ICCV 2021
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

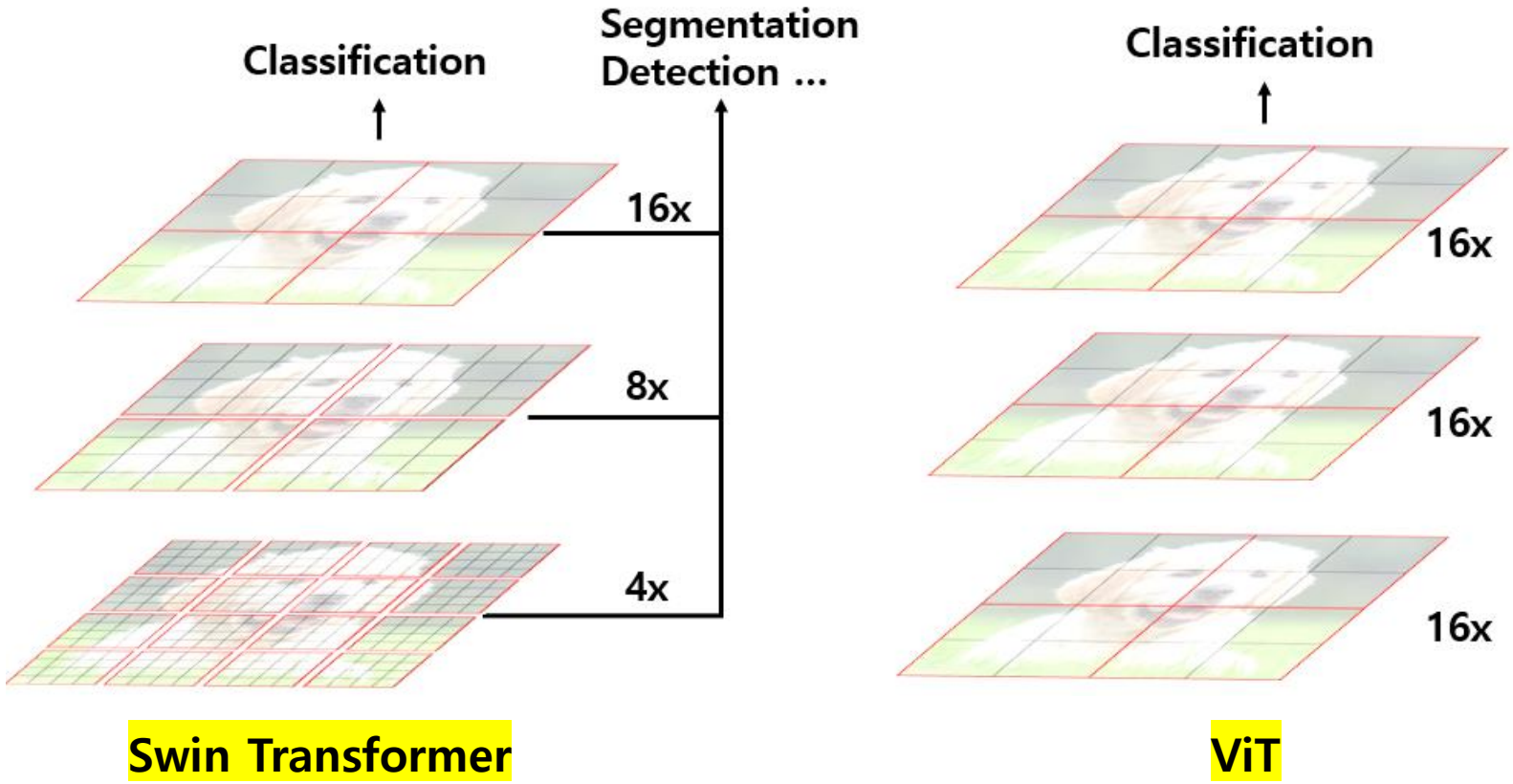
2022.08.05

논문 리뷰

배성훈

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows (ICCV 2021)

