Slide-Transformer: Hierarchical Vision Transformer with Local Self-Attention Supplementary Material

A. Model Architectures

We summarize the architectures of five Transformer models adopted in the main paper, including PVT [11], PVTv2 [12], Swin Transformer [8], CSwin Transformer [3], NAT [4] in Tab.5-10. For fair comparison, we only substitute the original self-attention blocks at early stages of the baseline models with our proposed *Slide Attention*, while the remaining blocks, training configurations, and model structure (width and depth) are kept unchanged.

B. Dataset and Training Setup

ImageNet. ImageNet 2012 [2] comprises 1.28 million training images and 50,000 validation images from 1000 different classes. For all baseline models, we follow the training configurations in the original paper, with adamw optimizer, 300 epoch training, and data augmentation settings follow DeiT [10]. For CSwin-Transformer, we follow the original setting and use exponential moving average (EMA) [9] with the same ema decay rate.

COCO. COCO dataset [7] is a standard object detection benchmark and we use a subset of 80k samples as training set and 35k for validation. For all baseline models, we train the network by adamw. Backbone networks are respectively pretrained on ImageNet dataset following the same training configurations in the original paper. We follow the "1x" learning schedule to train the whole network for 12 epochs and divide the learning rate by 10 at the 8th and 11th epoch respectively. For several models, we follow the configurations in the original paper, and additionally experiment "3x" schedule with 36 epochs. We apply standard data augmentation, that is resize, random flip and normalize. Learning rate is set at 0.01 and linear warmup is used in the first 500 iterations. We follow the "1x" learning schedule training the whole network for 12 epochs and divide the learning rate by 10 at the 8th and 11th epoch respectively. For several transformer-based models, we follow the configurations in the original paper, and test with "3x" schedule. All mAP results in the main paper are tested with input image size (3, 1333, 800).

ADE20K. ADE20K [14] is a widely-used semantic segmentation dataset, containing 150 categories. ADE20K has 25K images, with 20K for training, 2K for validation, and another 3K for testing. For baseline models, we follow the training configurations in their original paper respectively. For Semantic FPN [5], we optimize the models using AdamW with an initial learning rate of 1e-4 for 80k iterations. For UperNet [13], we use the AdamW optimizer with an initial learning rate of 6e-5 and a linear warmup of 1,500 iterations. Models are trained for a total of 160K iterations. We randomly resize and crop the image to 512 × 512 for training, and re-scale to have a shorter side of 512 pixels during testing.

RetinaNet Object Detection on COCO (Sch. 1x)

Method	FLOPs	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
PVT-T	221G	36.7	56.9	38.9	22.6	38.8	50.0
Slide-PVT-T	200G	40.1	61.1	42.2	25.9	43.3	54.2
PVT-S	286G	38.7	59.3	40.8	21.2	41.6	54.4
Slide-PVT-S	251G	42.4	63.9	45.0	26.8	45.6	56.9
PVT-M	373G	41.9	63.1	44.3	25.0	44.9	57.6
Slide-PVT-M	338G	43.5	64.7	46.1	26.3	47.1	58.5
PVTV2-B0	178G	37.2	57.2	39.5	23.1	40.4	49.7
Slide-PVTv2-B0	167G	37.6	57.9	40.1	22.9	40.4	50.2
PVTV2-B1	225G	41.2	61.9	43.9	25.4	44.5	54.3
Slide-PVTv2-B1	204G	41.5	62.3	44.0	26.0	44.8	54.9
PVTV2-B2	291G	44.6	65.6	47.6	27.4	48.8	58.6
Slide-PVTv2-B2	255G	45.0	66.2	48.4	28.8	48.8	59.7
PVTV2-B3	379G	45.9	66.8	49.3	28.6	49.8	61.4
Slide-PVTv2-B3	343G	46.8	67.7	50.3	30.5	51.1	61.6

Table 1. Results on COCO object detection with RetinaNet [6]. The FLOPs are computed over backbone, FPN, and detection head with an input resolution of 1280×800.

