Swin Transformer : Hierarchical Vision Transformer using Shifted Windows

Ze Liu, Yutong Lin, Yue Cao, Han Hu,
Yixuan Wei, Zheng Zhang, Stephen Lin, Baining Guo

ICCV 2021

Paper Presentation by : Li Hui Cham, Liujie Zheng

ViT vs CNN

- ViT captures global dependencies (self-attention) while CNN appreciate locality.
  - ViT has weaker inductive bias than CNN
- When train on mid-sized dataset, ResNet-like architectures perform better; however, ViT approached or surpassed the SOTA models’ performance with larger dataset
  - Computational demands can limit ViT efficiency in limited resources

Motivation

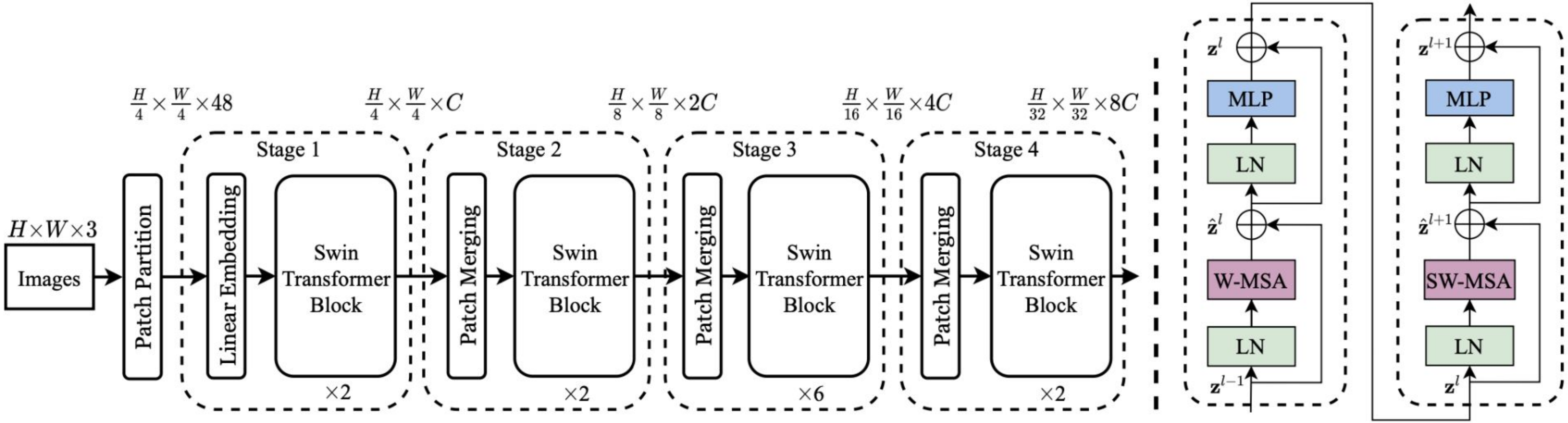
- ViT fails to process high-resolution image without having its **computational complexity scales to quadratic** of image size (global self-attention)
    - Standard ViT produces feature maps of a single **low resolution**
    - Tasks like semantic segmentation requires dense prediction at pixel level
- We want to expand the applicability of Transformer to serve as a **general purpose backbone** for computer vision

Question

How can we improve efficiency and greater accuracy with Transformer-based model ?

- Efficiency - lower computational complexity
- Accuracy - higher resolution, dense prediction

Swin Transformer Architecture



(a) Architecture

(b) Two Successive Swin Transformer Blocks

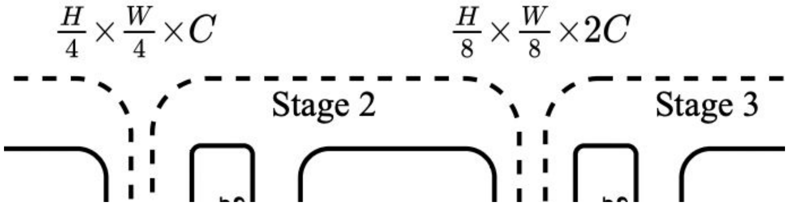
Swin Transformer Architecture

Key techniques :

