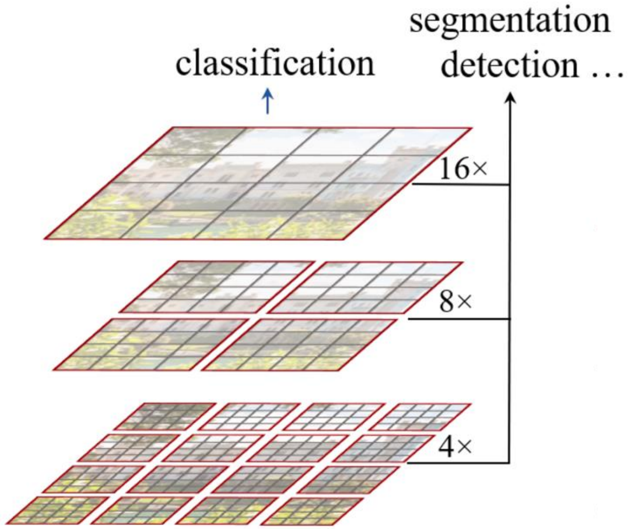
Variants of Vision Transformer

Swin Transformer

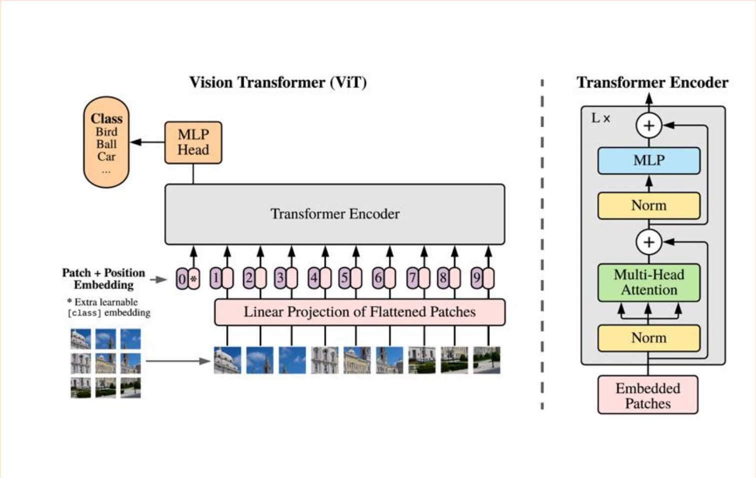


https://arxiv.org/pdf/2103.14030.pdf

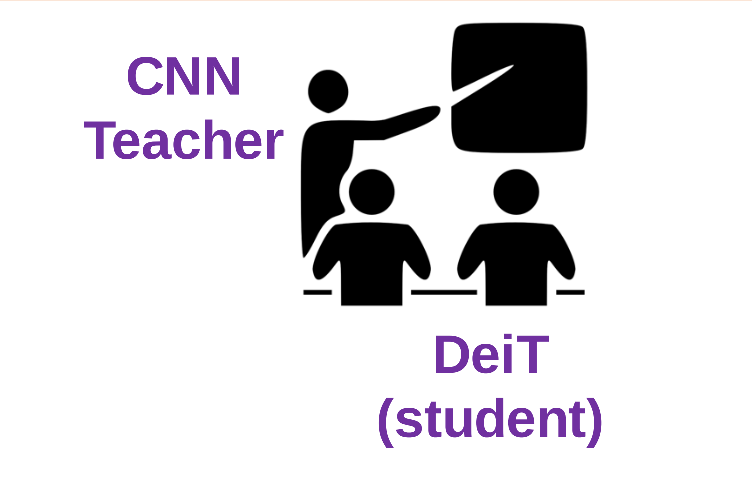
Vision Transformers Series

Vision Transformers

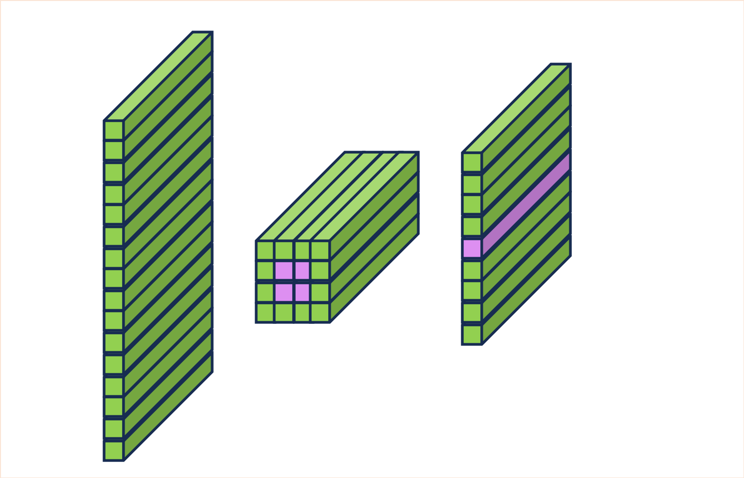
What we have covered so far:



https://arxiv.org/pdf/2010.11929.pdf



https://arxiv.org/pdf/2012.12877.pdf



https://arxiv.org/pdf/2101.11986.pdf

ViT

- Needs very large labeled data for (pre-) training

DeiT

Tokens-to-Token ViT

A ViT variant that can be trained on mid-size image dataset with superior performance than CNNs

Swin Transformer Overview

- Making transformer as a general-purpose backbone for computer vision

Recognized challenges of using transformers in computer vision:

- Visual tokens vary significantly in scale and resolution
- Computational cost of attention on high resolution images

Main idea:

- Hierarchical transformer architecture
- Shifted window based self-attention

Computational Complexity:
Global vs. Local Attention

Input image

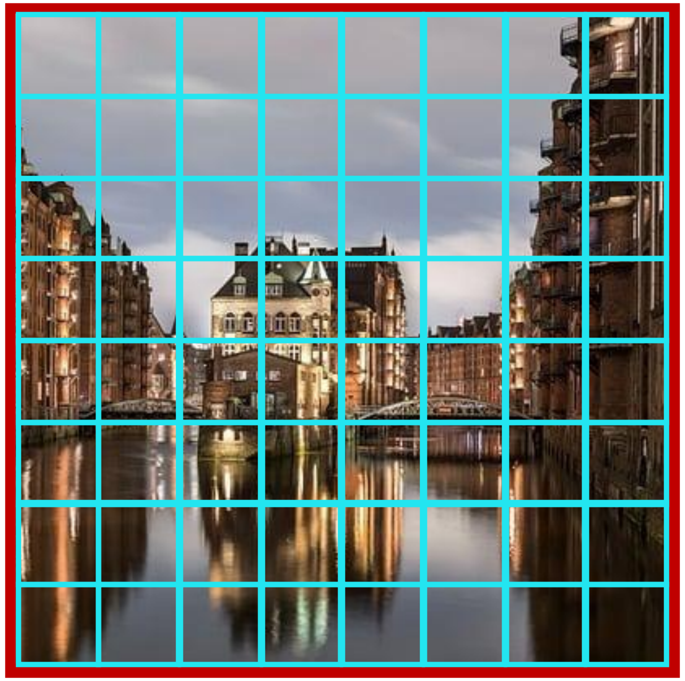