C. Additional Downstream Experiments

We also provide additional experiment results on semantic segmentation and object detection, and show in Tab.1, Tab.3 and Tab.4. Similar pattern can be observed: (1)

Method	Params	Flops	Top-1
PVT-T [11]	13.2M	1.9G	75.1
Slide-PVT-T	12.2M	2.0G	**78.0** (+2.9)
PVT-S	24.5M	3.8G	79.8
Slide-PVT-S	22.7M	4.0G	**81.7** (+1.9)
PVT-M	44.2M	6.7G	81.2
Slide-PVT-M	42.5M	9.8G	**82.9** (+1.7)
PVT-L	61.4M	6.7G	81.7
Slide-PVT-L	59.8M	9.8G	**83.9** (+2.2)
PVTv2-B0 [12]	3.4M	0.6G	70.5
Slide-PVTv2-B0	3.3M	0.6G	**71.4** (+0.9)
PVTv2-B1	13.1M	2.1G	78.7
Slide-PVTv2-B1	13.0M	2.2G	**79.5** (+0.7)
PVTv2-B2	25.4M	4.0G	82.0
Slide-PVTv2-B2	22.8M	4.2G	**82.7** (+0.7)
PVTv2-B3	45.2M	6.9G	83.2
Slide-PVTv2-B3	42.5M	7.1G	**83.8** (+0.6)
PVTv2-B4	62.6 M	10.1G	83.6
Slide-PVTv2-B4	59.8M	10.3G	**84.2** (+0.6)
PVTv2-B5	82.0M	11.8G	83.8
Slide-PVTv2-B5	78.9M	12.1G	**84.3** (+0.5)
Swin-T [8]	29M	4.5G	81.3
Slide-Swin-T	29M	4.6G	**82.3** (+1.0)
Swin-S	50M	8.7G	83.0
Slide-Swin-S	51M	8.9G	**83.7** (+0.7)
Swin-B	88M	15.4G	83.5
Slide-Swin-B	89M	15.5G	**84.2** (+0.7)
CSwin-T [3]	23M	4.3G	82.8
Slide-CSwin-T	23M	4.3G	**83.2** (+0.4)
CSwin-S	35M	6.9G	83.6
Slide-CSwin-S	35M	6.9G	**84.0** (+0.4)
CSwin-B	78M	15.0G	84.2
Slide-CSwin-B	78M	15.0G	**84.7** (+0.5)
NAT-M [4]	20M	2.7G	81.8
Slide-NAT-M	20M	2.7G	**82.4** (+0.6)
NAT-T	28M	4.3G	83.2
Slide-NAT-T	28M	4.3G	**83.6** (+0.4)
NAT-S	51M	7.8G	83.7
Slide-NAT-S	51M	7.8G	**84.3** (+0.6)

Table 2. Comparisons of slide attention with other vision transformer backbones on FLOPs, parameters, accuracy on the ImageNet-1K classification task.

Semantic Segmentation on ADE20K					
Backbone	Method	FLOPs	#Params	mIoU	mAcc
PVT-T	S-FPN	158G	17M	36.57	46.72
Slide-PVT-T	S-FPN	136G	16M	**38.43**	50.05
PVT-S	S-FPN	225G	28M	41.95	53.02
Slide-PVT-S	S-FPN	188G	26M	**42.47**	54.00
PVT-M	S-FPN	315G	48M	42.91	53.80
Slide-PVT-M	S-FPN	278G	46M	**43.97**	55.58
Swin-T	UperNet	945G	60M	44.51	55.61
Slide-Swin-T	UperNet	946G	60M	**45.67**	57.13
Swin-S	UperNet	1038G	81M	47.64	58.78
Slide-Swin-S	UperNet	1038G	81M	**48.46**	60.18
Swin-B	UperNet	1188G	121M	48.13	59.13
Slide-Swin-B	UperNet	1188G	121M	**48.58**	60.26

Table 3. Results of semantic segmentation. The FLOPs are computed over encoders and decoders with an input image at the resolution of 512×2048. S-FPN is short for SemanticFPN [5] model.

Our module achieves consistent improvements over baseline models; (2) The improvements on small objects are more significant, indicating the effectiveness of local inductive bias in our slide attention module. We also test the runtime for segmentation and detection on a RTX3090 GPU, and show the results below. Similar to classification, our method achieves a better trade-off than baselines.

