Lab Seminar

Swin Transformer / Video Swin Transformer
: Hierarchical Vision Transformer using Shifted Windows

University Of Seoul.
Machine Learning & Artificial Intelligence
Lab.

Junhyuk So
Undergraduate Student

MLAI
University of Seoul

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Ze Liu et al., ICCV 2021, Best Paper Award

- State of the Art | Object Detection on COCO test-dev (using additional training data)
- State of the Art | Instance Segmentation on COCO test-dev (using additional training data)
- State of the Art | Object Detection on COCO minival (using additional training data)
- State of the Art | Instance Segmentation on COCO minival (using additional training data)
- Ranked #8 | Semantic Segmentation on ADE20K (using additional training data)
- Ranked #9 | Semantic Segmentation on ADE20K val
- State of the Art | Action Recognition on Something-Something V2 (using additional training data)
- Ranked #2 | Action Classification on Kinetics-400 (using additional training data)
- Ranked #2 | Action Classification on Kinetics-600 (using additional training data)

Outline

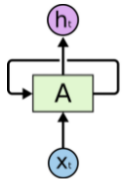
- Background
  - Transformer
  - Visual Transformer
- Swin Transformer
  - Motivation
  - Method
  - Experiments
- Video Swin Transformer
  - Method
  - Experiments
- Conclusion

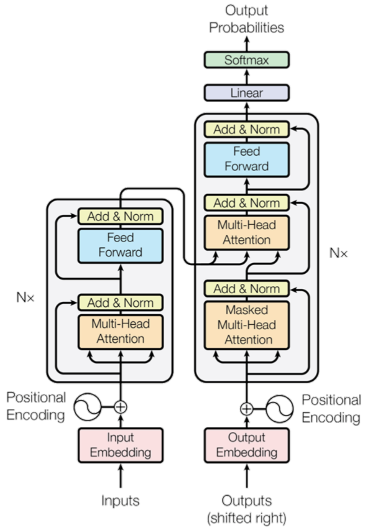
Background

Transformer

- Transformer is introduced to **solve problems** of **recurrent model**
  - Long term dependancy
  - Non-parallable structure

- By Using,
  - (self) **Attention** Mechanism
  - **Multi-head** Attention
  - Position Wise **Feed Forward**
  - **Positional Encoding**





MLAI
University of Seoul

Source : Attention is All you need

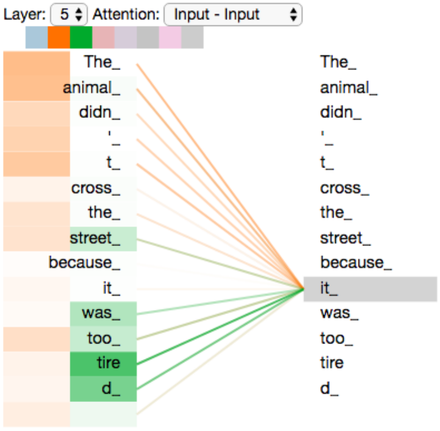
Background

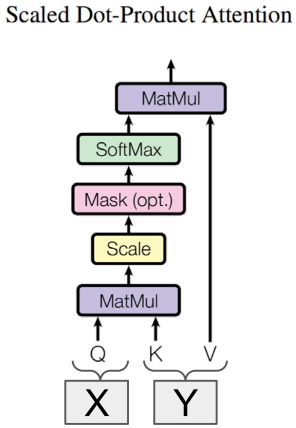
Attention

- To decide where to **Attend**
- 1. Transformation of layer map

$$Q = XW^Q, \quad K = YW^K, \quad V = YW^V,$$

  - if self-attention , X=Y
- 2. attention layer
  - Compare Every Q to Every K
    - by simple dot-product
  - Normalize and Softmax
    - Softmax with Q-dimension
  - Assign them(Multiply) to Value





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

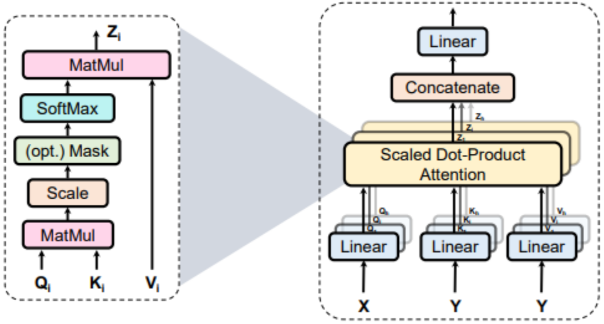
Source : Attention is All you need

Background

Multi Head Attention

- Single Head Attention may have restricted feature subspace
- Solution
  - Use Multiple Feature Subspace by Independent Heads

$$
\begin{aligned}
&Q_i = XW^{Q_i},\ K_i = XW^{K_i},\ V_i = XW^{V_i},\\
&Z_i = \mathrm{Attention}(Q_i, K_i, V_i),\ i = 1 \ldots h, \qquad (3)\\
&\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(Z_1, Z_2, ..., Z_h)W^O,
\end{aligned}
$$



