Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Ze Liu, Yutong Lin, Yue Cao1, Han Hu, Yixuan Wei, Zheng Zhang,
Stephen Lin,Baining Guo
Presenter: Yilin Wang

2022/3/29

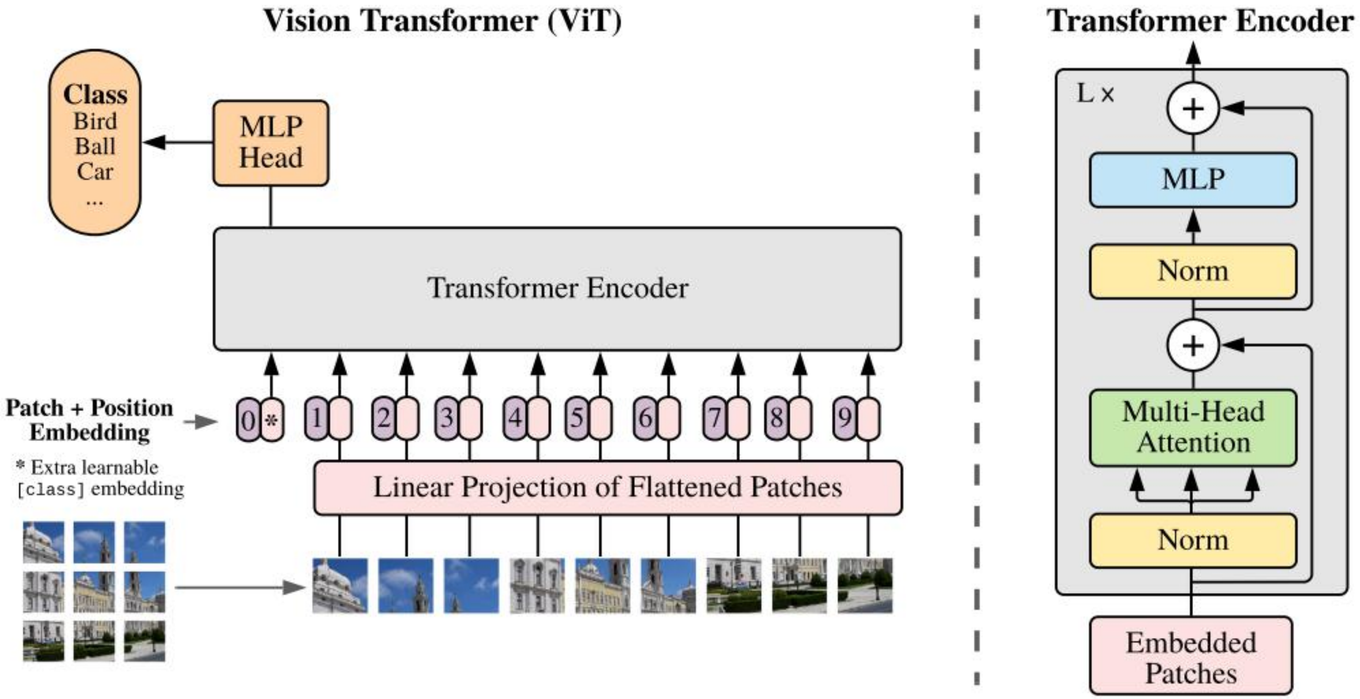
Background

- ViT (2020)
  - **Standard Transformer**
  - **Global instead of local**

➢single low resolution (14 x 14 / 16 x 16 patches)

- semantic segmention
- object detection

➢quadratic complexity



$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}};\, \mathbf{x}_p^1\mathbf{E};\, \mathbf{x}_p^2\mathbf{E};\cdots;\, \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}, \qquad \mathbf{E}\in\mathbb{R}^{(P^2\cdot C)\times D},\ \mathbf{E}_{pos}\in\mathbb{R}^{(N+1)\times D} \quad (1)$$

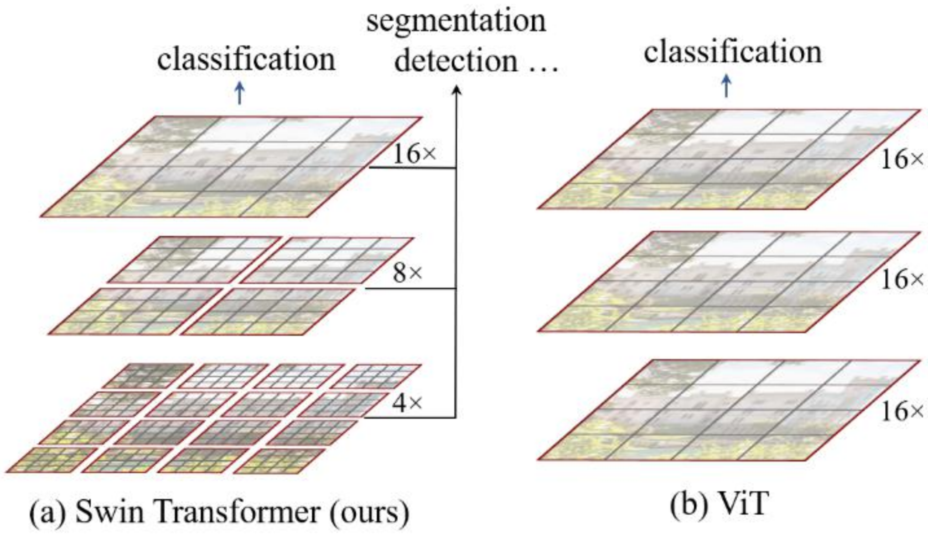
$$\mathbf{z'}_\ell = \mathrm{MSA}(\mathrm{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, \qquad \ell = 1\ldots L \quad (2)$$