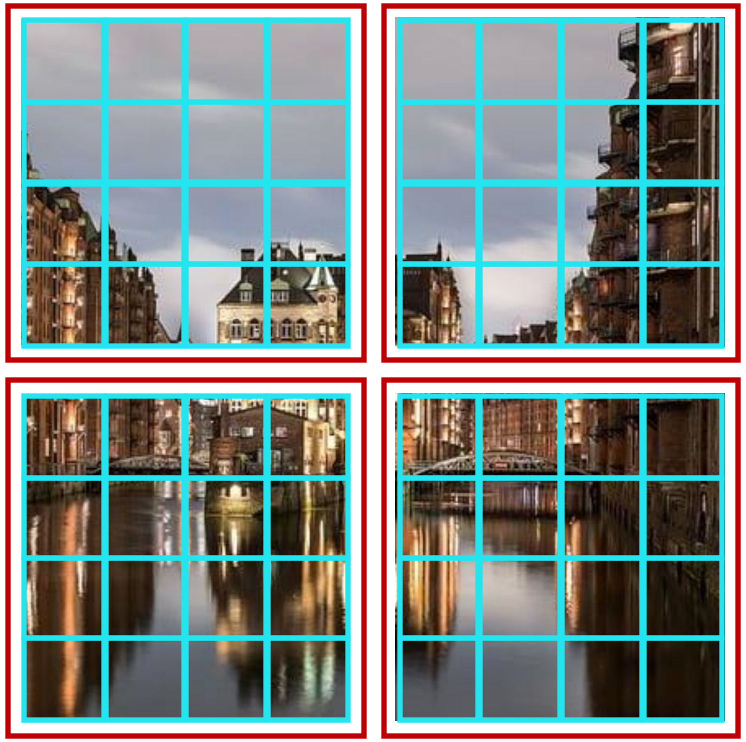
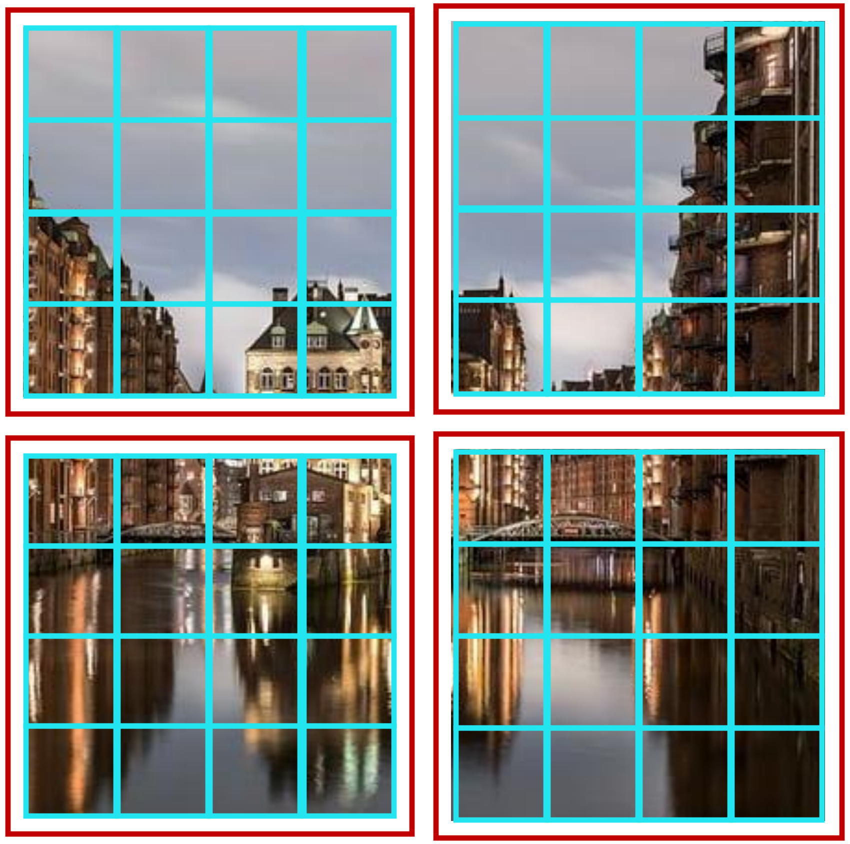


Image tokenized to N patches
→ Global attention has quadratic complexity $O(N^2)$