Source : Attention is All you need

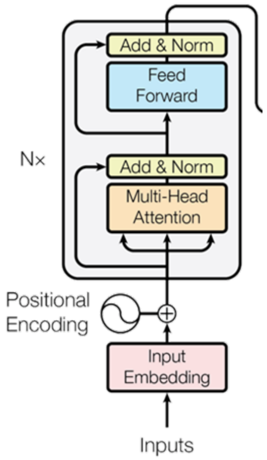
Background

Position Wise Feed Forward Network

- The Output of MHSA is then fed to Two Successive FFN

$$\mathrm{FFN}(x) = \mathrm{RELU}(W_1 x + b_1)W_2 + b_2.$$

- Position Wise ?
  - [Batch x N_seq x **seq_emb1**] -> [Batch x N_seq x **seq_emb2**]
  - 512 -> 2048 -> 512



Source : Attention is All you need

Background

Positional Encoding

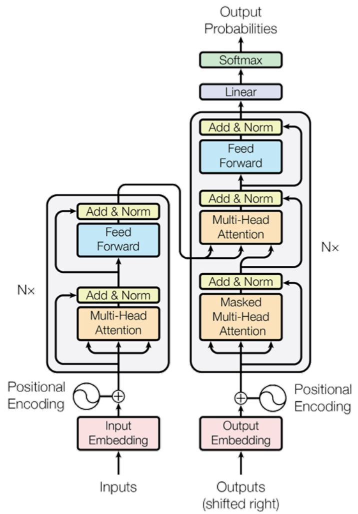
- **Attention** Mechanism handle All input sequence **identically**
  -> the **order** of sequence is **neglected**

- Solution
  - add some positional information to input embedding

$$PE_{(pos,i)} = \begin{cases} \sin(pos \cdot \omega_k) & \text{if} \quad i = 2k \\ \cos(pos \cdot \omega_k) & \text{if} \quad i = 2k+1, \end{cases}$$

$$\omega_k = \frac{1}{10000^{2k/d}}, \quad k = 1, \cdots, d/2,$$

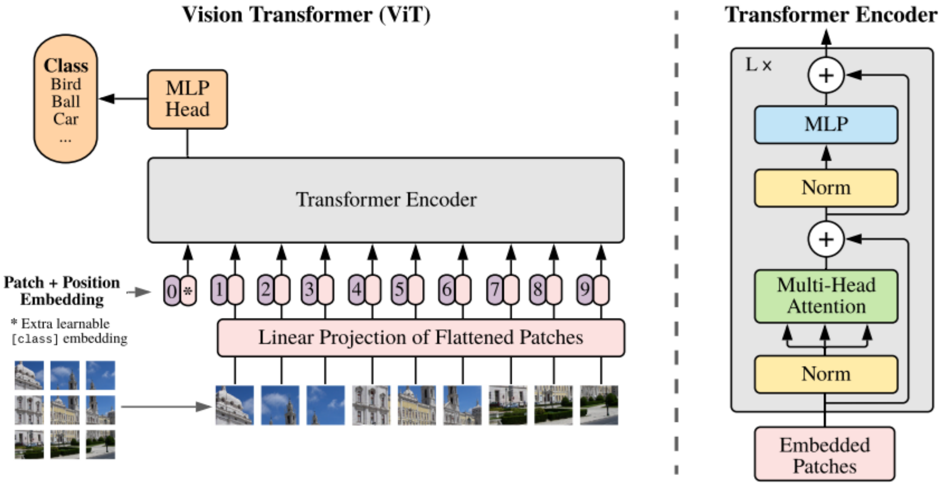
- where i,d are index, length



Source : Attention is All you need

Background

Vision Transformer



- Almost Same Architecture
  - Work Token -> Image Patch & Linear Projection
  - Normalization first

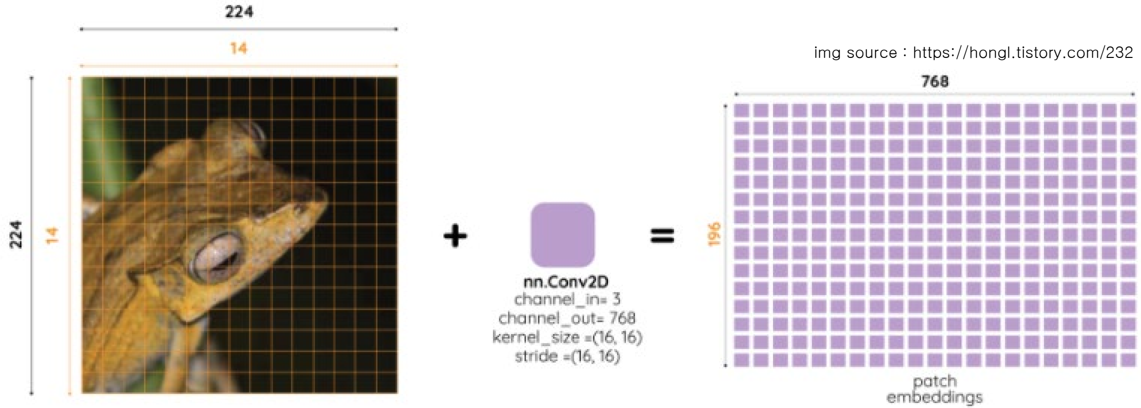
Source : An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale (https://arxiv.org/abs/2010.11929)