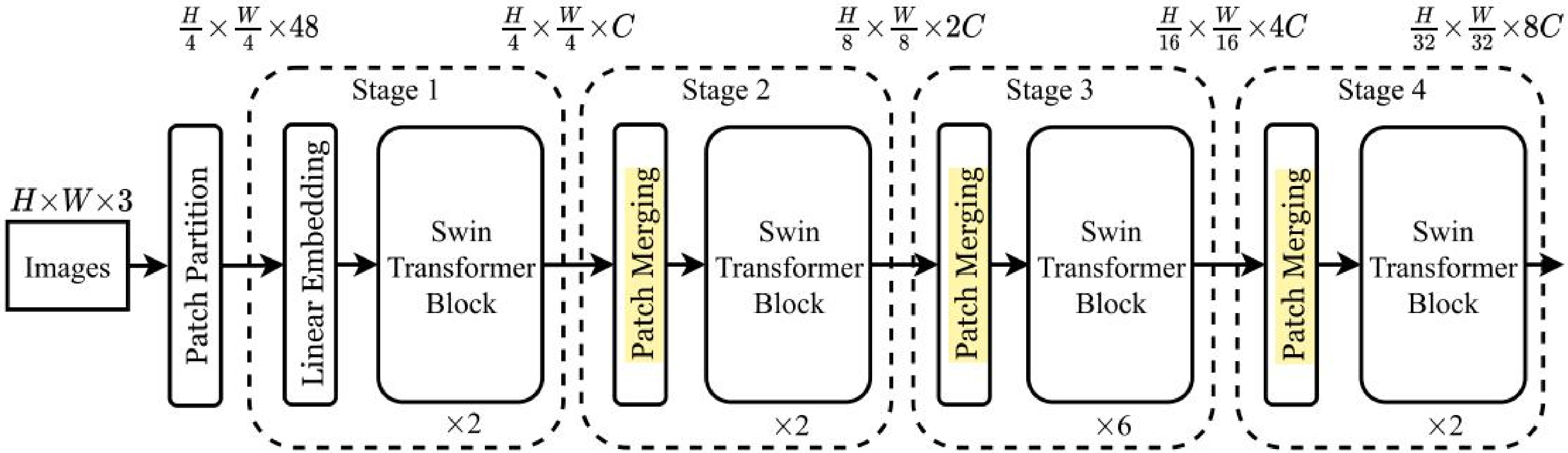
$$\mathbf{z}_\ell = \mathrm{MLP}(\mathrm{LN}(\mathbf{z'}_\ell)) + \mathbf{z'}_\ell, \qquad \ell = 1\ldots L \quad (3)$$