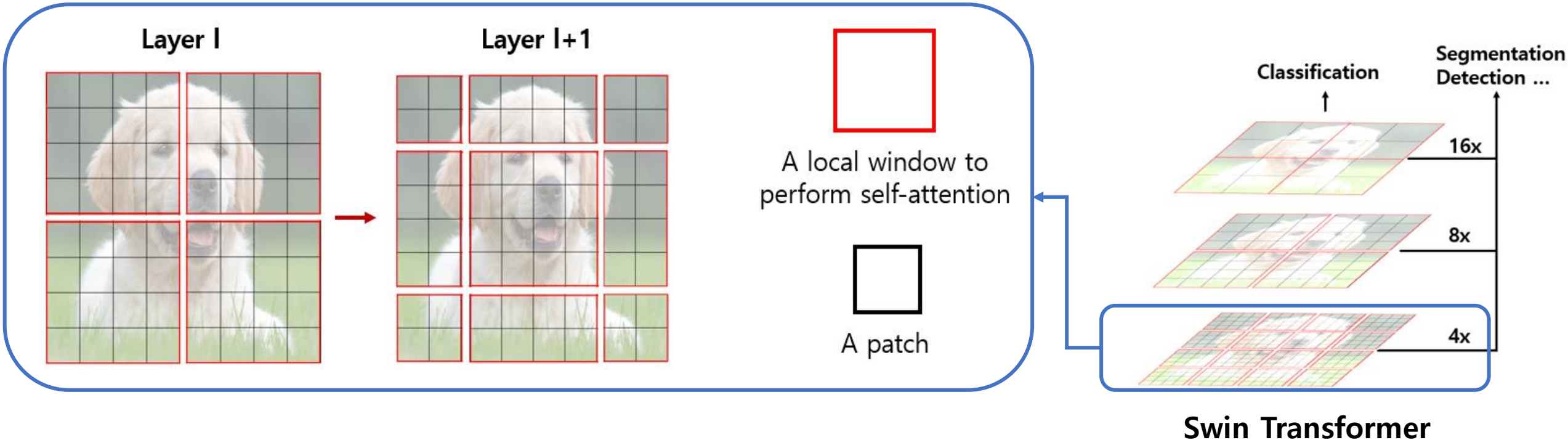
- **Research Background:**
  - ViT는 Image를 **Fixed sale patch**로 쪼개어 **이미지 해상도(크기)가 증가**하면 **self-attention이 어려움 (Quadratic)**
  - Quadratic하게 증가하는 연산량은 **학습에 오랜 시간, 많은 비용**이 들고 여러 CV의 scale에 민감한 task를 잘 처리하지 못함
  - 이를 해결하기 위해 Transformer 기반의 backbone에 CNN vision task에서 수행하는 역할 일부를 수용해 **Swin Transformer** 제안 (**Hierarchical feature map, Shifted window (Swin) block**)



Swin Transformer: Hierarchical Vision Transformer using Shifted Windows (ICCV 2021)

