applying a global average pooling layer on the output feature map of the last stage, followed by a linear classifier. We find this strategy to be as accurate as using an additional class token as in ViT [10] and DeiT [21]. In evaluation, the top-1 accuracy using a single crop is reported.

Regular ImageNet-1K training The training settings mostly follow [21]. For all model variants, we adopt a default input image resolution of 224². For other resolutions such as 384², we fine-tune the models trained at 224² resolution, instead of training from scratch, to reduce GPU consumption.

When training from scratch with a 224² input, we employ an AdamW [15] optimizer for 300 epochs using a cosine decay learning rate scheduler with 20 epochs of linear warm-up. A batch size of 1024, an initial learning rate of 0.001, a weight decay of 0.05, and gradient clipping with a max norm of 1 are used. We include most of the augmentation and regularization strategies as [21] in training, including RandAugment [9], Mixup [26], Cutmix [25], random erasing [28] and stochastic depth [14], but not repeated augmentation [13] and Exponential Moving Average (EMA) [17] which do not enhance performance. Note that this is contrary to [21] where repeated augmentation is crucial to stabilize the training of ViT. An increasing degree of stochastic depth augmentation is employed for larger models, i.e. 0.2, 0.3, 0.5 for Swin-T, Swin-S, and Swin-B, respectively.

For fine-tuning on input with larger resolution, we employ an adamW [15] optimizer for 30 epochs with a constant learning rate of 10⁻⁶, weight decay of 10⁻⁸, and the same data augmentation and regularizations as the first stage except for setting the stochastic depth ratio to 0.1.

ImageNet-22K pre-training We also pre-train on the larger ImageNet-22K dataset, which contains 14.2 million images and 22K classes. The training is done in two stages. For the first stage with 224² input, we employ an AdamW optimizer for 90 epochs using a linear decay learning rate scheduler with a 5-epoch linear warm-up. A batch size of 4096, an initial learning rate of 0.001, and a weight decay of 0.01 are used. In the second stage of ImageNet-1K fine-tuning with 224²/384² input, we train the models for 30 epochs with a batch size of 1024, a constant learning rate of 10⁻⁵, and a weight decay of 10⁻⁸.

A2.2. Object detection on COCO

For an ablation study, we consider four typical object detection frameworks: Cascade Mask R-CNN [12, 2], ATSS [27], RepPoints v2 [7], and Sparse RCNN [18] in mmdetection [6]. For these four frameworks, we utilize the same settings: multi-scale training [4, 18] (resizing the input such that the shorter side is between 480 and 800 while the longer side is at most 1333), AdamW [16] optimizer (initial learning rate of 0.0001, weight decay of 0.05, and batch size of 16), and 3x schedule (36 epochs with the learning rate decayed by 0.1× at epochs 27 and 33).

For system-level comparison, we adopt an improved HTC [5] (denoted as HTC++) with instaboost [11], stronger multi-scale training [3] (resizing the input such that the shorter side is between 400 and 1400 while the longer side is at most 1600), 6x schedule (72 epochs with the learning rate decayed at epochs 63 and 69 by a factor of 0.1), soft-NMS [1], and an extra global self-attention layer appended at the output of last stage and ImageNet-22K pre-trained model as initialization. We adopt stochastic depth with ratio of 0.2 for all Swin Transformer models.

A2.3. Semantic segmentation on ADE20K

ADE20K [29] is a widely-used semantic segmentation dataset, covering a broad range of 150 semantic categories. It has 25K images in total, with 20K for training, 2K for validation, and another 3K for testing. We utilize UperNet [23] in mmsegmentation [8] as our base framework for its high efficiency.

In training, we employ the AdamW [16] optimizer with an initial learning rate of 6 × 10⁻⁵, a weight decay of 0.01, a scheduler that uses linear learning rate decay, and a linear warm-up of 1,500 iterations. Models are trained on 8 GPUs with 2 images per GPU for 160K iterations. For augmentations, we adopt the default setting in mmsegmentation of random horizontal flipping, random re-scaling within ratio range [0.5, 2.0] and random photometric distortion. Stochastic depth with ratio of 0.2 is applied for all Swin Transformer models. Swin-T, Swin-S are trained on the standard setting as the previous approaches with an input of 512×512. Swin-B and Swin-L with ½ indicate that these two models are pre-trained on ImageNet-22K, and trained with the input of 640×640.

In inference, a multi-scale test using resolutions that are [0.5, 0.75, 1.0, 1.25, 1.5, 1.75]× of that in training is em-
A3. More Experiments

A3.1. Image classification with different input size

Table 2 lists the performance of Swin Transformers with different input image sizes from 224^2 to 384^2. In general, a larger input resolution leads to better top-1 accuracy but with slower inference speed.

A3.2. Different Optimizers for ResNe(X)t on COCO

Table 3 compares the AdamW and SGD optimizers of the ResNe(X)t backbones on COCO object detection. The Cascade Mask R-CNN framework is used in this comparison. While SGD is used as a default optimizer for Cascade Mask R-CNN framework, we generally observe improved accuracy by replacing it with an AdamW optimizer, particularly for smaller backbones. We thus use AdamW for ResNe(X)t backbones when compared to the proposed Swin Transformer architectures.

A3.3. Swin MLP-Mixer

We apply the proposed hierarchical design and the shifted window approach to the MLP-Mixer architectures [19], referred to as Swin-Mixer. Table 4 shows the performance of Swin-Mixer compared to the original MLP-Mixer architectures [19] and a follow-up approach, i.e., ResMLP [19]. Swin-Mixer performs significantly better than MLP-Mixer (81.3% vs. 76.4%) using slightly smaller computation budget (10.4G vs. 12.7G). It also has better speed accuracy trade-off compared to ResMLP [20]. These results indicate the proposed hierarchical design and the shifted window approach are generalizable.
References


[27] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and
