Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Machine Learning Course Team 16

310511067 陳品樺　310511061 林彥廷

310510179 王縮晴　309652008 廖家緯

Outline

1. **Introduction:** motivation and history
2. **Method:** network architecture
3. **Experiment:** classification, detection, and segmentation
4. **Conclusion**

Introduction: motivation

From NLP to CV

Use transformers architecture from NLP to CV

Scale problem

Visual tokens can vary a lot in scale

Computation

Reduce computational of high-resolution image

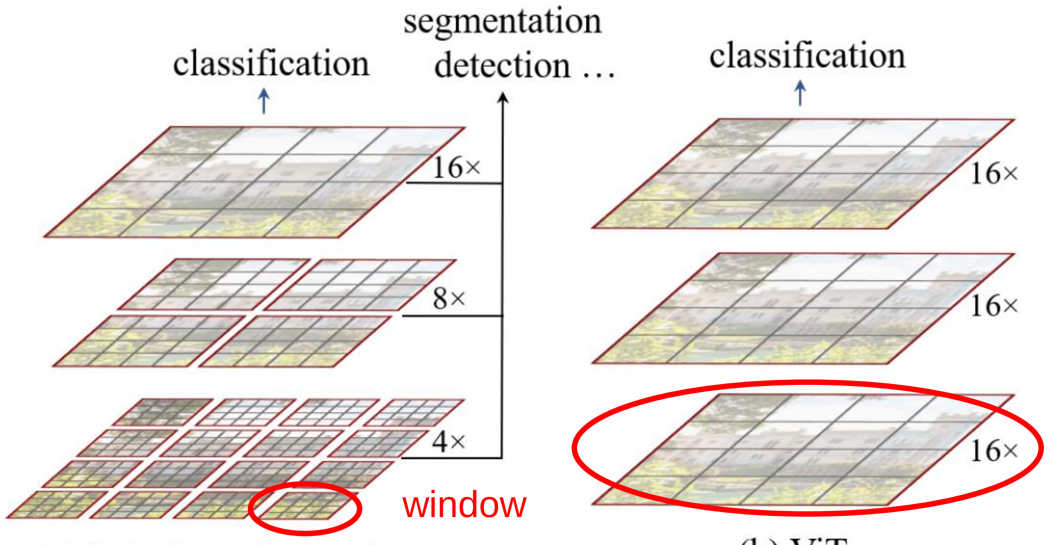
Introduction: swin transformer

Hierarchical

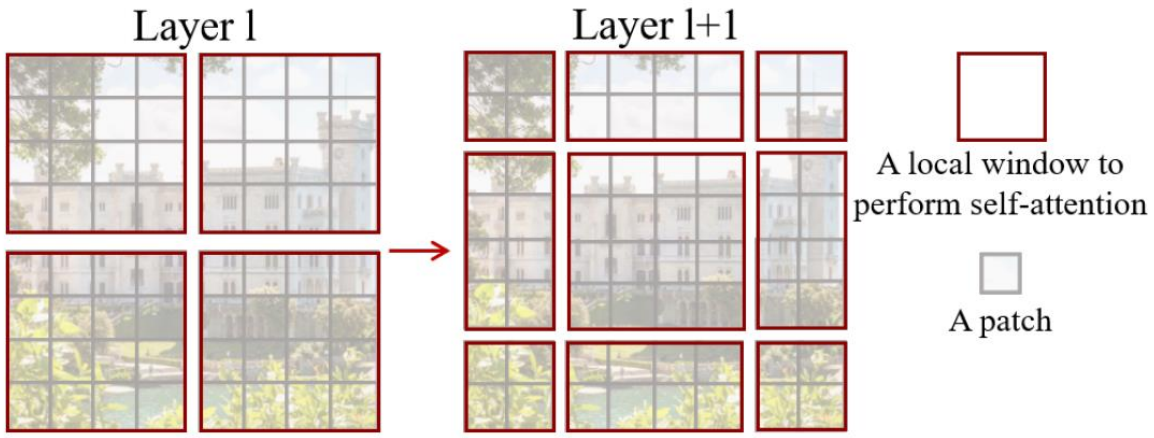
reduce self-attention complexity to linear time