Background

ViT (Vision Transformer)

- Patch Partioning & Linear Projection
  - H*W*C -> (H/16)x(W/16)*3
  - Mix accross channel dim.
  - Can be implmeneted by simple Conv2D.

img source : https://hongl.tistory.com/232



Source : An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale (https://arxiv.org/abs/2010.11929)

Swin Transformer

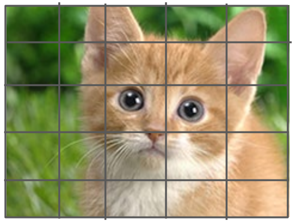
Motivation

- problems of original ViT
  - No consideration for Vision domain.
    - Scale-Problem
  - Computational Inefficiency
    - Quadratic Computation
    - Memory Inefficiency

Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Scale problem

- Size of object can vary depending on Location of Camera
  - Same Patch size ignores this issue.

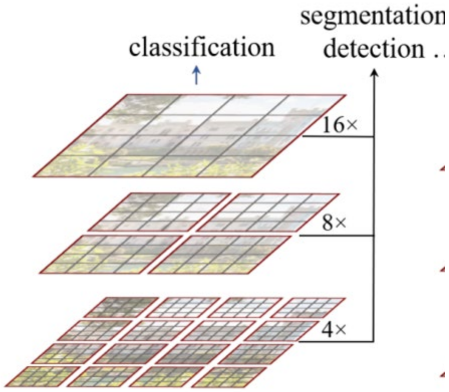


Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

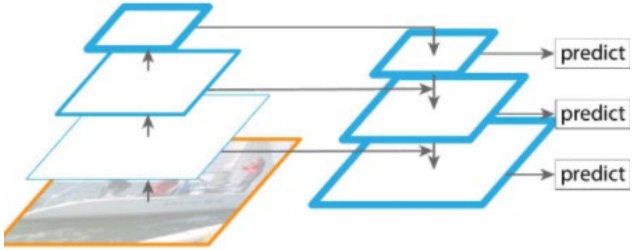
Scale problem

Solution

- Use Different Patch Size!
- Aggregate differently scaled feature map to make final prediction.
  - E.g) FPN



(a) Swin Transformer (ours)



(d) Feature Pyramid Network

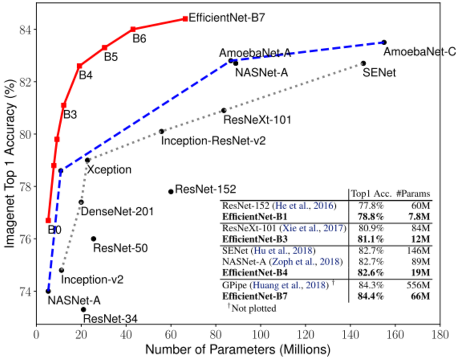
Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Quadratic Computation

- Attention Mechnaism -> Quadratic Complexity
  - hard to use larger image size (resolution)

$$A(Q, K, V) = softmax(QK^T)V$$

- But why we need larger image Size ?
  - Vision tasks tend to generalize well when use bigger image size
    - E.g) EfficientNet B0 : 224, EfficientNetB7 : 600



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**78.8%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.1%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.6%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.4%**	**66M**

† Not plotted

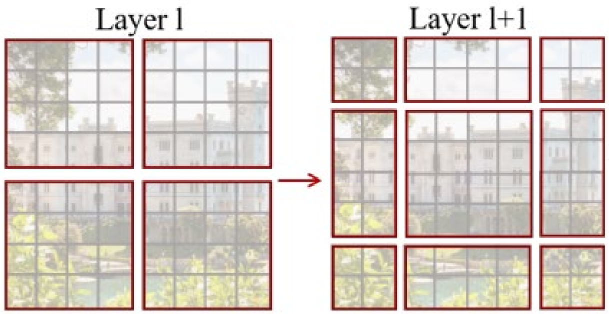
Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Quadratic Computation

- Solution
  - **Local Attention** (W – MSA)
    - Attend only in windows

- Can have **Linear** computational complexity :)
- Can add inductive bias :)?