$$\mathbf{y} = \mathrm{LN}(\mathbf{z}_L^0) \quad (4)$$

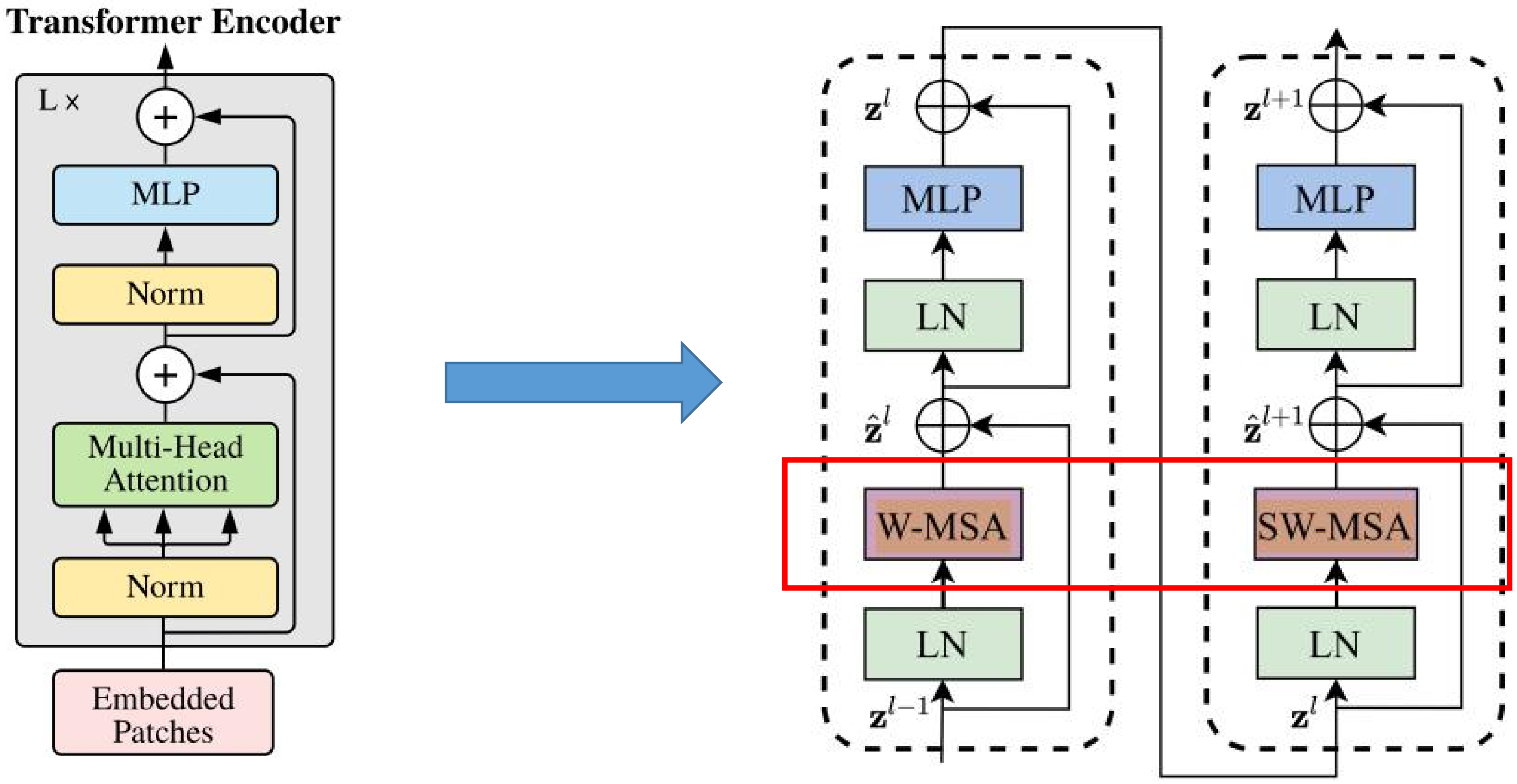
Hierarchical Architecture

- Patch Partition （P=4）
- Patch Merging * 3





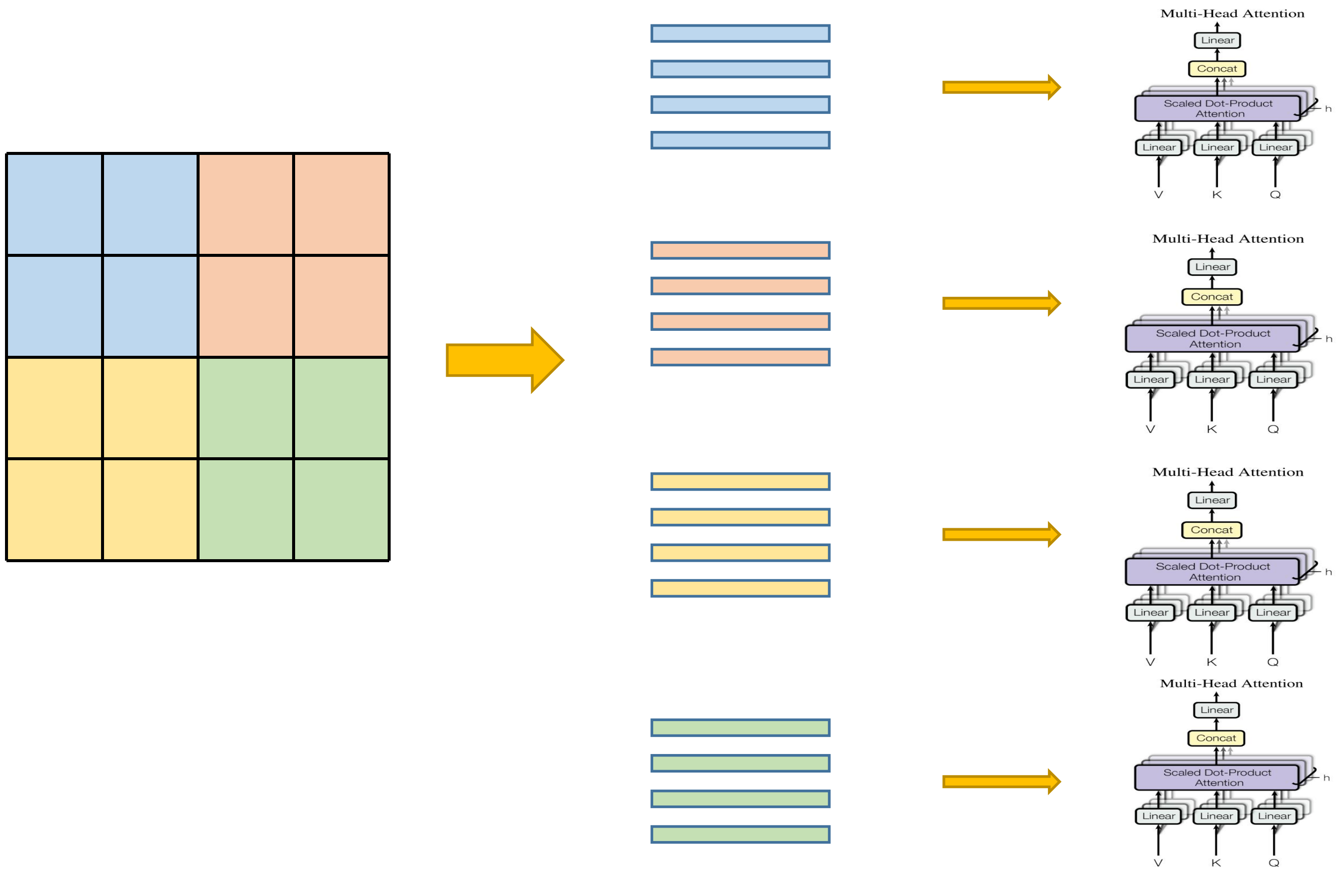
Swin Transformer Block

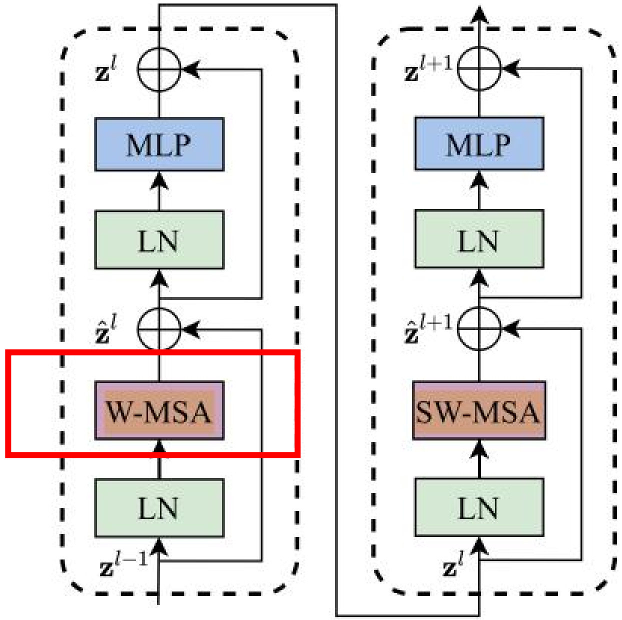


Shifted Window based Self-Attention

Shifted Window based Self-Attention

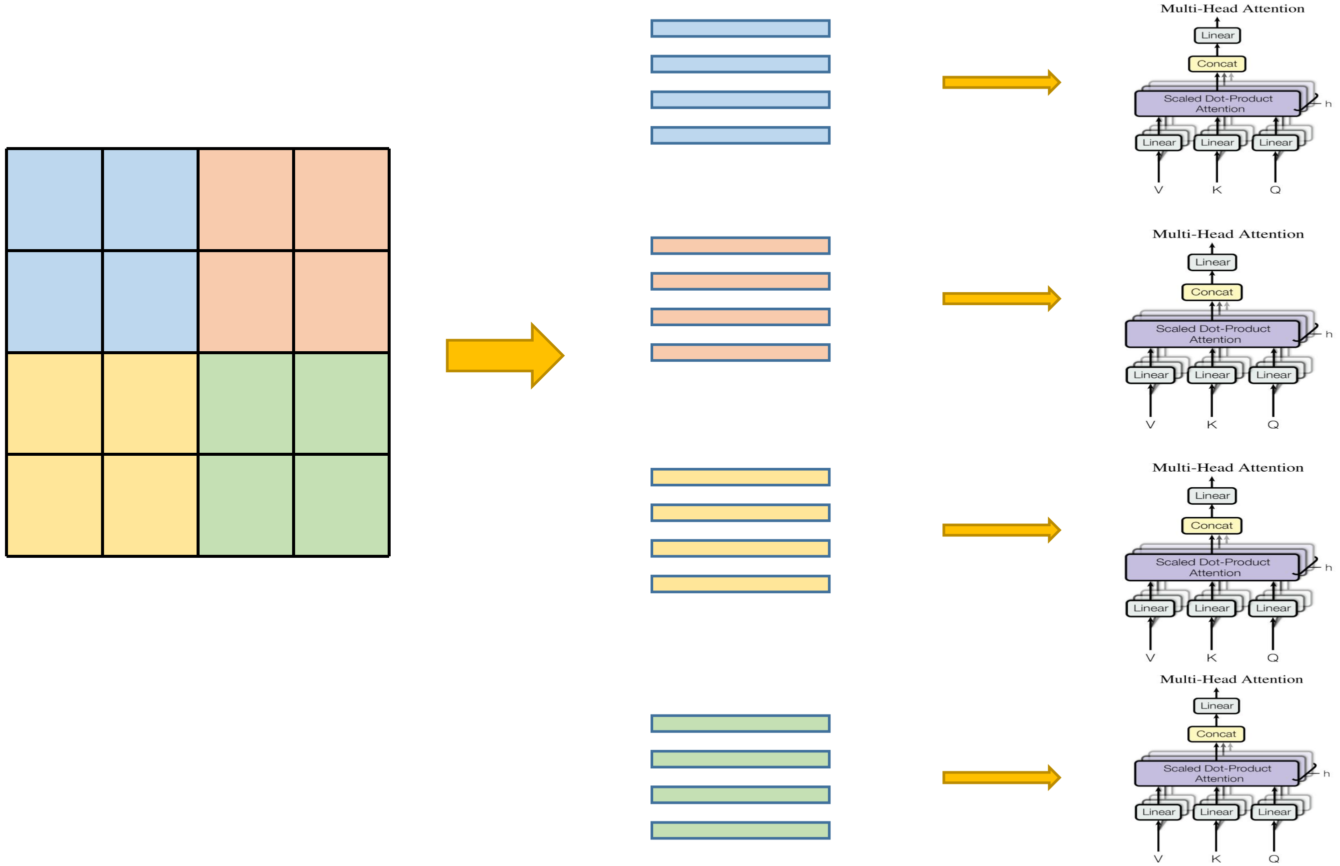
- Self-attention in non-overlapped windows (W-MSA)
  - Attention in local windows containing M x M patches