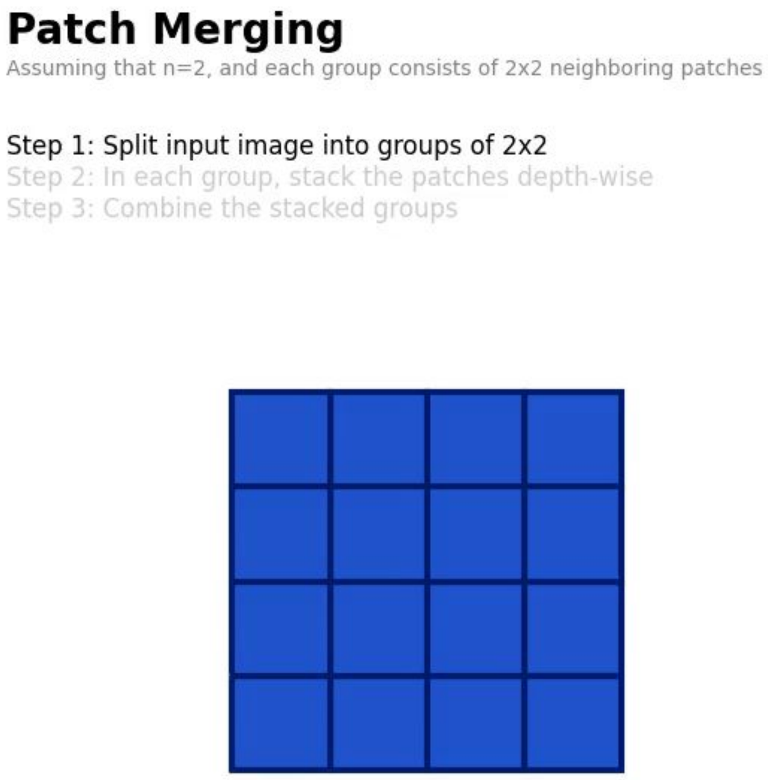
- Patch Merging
- **S**hifted **Win**dow Based Self Attention
    - Window based self-attention
    - Shifted window partitioning

Patch Merging

- To produce **hierarchical feature maps**
- Patch merging operation downsamples the input by **a factor of n** by **grouping nxn patches** and concatenating the patches depth-wise.
- H x W x C to (H/n) x (W/n) x (n^2*C)
- Lastly, apply a linear embedding layer to reduce dimension
  - (H/n) x (W/n) x $(n*C)$



C is the channel number of the hidden layers



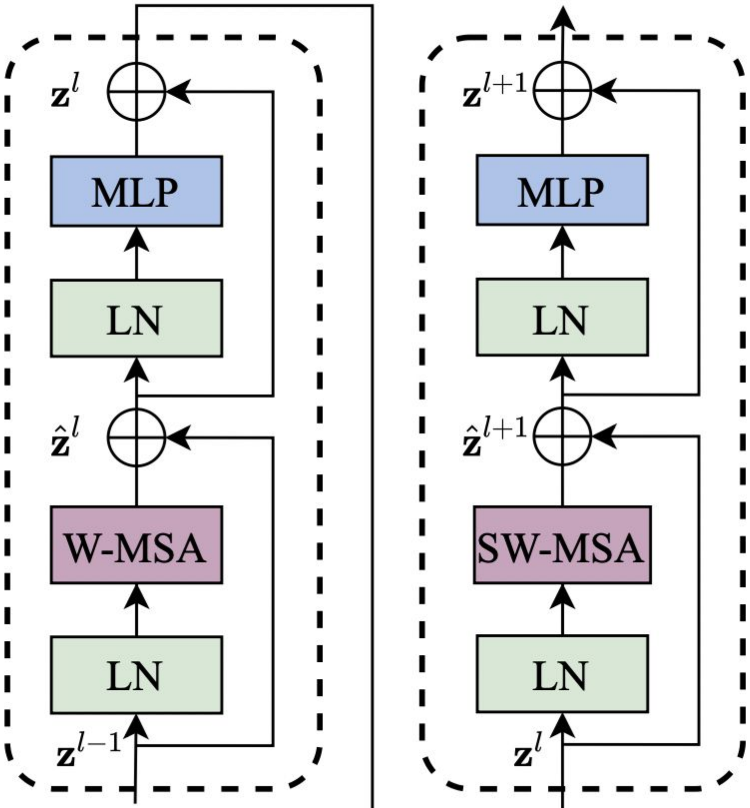
Patch merging operation downsamples the input by **a factor of n** by **grouping nxn patches** and concatenating the patches depth-wise.

a ‘patch’ refers to the smallest unit in a feature map.