- Can't handle **long range**-dependancy :(
  - **Use Shifted Window (SW-MSA)**



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

Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Overall Architecture

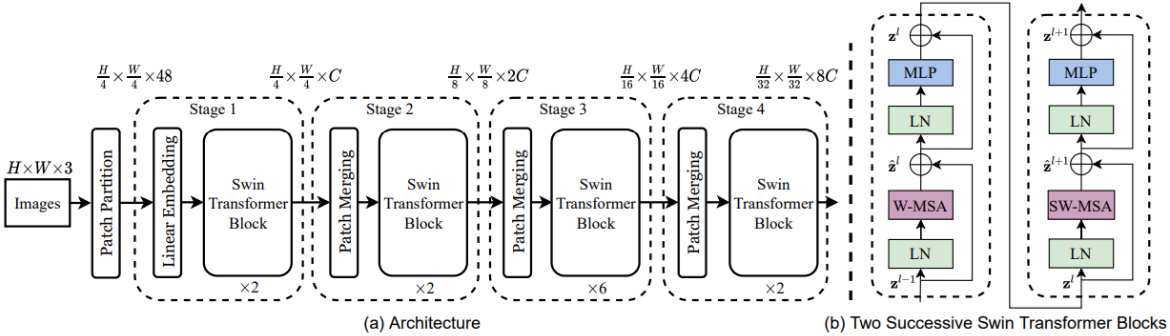
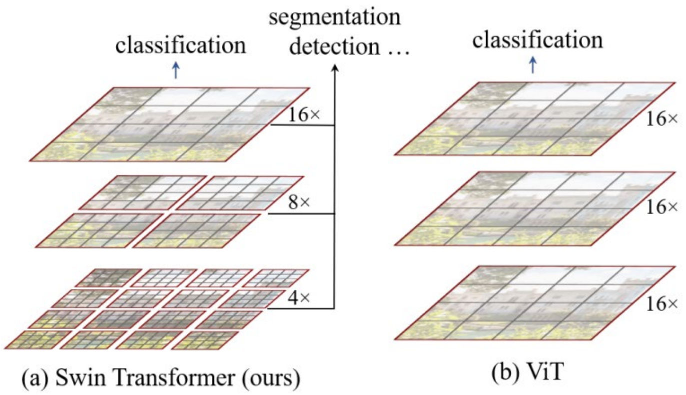


Figure 3. (a) The architecture of a Swin Transformer (Swin-T); (b) two successive Swin Transformer Blocks (notation presented with Eq. (3)). W-MSA and SW-MSA are multi-head self attention modules with regular and shifted windowing configurations, respectively.



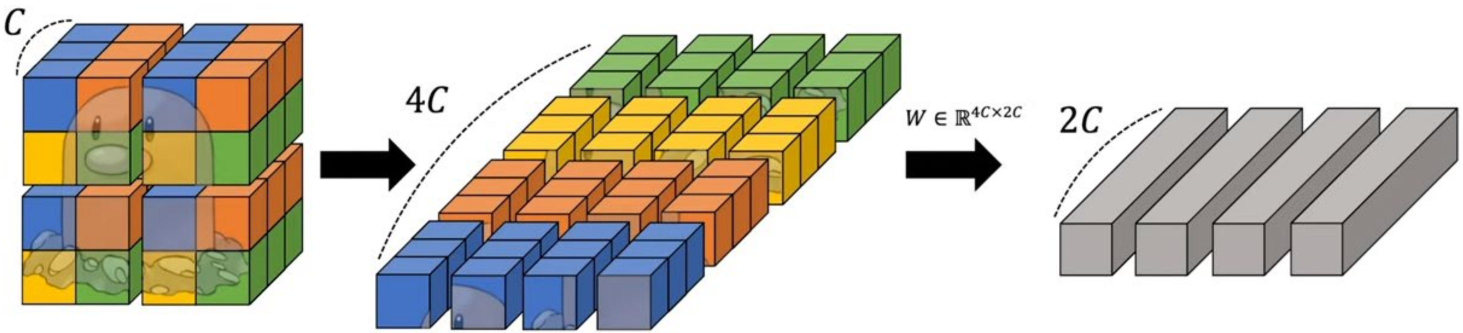
Details

- Patch Merging Block
- Relative Positional Bias
- Cyclic Shifted Batch Computing

Method

Patch Merging Block

- If we simply merge 2x2 Neighbor patches..
  - 1C -> 4C
- Use Linear projection across Channel dimension.
  - 4C -> 2C



Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Method

Relative Position Bias

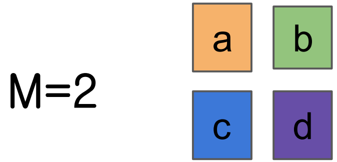
Relative position bias In computing self-attention, we follow [49, 1, 32, 33] by including a relative position bias $B \in \mathbb{R}^{M^2 \times M^2}$ to each head in computing similarity:

$$\text{Attention}(Q, K, V) = \text{SoftMax}(QK^T/\sqrt{d} + B)V, \quad (4)$$

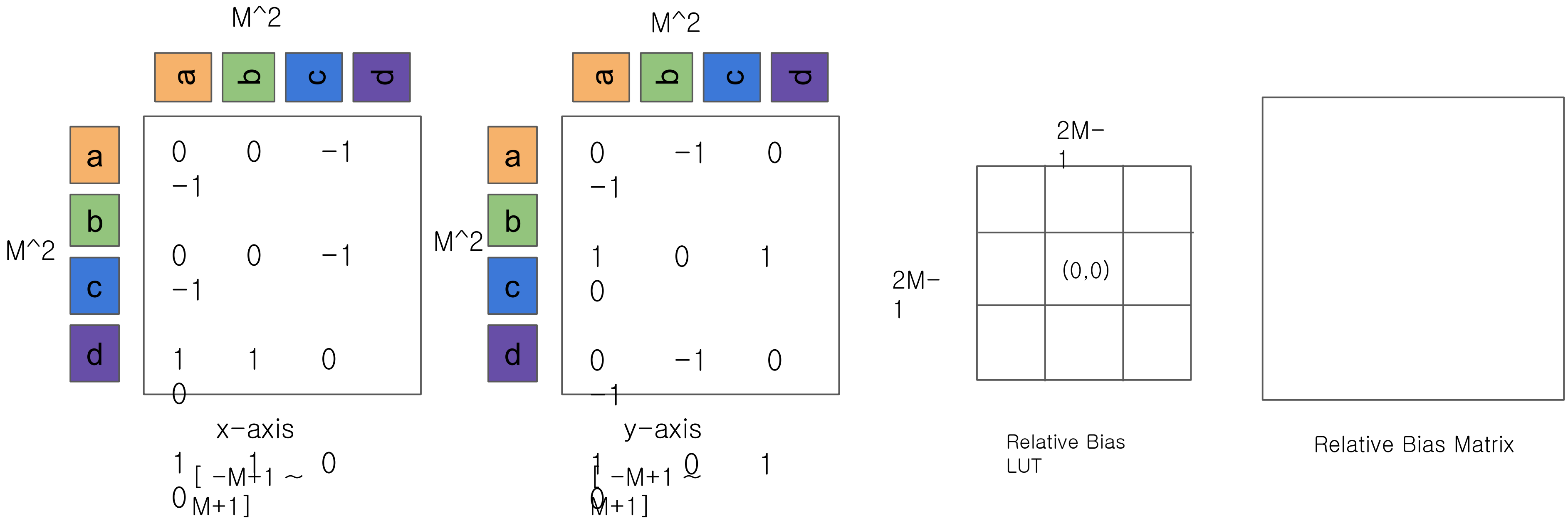
where $Q, K, V \in \mathbb{R}^{M^2 \times d}$ are the *query*, *key* and *value* matrices; d is the *query*/*key* dimension, and M^2 is the number of patches in a window. Since the relative position along each axis lies in the range $[-M+1, M-1]$, we parameterize a smaller-sized bias matrix $\hat{B} \in \mathbb{R}^{(2M-1)\times(2M-1)}$, and values in B are taken from $\hat{B}$.

Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows
(https://arxiv.org/pdf/2103.14030.pdf)