Shifted Windows

connect the patch in different windows



(a) Swin Transformer (ours) (b) ViT



Introduction: history of swin transformer in 2021

Release code to GitHub

4/12

5/12

Transformer-SSL

Used as a backbone for Self-Supervised Learning

6/25

Video Swin Transformer

SOTA accuracy on video recognition benchmarks

7/3

Add MLP

replacing all multi-head self-attention blocks by MLP layers

8/9

Soft Teacher

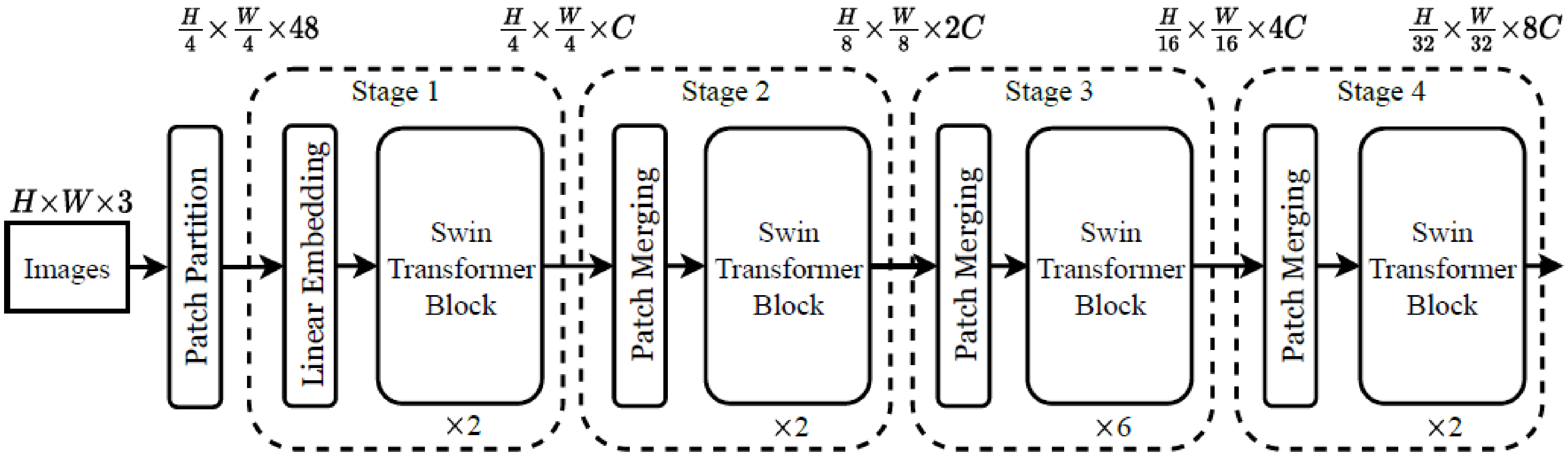
end-to-end semi-supervisd object detection method