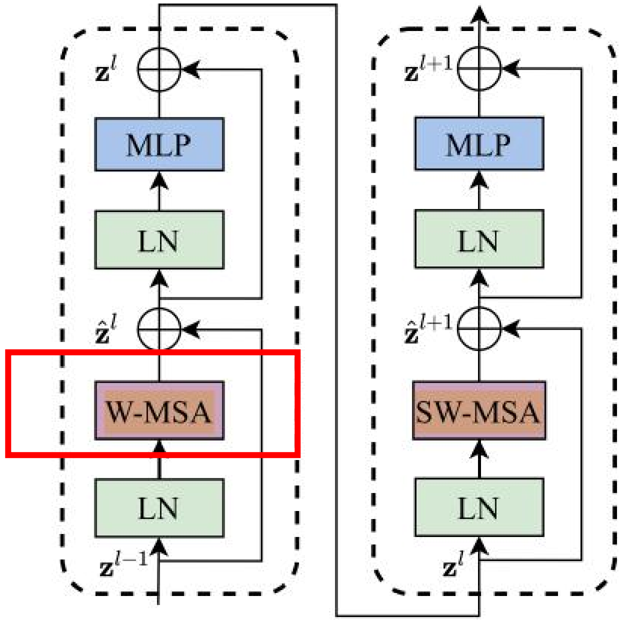




Shifted Window based Self-Attention

- Self-attention in non-overlapped windows (W-MSA)
  - Attention in local windows containing M x M patches





$\Omega(h^2w^2)$

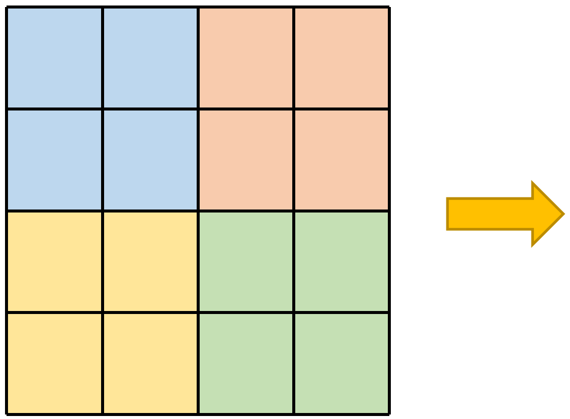
$$\Omega(\mathrm{MSA}) = 4hwC^2 + 2(hw)^2C, \quad (1)$$

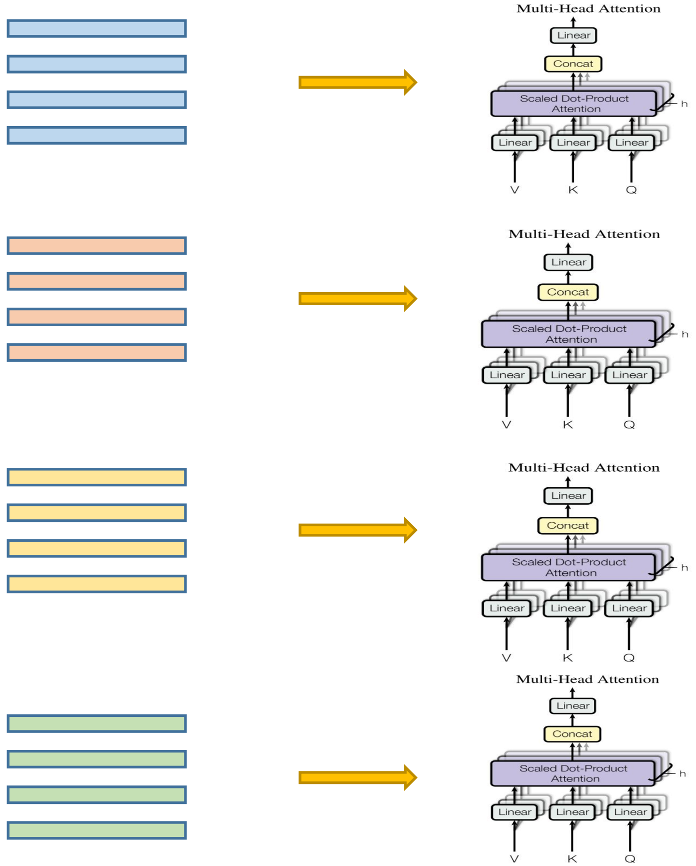
$$\Omega(\text{W-MSA}) = 4hwC^2 + 2M^2hwC, \quad (2)$$