Method

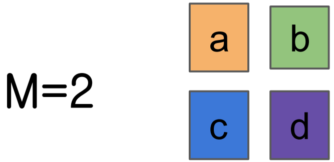


Relative Position Bias

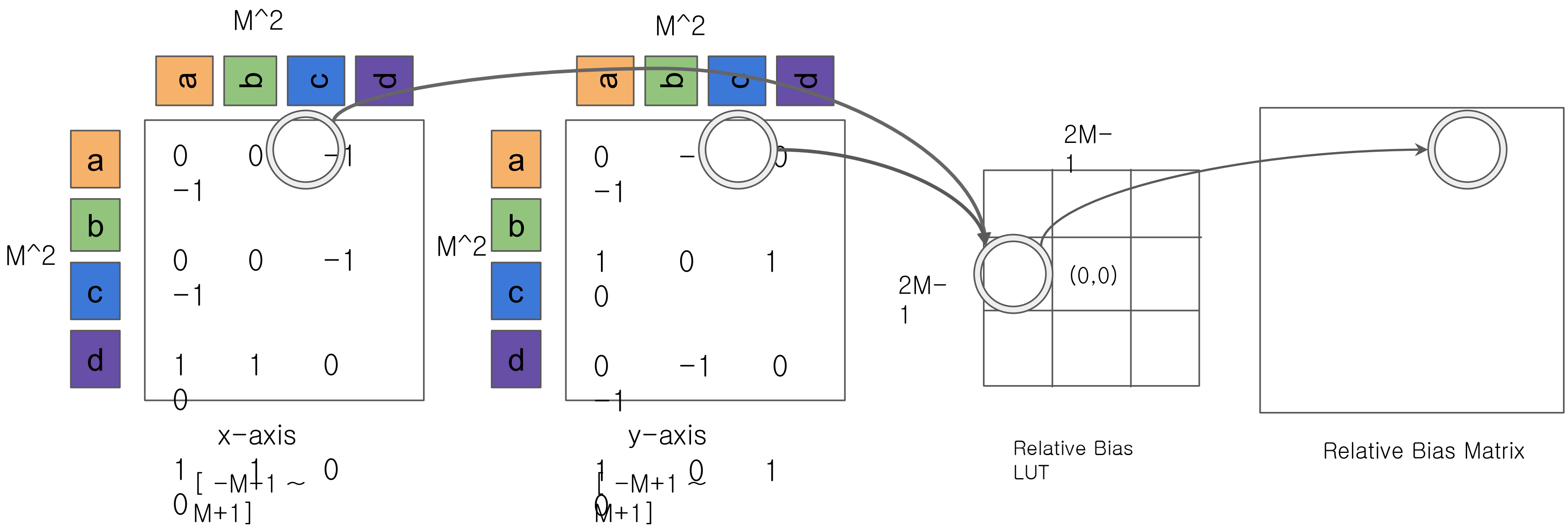


Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Method



Relative Position Bias



Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Cyclic Shifted Batch Computing

- Problem : If we use SW-MSA, Window sizes are not same
  - Naive solution : Pad every small windows
    - But this make # Patch N+1,N+1. → More Computation !
- Solution : Cyclic Shift and Masked MSA

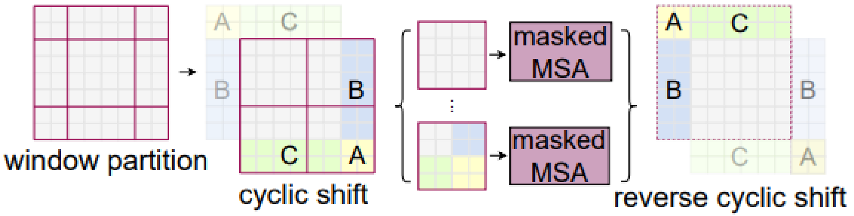


Figure 4. Illustration of an efficient batch computation approach for self-attention in shifted window partitioning.

Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Cyclic Shifted Batch Computing

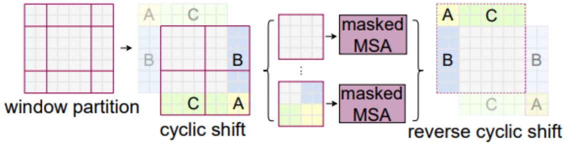
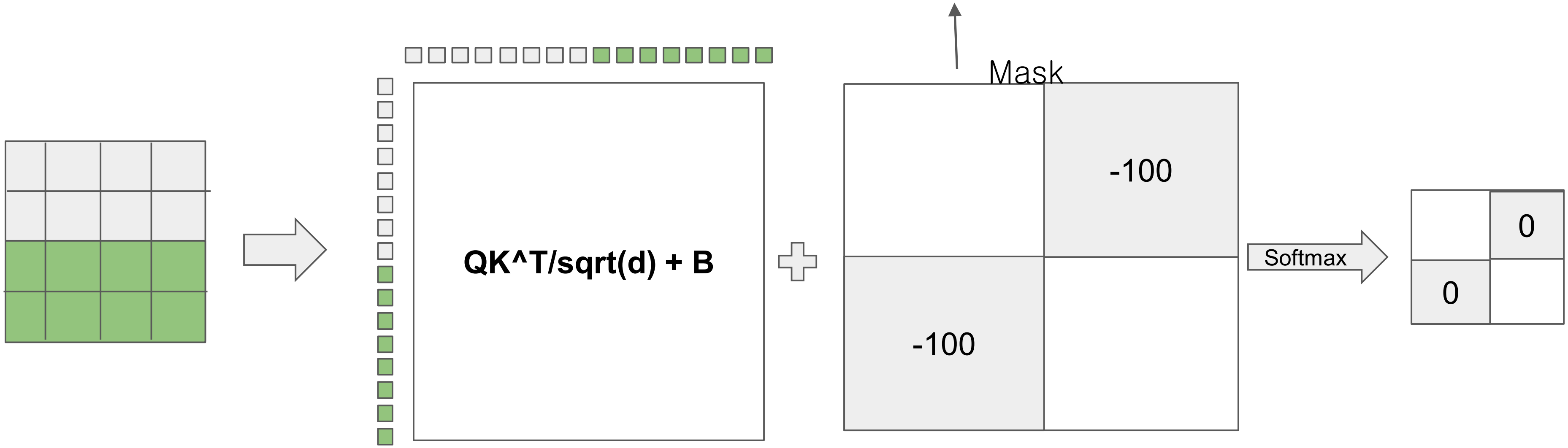


Figure 4. Illustration of an efficient batch computation approach for self-attention in shifted window partitioning.