10/12

ICCV 2021 best paper award

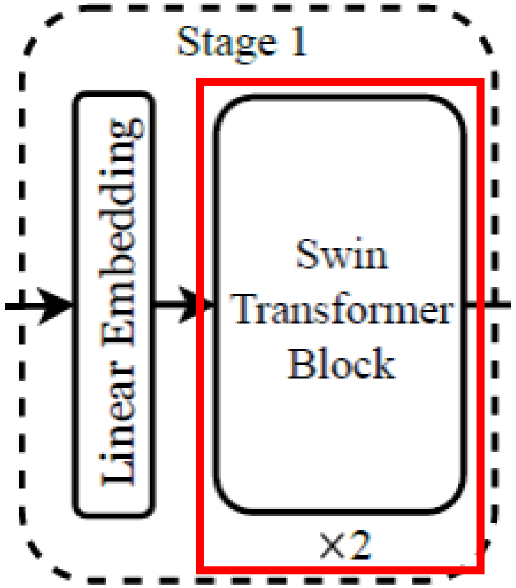
Method: network architecture

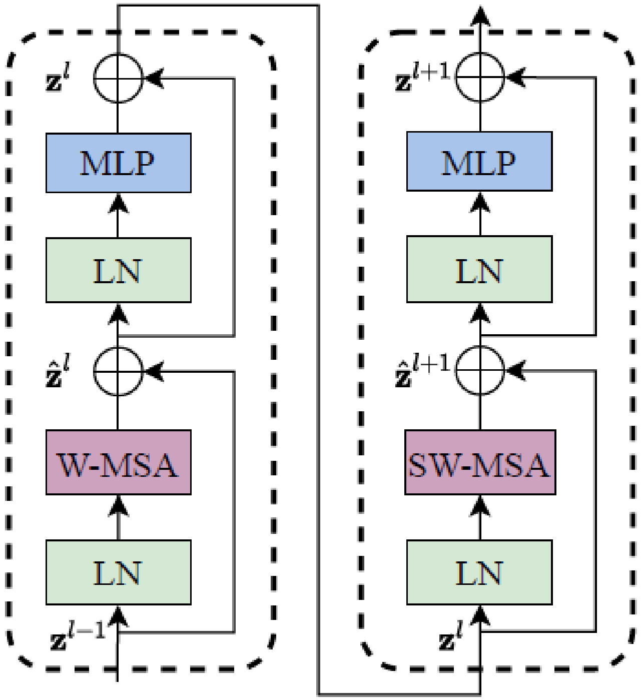
- Partition H x W image into 4 x 4 patches
- This architecture is composed of 4 stages, one for each scale
- Linear embedding layer compute the features of each token
- Patch merging layer reduce the number of patches for multi-scale learning



Architecture: swin transformer block

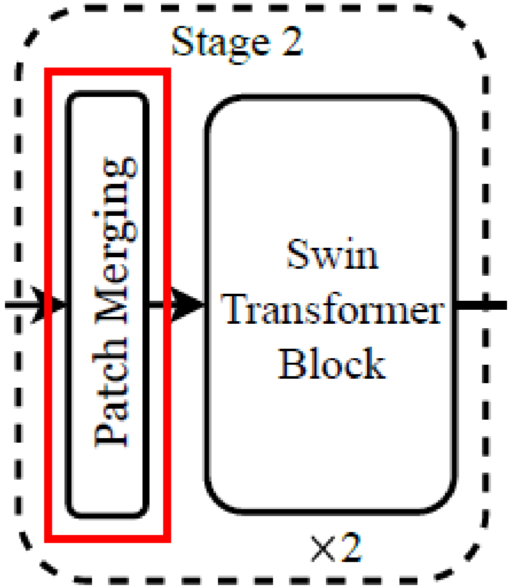
- **W-MSA:** Window based Multi-head Self Attention
- **SW-MSA:** Shifted Window based Multi-head Self Attention
- **LN:** Layer normalization
- **MLP:** Multilayer Perceptron (Linear-GELU-Linear-Dropout)