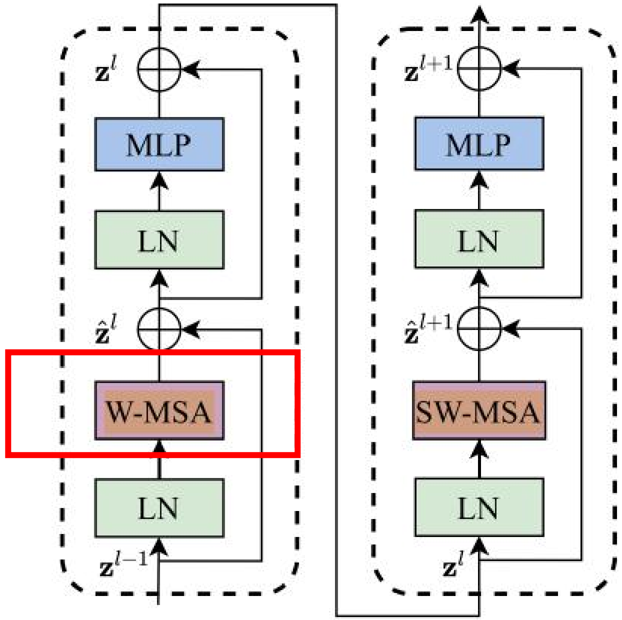
$\Omega(hw)$

Shifted Window based Self-Attention

- Self-attention in non-overlapped windows (W-MSA)
  - Attention in local windows containing M x M patches







Limited modeling power: lacks connection across windows

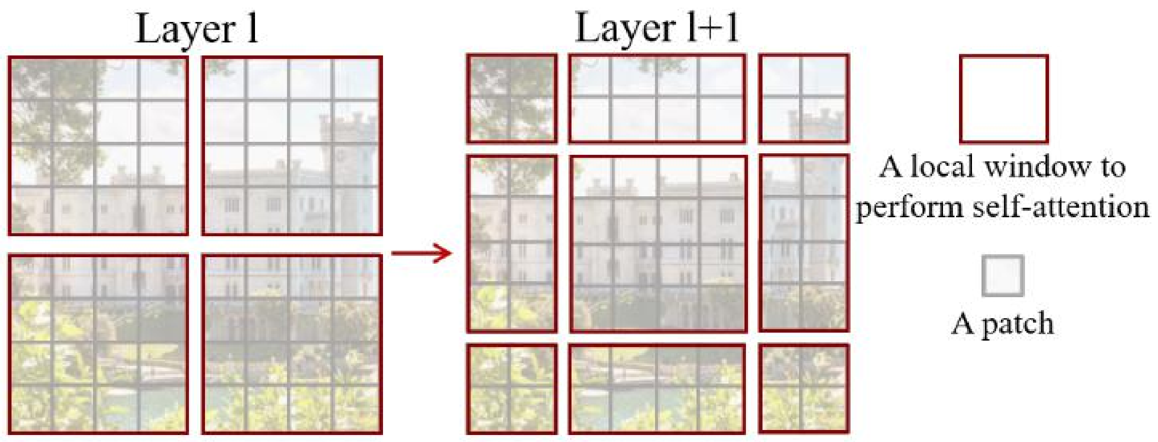
$\Omega(h^2w^2)$

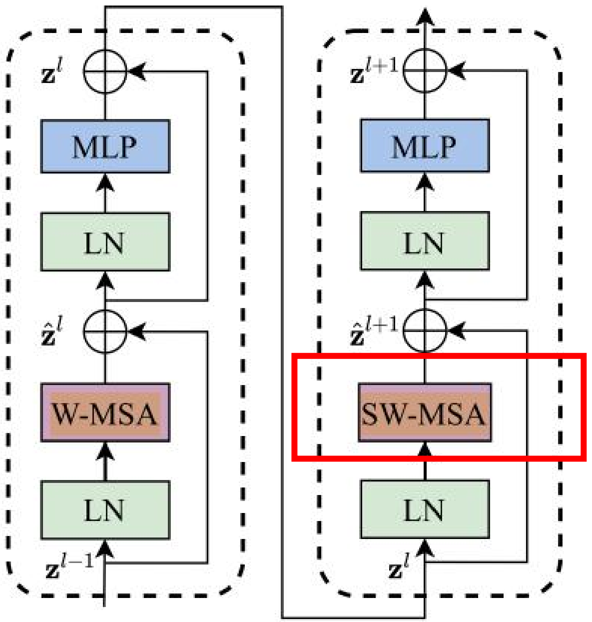
$$\Omega(\mathrm{MSA}) = 4hwC^2 + 2(hw)^2C, \quad (1)$$
$$\Omega(\text{W-MSA}) = 4hwC^2 + 2M^2hwC, \quad (2)$$

$\Omega(hw)$

Shifted Window based Self-Attention

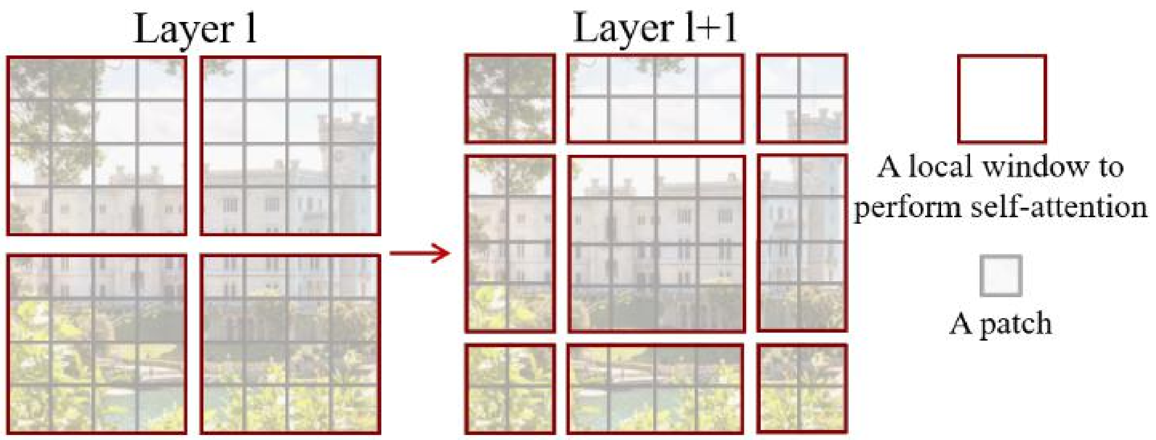
- Shifted windows partitioning (SW-MSA)
  - Shift the regular windows of W-MSA by $\left(\left\lfloor\frac{M}{2}\right\rfloor, \left\lfloor\frac{M}{2}\right\rfloor\right)$