Masked
MSA

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



Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Experiments

Datasets

- Classification
  - Imagenet 1k
  - Imagenet 1k – ImageNet 22K Pretrained
- Detection
  - COCO 2017
- Segmentation
  - ADE20K

Settings

- Optimizer , LR
  - AdamW (1e-3, bs 1024)
- Augmentation
  - RandAug, Mixup, Cutmix, Cutout
  - Stochastic Depth.
  - No Repeated Aug.
- Regularization
  - Weight decay, Gradient clipping

Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Experiments

Shifted Windows w/o, Positional Embedding w/po

	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).

Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Experiments

Imagenet 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

(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

Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Experiments

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

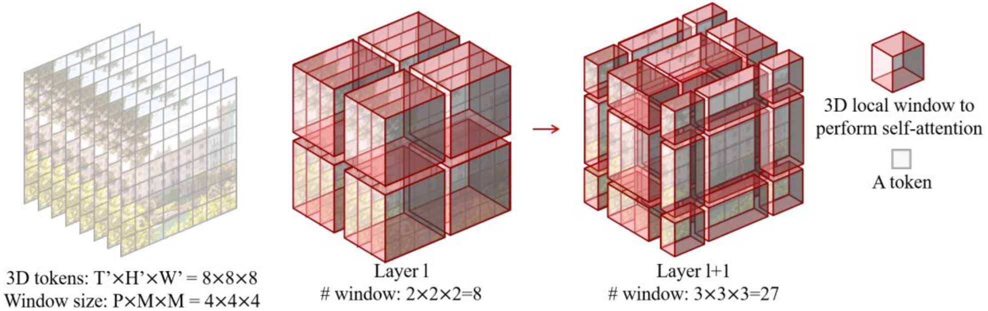
Source : Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Video Swin Transformer

Ze Liu et al, 2021 Arxiv

Source : Video Swin Transformer:

Motivation



- Spatio-temporal adaption of Swin Transformer
- strictly follows the hierarchical structure of the original Swin
  - but extends the scope of local attention computation
    from only the spatial domain to the spatiotemporal domain

MLAI
University of Seoul

Source : Video Swin Transformer:

Method

Overall Architecture

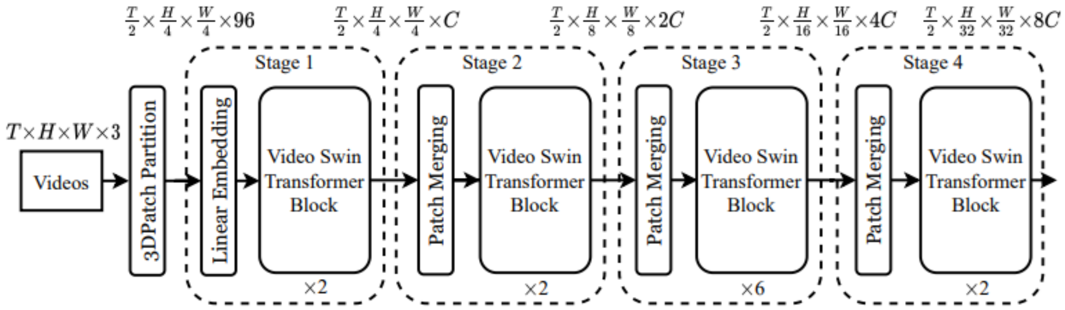
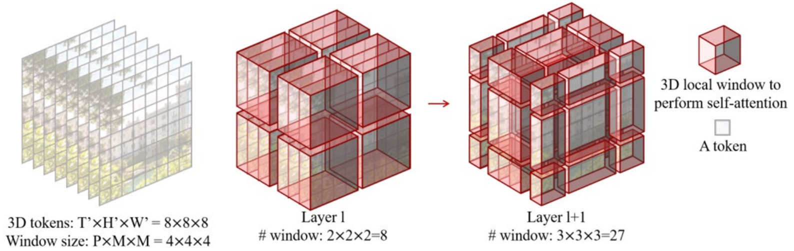


Figure 1: Overall architecture of Video Swin Transformer (tiny version, referred to as Swin-T).



