ViT & Its Variants

10.30

Content

1.Development

2017.6 | Transformer
Solely based on attention mechanism, the Transformer is proposed and shows great performance on NLP tasks.

2018.10 | BERT
Pre-training transformer models begin to be dominated in the field of NLP.

2020.5 | GPT-3
A huge transformer with 170B parameters, takes a big step towards general NLP model.

2020.5 | DETR
A simple yet effective framework for high-level vision by viewing object detection as a direct set prediction problem.

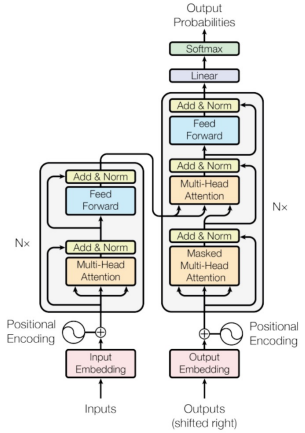
2020.7 | iGPT
The transformer model for NLP can also be used for image pre-training.

2020.10 | ViT
Pure transformer architectures work well for visual recognition.

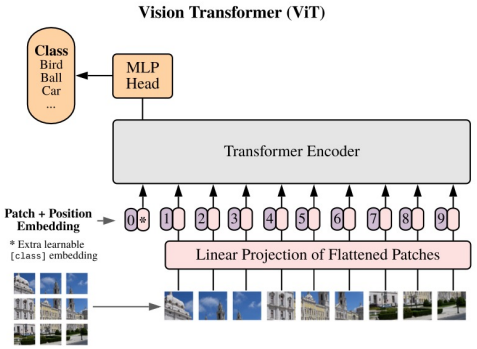
2020.12 | IPT
The first transformer model for low-level vision by combining multi-tasks.

2021 | ViT Variants
Variants of ViT models, e.g., DeiT, PVT, TNT, and Swin.

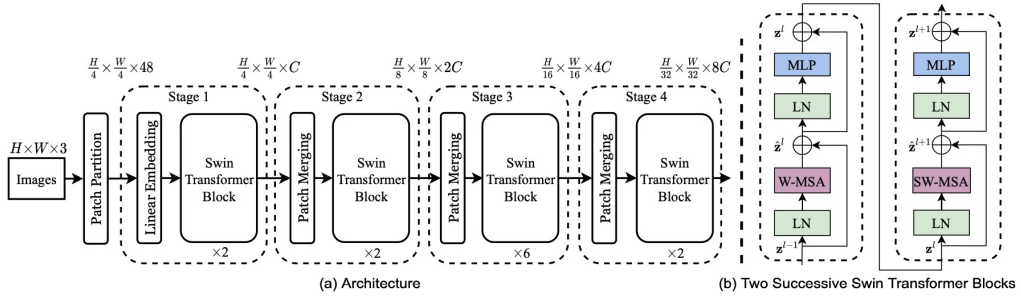
■ Transformer



■ ViT

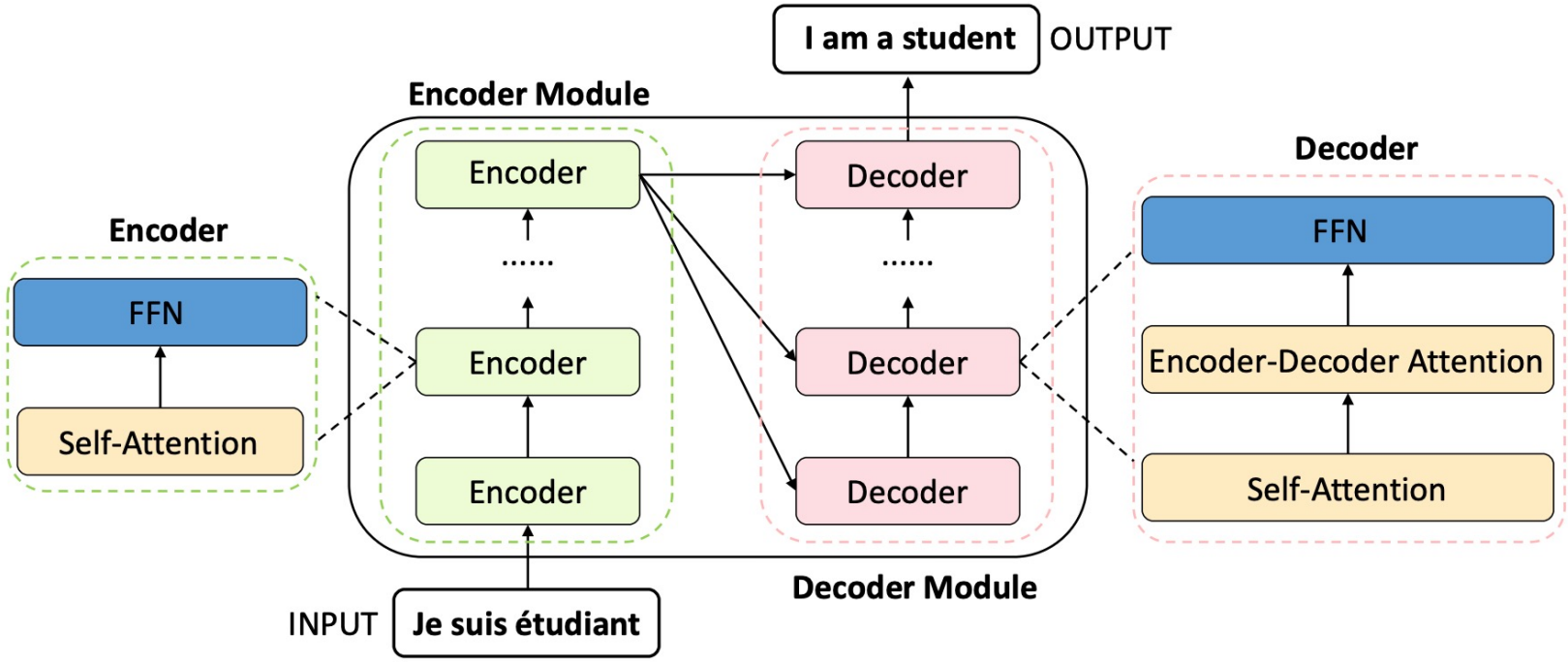


■ Swin Transformer



2. Architecture Of Transformer

- Transformer was first used in the field of NLP on machine translation tasks.
- Transformer consists if an encoder module and a decoder module.
- Each encoder is composed of a Multi-head self-attention layer and a feed-forward neural network. Layernorm is applied before every block, and residual connections after every block.