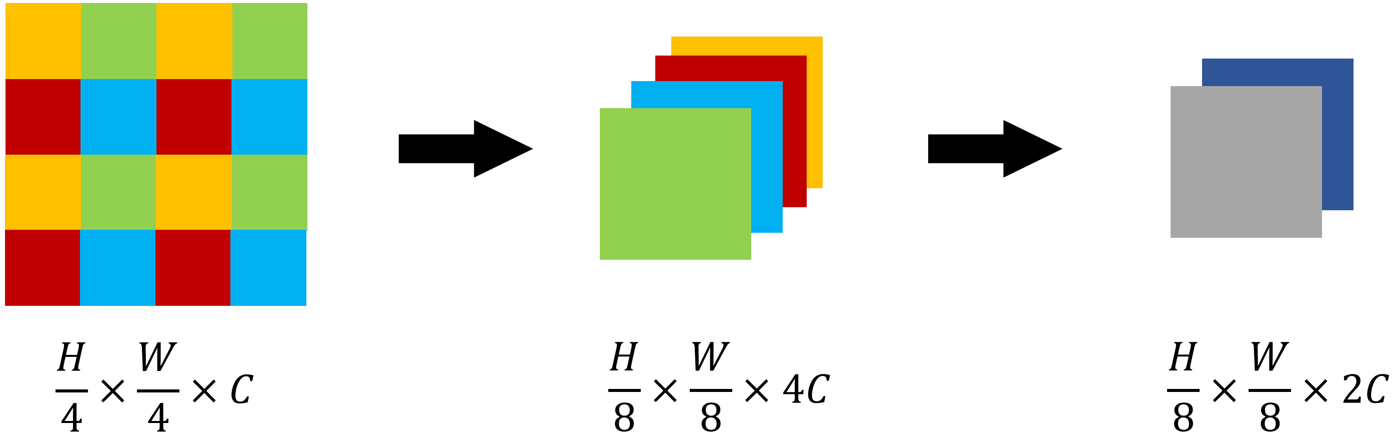




Architecture: patch merging layer

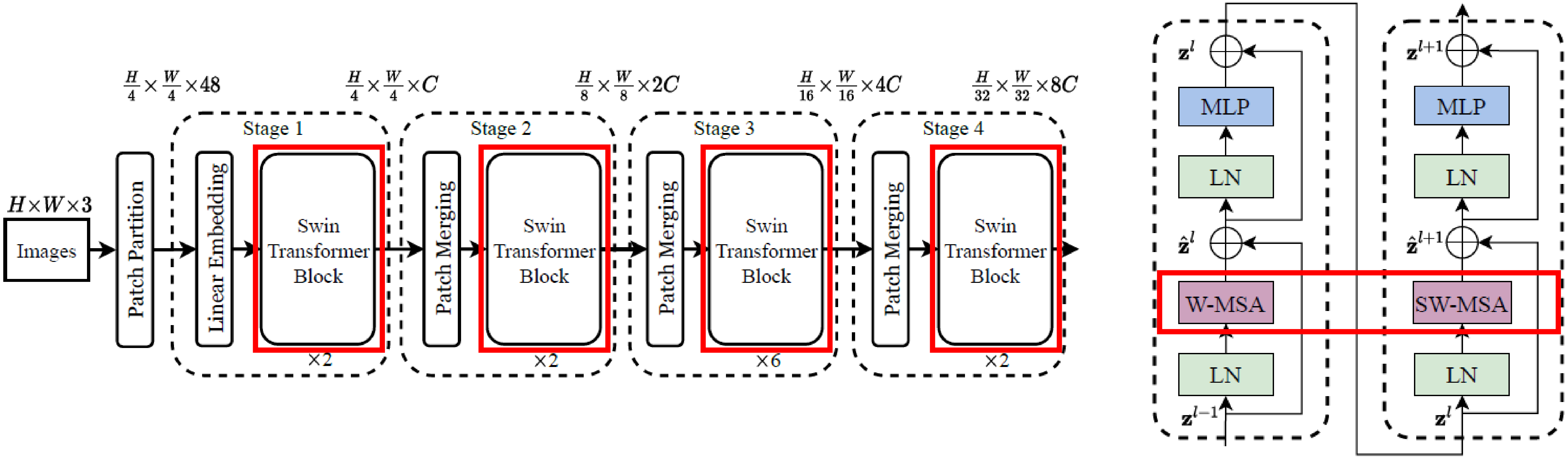
- It concatenates the features of each group of 2 × 2 neighboring patches
- A linear layer is applied to reduce the dimension from 4C to 2C