Windows of size 4×4
→ Local attention has linear complexity $O(N)$

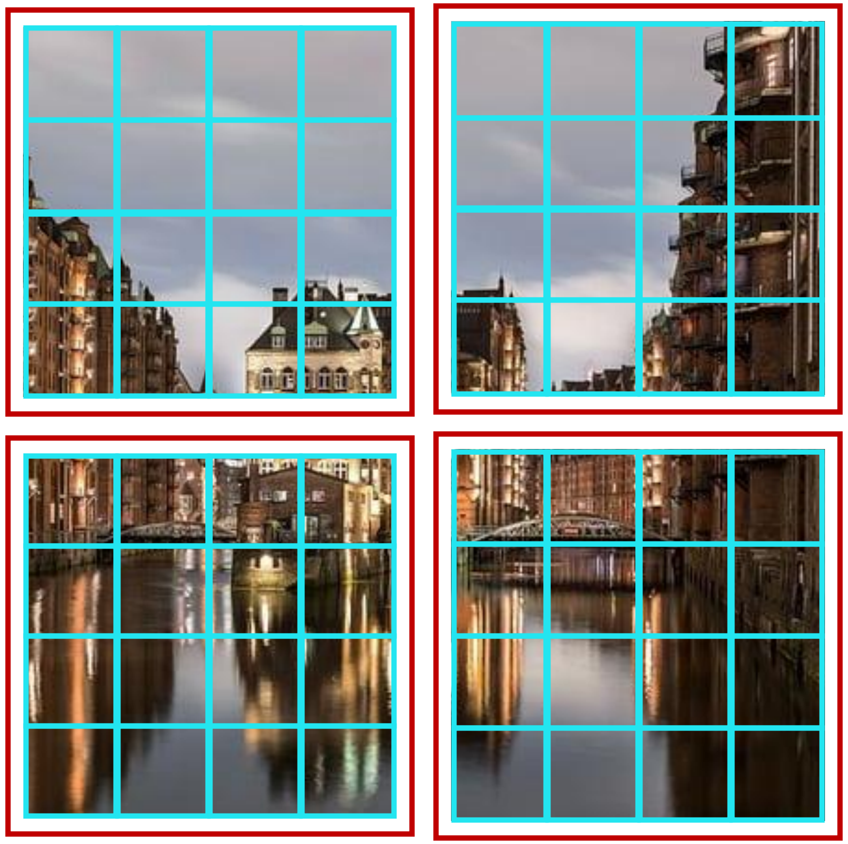
Window-based Attention



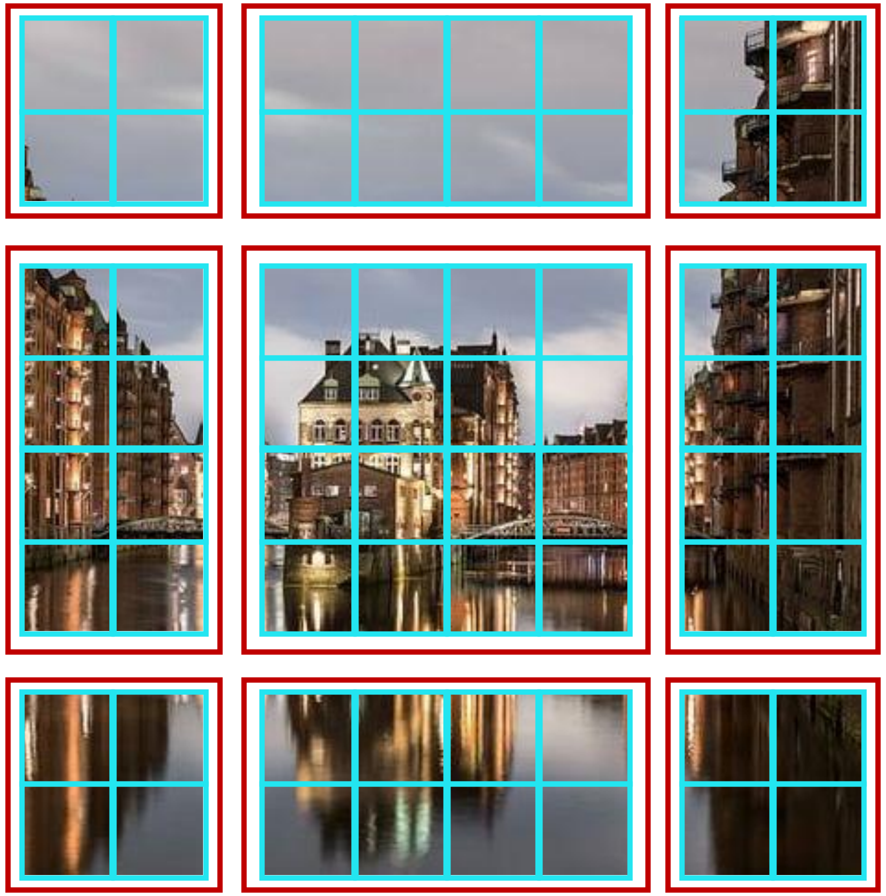
W-MSA: performing MSA in each local window

- Efficiency: Linear w.r.t. effective sequence length $O(N)$
- Lacking interactions across windows
  ➔ limited modeling power