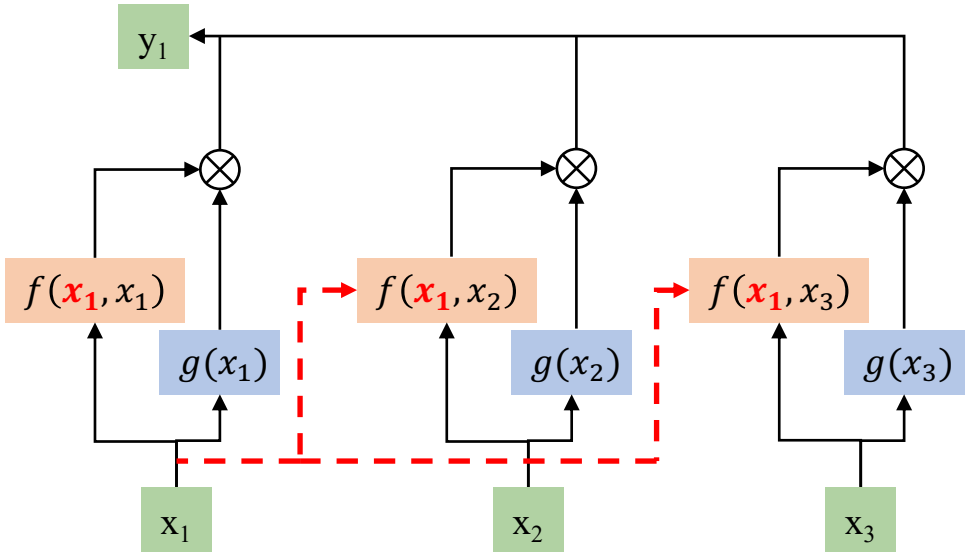
2.1 Formulation Of Self-attention

- Self-attention module computes the responses at each position in a sequence by estimating attention scores to all positions and gathering the corresponding embeddings based on the scores accordingly.

Given an input signal $\mathbf{X} \in \mathbf{R}^{\mathbf{n}\times\mathbf{d}}$, where $\mathbf{n}$ is the number of sequences and $\mathbf{d}$ is the number of channels.

$$y_i = \sum_{\forall j} f(x_i, x_j) g(x_j)$$

- Where $\mathbf{x_i} \in \mathbf{R}^{\mathbf{1}\times\mathbf{d}}$, $\mathbf{y_i} \in \mathbf{R}^{\mathbf{1}\times\mathbf{d}}$ indicate the $\mathbf{i}^{\mathbf{th}}$ position of the input signal $\mathbf{X}$ and output signal $\mathbf{Y}$, respectively.
- Subscript $\mathbf{j}$ is the index that enumerates **all positions**.
- $\mathbf{f}(\cdot)$ computes a representing relationship between i and all j.
- $\mathbf{g}(\cdot)$ computes a representing of the input signal at position j.



2.1 Formulation Of Self-attention

There are many choices for the pairwise function $\mathbf{f}(\cdot)$. For example, $\mathbf{f}(\cdot)$ can be formulated as:

$$f(x_i, x_j) = \frac{e^{\theta(x_i)\emptyset(x_j)^T}}{\sum_j e^{\theta(x_i)\emptyset(x_j)^T}} = \alpha'_{i,j}$$

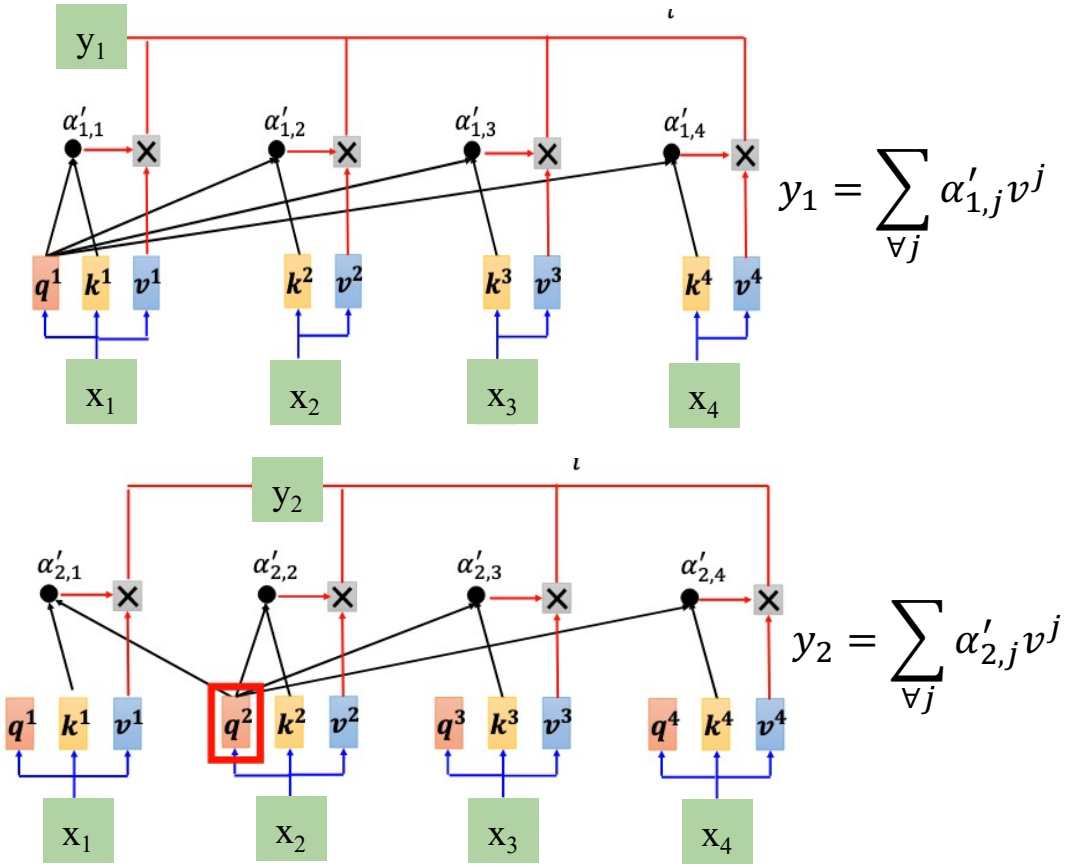
- Where $\boldsymbol{\theta}(\cdot)$ and $\emptyset(\cdot)$ can be any embedding layers. If we consider the $\boldsymbol{\theta}(\mathbf{X}) = \mathbf{XW_q}$, $\emptyset(\mathbf{X}) = \mathbf{XW_k}$, $\mathbf{g}(\mathbf{X}) = \mathbf{XW_v}$. $\mathbf{W_\theta} \in \mathbf{R^{d \times d_{model}}}$, $\mathbf{W_\emptyset} \in \mathbf{R^{d \times d_{model}}}$, $\mathbf{W_g} \in \mathbf{R^{d \times d_{model}}}$.