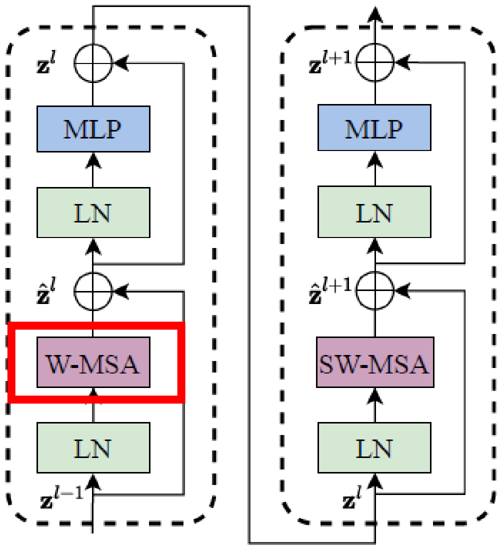


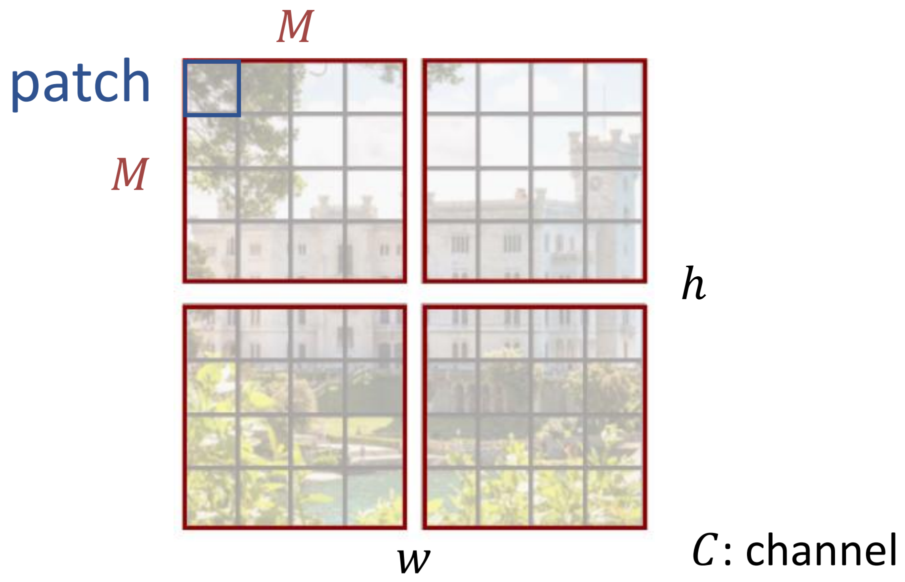
CNN	Swin
Convolution	W-MSA
Pooling	Patch Merging