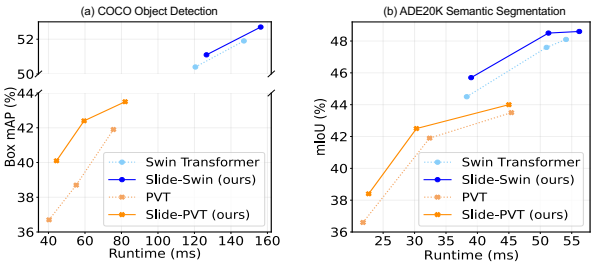


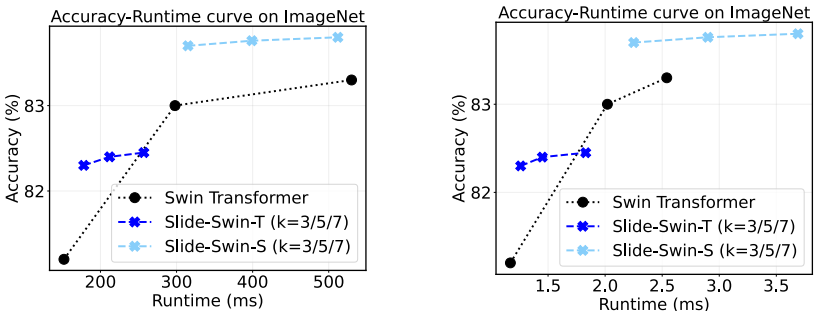
Figure 1. Runtime for object detection and segmentation.

D. Full Classification Results

Due to the page limit, we only present representative ImageNet classification results in Figure 5 of main paper. Here, we show the full classification results when adapting our module on all model sizes of baseline models in Tab.2.

E. Ablation Study on the Window Size

We also conduct ablation studies on the window size of our local attention, and show runtime-performance comparison on both iPhone 12 and RTX3090 GPU. Our experiments are all based on Swin-Transformer-T and Swin-Transformer-S, and only the self-attention modules in the



(a) iPhone 12 **(b) GPU (RTX 3090)**

Figure 2. Ablation study on the window size k. We show runtime comparison on iPhone 12 and RTX 3090 GPU.

(a) Mask R-CNN Object Detection & Instance Segmentation on COCO

Method	FLOPs	#Param	Schedule	AP^b	AP^b_{50}	AP^b_{75}	AP^b_s	AP^b_m	AP^b_l	AP^m	AP^m_{50}	AP^m_{75}	AP^m_s	AP^m_m	AP^m_l
PVT-T	240G	33M	1x	36.7	59.2	39.3	21.6	39.2	49.0	35.1	56.7	37.3	19.5	37.4	48.5
Slide-PVT-T	219G	32M	1x	40.4	63.4	43.8	25.3	42.8	53.0	38.1	60.4	41.0	20.0	40.1	55.2
PVT-S	305G	44M	1x	40.4	62.9	43.8	22.9	43.0	55.4	37.8	60.1	40.3	20.4	40.3	53.6
Slide-PVT-S	269G	42M	1x	42.8	65.9	46.7	26.6	45.5	57.3	40.1	63.1	43.1	20.3	42.4	59.0
PVT-M	392G	64M	1x	42.0	64.4	45.6	24.4	44.9	57.9	39.0	61.6	42.1	21.3	42.0	55.2
Slide-PVT-M	357G	62M	1x	44.4	66.9	48.6	28.9	47.0	59.4	0.408	63.9	43.8	25.0	43.5	55.9
PVTv2-B0	196G	23M	1x	38.2	60.5	40.7	22.9	40.9	49.6	36.2	57.8	38.6	18.0	38.4	51.9
Slide-PVTv2-B0	185G	23M	1x	38.8	60.9	41.9	23.7	41.5	50.7	36.4	58.0	38.8	20.6	39.1	49.5
PVTv2-B1	244G	34M	1x	41.8	64.3	45.9	26.4	44.9	54.3	38.8	61.2	41.6	20.2	41.3	56.1
Slide-PVTv2-B1	222G	33M	1x	42.6	65.3	46.8	27.4	45.6	55.7	39.7	62.6	42.6	24.1	42.9	53.7
PVTv2-B2	309G	45M	1x	45.3	67.1	49.6	28.8	48.4	59.5	41.2	64.2	44.4	22.0	43.7	59.4
Slide-PVTv2-B2	274G	43M	1x	46.0	68.2	50.3	28.8	49.4	61.0	41.9	65.1	45.4	24.6	45.2	57.2
PVTv2-B3	397G	65M	1x	47.0	68.1	51.7	30.2	50.4	62.4	42.5	65.7	45.7	23.2	45.3	61.5
Slide-PVTv2-B3	362G	63M	1x	47.8	69.5	52.6	30.2	51.3	62.8	43.2	66.5	46.6	26.1	46.3	58.7
Swin-T	267G	48M	1x	43.7	66.6	47.7	28.5	47.0	57.3	39.8	63.3	42.7	24.2	43.1	54.6
Slide-Swin-T	268G	48M	1x	44.3	67.2	48.5	28.9	47.8	57.0	40.3	63.9	43.0	24.3	44.0	54.5
Swin-T	267G	48M	3x	46.0	68.1	50.3	31.2	49.2	60.1	41.6	65.1	44.9	25.9	45.1	56.9
Slide-Swin-T	268G	48M	3x	46.8	69.0	51.6	31.7	50.4	60.1	42.3	66.0	45.8	23.5	45.8	60.8