$$y_i = \sum_{\forall j} \frac{e^{x_i W_q W_k^T x_j^T}}{\sum_j e^{x_i W_q W_k^T x_j^T}} x_j W_v = \sum_{\forall j} \frac{e^{q_i k_j}}{\sum_j e^{q_i k_j}} v_j$$

$$Y = softmax\left(XW_q W_k^T X^T\right) XW_v$$

- $\mathbf{Q} = \boldsymbol{\theta}(\mathbf{X}) = \mathbf{XW_q}$, $\mathbf{K} = \emptyset(\mathbf{X}) = \mathbf{XW_k}$, $\mathbf{V} = \mathbf{g}(\mathbf{X}) = \mathbf{XW_v}$

$$Y = softmax(QK^T)V$$



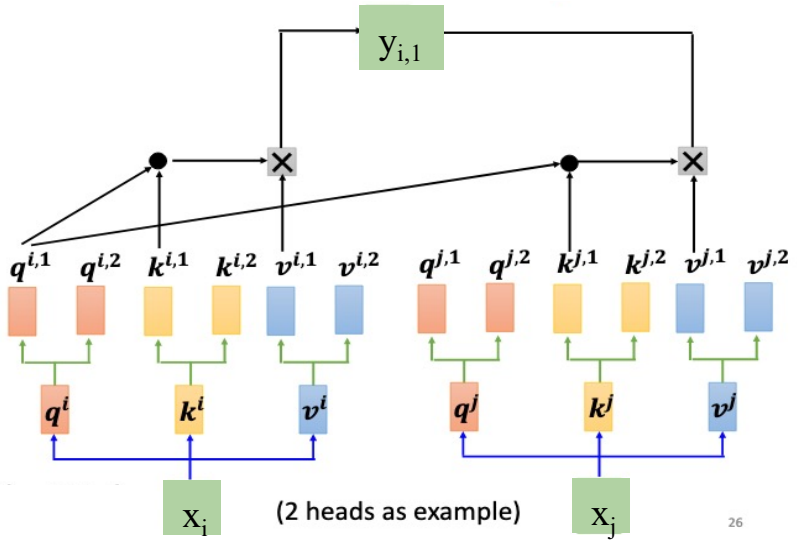
$$y_1 = \sum_{\forall j} \alpha'_{1,j} v^j$$

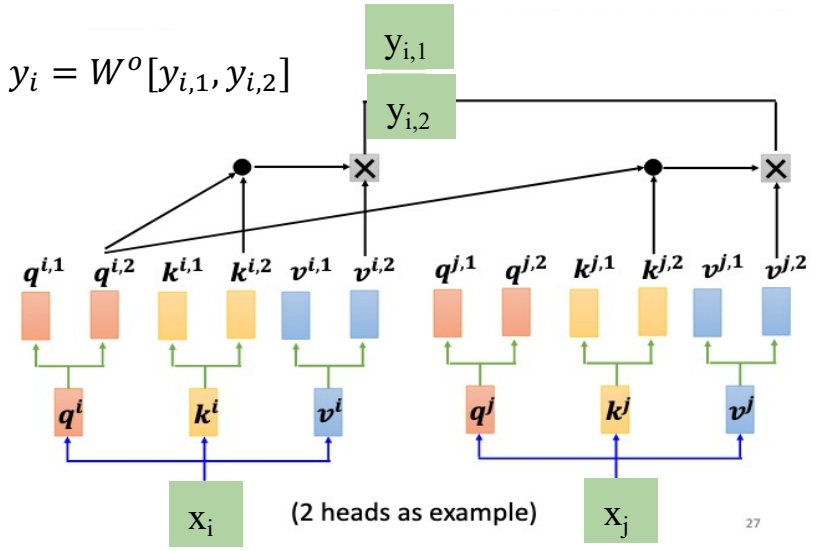
$$y_2 = \sum_{\forall j} \alpha'_{2,j} v^j$$

2.2 Formulation Of Multi-head Attention

- Given an input vector $\mathbf{x_i}$ and the number of heads $\mathbf{h}$.
- The input vector $\mathbf{x_i}$ is first transformed into three different groups of vector: query group $\mathbf{q}^i$, key group $\mathbf{k}^i$, value group $\mathbf{v}^i$.
- In each group, there are $\mathbf{h}$ vectors with dimension $\mathbf{d_{q'} = d_{k'} = d_{v'} = d_{model}/h}$.
- The vectors derived from different inputs and packed together into three groups of matrices: $\{Q_i\}_{i=1}^h$, $\{K_i\}_{i=1}^h$ and $\{V_i\}_{i=1}^h$.

$$MultiHead(Q', K', V') = Concat(head_1, \dots, head_n)W^o, where\ head_i = Attention(Q_i, K_i, V_i)$$





2.3 Other Key Concepts

I. Positional Encoding

- Representing the position of each input sequence as a vector.
- Position vector X_{pe} has the same dimension as input sequence X.
- The input of transformers will be the sum of X_{pe} and X.

$$X' = X + X_{pe}, \qquad X \in R^{B\times n\times d}, X_{pe} \in R^{B\times n\times d}$$

- Where B: batch size, n: length of sequence, d: dimension of each vector.

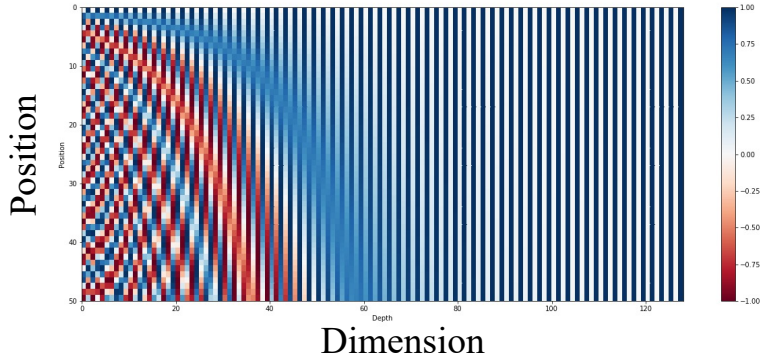
① Sinusoidal Position Encoding

pos: the position in an input sequence, **i**: the dimension of a vector

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

■ $X_{pe}: n = 50, d = 128$



② Learned Positional Embedding

Random initialization X_{pe} and X_{pe} can be learned.