Swin transformer block



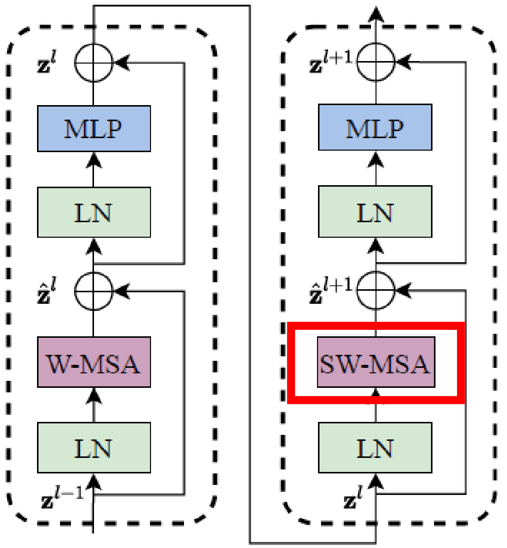
Swin transformer block: W-MSA layer

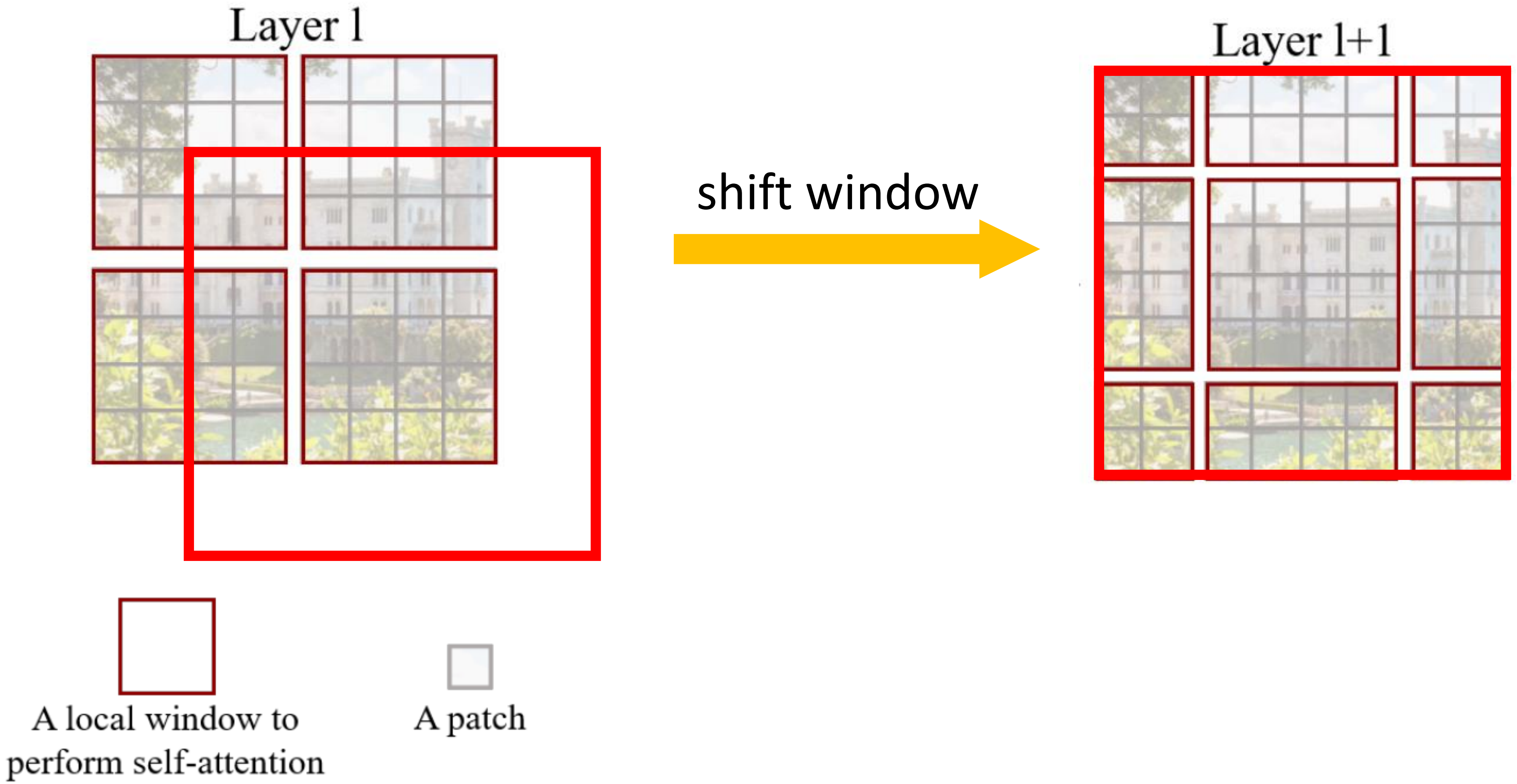




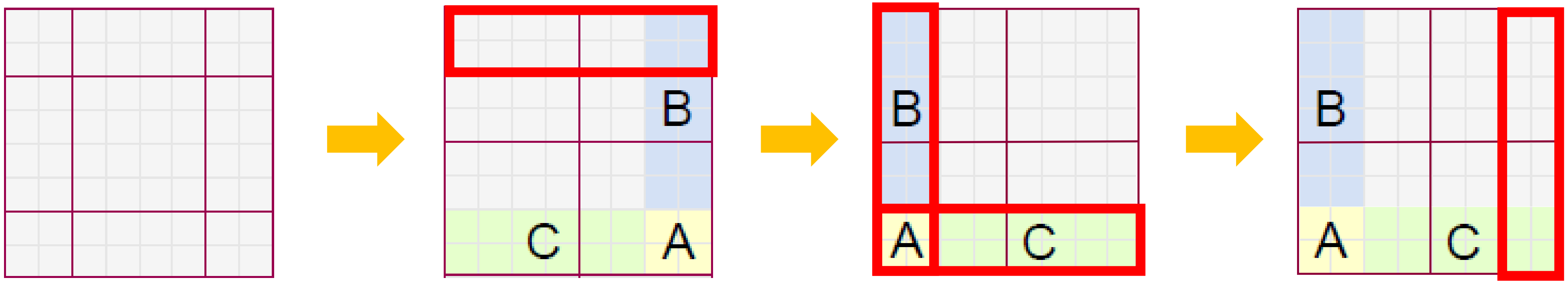
	ViT	Swin
Shape of Q, K, V	$\mathbb{R}^{hw \times C}$	$\mathbb{R}^{M^2 \times C}$
Compute Q, K, V	$\Omega(hwC^2)$	$\Omega(hwC^2)$
Compute QK^T	$\Omega((hw)^2C)$	$\Omega(M^2hwC)$
Compute softmax $\left(\frac{QK^T}{\sqrt{dk}}\right)V$	$\Omega((hw)^2C)$	$\Omega(M^2hwC)$