(b) Cascade Mask R-CNN Object Detection & Instance Segmentation on COCO

Method	FLOPs	#Param	Schedule	AP^b	AP^b_{50}	AP^b_{75}	AP^b_s	AP^b_m	AP^b_l	AP^m	AP^m_{50}	AP^m_{75}	AP^m_s	AP^m_m	AP^m_l
Swin-T	745G	86M	1x	48.1	67.1	52.2	30.4	51.5	63.1	41.7	64.4	45.0	24.0	45.2	56.9
Slide-Swin-T	747G	86M	1x	48.6	67.7	52.7	32.1	52.2	63.5	41.9	65.0	45.2	23.2	45.3	60.9
Swin-T	745G	86M	3x	50.4	69.2	54.7	33.8	54.1	65.2	43.7	66.6	47.3	27.3	47.5	59.0
Slide-Swin-T	747G	86M	3x	51.1	69.8	55.4	35.2	54.4	65.8	44.3	67.4	48.0	28.0	48.0	59.2
Swin-S	838G	107M	3x	51.9	70.7	56.3	35.2	55.7	67.7	45.0	68.2	48.8	28.8	48.7	60.6
Slide-Swin-S	838G	107M	3x	52.5	71.3	57.2	35.6	56.1	68.0	45.4	68.9	49.6	29.1	49.2	60.6
Swin-B	981G	145M	3x	51.9	70.5	56.4	35.4	55.2	67.4	45.0	68.1	48.9	28.9	48.3	60.4
Slide-Swin-B	983G	145M	3x	52.7	71.2	57.2	37.0	56.1	68.0	45.5	68.8	49.6	30.1	48.8	60.9

Table 4. Results on COCO dataset. The FLOPs are computed over backbone, FPN and detection head with input resolution of 1280×800.

first two stages are substituted with slide attention. It can be observed that increasing the window size brings marginal improvements on the model performance while resulting in huge increase on the inference time. Similar results are also observed in other works like [1]. Therefore, we consider using window size $k = 3$ for all the results shown in the main paper. We see the ineffectiveness of increasing window size as a future direction and where a better performance-efficiency trade-off will be more valuable.

F. Limitations

As we have stated above, the increasing of window size in slide attention only results in marginal improvements. We believe a better performance-efficiency trade-off with larger window size worth further investigation, and we see this as an important future direction.

References

[1] Moab Arar, Ariel Shamir, and Amit H Bermano. Learned queries for efficient local attention. In *Conference on Computer Vision and Pattern Recognition*, 2022. 4

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition*, 2009. 1

[3] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows. In *Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2

[4] Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. Neighborhood attention transformer. In *arXiv preprint arXiv:2204.07143*, 2022. 1, 2

[5] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *Conference on Computer Vision and Pattern Recognition*, 2019. 1, 2

[6] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *International Conference on Computer Vision*, 2017. 1

[7] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014. 1

[8] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *International Conference on Computer Vision*, 2021. 1, 2

[9] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. In *Journal on Control and Optimization*, 1992. 1

[10] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, 2021. 1

[11] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *International Conference on Computer Vision*, 2021. 1, 2

[12] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvt v2: Improved baselines with pyramid vision transformer. In *Computational Visual Media*, 2022. 1, 2

[13] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *European Conference on Computer Vision*, 2018. 1