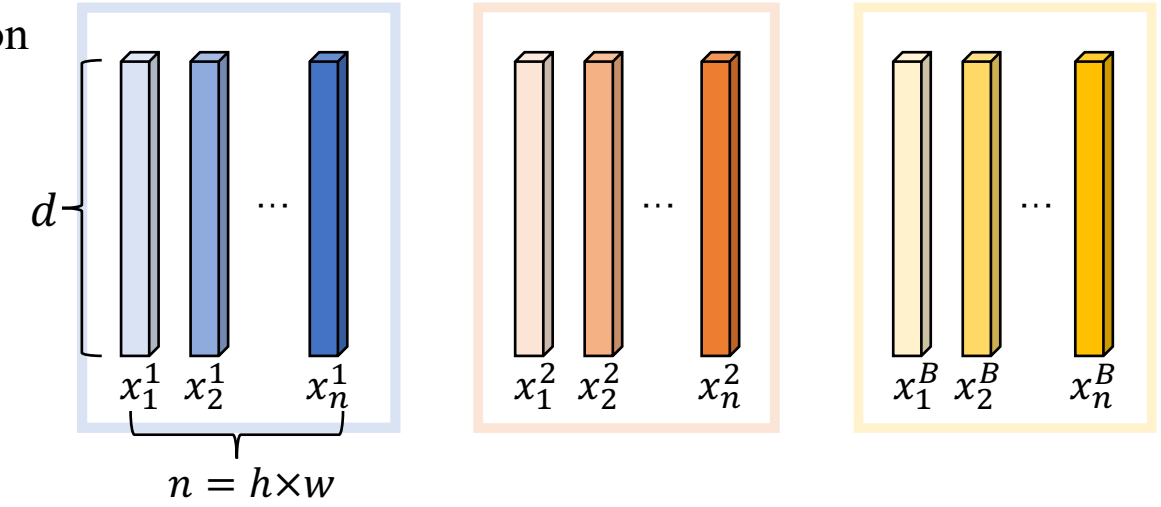
2.3 Other Key Concepts

II. Layer Norm

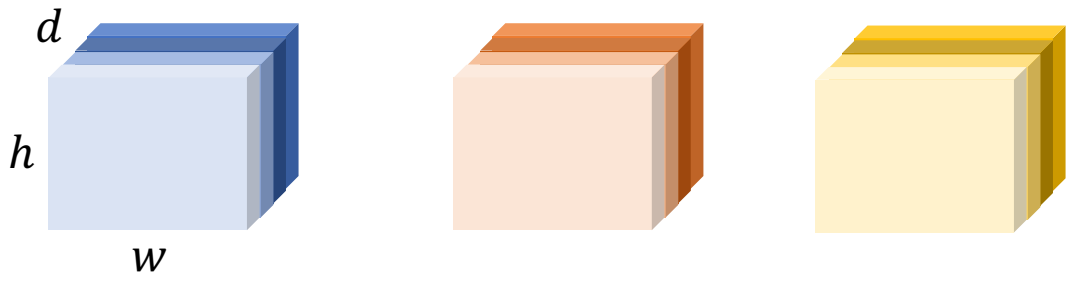
- Let **B**: batch size, **n**: number of sequence, **d**: dimension of vector.

$$\mu^b = \frac{1}{H}\sum_{i=1}^{n}\sum_{j=1}^{d} x_{i,j}^b, \qquad H = n \times d$$

$$\sigma^b = \sqrt{\frac{1}{H}\sum_{i=1}^{n}\sum_{j=1}^{d}\left(x_{i,j}^b - \mu^b\right)^2} \qquad \hat{x}_{i,j}^b = \frac{x_{i,j}^b - \mu^b}{\sqrt{(\sigma^b)^2 + \epsilon}}$$



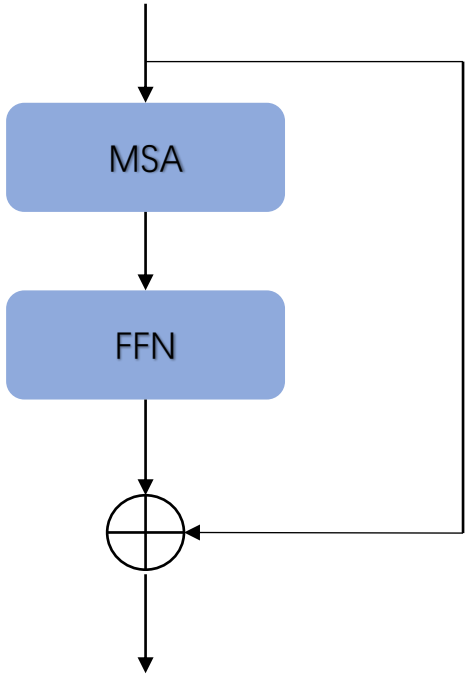
- $\mathbf{x_i} \in \mathbf{R^{1\times d}}$ indicates the i^{th} position of the input sequence **X**.
- **b** indicates the b^{th} sample in a batch.
- **j** indicates the j^{th} element in a vector.



