Switching between CNNs and Transformers

Wangmeng Zuo

Center on Machine Learning Research
Harbin Institute of Technology

Content

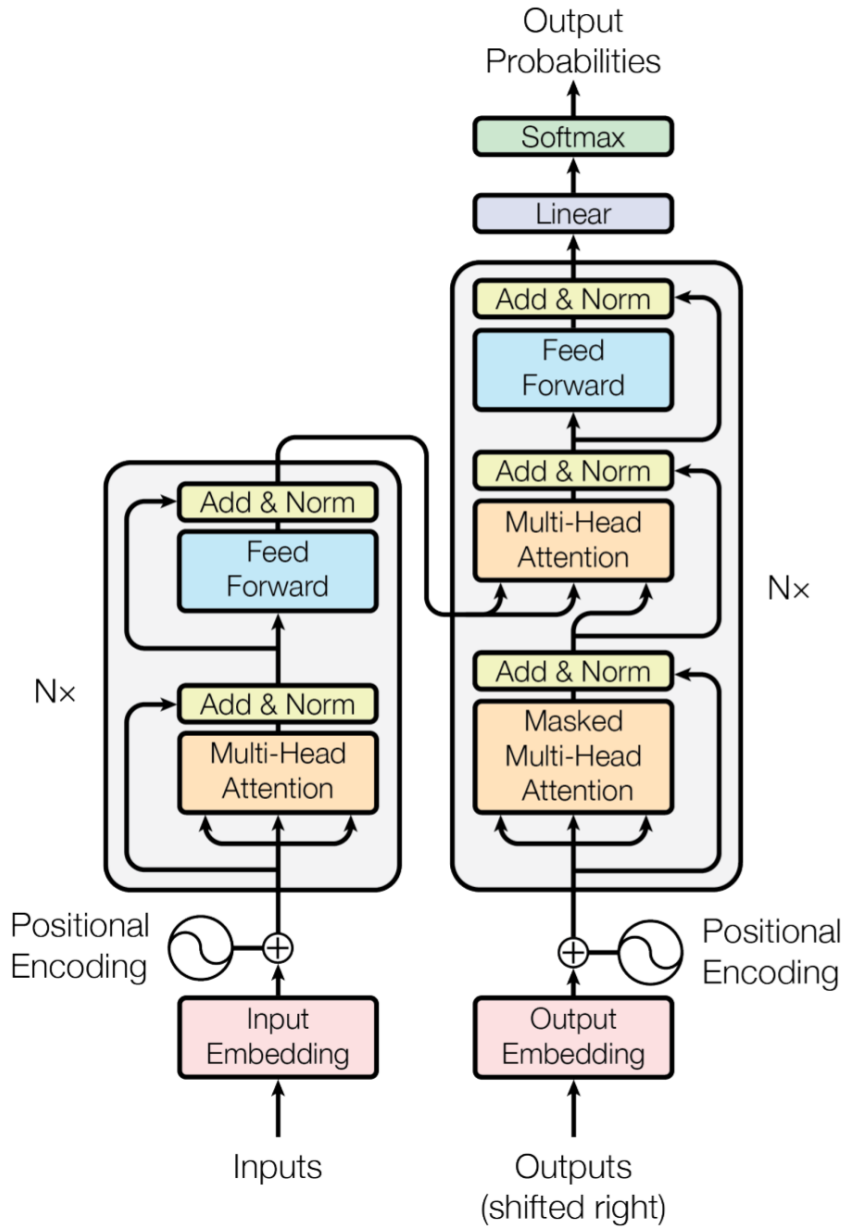
- CNN -> Vision Transformers

- Vision Transformers -> CNN

Transformer

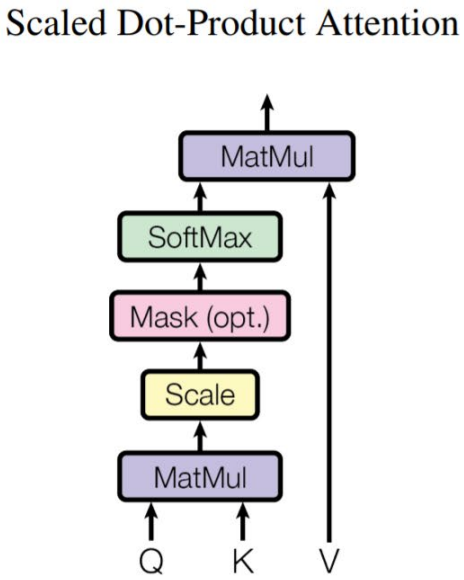
- NLP