Swin transformer block: SW-MSA layer





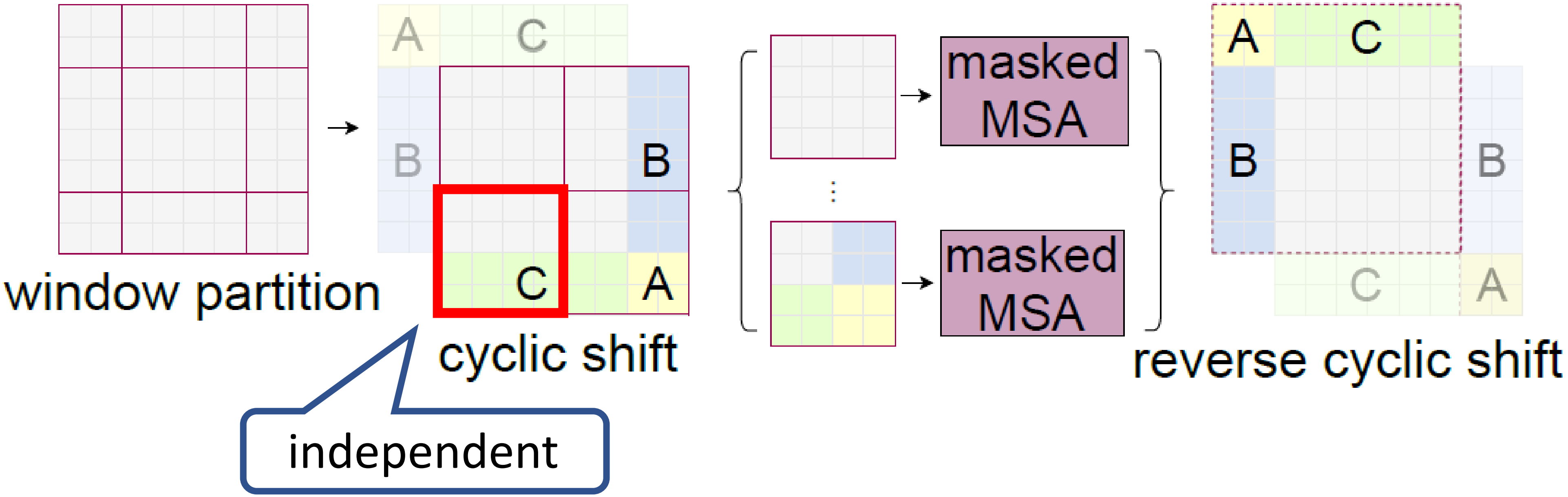
Swin transformer block: SW-MSA layer



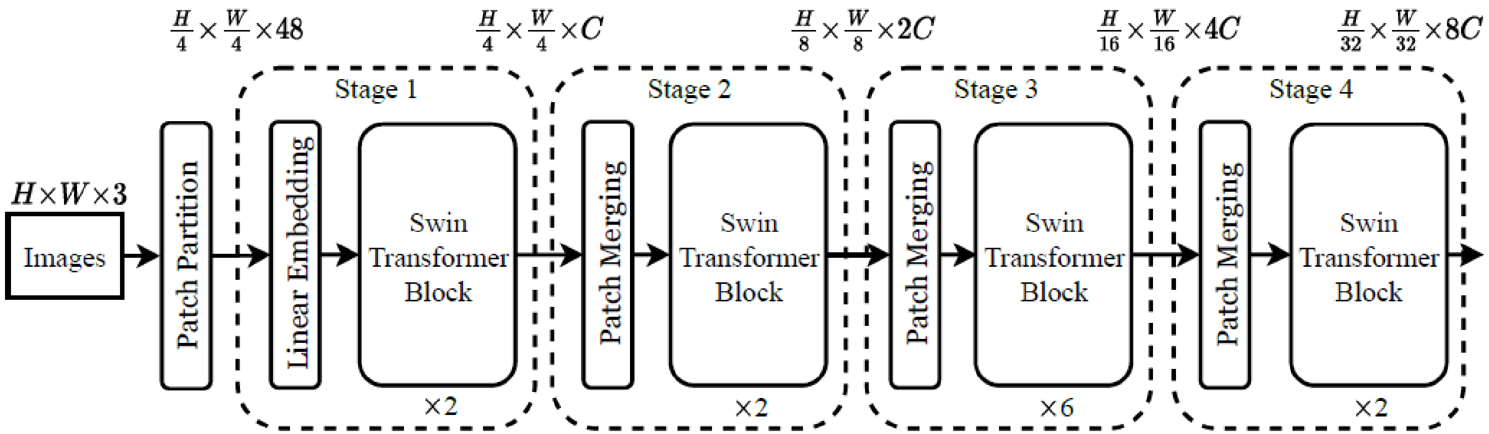
Cyclic Shift

Swin transformer block: SW-MSA layer

1. Do the cyclic shift to the figure after window shift
2. Perform self-attention computations
3. Mask non-adjacent regions
4. Reverse the region of cyclic shift



