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 $n \times n$" indicates a concatenation of $n \times n$ 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 224^2. For other resolutions such as 384^2, we fine-tune the models trained at 224^2 resolution, instead of training from scratch, to reduce GPU consumption.

When training from scratch with a 224^2 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 of [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^{-5}, 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 224^2 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 224^2/384^2 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 $10\times$ 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 warmup 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 ‡ 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-

	downsp. rate (output size)	Swin-T	Swin-S	Swin-B	Swin-L
stage 1	4× (56×56)	concat 4×4, 96-d, LN	concat 4×4, 96-d, LN	concat 4×4, 128-d, LN	concat 4×4, 192-d, LN
		$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 96, head 3}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 96, head 3}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 128, head 4}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 192, head 6}\end{bmatrix} \times 2$
stage 2	8× (28×28)	concat 2×2, 192-d , LN	concat 2×2, 192-d , LN	concat 2×2, 256-d , LN	concat 2×2, 384-d , LN
		$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 192, head 6}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 192, head 6}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 256, head 8}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 384, head 12}\end{bmatrix} \times 2$
stage 3	16× (14×14)	concat 2×2, 384-d , LN	concat 2×2, 384-d , LN	concat 2×2, 512-d , LN	concat 2×2, 768-d , LN
		$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 384, head 12}\end{bmatrix} \times 6$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 384, head 12}\end{bmatrix} \times 18$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 512, head 16}\end{bmatrix} \times 18$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 768, head 24}\end{bmatrix} \times 18$
stage 4	32× (7×7)	concat 2×2, 768-d , LN	concat 2×2, 768-d , LN	concat 2×2, 1024-d , LN	concat 2×2, 1536-d , LN
		$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 768, head 24}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 768, head 24}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 1024, head 32}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{win. sz. } 7\times7,\\ \text{dim 1536, head 48}\end{bmatrix} \times 2$

Table 1. Detailed architecture specifications.

	Swin-T		Swin-S		Swin-B	
input size	top-1 acc	throughput (image / s)	top-1 acc	throughput (image / s)	top-1 acc	throughput (image / s)
224^2	81.3	755.2	83.0	436.9	83.3	278.1
256^2	81.6	580.9	83.4	336.7	83.7	208.1
320^2	82.1	342.0	83.7	198.2	84.0	132.0
384^2	82.2	219.5	83.9	127.6	84.5	84.7

Table 2. Swin Transformers with different input image size on ImageNet-1K classification.

Backbone	Optimizer	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
R50	SGD	45.0	62.9	48.8	38.5	59.9	41.4
	AdamW	46.3	64.3	50.5	40.1	61.7	43.4
X101-32x4d	SGD	47.8	65.9	51.9	40.4	62.9	43.5
	AdamW	48.1	66.5	52.4	41.6	63.9	45.2
X101-64x4d	SGD	48.8	66.9	53.0	41.4	63.9	44.7
	AdamW	48.3	66.4	52.3	41.7	64.0	45.1

Table 3. Comparison of the SGD and AdamW optimizers for ResNe(X)t backbones on COCO object detection using the Cascade Mask R-CNN framework.

ployed. When reporting test scores, both the training images and validation images are used for training, following common practice [24].

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.

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
MLP-Mixer-B/16 [19]	224^2	59M	12.7G	-	76.4
ResMLP-S24 [20]	224^2	30M	6.0G	715	79.4
ResMLP-B24 [20]	224^2	116M	23.0G	231	81.0
Swin-T/D24 (Transformer)	256^2	28M	5.9G	563	81.6
Swin-Mixer-T/D24	256^2	20M	4.0G	807	79.4
Swin-Mixer-T/D12	256^2	21M	4.0G	792	79.6
Swin-Mixer-T/D6	256^2	23M	4.0G	766	79.7
Swin-Mixer-B/D24 (no shift)	224^2	61M	10.4G	409	80.3
Swin-Mixer-B/D24	224^2	61M	10.4G	409	81.3

Table 4. Performance of Swin MLP-Mixer on ImageNet-1K classification. D indictes the number of channels per head. Throughput is measured using the GitHub repository of [22] and a V100 GPU, following [21].

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

[1] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S. Davis. Soft-nms – improving object detection with one line of code. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1

[2] 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. 1

[3] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019. 1

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. 1

[5] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4974–4983, 2019. 1

[6] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 1

[7] Yihong Chen, Zheng Zhang, Yue Cao, Liwei Wang, Stephen Lin, and Han Hu. Reppoints v2: Verification meets regression for object detection. In *NeurIPS*, 2020. 1

[8] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. `https://github.com/open-mmlab/mmsegmentation`, 2020. 1

[9] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020. 1

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1

[11] Hao-Shu Fang, Jianhua Sun, Runzhong Wang, Minghao Gou, Yong-Lu Li, and Cewu Lu. Instaboost: Boosting instance segmentation via probability map guided copy-pasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 682–691, 2019. 1

[12] 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. 1

[13] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8129–8138, 2020. 1

[14] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016. 1

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

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

[17] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992. 1

[18] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, et al. Sparse r-cnn: End-to-end object detection with learnable proposals. *arXiv preprint arXiv:2011.12450*, 2020. 1

[19] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. Mlp-mixer: An all-mlp architecture for vision, 2021. 2

[20] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Gautier Izacard, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, and Hervé Jégou. Resmlp: Feedforward networks for image classification with data-efficient training, 2021. 2

[21] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020. 1, 2

[22] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. 2

[23] 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)*, pages 418–434, 2018. 1

[24] Minghao Yin, Zhuliang Yao, Yue Cao, Xiu Li, Zheng Zhang, Stephen Lin, and Han Hu. Disentangled non-local neural networks. In *Proceedings of the European conference on computer vision (ECCV)*, 2020. 2

[25] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6023–6032, 2019. 1

[26] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 1

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

anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9759–9768, 2020. 1

[28] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13001–13008, 2020. 1

[29] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal on Computer Vision*, 2018. 1