[14] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. In *International Journal of Computer Vision*. Springer, 2019. 1

stage	output	Slide-PVT-T		Slide-PVT-S		Slide-PVT-M	
		Slide Attention	PVT Block	**Slide Attention**	PVT Block	**Slide Attention**	PVT Block
res1	56×56	Conv1×1, stride=4, 64, LN					
		$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 64 \\ \text{head } 1 \end{bmatrix} \times 2$	None	$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 64 \\ \text{head } 1 \end{bmatrix} \times 3$	None	$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 64 \\ \text{head } 1 \end{bmatrix} \times 3$	None
res2	28×28	Conv1×1, stride=2, 128, LN					
		$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 128 \\ \text{head } 2 \end{bmatrix} \times 2$	None	$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 128 \\ \text{head } 2 \end{bmatrix} \times 3$	None	$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 128 \\ \text{head } 2 \end{bmatrix} \times 3$	None
res3	14×14	Conv1×1, stride=2, 320, LN					
		None	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 256 \\ \text{head } 5 \end{bmatrix} \times 2$	None	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 256 \\ \text{head } 5 \end{bmatrix} \times 6$	None	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 256 \\ \text{head } 5 \end{bmatrix} \times 18$
res4	7×7	Conv1×1, stride=2, 512, LN					
		None	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 512 \\ \text{head } 8 \end{bmatrix} \times 2$	None	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 512 \\ \text{head } 8 \end{bmatrix} \times 3$	None	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 512 \\ \text{head } 8 \end{bmatrix} \times 3$

Table 5. Architectures of Slide-PVT models.

stage	output	Slide-PVTv2-B0		Slide-PVTv2-B1		Slide-PVTv2-B2	
		Slide Attention	PVTv2 Block	**Slide Attention**	PVTv2 Block	**Slide Attention**	PVTv2 Block
res1	56×56	Conv4×4, stride=4, 32, LN		Conv4×4, stride=4, 64, LN			
		$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 32 \\ \text{head } 1 \end{bmatrix} \times 2$	None	$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 64 \\ \text{head } 1 \end{bmatrix} \times 2$	None	$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 64 \\ \text{head } 1 \end{bmatrix} \times 3$	None
res2	28×28	Conv1×1, stride=2, 64, LN		Conv1×1, stride=2, 128, LN			
		$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 64 \\ \text{head } 2 \end{bmatrix} \times 2$	None	$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 128 \\ \text{head } 2 \end{bmatrix} \times 2$	None	$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 128 \\ \text{head } 2 \end{bmatrix} \times 3$	None
res3	14×14	Conv2×2, stride=2, 160, LN		Conv2×2, stride=2, 320, LN			
		None	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 160 \\ \text{head } 5 \end{bmatrix} \times 2$	None	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 320 \\ \text{head } 5 \end{bmatrix} \times 2$	$\begin{bmatrix} \text{win } 3\times3 \\ \text{dim } 320 \\ \text{head } 5 \end{bmatrix} \times 2$	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 320 \\ \text{head } 5 \end{bmatrix} \times 4$
res4	7×7	Conv2×2, stride=2, 256, LN		Conv2×2, stride=2, 512, LN			
		None	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 512 \\ \text{head } 8 \end{bmatrix} \times 2$	None	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 512 \\ \text{head } 8 \end{bmatrix} \times 2$	None	$\begin{bmatrix} \text{win } 7\times7 \\ \text{dim } 512 \\ \text{head } 8 \end{bmatrix} \times 3$

Table 6. Architectures of Slide-PVTv2 models (Part1).

stage	output	Slide-PVTv2-B3		Slide-PVTv2-B4		Slide-PVTv2-B5	
		Slide Attention	PVTv2 Block	**Slide Attention**	PVTv2 Block	**Slide Attention**	PVTv2 Block
res1	56×56	Conv4×4, stride=4, 64, LN					
		win 3×3, dim 64, head 1 $\times3$	None	win 3×3, dim 64, head 1 $\times2$	None	win 3×3, dim 64, head 1 $\times3$	None
res2	28×28	Conv2×2, stride=2, 128, LN					
		win 3×3, dim 128, head 2 $\times3$	None	win 3×3, dim 128, head 2 $\times8$	None	win 3×3, dim 128, head 2 $\times6$	None
res3	14×14	Conv2×2, stride=2, 320, LN					
		win 3×3, dim 320, head 5 $\times10$	win 7×7, dim 320, head 5 $\times8$	win 3×3, dim 320, head 5 $\times15$	win 7×7, dim 320, head 5 $\times12$	win 3×3, dim 320, head 5 $\times20$	win 7×7, dim 320, head 5 $\times20$
res4	7×7	Conv1×1, stride=2, 512, LN					
		None	win 7×7, dim 512, head 8 $\times3$	None	win 7×7, dim 512, head 8 $\times2$	None	win 7×7, dim 512, head 8 $\times3$

Table 7. Architectures of Slide-PVTv2 models (Part2).