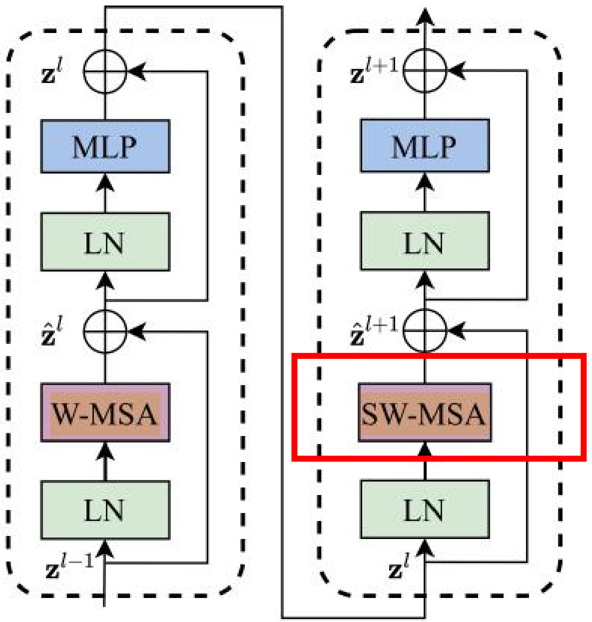


Shifted Window based Self-Attention

- Shifted windows partitioning (SW-MSA)
  - Shift the regular windows of W-MSA by $\left(\left\lfloor\frac{M}{2}\right\rfloor, \left\lfloor\frac{M}{2}\right\rfloor\right)$



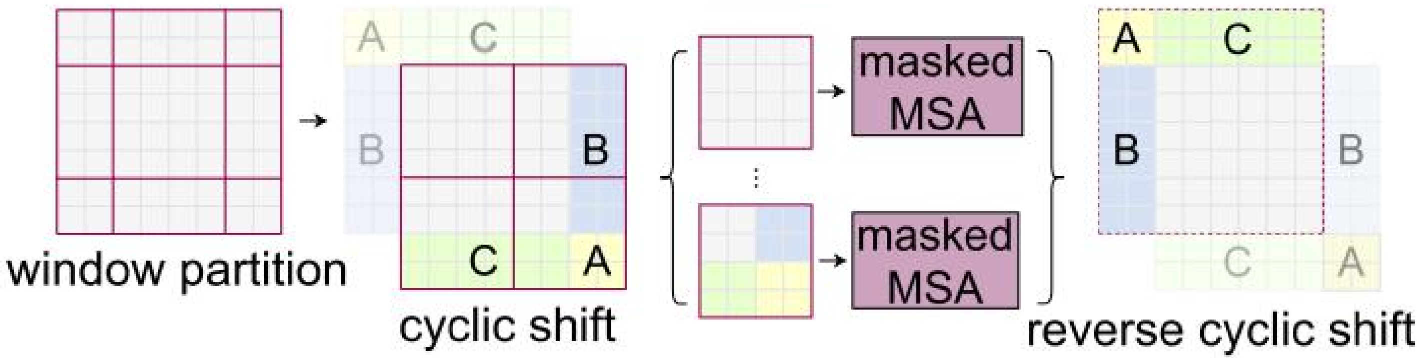
2 x 2 $\xrightarrow{\text{x 2.25}}$ 3 x 3

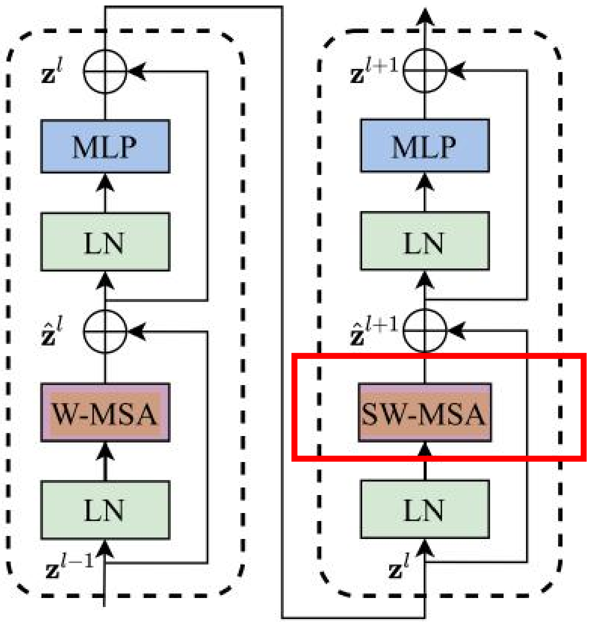


Considerable increase in the amount of windows

Shifted Window based Self-Attention

- Efficient batch computation for shifted configuration
  - masked-MSA: limit self-attention computation within only adjacent sub-windows
  - Maintain the number of batched windows as the same as that of regular window partitioning

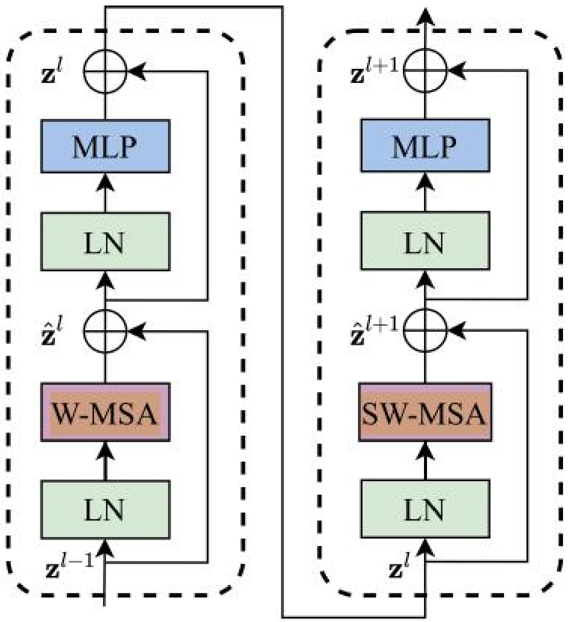




Shifted Window based Self-Attention

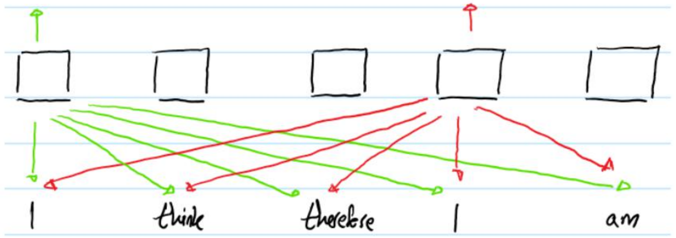
- Shifted window partitioning in successive blocks
  - Alternates between two partitioning configurations