Shifted Window-based Attention

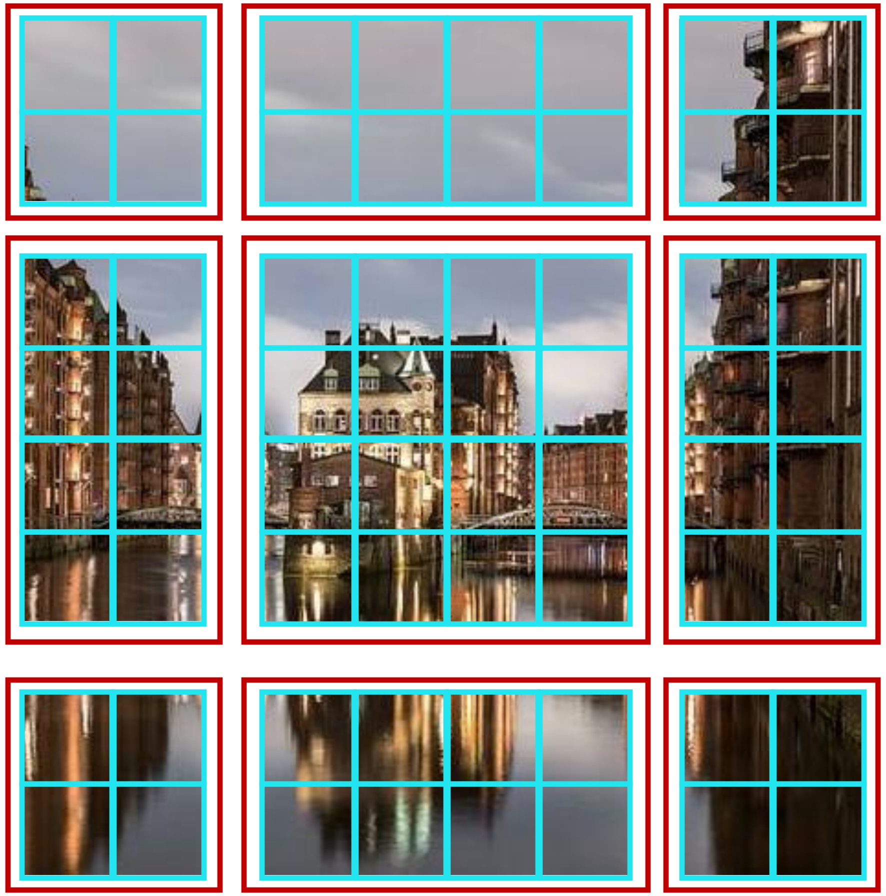


W-MSA: Attention in regular windows



SW-MSA: Attention in shifted windows

Shifted Window-based Attention



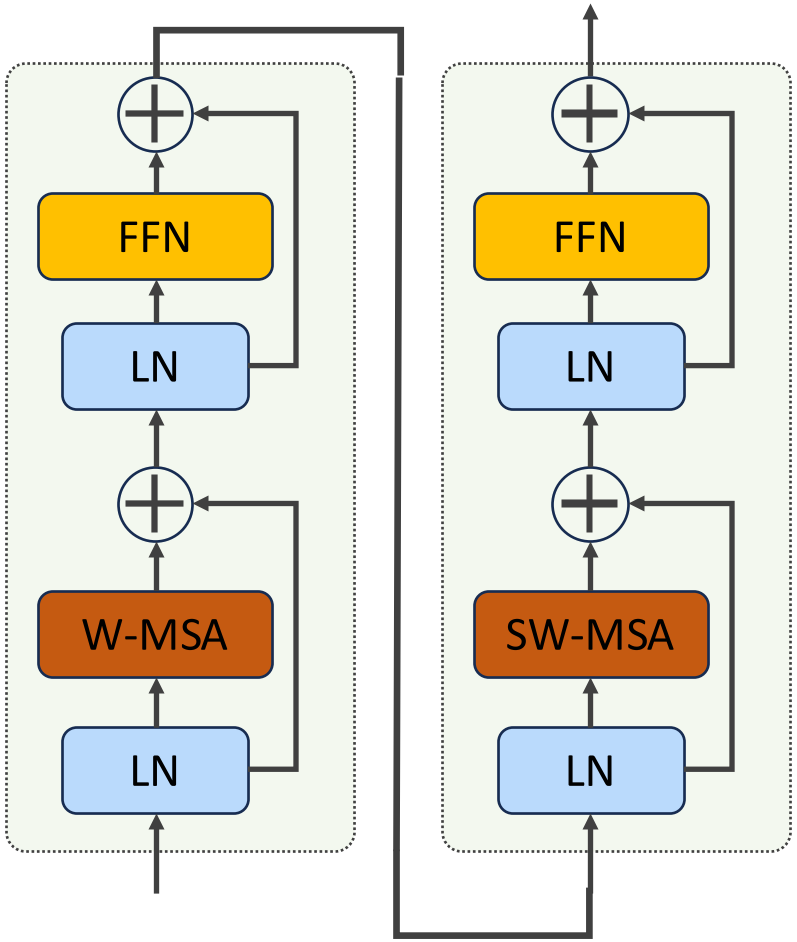
SW-MSA: Attention in shifted windows

- All queries within a window share the same key set
  - ➔ facilitate memory access
  - ➔ lower latency

- Another side-effect: Increasing number of windows (from 4 to 9)