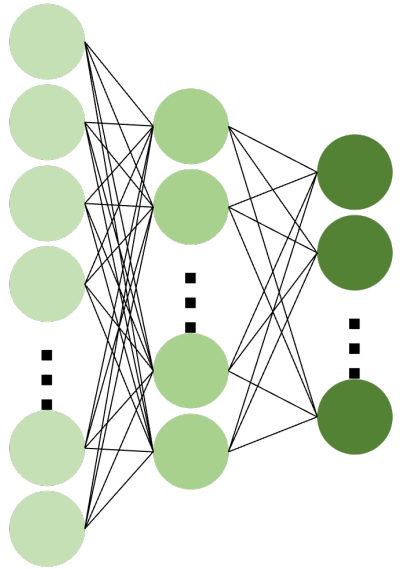
2.3 Other Key Concepts

III. Residual Connection

$$X' = X + f(X)$$



IV. FFN(Feed Forward Network) MLP(Multi layer Perception)

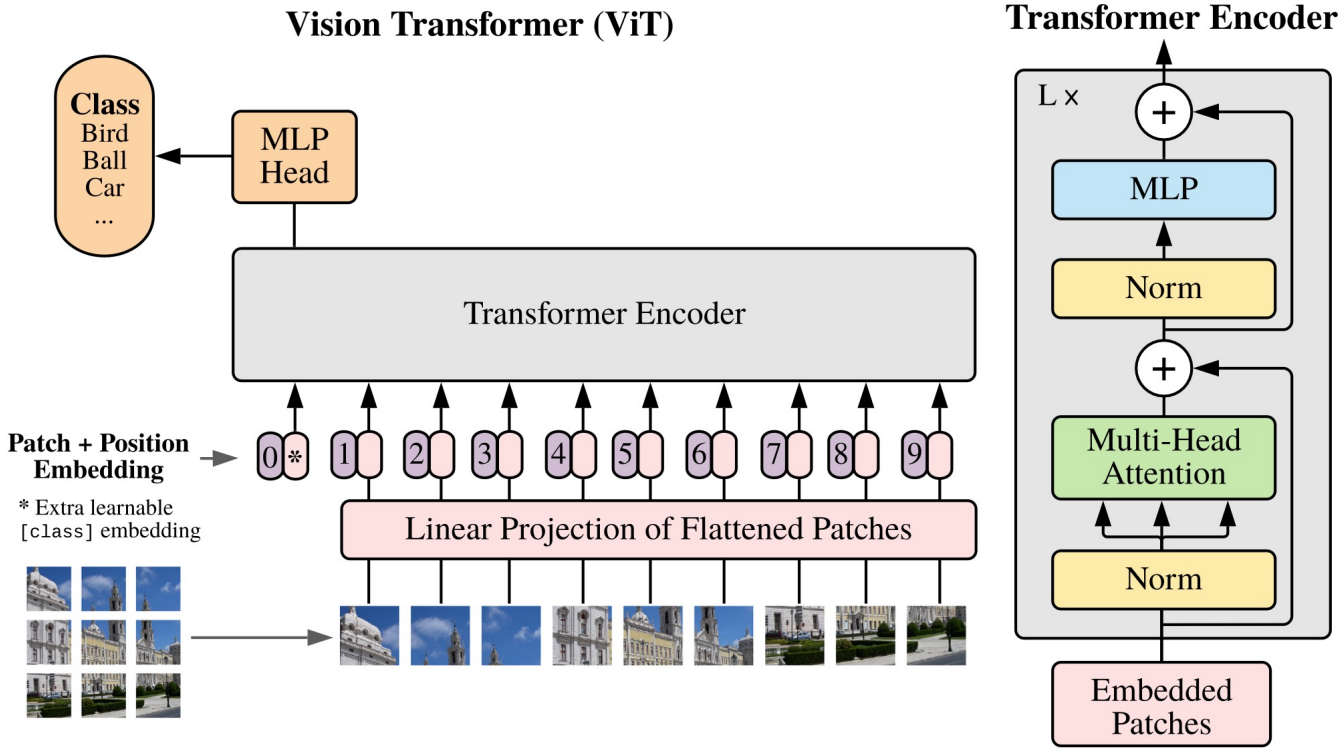


AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

CVPR 2020

3.1 Architecture Of ViT

- ViT is a pure transformer that performs well on image classification task when applied directly to the sequences of image patches.
- ViT follows the original Transformer, but only contains encoder layers.
- Each encoder is composed of a Multi-head self-attention layer and a feed-forward neural network. Layernorm is applied before every block, and residual connections after every block.



- Linear Projection of Flattened Patches

  Patch size: $(n, 16,16,3)$
  $\rightarrow (n, 16\times16\times3)$
  $\rightarrow (n, d_{model})$
  $\rightarrow (1+n, d_{model})$

- Patch + Position Embedding

  $X: (1+n, d_{model})$, $X_{pe}: (1+n, d_{model})$
  Embedded Patches: $X' = X + X_{pe}$

- Encoder Output

  $Y: (1+n, d_{model})$
  take out class token from Y
  Class token: $(1, d_{model})$

3.1 ViT Experiments

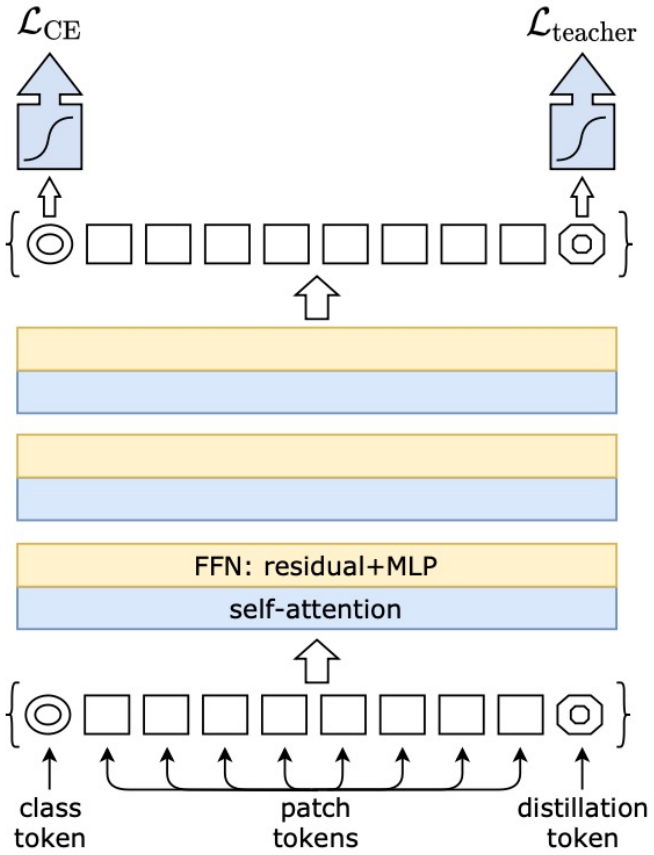
	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	–
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	–
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	–
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	–
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

Training **d**ata-**e**fficient **i**mage **t**ransformers & distillation through attention

PMLR 2021

3.2 Architecture Of DeiT

- High-performing vision transformers are pre-trained with hundreds of millions of images using a large infrastructure, thereby limiting their adoption.
- DeiT introduces a new distillation procedure based on a distillation token, and the model pre-trained on Imagenet are competitive when transferred to downstream tasks.



Let $\boldsymbol{Z_t}$ be the logits of the teacher model, $\boldsymbol{Z_s}$ the logits of the student model. We denote by $\boldsymbol{\tau}$ the temperature for the distillation, $\boldsymbol{\lambda}$ the coefficient balancing the Kullback–Leibler divergence loss (**KL**) and the cross-entropy ($\boldsymbol{L_{CE}}$) on ground truth labels **y**, and $\boldsymbol{\psi}$ the softmax function.

- Soft distillation

$$\mathcal{L}_{\text{global}} = (1-\lambda)\mathcal{L}_{\text{CE}}(\psi(Z_s), y) + \lambda\tau^2 \text{KL}(\psi(Z_s/\tau), \psi(Z_t/\tau)).$$

- Hard distillation —— $y_t = argmax_c Z_t(c)$

$$\mathcal{L}_{\text{global}}^{\text{hardDistill}} = \frac{1}{2}\mathcal{L}_{\text{CE}}(\psi(Z_s), y) + \frac{1}{2}\mathcal{L}_{\text{CE}}(\psi(Z_s), y_t).$$

- Hard distillation + Label smoothing

$$\hat{y}_t(i) = \begin{cases} 1-\epsilon & i = target \\ \epsilon/(K-1) & i \neq target \end{cases}, \epsilon = 0.1$$

0	0.025
1	0.9
0	0.025
0	0.025
0	0.025

3.2 Training/Test procedure Of DeiT

- **Training**

Teacher Model is RegNetY-16GF and let its output : y. In training process, the model has two output: Output of class token: y_{ct} and output of distillation token: y_{dt}.