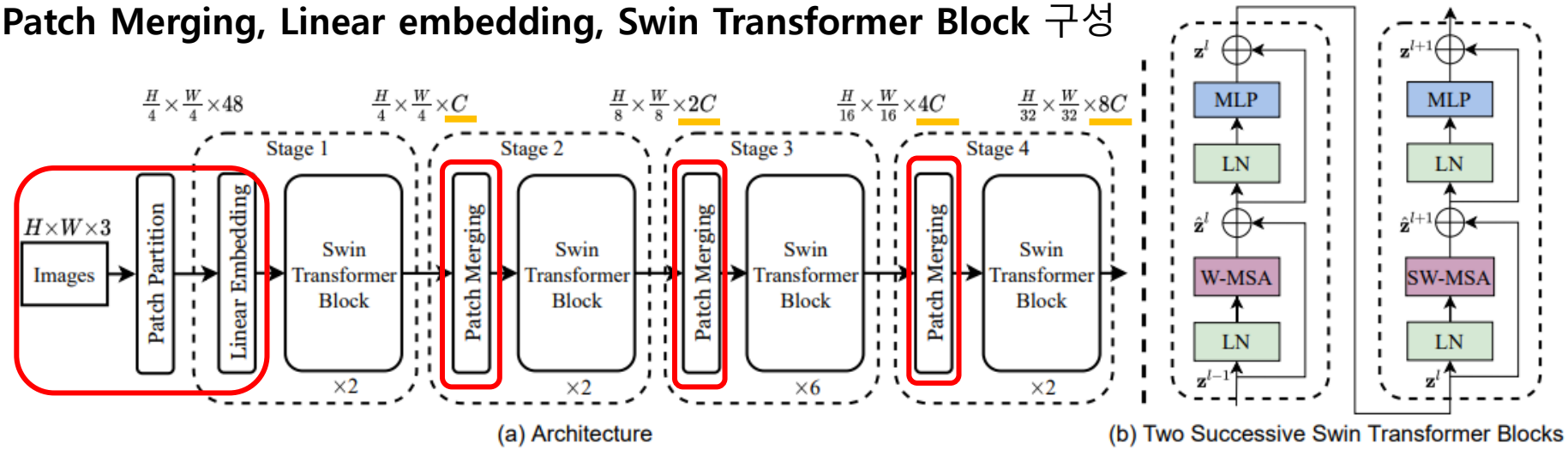
- **Research Background:**
  - Swin Transformer는 더 작은 단위의 patch로 시작해 점점 patch들을 merge하는 방식을 사용
  - Window내의 patch들끼리 만 self-attention 수행해 **연산량이 Linear하게 증가**
  - 각 계층마다 Representation을 가져 multi-scale entity를 다루는 비전 분야에서 좋은 성능을 보임 (Segmentation)

*Window: M개의 인접한 patch로 구성된 patch set

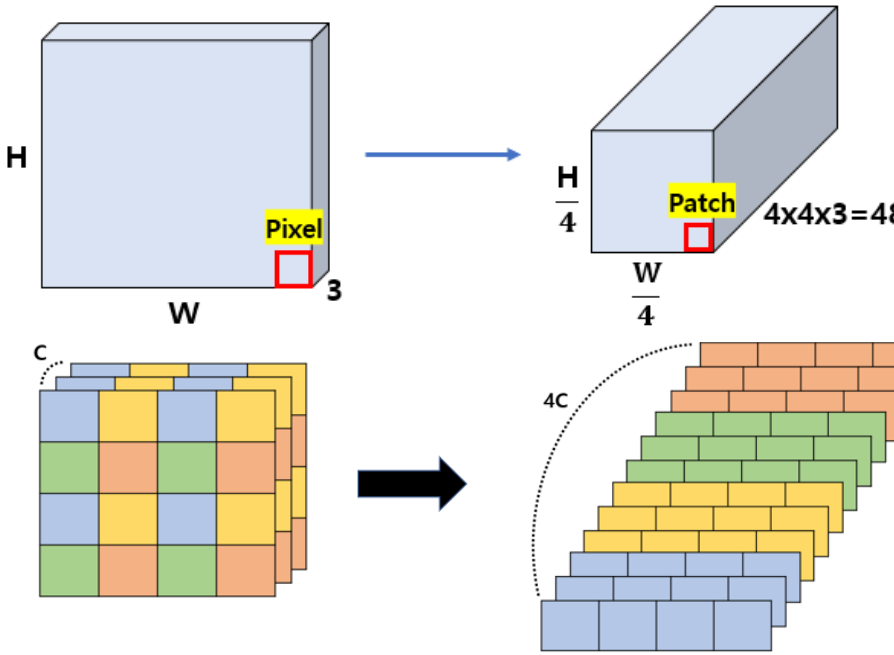


Swin Transformer: Hierarchical Vision Transformer using Shifted Windows (ICCV 2021)