stage	output	Slide-Swin-T		Slide-Swin-S		Slide-Swin-B	
		Slide Attention	Swin Block	**Slide Attention**	Swin Block	**Slide Attention**	Swin Block
res1	56×56	concat 4×4, 96, LN		concat 4×4, 96, LN		concat 4×4, 128, LN	
		win 3×3, dim 96, head 3 $\times2$	None	win 3×3, dim 96, head 3 $\times2$	None	win 3×3, dim 128, head 3 $\times2$	None
res2	28×28	concat 4×4, 192, LN		concat 4×4, 192, LN		concat 4×4, 256, LN	
		win 3×3, dim 192, head 6 $\times2$	None	win 3×3, dim 192, head 6 $\times2$	None	win 3×3, dim 256, head 6 $\times2$	None
res3	14×14	concat 4×4, 384, LN		concat 4×4, 384, LN		concat 4×4, 512, LN	
		None	win 7×7, dim 384, head 12 $\times6$	None	win 7×7, dim 384, head 12 $\times18$	None	win 7×7, dim 512, head 12 $\times18$
res4	7×7	concat 4×4, 768, LN		concat 4×4, 768, LN		concat 4×4, 1024, LN	
		None	win 7×7, dim 768, head 24 $\times2$	None	win 7×7, dim 768, head 24 $\times2$	None	win 7×7, dim 1024, head 24 $\times2$

Table 8. Architectures of Slide-Swin models.

stage	output	Slide-CSwin-T		Slide-CSwin-S		Slide-CSwin-B	
		Slide Attention	CSwin Block	**Slide Attention**	CSwin Block	**Slide Attention**	CSwin Block
res1	56×56	Conv7×7, stride=4, 64, LN				Conv7×7, stride=4, 96, LN	
		win 3×3, dim 64, head 2 $\times1$	None	win 3×3, dim 64, head 2 $\times2$	None	win 3×3, dim 96, head 4 $\times2$	None
res2	28×28	Conv7×7, stride=4, 128, LN				Conv7×7, stride=4, 192, LN	
		win 3×3, dim 128, head 4 $\times2$	None	win 3×3, dim 128, head 4 $\times4$	None	win 3×3, dim 192, head 8 $\times4$	None
res3	14×14	Conv7×7, stride=4, 256, LN				Conv7×7, stride=384, LN	
		win 3×3, dim 256, head 8 $\times5$	win 7×7, dim 256, head 8 $\times16$	win 3×3, dim 256, head 8 $\times8$	win 7×7, dim 256, head 8 $\times24$	win 3×3, dim 384, head 16 $\times14$	win 7×7, dim 384, head 16 $\times18$
res4	7×7	Conv7×7, stride=4, 512, LN				Conv7×7, stride=4, 768, LN	
		None	win 7×7, dim 512, head 16 $\times1$	None	win 7×7, dim 512, head 16 $\times2$	None	win 7×7, dim 768, head 32 $\times2$

Table 9. Architectures of Slide-CSwin models.

stage	output	Slide-NAT-Mini		Slide-NAT-Tiny		Slide-NAT-Small	
		Slide Attention	Swin Block	**Slide Attention**	Swin Block	**Slide Attention**	Swin Block
res1	56×56	2 * Conv3×3, stride=2, 64, LN				2 * Conv3×3, stride=2, 96, LN	
		win 3×3, dim 64, head 2 $\times3$	None	win 3×3, dim 64, head 2 $\times3$	None	win 3×3, dim 96, head 3 $\times3$	None
res2	28×28	Conv3×3, stride=2, 128, LN				Conv3×3, stride=2, 192, LN	
		win 3×3, dim 128, head 4 $\times4$	None	win 3×3, dim 128, head 4 $\times4$	None	win 3×3, dim 192, head 6 $\times4$	None
res3	14×14	Conv3×3, stride=2, 256, LN				Conv3×3, stride=2, 384, LN	
		None	win 7×7, dim 256, head 8 $\times6$	win 3×3, dim 256, head 8 $\times10$	win 7×7, dim 256, head 8 $\times8$	win 3×3, dim 384, head 12 $\times10$	win 7×7, dim 384, head 12 $\times8$
res4	7×7	Conv3×3, stride=2, 512, LN				Conv3×3, stride=2, 768, LN	
		None	win 7×7, dim 512, head 16 $\times5$	None	win 7×7, dim 512, head 16 $\times5$	None	win 7×7, dim 768, head 24 $\times5$

Table 10. Architectures of Slide-NAT models.