- Training Loss: $L_{total} = \alpha L_{base} + (1-\alpha)L_{dist}$
- If distillation procedure uses Soft-distillation : $L_{total} = \alpha L_{CE}(y_{ct}, y) + (1-\alpha)L_{KL}(y_{dt}, y_{teacher})$
- If distillation procedure uses Hard-distillation: $L_{total} = \alpha L_{CE}(y_{ct}, y) + (1-\alpha)L_{CE}(y_{dt}, argmax(y_{teacher}))$
- If using Label-smoothing Trick: $L_{base}(\cdot) = L_{CE}(y_{ct}, label\_smoothing(y))$

$$\text{And : } L_{total} = \alpha L_{CE}\big(y_{ct}, label\_smoothing(y)\big) + (1-\alpha)L_{dist}$$

- **Testing**

Model Output: $y = \frac{1}{2}y_{ct} + \frac{1}{2}y_{dt}$

3.2 DeiT Experiments

DeiT: method ↓	supervision label	supervision teacher	ImageNet top-1 (%) Ti 224	S 224	B 224	B↑384
no distillation	✓	✗	72.2	79.8	81.8	83.1
usual distillation	✗	soft	72.2	79.8	81.8	83.2
hard distillation	✗	hard	74.3	80.9	83.0	84.0
class embedding	✓	hard	73.9	80.9	83.0	84.2
distil. embedding	✓	hard	74.6	81.1	83.1	84.4
DeiT⚗: class+distil.	✓	hard	74.5	81.2	83.4	84.5

Model	ImageNet	CIFAR-10	CIFAR-100	Flowers	Cars	iNat-18	iNat-19	im/sec
EfficientNet-B7	84.3	98.9	91.7	98.8	94.7	_	_	55.1
ViT-B/32	73.4	97.8	86.3	85.4	_	_	_	394.5
ViT-B/16	77.9	98.1	87.1	89.5	_	_	_	85.9
ViT-L/32	71.2	97.9	87.1	86.4	_	_	_	124.1
ViT-L/16	76.5	97.9	86.4	89.7	_	_	_	27.3
DeiT-B	81.8	99.1	90.8	98.4	92.1	73.2	77.7	292.3
DeiT-B↑384	83.1	99.1	90.8	98.5	93.3	79.5	81.4	85.9
DeiT-B⚗	83.4	99.1	91.3	98.8	92.9	73.7	78.4	290.9
DeiT-B⚗↑384	84.4	99.2	91.4	98.9	93.9	80.1	83.0	85.9

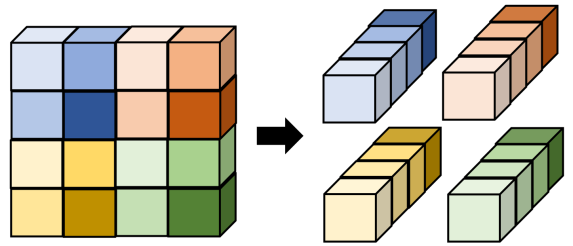
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

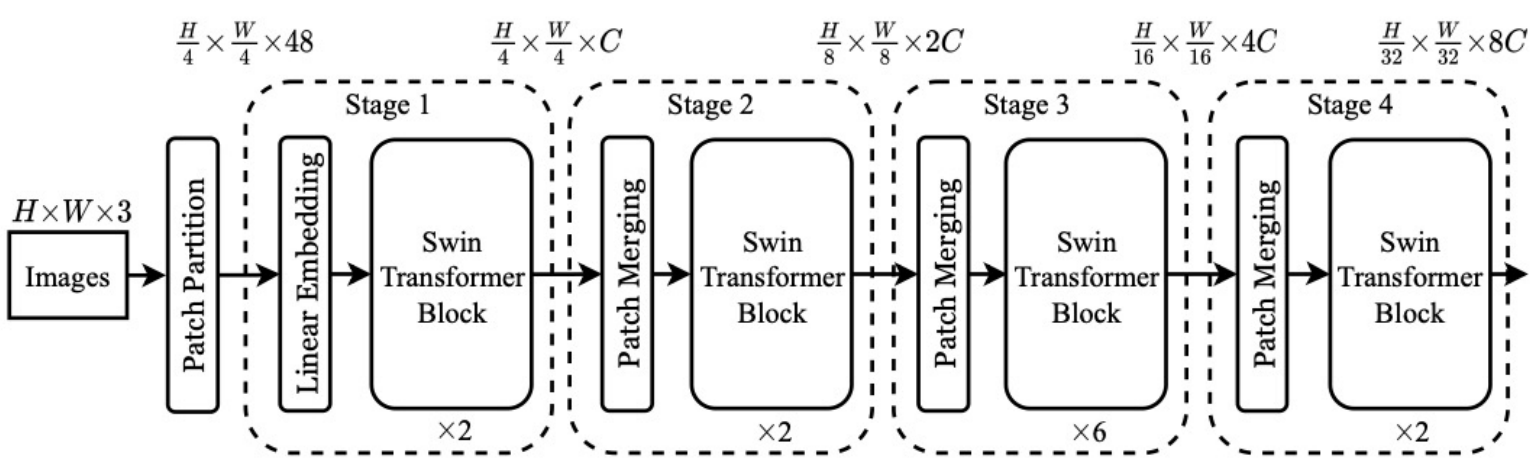
ICCV 2021 Best Paper

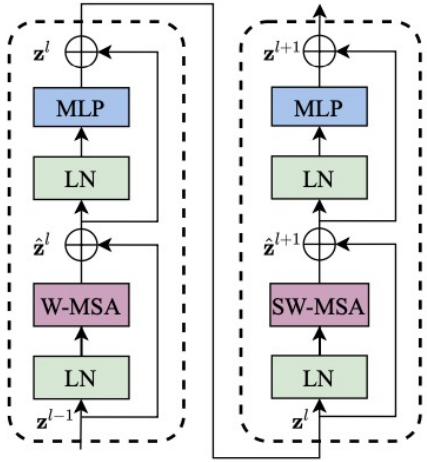
3.3 Architecture Of Swin Transformer

- Swin Transformer, that capably serves as a general-purpose backbone for computer vision, has a hierarchical architecture.
- In Swin Transformer , representation is computed with Shifted windows.