Solution: Using a cyclic shift and masking for efficient batch computation

Swin Transformer Blocks

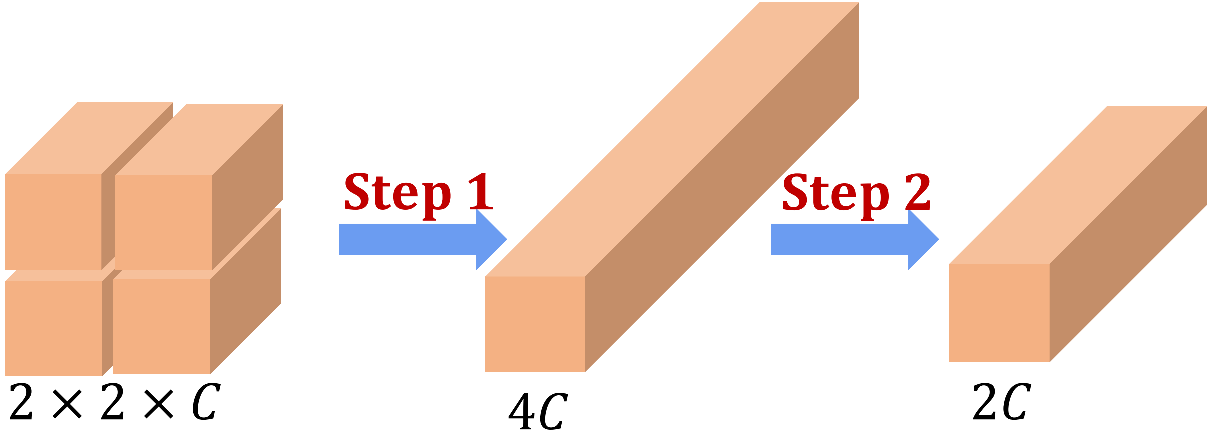


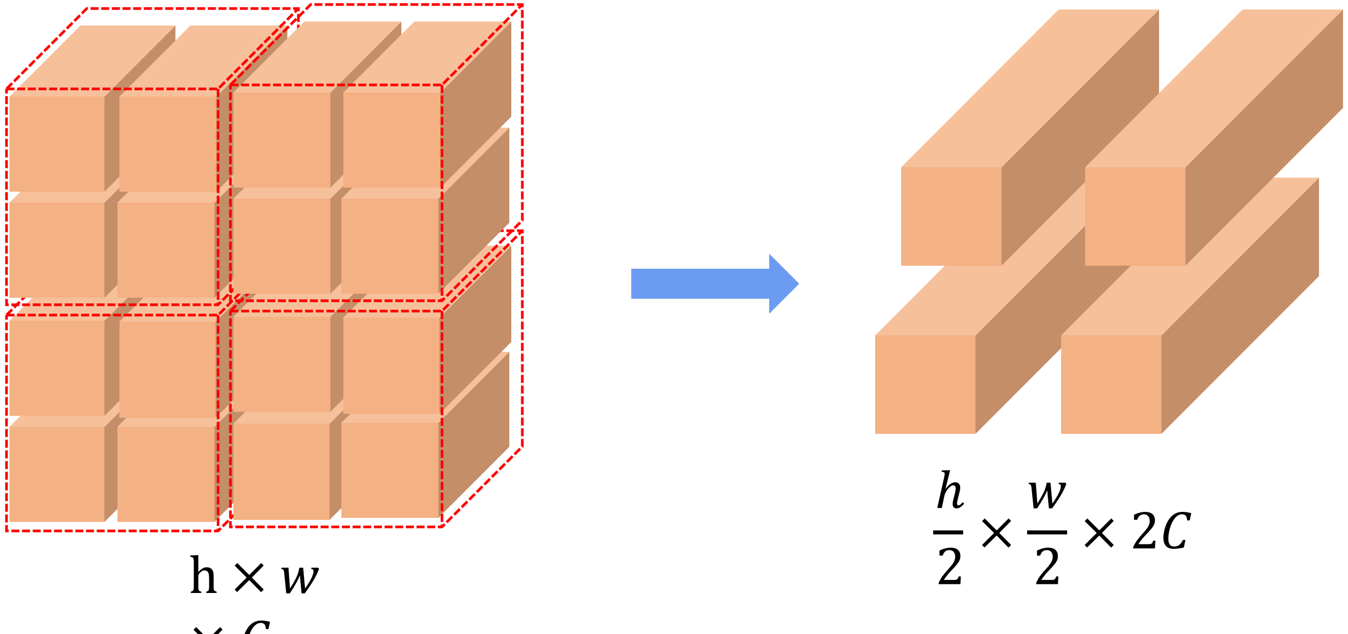
- Alternating regular window and shifted window attention in consecutive transformer layers

  - ➔ Efficient attention with linear complexity