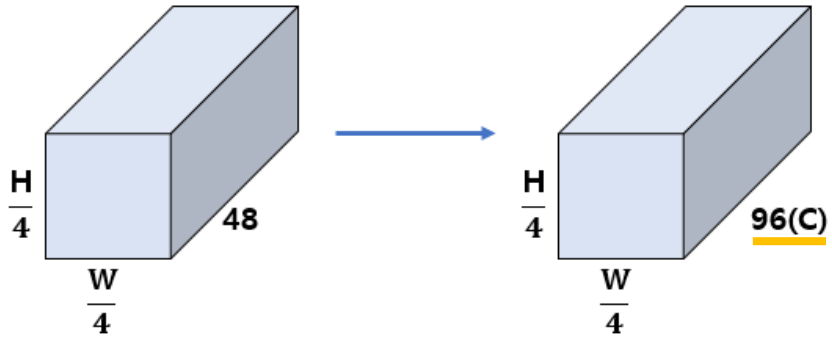
- **Method:**
  - **Patch partition, Patch Merging, Linear embedding, Swin Transformer Block 구성**



Stage 1: Patch partition/Merging



Stage 1: Linear Embedding



Patch Partition/Merging의 출력이 Linear layer에서 C의 dimension이 됨.
C=(96, 192, 384, 768)

<Tiny model 기준 stage마다>

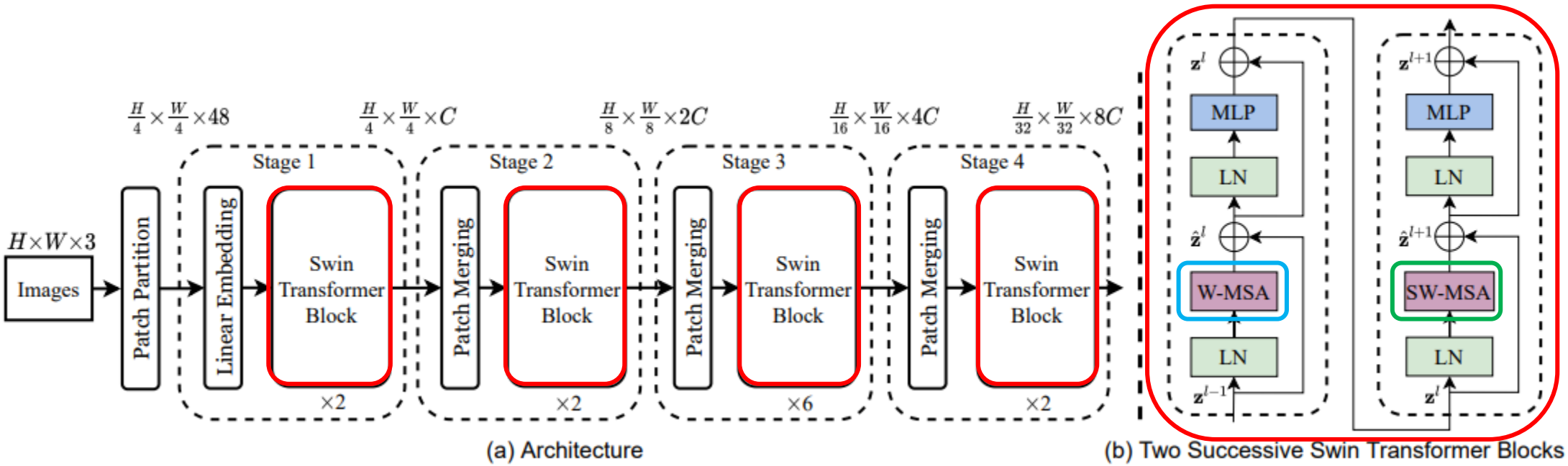
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows (ICCV 2021)