Source : https://towardsdatascience.com/a-comprehensive-guide-to-swin-transformer-64965f89d14c

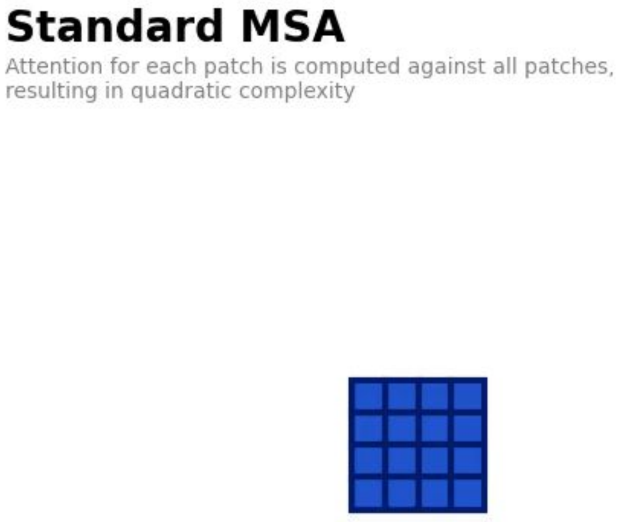
Swin Transformer Block

- Replaces standard multi-head self-attention module in ViT with W-MSA and SW-MSA
- W : window ; SW : shifted window



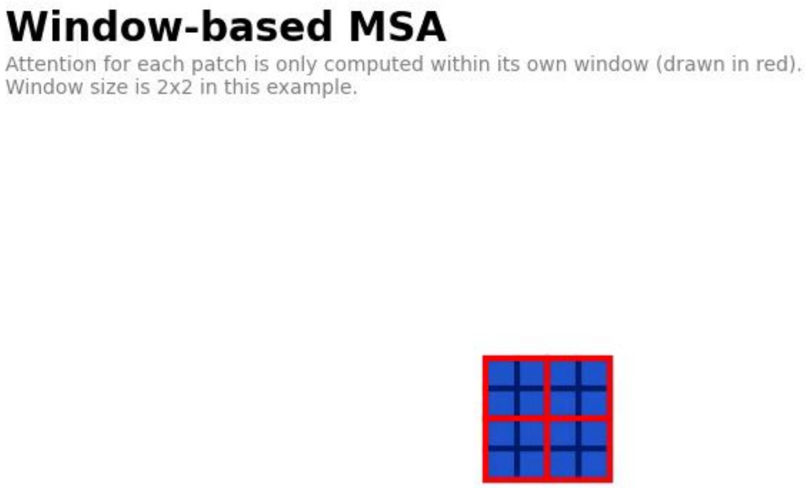
Window based self attention

- Multi self-attention (MSA) in standard ViT performs **global self-attention**
- Attention for each patch is computed against all patches
- Results in **quadratic computation complexity** wrt image size → **not suitable for high resolution image**



Window-based self-attention (W-MSA)

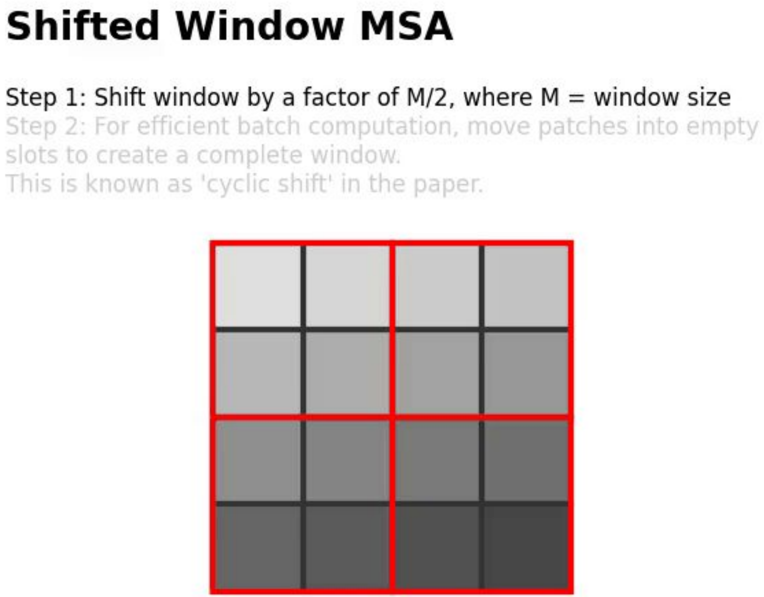
- Self-attention is computed within **local** window
- Windows partitions the image and are **non-overlapping**
- Fixed window size therefore **linear computational complexity**