  - ➔ Enables long-range interactions across windows
  - ➔ Increase modeling capability

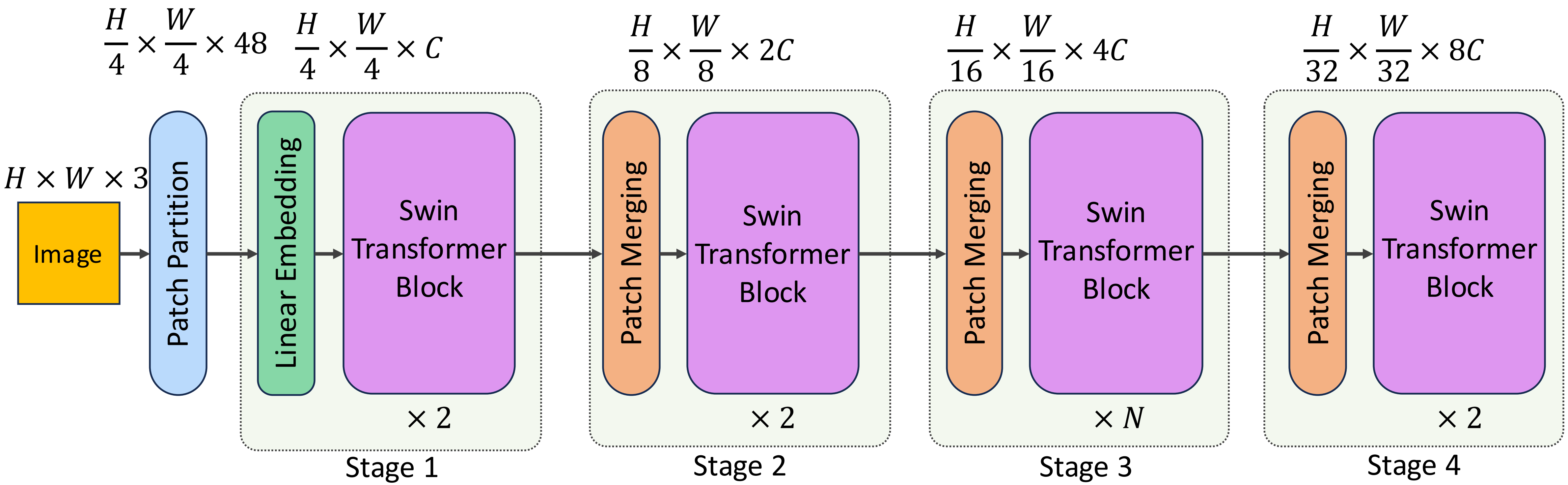
Patch merging layers for down-sampling

- Input: $h \times w \times C$ patches
- Concatenate the embedding layers in local 2×2 neighborhood
- Apply a linear layer to reduce dimensions