- **Method:**

  <Swin Transformer Block> **Window Multihead Self-attention (W-MSA)**

  - 현재 window에 있는 패치끼리 만 Self-Attention 수행
  - 이는 주변 픽셀들끼리 서로 연관성을 높여 computational complexity를 해결 (**Quadratic -> Linear**)
  - **M(Window size) < hw(image size)** 때문에 **W-MSA**의 연산량 < **MSA**의 연산량

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



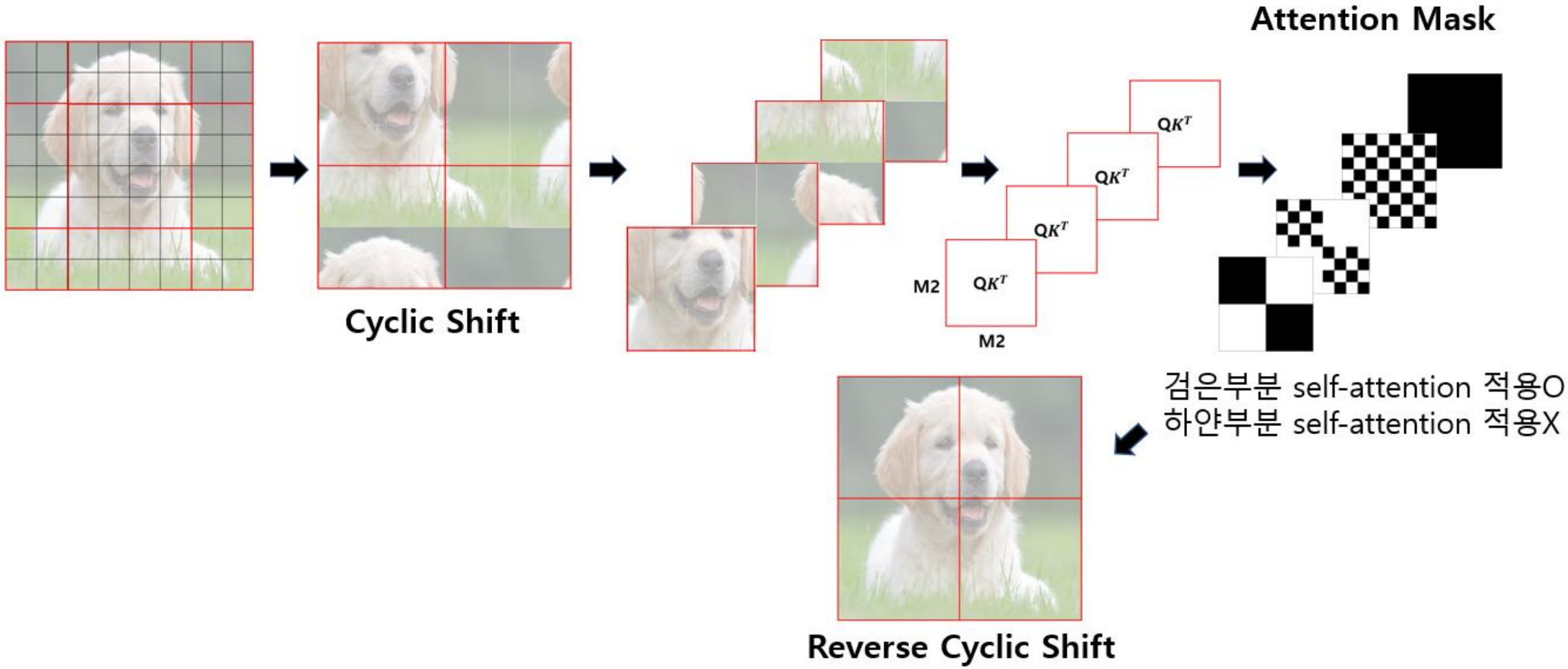
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows (ICCV 2021)