Right : An image is partitioned into 4 windows (red), each window has 2x2 patches (“window size”)

Shifted Window Partitioning

- W-MSA is efficient but limits the modelling power of the network
- **Shifted Window MSA (SW-MSA)** introduce **cross-window connections**
- Displace the window by a factor of ***M*/2 pixels** (M is window size) towards the bottom right direction



Cyclic Shift

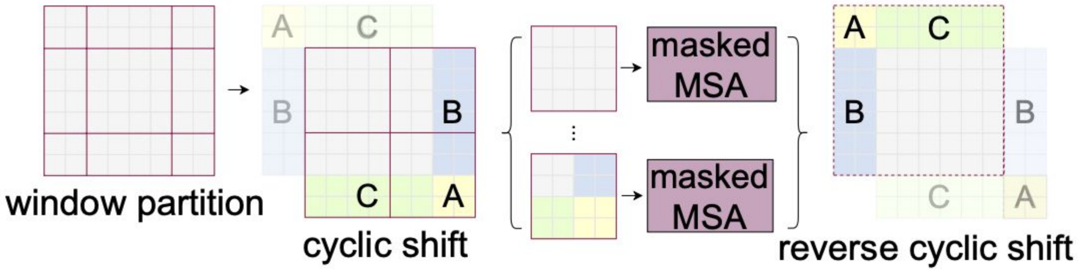


Figure 4. Illustration of an efficient batch computation approach for self-attention in shifted window partitioning.

- Shift results in **"isolated" patches and incomplete windows**
- **Cyclic shift** propose a more efficient batch computation
- A batched window consists of **non-adjacent sub-windows** in the original feature map
- **Masking mechanism** is employed to limit self-attention to within each sub-window

Image Classification on ImageNet-1K

a) trained on ImageNet-1K for 300 epochs
b) pre-trained on ImageNet-22K (22K classes) for 90 epochs then fine-tuned on ImageNet-1K for 30 epochs

(a) Regular ImageNet-1K trained models

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
RegNetY-4G [48]	224^2	21M	4.0G	1156.7	80.0
RegNetY-8G [48]	224^2	39M	8.0G	591.6	81.7
RegNetY-16G [48]	224^2	84M	16.0G	334.7	82.9
EffNet-B3 [58]	300^2	12M	1.8G	732.1	81.6
EffNet-B4 [58]	380^2	19M	4.2G	349.4	82.9
EffNet-B5 [58]	456^2	30M	9.9G	169.1	83.6
EffNet-B6 [58]	528^2	43M	19.0G	96.9	84.0
EffNet-B7 [58]	600^2	66M	37.0G	55.1	84.3
ViT-B/16 [20]	384^2	86M	55.4G	85.9	77.9
ViT-L/16 [20]	384^2	307M	190.7G	27.3	76.5
DeiT-S [63]	224^2	22M	4.6G	940.4	79.8
DeiT-B [63]	224^2	86M	17.5G	292.3	81.8
DeiT-B [63]	384^2	86M	55.4G	85.9	83.1
Swin-T	224^2	29M	4.5G	755.2	81.3
Swin-S	224^2	50M	8.7G	436.9	83.0
Swin-B	224^2	88M	15.4G	278.1	83.5
Swin-B	384^2	88M	47.0G	84.7	84.5

(b) ImageNet-22K pre-trained models

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
R-101x3 [38]	384^2	388M	204.6G	-	84.4
R-152x4 [38]	480^2	937M	840.5G	-	85.4
ViT-B/16 [20]	384^2	86M	55.4G	85.9	84.0
ViT-L/16 [20]	384^2	307M	190.7G	27.3	85.2
Swin-B	224^2	88M	15.4G	278.1	85.2
Swin-B	384^2	88M	47.0G	84.7	86.4
Swin-L	384^2	197M	103.9G	42.1	87.3

Object Detection on COCO 2017

a) Swin-T brings consistent +3.4~4.2 box AP gains over ResNet-50, with slightly larger model size, FLOPs and latency

b) Swin achieves significant gains over other backbones which has similar model size, FLOPs and latency.