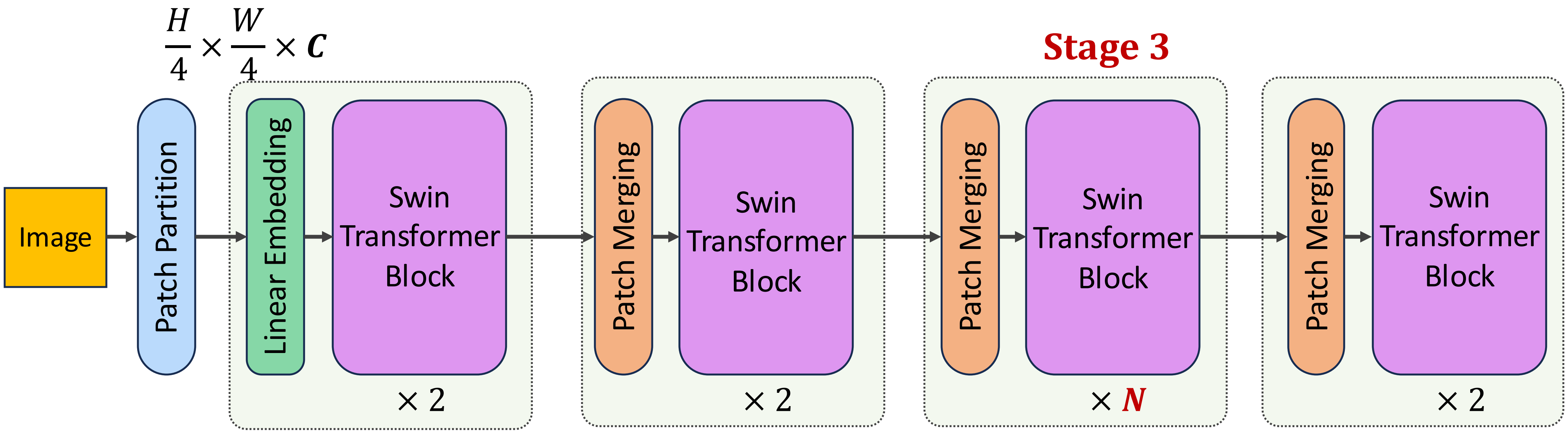
Swin Transformer Architecture



Patch size:
4×4

- Using relative position embedding
- Each Swin transformer block contains a pair of alternative regular window and shifted window sub-blocks
- Down-sampling starts from stage 2

Architecture variants



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

Experiments

Classification

Exp1: Training from scratch on ImageNet-1K

Exp2: Pre-train on ImageNet-22K, then finetune on ImageNet-1K

Object Detection

COCO 2017 dataset

118K training, 5K validation, and 20K test

Semantic Segmentation

ADE20K dataset

150 categories
20K training, 2K validation, and 3K test

Image 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

https://arxiv.org/pdf/2103.14030.pdf

Exp1: Training models from scratch on ImageNet-1k

- Similar performance compared to ConvNet models
- Slightly better than DeiT

Exp2: Pre-training on ImageNet-22k, and finetune on ImageNet-1k

- Swin transformer models outperform their counterparts

Objection Detection & Instance 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

https://arxiv.org/pdf/2103.14030.pdf

- COCO 2017 dataset
- Multiscale training:
  - Smallest dimension between 450 – 800
  - Largest dimension at most 1333

- Table (a): Comparing Swin-T and ResNEt-50 using different detection methods
- Table (b): fixed method (Mask R-CNN), comparing different backbones

Semantic Segmentation

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

https://arxiv.org/pdf/2103.14030.pdf

- ADE20K dataset
- Using UPerNet framework with Swin backbone

- Swin-S achieves +5 mIoU compared to DeiT-S (with similar computation cost)
- Pretrained Swin-L outperforms the previous best model (SETR)

Swin Transformer

- A general-purpose transformer backbone for computer vision
  - Hierarchical representations
  - Shifted window based self-attention

Thanks for watching