Details

- No patch merging between T-dimension
- 3D Relative Position Bias

Method

3D Relational Position Bias

3D Relative Position Bias Numerous previous works [31, 2, 16, 17] have shown that it can be advantageous to include a relative position bias to each head in self-attention computation. Thus, we follow [28] by introducing 3D relative position bias $B \in \mathbb{R}^{P^2 \times M^2 \times M^2}$ for each head as

$$\text{Attention}(Q, K, V) = \text{SoftMax}(QK^T/\sqrt{d} + B)V, \tag{2}$$

where $Q, K, V \in \mathbb{R}^{PM^2 \times d}$ are the *query*, *key* and *value* matrices; d is the dimension of *query* and *key* features, and PM^2 is the number of tokens in a 3D window. Since the relative position along each axis lies in the range of $[-P+1, P-1]$ (temporal) or $[-M+1, M-1]$ (height or width), we parameterize a smaller-sized bias matrix $\hat{B} \in \mathbb{R}^{(2P-1)\times(2M-1)\times(2M-1)}$, and values in B are taken from $\hat{B}$.

3D Bias LUT

MLAI
University of Seoul

Source : Video Swin Transformer:

Experiments

Table 1: Comparison to state-of-the-art on Kinetics-400. "384↑" signifies that the model uses a large spatial resolution of 384×384. "Views" indicates # temporal clip × # spatial crop. The magnitude are Giga (10^9) and Mega (10^6) for FLOPs and Param respectively.

Method	Pretrain	Top-1	Top-5	Views	FLOPs	Param
R(2+1)D [37]	-	72.0	90.0	10 × 1	75	61.8
I3D [6]	ImageNet-1K	72.1	90.3	-	108	25.0
NL I3D-101 [40]	ImageNet-1K	77.7	93.3	10 × 3	359	61.8
ip-CSN-152 [36]	-	77.8	92.8	10 × 3	109	32.8
CorrNet-101 [39]	-	79.2	-	10 × 3	224	-
SlowFast R101+NL [13]	-	79.8	93.9	10 × 3	234	59.9
X3D-XXL [12]	-	80.4	94.6	10 × 3	144	20.3
MViT-B, 32×3 [10]	-	80.2	94.4	1 × 5	170	36.6
MViT-B, 64×3 [10]	-	81.2	95.1	3 × 3	455	36.6
TimeSformer-L [3]	ImageNet-21K	80.7	94.7	1 × 3	2380	121.4
ViT-B-VTN [29]	ImageNet-21K	78.6	93.7	1 × 1	4218	11.04
ViViT-L/16x2 [1]	ImageNet-21K	80.6	94.7	4 × 3	1446	310.8
ViViT-L/16x2 320 [1]	ImageNet-21K	81.3	94.7	4 × 3	3992	310.8
ip-CSN-152 [36]	IG-65M	82.5	95.3	10 × 3	109	32.8
ViViT-L/16x2 [1]	JFT-300M	82.8	95.5	4 × 3	1446	310.8
ViViT-L/16x2 320 [1]	JFT-300M	83.5	95.5	4 × 3	3992	310.8
ViViT-H/16x2 [1]	JFT-300M	84.8	95.8	4 × 3	8316	647.5
Swin-T	ImageNet-1K	78.8	93.6	4 × 3	88	28.2
Swin-S	ImageNet-1K	80.6	94.5	4 × 3	166	49.8
Swin-B	ImageNet-1K	80.6	94.6	4 × 3	282	88.1
Swin-B	ImageNet-21K	82.7	95.5	4 × 3	282	88.1
Swin-L	ImageNet-21K	83.1	95.9	4 × 3	604	197.0
Swin-L (384↑)	ImageNet-21K	84.6	96.5	4 × 3	2107	200.0
Swin-L (384↑)	ImageNet-21K	**84.9**	**96.7**	10 × 5	2107	200.0

Video Classification

- Pretraining on ImageNet
  - Directly duplicate weights in Swin Twice
  - Multiply whole matrix 0.5

Experiments

Ablation Study

Temporal dimension/window size

Table 5: Ablation study on temporal dimension of 3D tokens and temporal window size with Swin-T on K400.

temporal dimension	Window size	Top 1	Top 5	FLOPs	Param
16	16×7×7	79.1	93.8	106	28.5
8	8×7×7	78.5	93.2	44	28.2
4	4×7×7	76.7	92.5	20	28.0
16	16×7×7	79.1	93.8	106	28.5
16	8×7×7	78.8	93.6	88	28.2
16	4×7×7	78.6	93.4	79	28.0

Temporal shifting Effect

Table 6: Ablation study on the 3D shifted window approach with Swin-T on K400.

	Top-1	Top-5
w. 3D shifting	78.8	93.6
w/o temporal shifting	78.5	93.5
w/o 3D shifting	78.1	93.3

Source : Video Swin Transformer:

Conclusion

Swin Transformer

- Solve scale problem by Dynamic Patch Size
- Solve Quadratic Time Complexity probelm by Local Attention
    - Solve Long range dependancy problem by Shifted Window

Future research ?

- Temporal Hierarchy
- Temporal scale problem

Q&A