(a) Various frameworks

Method	Backbone	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	#param.	FLOPs	FPS
Cascade Mask R-CNN	R-50	46.3	64.3	50.5	82M	739G	18.0
	Swin-T	**50.5**	**69.3**	**54.9**	86M	745G	15.3
ATSS	R-50	43.5	61.9	47.0	32M	205G	28.3
	Swin-T	**47.2**	**66.5**	**51.3**	36M	215G	22.3
RepPointsV2	R-50	46.5	64.6	50.3	42M	274G	13.6
	Swin-T	**50.0**	**68.5**	**54.2**	45M	283G	12.0
Sparse R-CNN	R-50	44.5	63.4	48.2	106M	166G	21.0
	Swin-T	**47.9**	**67.3**	**52.3**	110M	172G	18.4

(b) Various backbones w. Cascade Mask R-CNN

	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	param	FLOPs	FPS
DeiT-S†	48.0	67.2	51.7	41.4	64.2	44.3	80M	889G	10.4
R50	46.3	64.3	50.5	40.1	61.7	43.4	82M	739G	18.0
Swin-T	**50.5**	**69.3**	**54.9**	**43.7**	**66.6**	**47.1**	86M	745G	15.3
X101-32	48.1	66.5	52.4	41.6	63.9	45.2	101M	819G	12.8
Swin-S	**51.8**	**70.4**	**56.3**	**44.7**	**67.9**	**48.5**	107M	838G	12.0
X101-64	48.3	66.4	52.3	41.7	64.0	45.1	140M	972G	10.4
Swin-B	**51.9**	**70.9**	**56.5**	**45.0**	**68.4**	**48.7**	145M	982G	11.6

Object Detection on COCO 2017

c) The best Swin model achieves 58.7 box AP and 51.1 mask AP on COCO test-dev, surpassing the previous best results by +2.7 box AP (Copy-paste without external data) and +2.6 mask AP (DetectoRS).

(c) System-level Comparison

Method	mini-val		test-dev		#param.	FLOPs
	AP^{box}	AP^{mask}	AP^{box}	AP^{mask}		
RepPointsV2* [12]	-	-	52.1	-	-	-
GCNet* [7]	51.8	44.7	52.3	45.4	-	1041G
RelationNet++* [13]	-	-	52.7	-	-	-
SpineNet-190 [21]	52.6	-	52.8	-	164M	1885G
ResNeSt-200* [78]	52.5	-	53.3	47.1	-	-
EfficientDet-D7 [59]	54.4	-	55.1	-	77M	410G
DetectoRS* [46]	-	-	55.7	48.5	-	-
YOLOv4 P7* [4]	-	-	55.8	-	-	-
Copy-paste [26]	55.9	47.2	56.0	47.4	185M	1440G
X101-64 (HTC++)	52.3	46.0	-	-	155M	1033G
Swin-B (HTC++)	56.4	49.1	-	-	160M	1043G
Swin-L (HTC++)	57.1	49.5	57.7	50.2	284M	1470G
Swin-L (HTC++)*	**58.0**	**50.4**	**58.7**	**51.1**	284M	-

Semantic Segmentation on ADE20K

Swin-S is +5.3 mIoU higher than DeiT-S with similar computation cost. It is also +4.4 mIoU higher than ResNet-101, and +2.4 mIoU higher than ResNeSt-101.

Swin-L model with ImageNet-22K pre-training achieves 53.5 mIoU on the val set, surpassing the previous best model by +3.2 mIoU (50.3 mIoU by SETR which has a larger model size).