$$\hat{\mathbf{z}}^l = \text{W-MSA}\left(\text{LN}\left(\mathbf{z}^{l-1}\right)\right) + \mathbf{z}^{l-1},$$
$$\mathbf{z}^l = \text{MLP}\left(\text{LN}\left(\hat{\mathbf{z}}^l\right)\right) + \hat{\mathbf{z}}^l,$$
$$\hat{\mathbf{z}}^{l+1} = \text{SW-MSA}\left(\text{LN}\left(\mathbf{z}^l\right)\right) + \mathbf{z}^l,$$
$$\mathbf{z}^{l+1} = \text{MLP}\left(\text{LN}\left(\hat{\mathbf{z}}^{l+1}\right)\right) + \hat{\mathbf{z}}^{l+1},$$



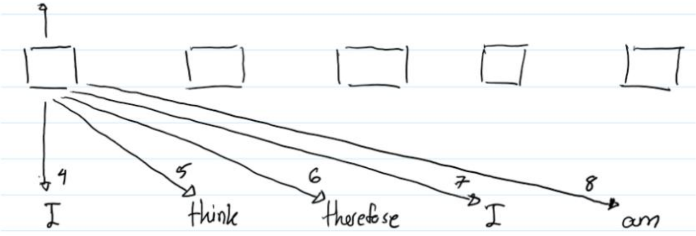
Shifted Window based Self-Attention

- Relative position bias



Absolute position embeddings

$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

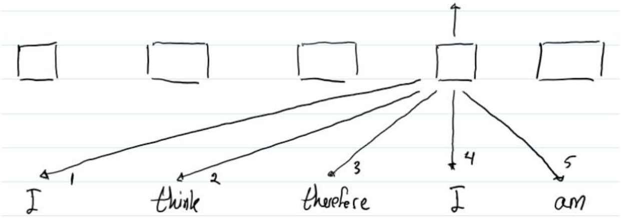


Relative position embeddings

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

$$B \in \mathbb{R}^{M^2 \times M^2}$$

$$[-M+1, M-1]$$



Results

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
(a) Regular ImageNet-1K trained models					
RegNetY-4G [44]	224^2	21M	4.0G	1156.7	80.0
RegNetY-8G [44]	224^2	39M	8.0G	591.6	81.7
RegNetY-16G [44]	224^2	84M	16.0G	334.7	82.9
ViT-B/16 [19]	384^2	86M	55.4G	85.9	77.9
ViT-L/16 [19]	384^2	307M	190.7G	27.3	76.5
DeiT-S [57]	224^2	22M	4.6G	940.4	79.8
DeiT-B [57]	224^2	86M	17.5G	292.3	81.8
DeiT-B [57]	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 [34]	384^2	388M	204.6G	-	84.4
R-152x4 [34]	480^2	937M	840.5G	-	85.4
ViT-B/16 [19]	384^2	86M	55.4G	85.9	84.0
ViT-L/16 [19]	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

Table 1. Comparison of different backbones on ImageNet-1K classification. Throughput is measured using the GitHub repository of [62] and a V100 GPU, following [57].

Image Classification

(a) Various frameworks

Method	Backbone	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	#param.	FLOPs	FPS
Cascade	R-50	46.3	64.3	50.5	82M	739G	18.0
Mask R-CNN	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-50	44.5	63.4	48.2	106M	166G	21.0
R-CNN	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

(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	-	-	-
DetectoRS* [42]	-	-	55.7	48.5	-	-
YOLOv4 P7* [4]	-	-	55.8	-	-	-
Copy-paste [23]	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	-

Table 2. Results on COCO object detection and instance segmentation. †denotes that additional decovolution layers are used to produce hierarchical feature maps. * indicates multi-scale testing.

Object Detection

ADE20K Method	Backbone	val mIoU	test score	#param.	FLOPs	FPS
DLab.v3+ [11]	ResNet-101	44.1	-	63M	1021G	16.0
DNL [65]	ResNet-101	46.0	56.2	69M	1249G	14.8
OCRNet [67]	ResNet-101	45.3	56.0	56M	923G	19.3
UperNet [63]	ResNet-101	44.9	-	86M	1029G	20.1
OCRNet [67]	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 [73]	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

Table 3. Results of semantic segmentation on the ADE20K val and test set. † indicates additional deconvolution layers are used to produce hierarchical feature maps. ‡ indicates that the model is pre-trained on ImageNet-22K.

Semantic Segmentation

Ablation Study

	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**

Table 4. Ablation study on the *shifted windows* approach and different position embedding methods on three benchmarks, using the Swin-T architecture. w/o shifting: all self-attention modules adopt regular window partitioning, without *shifting*; abs. pos.: absolute position embedding term of ViT; rel. pos.: the default settings with an additional relative position bias term (see Eq. (4)); app.: the first scaled dot-product term in Eq. (4).

method	MSA in a stage (ms)				Arch. (FPS)		
	S1	S2	S3	S4	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

Table 5. Real speed of different self-attention computation methods and implementations on a V100 GPU.

Shifted windows configuration

Batch computation based on cyclic shift

Swin Transformer V2: Scaling Up Capacity and Resolution

Ze Liu* Han Hu*† Yutong Lin Zhuliang Yao Zhenda Xie Yixuan Wei Jia Ning
Yue Cao Zheng Zhang Li Dong Furu Wei Baining Guo
Microsoft Research Asia
{v-zeliu1,hanhu,t-yutonglin,t-zhuyao,t-zhxie,t-yixuanwei,v-jianing}@microsoft.com
{yuecao,zhez,lidong1,fuwei,bainguo}@microsoft.com