- Linear Embedding: $(\frac{H}{4} \times \frac{W}{4}, 48) \rightarrow (\frac{H}{4} \times \frac{W}{4}, C)$
- Patch Merging: $(\frac{H}{4} \times \frac{W}{4}, C) \rightarrow (\frac{H}{8} \times \frac{W}{8}, 4C) \rightarrow (\frac{H}{8} \times \frac{W}{8}, 2C)$



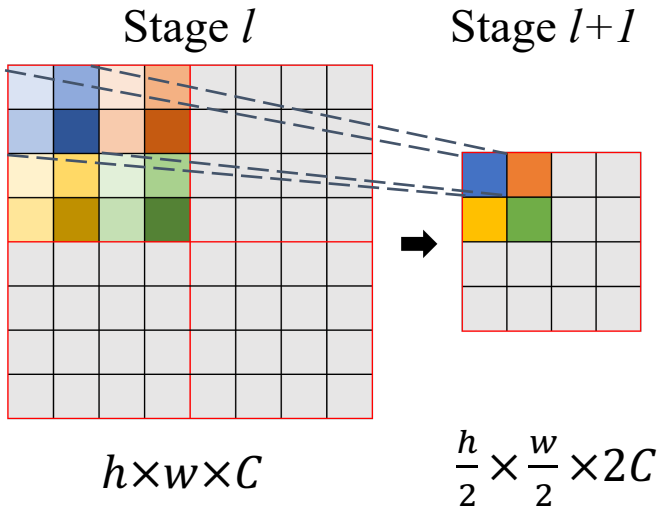


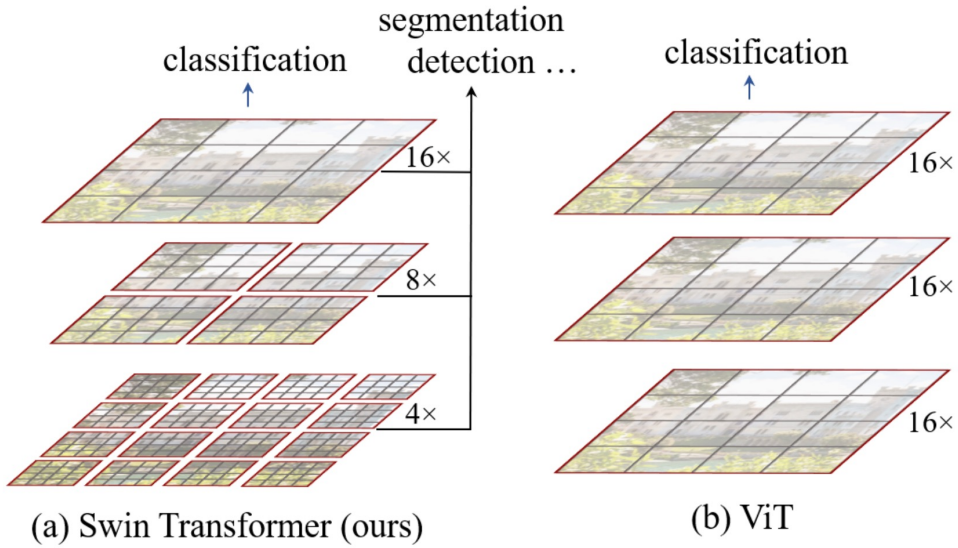


■ W-MSA: Regular Window Multi-head Self-attention
■ SW-MSA: Shifted Window Multi-head Self-attention

3.3 Why Hierarchical?

- The architecture of ViT is **unsuitable** for use as a general-purpose backbone network on **dense** vision tasks when input image **resolution is high**.
- As the structure deepens, the receptive field of patch in each layer gradually increases, eg., VGG and ResNet.



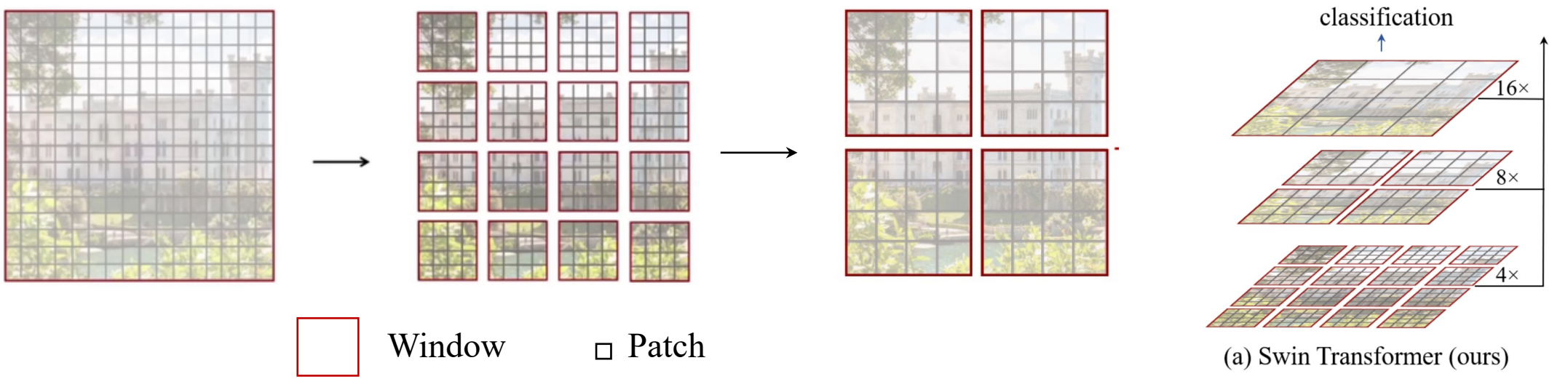


3.3 Window Multihead Self-attention

- Windows are arranged to evenly partition the image in a non-overlapping manner.
- The number of patches in each window is fixed, each window contains $M \times M$ patches.
- The computational complexity of global self-attention is quadratic to image size . Different from global self-attention, window is the unit for self-attention . So, it has linear complexity to image size.
- Supposing a global MSA module and a window based one on an image of $h \times w$ patches are:

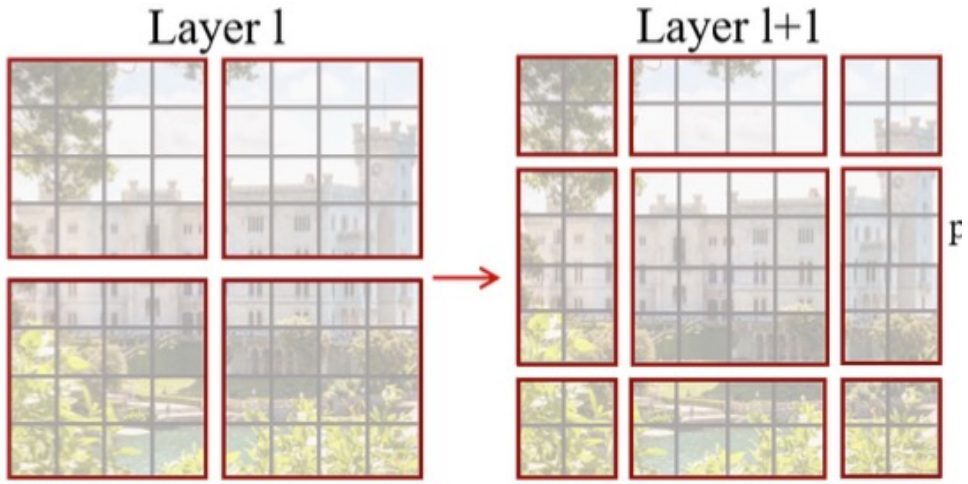
$$\Omega(\mathbf{MSA}) = 4hwC^2 + 2(hw)^2C, \quad (1)$$

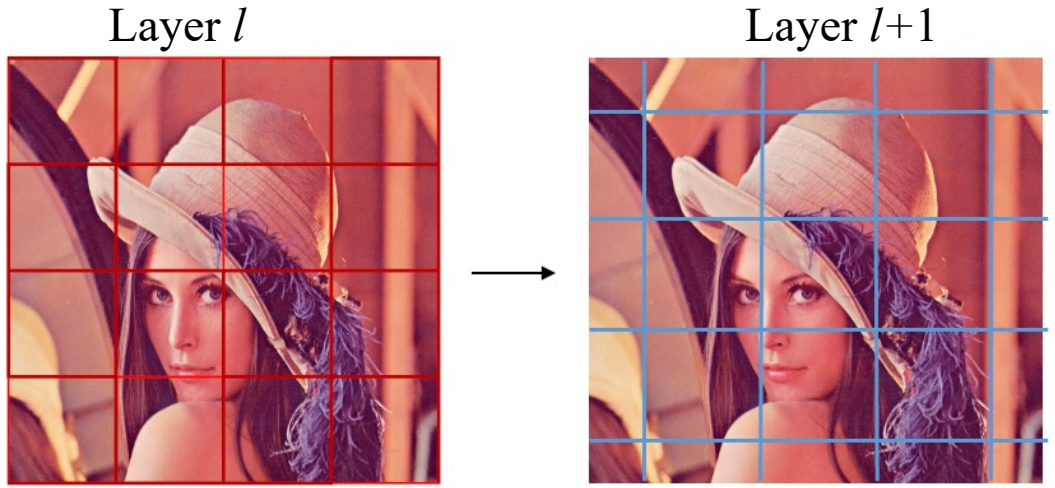
$$\Omega(\mathbf{W\text{-}MSA}) = 4hwC^2 + 2M^2hwC, \quad (2)$$



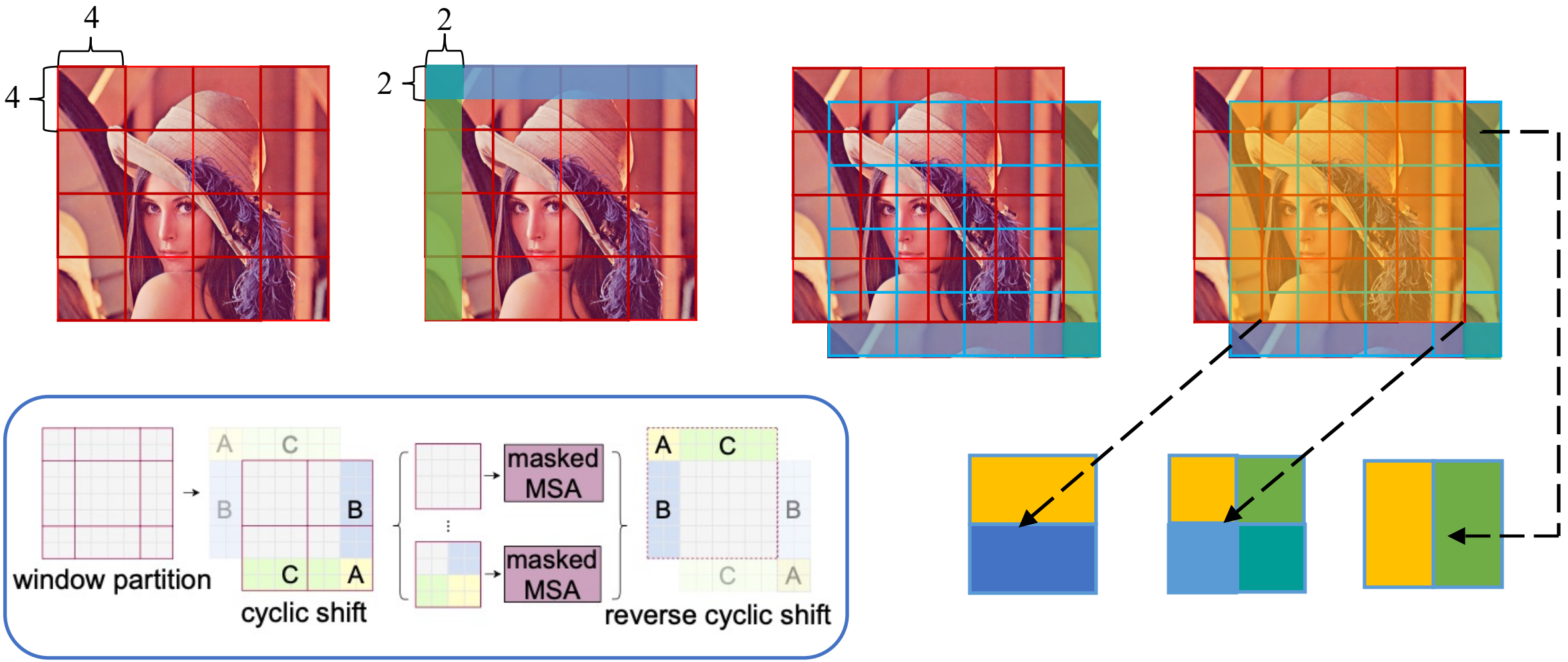
3.3 Shifted Window Multihead Self-attention

- The self-attention computation in the new windows crosses the boundaries of the previous windows in layer *l*, providing connections among them.





3.3 Shifted Window Multihead Self-attention



3.3 Experiment Of SwinT

Classification on ImageNet-1K

(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

Object detection on COCO

(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

3.3 Experiment Of SwinT

	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**

END

10.30