ADE20K Method	Backbone	val mIoU	test score	#param.	FLOPs	FPS
DANet [23]	ResNet-101	45.2	-	69M	1119G	15.2
DLab.v3+ [11]	ResNet-101	44.1	-	63M	1021G	16.0
ACNet [24]	ResNet-101	45.9	38.5	-		
DNL [71]	ResNet-101	46.0	56.2	69M	1249G	14.8
OCRNet [73]	ResNet-101	45.3	56.0	56M	923G	19.3
UperNet [69]	ResNet-101	44.9	-	86M	1029G	20.1
OCRNet [73]	HRNet-w48	45.7	-	71M	664G	12.5
DLab.v3+ [11]	ResNeSt-101	46.9	55.1	66M	1051G	11.9
DLab.v3+ [11]	ResNeSt-200	48.4	-	88M	1381G	8.1
SETR [81]	T-Large‡	50.3	61.7	308M	-	-
UperNet	DeiT-S†	44.0	-	52M	1099G	16.2
UperNet	Swin-T	46.1	-	60M	945G	18.5
UperNet	Swin-S	49.3	-	81M	1038G	15.2
UperNet	Swin-B‡	51.6	-	121M	1841G	8.7
UperNet	Swin-L‡	**53.5**	**62.8**	234M	3230G	6.2

Ablation Study

Swin-T with the shifted window partitioning outperforms the counterpart built on a single window partitioning

Swin-T with relative position bias outperforms out counterparts

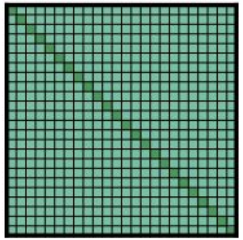
	ImageNet		COCO		ADE20k
	top-1	top-5	AP^{box}	AP^{mask}	mIoU
w/o shifting	80.2	95.1	47.7	41.5	43.3
shifted windows	**81.3**	**95.6**	**50.5**	**43.7**	**46.1**
no pos.	80.1	94.9	49.2	42.6	43.8
abs. pos.	80.5	95.2	49.0	42.4	43.2
abs.+rel. pos.	81.3	95.6	50.2	43.4	44.0
rel. pos. w/o app.	79.3	94.7	48.2	41.9	44.1
rel. pos.	**81.3**	**95.6**	**50.5**	**43.7**	**46.1**

Relative position bias:

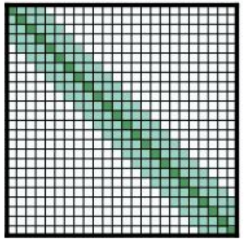
$$\text{Attention}(Q, K, V) = \text{SoftMax}(QK^T/\sqrt{d} + B)V, \quad (4)$$

Ablation Study

Shifted window attention is faster than sliding window attention and Performer (which is one of the fastest Transformer architectures) while having similar performance



(a) Full n^2 attention



(b) Sliding window attention

method	MSA in a stage (ms) S1	S2	S3	S4	Arch. (FPS) T	S	B
sliding window (naive)	122.5	38.3	12.1	7.6	183	109	77
sliding window (kernel)	7.6	4.7	2.7	1.8	488	283	187
Performer [14]	4.8	2.8	1.8	1.5	638	370	241
window (w/o shifting)	2.8	1.7	1.2	0.9	770	444	280
shifted window (padding)	3.3	2.3	1.9	2.2	670	371	236
shifted window (cyclic)	3.0	1.9	1.3	1.0	755	437	278

	Backbone	ImageNet top-1	top-5	COCO AP^{box}	AP^{mask}	ADE20k mIoU
sliding window	Swin-T	81.4	95.6	50.2	43.5	45.8
Performer [14]	Swin-T	79.0	94.2	-	-	-
shifted window	Swin-T	81.3	95.6	50.5	43.7	46.1

Summary

- **Novelty.** The Shifted Window Self Attention mechanism introduces linear computational complexity in ViT using local window while keeping the global representation of the image with cross-window connections.
- **Scalable.** The author modifies the self-attention mechanism on ViT instead of CNN architecture, Swin Transformer achieves the goal of a general purpose backbone for CV.
- **Fair comparison.** The authors are trying their best to make fair comparisons by comparing models with similar sizes and FLOPs.
- **Speed-accuracy trade off.** Comparing to previous state-of-the-art, Swin Transformer achieves better accuracy with similar speed on multiple tasks. Comparing to sliding window attention, shifted window attention maintains similar performance while being significantly faster.

Thank you