Contribution

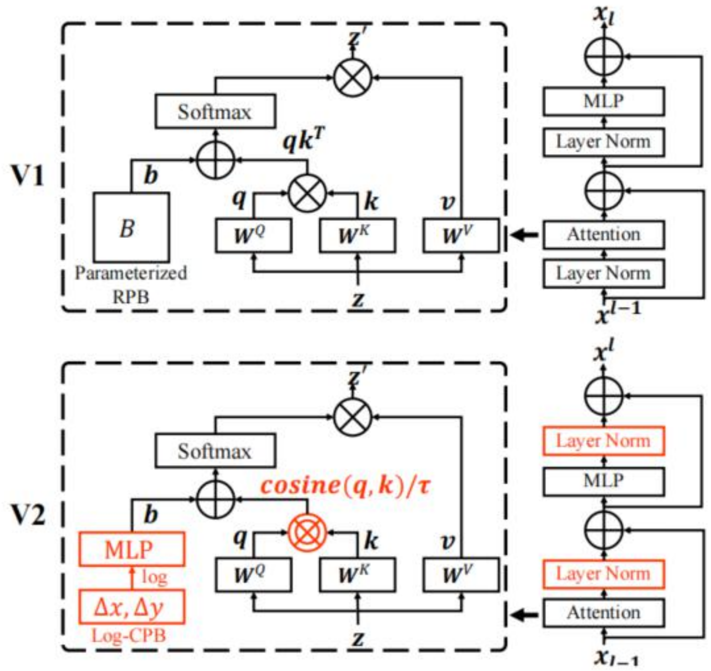
- scaling up to **3 billion parameters**
- resolution up to **1,536 x 1,536**
- open source of crucial implementation details of **saving GPU memory**

Adaptations from V1

- pre-norm -> post-norm
- dot product attention -> scaled cosine attention
- relative position bias -> log spaced continuous relative position bias

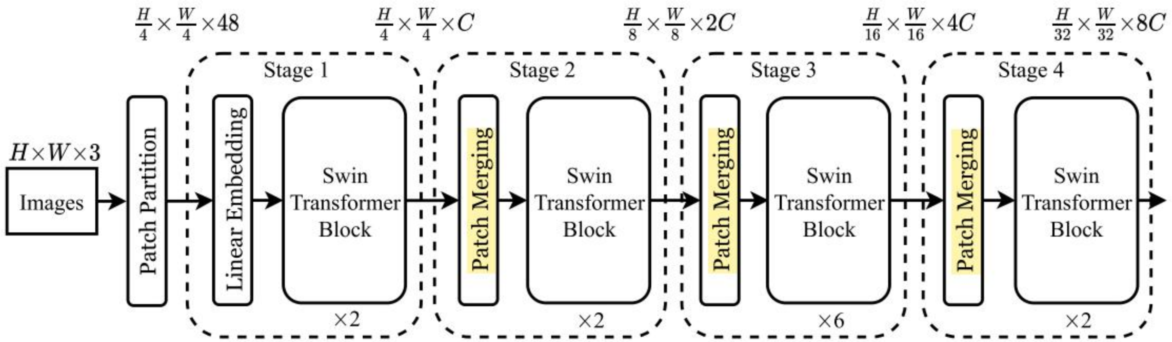
capacity

resolution



Implementation Tricks

- Zero-Redundancy Optimizer (ZeRO)
  - Regular: broadcast model parameters and optimization states to every GPU or a master node => too redundant for large model
  - ZeRO: the model parameters and the corresponding optimization states will be divided and distributed to multiple GPUs



- Activation check-pointing
  - Saving space of feature maps

- Sequential self-attention computation
  - Compute self-attention sequentially instead of batch-wise in first two stages

Results

Method	param	pre-train images	pre-train length (#im)	pre-train im size	pre-train time	fine-tune im size	ImageNet-1K-V1 top-1 acc	ImaegNet-1K-V2 top-1 acc
SwinV1-B	88M	IN-22K-14M	1.3B	224^2	$<30^\dagger$	384^2	86.4	76.58
SwinV1-L	197M	IN-22K-14M	1.3B	224^2	$<10^\dagger$	384^2	87.3	77.46
ViT-G [65]	1.8B	JFT-3B	164B	224^2	$>30k$	518^2	90.45	83.33
V-MoE [44]	14.7B*	JFT-3B	-	224^2	16.8k	518^2	90.35	-
CoAtNet-7 [11]	2.44B	JFT-3B	-	224^2	20.1k	512^2	**90.88**	-
SwinV2-B	88M	IN-22K-14M	1.3B	192^2	$<30^\dagger$	384^2	87.1	78.08
SwinV2-L	197M	IN-22K-14M	1.3B	192^2	$<20^\dagger$	384^2	87.7	78.31
SwinV2-G	3.0B	IN-22K-ext-70M	3.5B	192^2	$<0.5k^\dagger$	640^2	90.17	**84.00**

Table 2. Comparison with previous largest vision models on ImageNet-1K V1 and V2 classification. * indicates the sparse model; the "pre-train time" column is measured by the TPUv3 core days with numbers copied from the original papers. † That of SwinV2-G is estimated according to training iterations and FLOPs.

Method	train I(W) size	test I(W) size	mini-val (AP) box	mini-val (AP) mask	test-dev (AP) box	test-dev (AP) mask
CopyPaste [17]	1280(-)	1280(-)	57.0	48.9	57.3	49.1
SwinV1-L [35]	800(7)	ms(7)	58.0	50.4	58.7	51.1
YOLOR [53]	1280(-)	1280(-)	-	-	57.3	-
CBNet [32]	1400(7)	ms(7)	59.6	51.8	60.1	52.3
DyHead [10]	1200(-)	ms(-)	60.3	-	60.6	-
SoftTeacher [60]	1280(12)	ms(12)	60.7	52.5	61.3	53.0
SwinV2-L (HTC++)	1536(32)	1100(32)	58.8	51.1	-	-
		1100 (48)	58.9	51.2	-	-
		ms (48)	60.2	52.1	60.8	52.7
SwinV2-G (HTC++)	1536(32)	1100(32)	61.7	53.3	-	-
		1100 (48)	61.9	53.4	-	-
		ms (48)	**62.5**	**53.7**	**63.1**	**54.4**

Table 3. Comparison with previous best results on COCO object detection and instance segmentation. I(W) indicates the image and window size. ms indicate multi-scale testing is employed.

Method	train I(W) size	test I(W) size	mIoU
SwinV1-L [35]	640(7)	640(7)	53.5*
Focal-L [61]	640(40)	640(40)	55.4*
CSwin-L [14]	640(40)	640(40)	55.7*
MaskFormer [8]	640(7)	640(7)	55.6*
FaPN [22]	640(7)	640(7)	56.7*
BEiT [3]	640(40)	640(40)	58.4*
SwinV2-L (UperNet)	640(40)	640(40)	55.9*
SwinV2-G (UperNet)	640(40)	640(40)	59.1
		896 (56)	59.3
		896 (56)	**59.9***

Table 4. Comparison with previous best results on ADE20K semantic segmentation. * indicates multi-scale testing is used.

Thank you!