- **Method:**

  <Swin Transformer Block> **Shifted Window Multihead Self-attention (SW-MSA)**

  - 하지만 W-MSA는 Window를 고정해, 고정된 부분에서만 Self-attention을 수행해 멀리 떨어진 Patch들과 상호작용이 불가능
  - 이를 해결하기 위해, Window간의 연결을 추가하면서 적은 연산량을 유지하는 **Shifted Window Multihead Self-attention** 제안

Cyclic Shift와 Masked MSA 예시

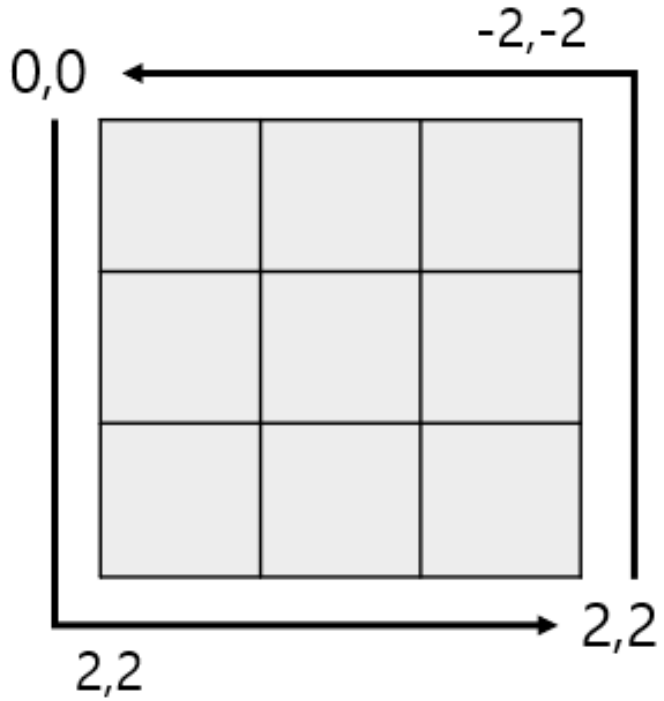


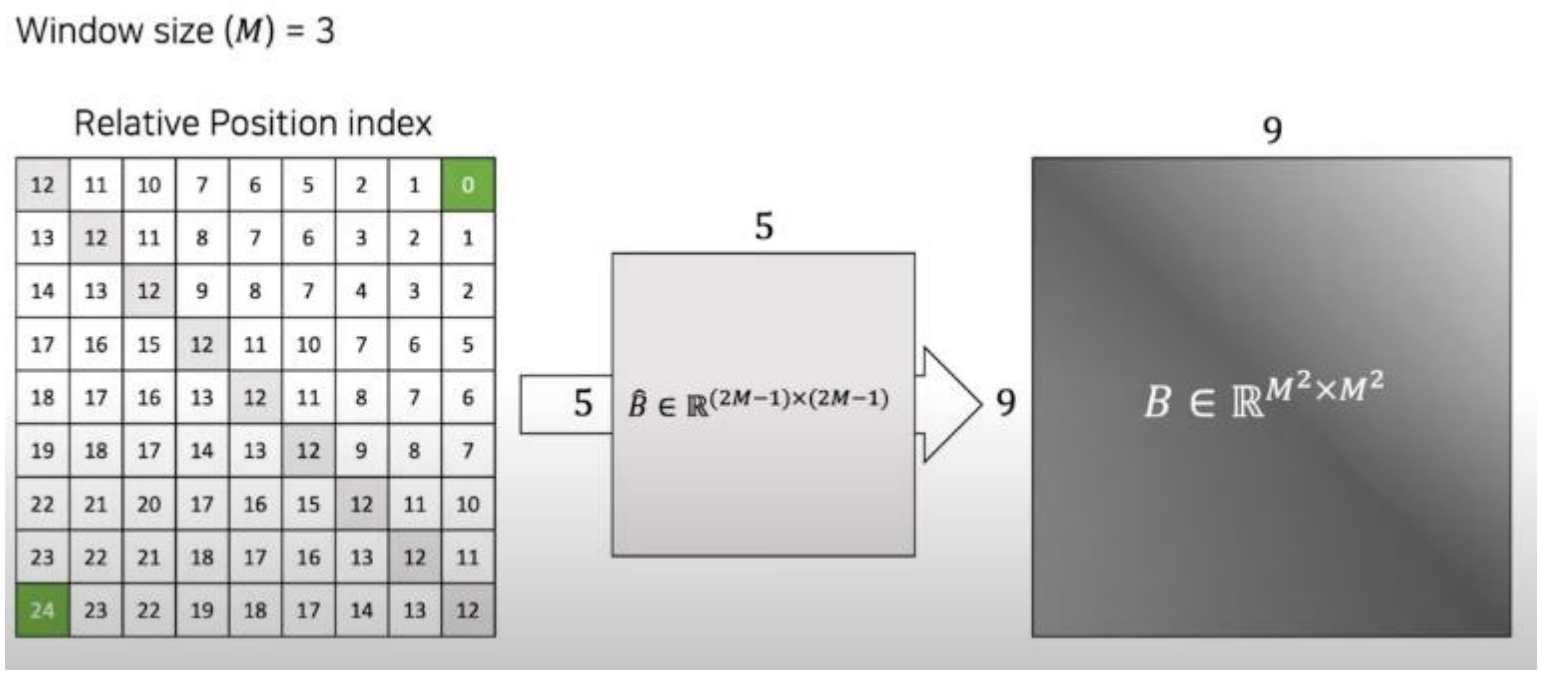
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows (ICCV 2021)

- **Method:**
  - ViT 와 다르게 Positional embedding 을 입력 부분에 추가하지 않고, **Relative Position Bias** 활용

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

Relative Position Bias: 기존 ViT에서 softmax를 취하기 전, B를 더함
Author: 기존의 Position Embedding에서의 절대좌표를 더하는 것보다 **상대좌표**를 더해주는 것이 더 좋은 방법





Swin Transformer: Hierarchical Vision Transformer using Shifted Windows (ICCV 2021)

- **Experiment:**
  - ImageNet Dataset에서 Swin Transformer는 ViT base model보다 parameter 수, FlOPS는 훨씬 적으면서 높은 성능을 보임
  - CNN 기반 모델 중 SOTA를 달성한 EfficientNet-B7보다 좋은 성능 보임
  - 다른 여러 Task(detection, segmentation 등)에서 SOTA를 달성

Comparison of different backbones on ImageNet-1K classification

(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

Results on COCO object detection and ADE20K semantic segmentation

(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

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

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows (ICCV 2021)

- **Experiment:**
  - 기존 shifted window, kernel 기반, padding 기반보다 제안한 cyclic shift가 가장 좋은 성능을 입증
  - 또한 relative position bias 를 단독으로 사용할 때 가장 좋은 성능을 달성해 효과를 입증함

Shifted windows approach and Different position embedding methods on three benchmarks
(using the Swin-T architecture)

	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**	Better Performance
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**	Better Performance

성능: Only W-MSA < SW-MSA + W-MSA

성능: 절대좌표(abs.pos), 절대+상대좌표(abs.+rel.pos).. < Only 상대좌표(rel.pos)

한줄평: Swin Transformer는 ViT 의 문제점이었던 많은 계산량과 이미지 resolution, scale의 variation이 고려되지 않았던 점(적은 inductive bias)을 개선했고 여러 CV task에서 좋은 성능을 달성했다. 기존의 ViT가 많은 양의 사전학습으로 다른 task에 활용이 어려울 것이라 생각했는데 이를 해결해주는 의미 있는 연구라고 생각한다.