Swin transformer models of different sizes



- Swin-T: $C = 96$, layer numbers = $\{2, 2, 6, 2\}$
- Swin-S: $C = 96$, layer numbers =$\{2, 2, 18, 2\}$
- Swin-B: $C = 128$, layer numbers =$\{2, 2, 18, 2\}$
- Swin-L: $C = 192$, layer numbers =$\{2, 2, 18, 2\}$

number of channels

number of swin transformer blocks in each stage

Experiments: image classification on ImageNet

- Comparison among RegNetY, EfficientNet, ViT, DeiT, and Swin-T

(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

Experiments: object detection on COCO dataset

- R-50 vs. Swin-T on detection with different detection frameworks

(a) Various frameworks

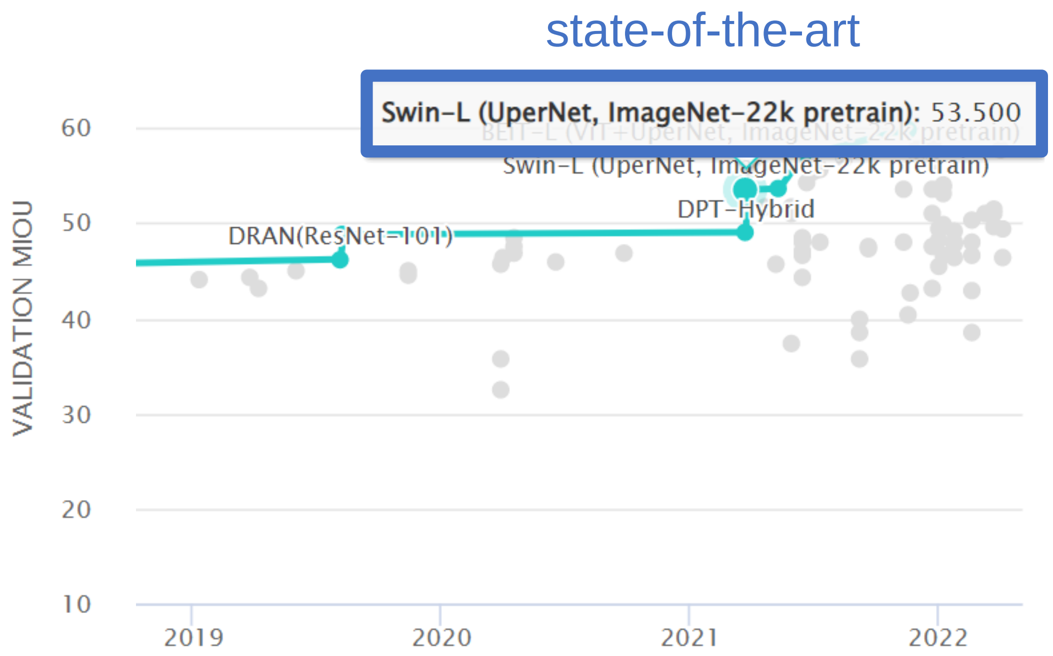
Method	Backbone	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	#param.	FLOPs	FPS
Cascade Mask R-CNN	R-50	46.3	64.3	50.5	82M	739G	18.0
	Swin-T	**50.5**	**69.3**	**54.9**	86M	745G	15.3
ATSS	R-50	43.5	61.9	47.0	32M	205G	28.3
	Swin-T	**47.2**	**66.5**	**51.3**	36M	215G	22.3
RepPointsV2	R-50	46.5	64.6	50.3	42M	274G	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

AP：average precision

Experiments: Semantic segmentation on ADE20K

- Comparison among different backbones on different segmentation frameworks

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



Conclusion

- Swin Transformer produces a hierarchical feature representation and has linear computational complexity with respect to input image size

- Swin Transformer achieves the state-of-the-art performance on COCO object detection and ADE20K semantic segmentation

Contribution

- 310511067 陳品樺: introduction, slides designing
- 309652008 廖家緯: network architecture, slides designing
- 310511061 林彥廷: network architecture, slides designing
- 310510179 王綰晴: experiments, slides designing

Thanks for your listening