- Encoder-decoder
- Feed Forward Network

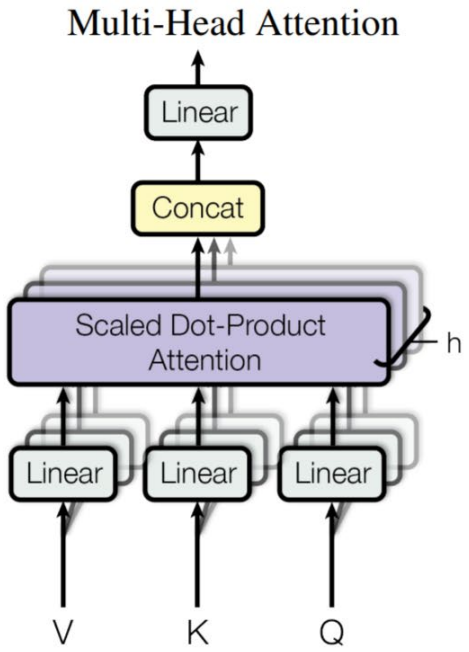
$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

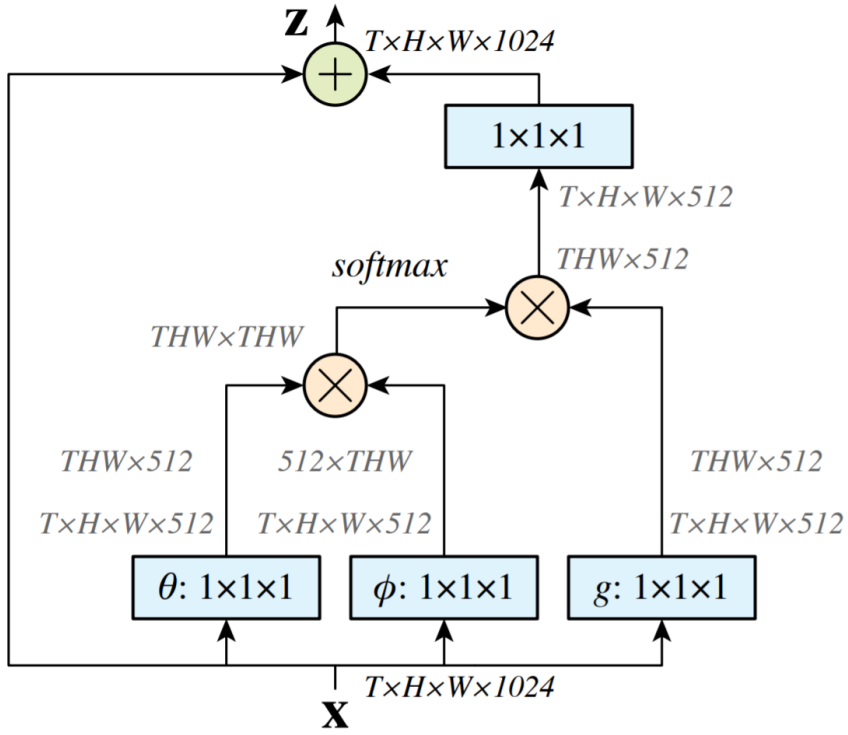


Spatial Self-Attention与Non-Local

- Spatial Self-Attention Non-Local Block







Spatial Self-Attention

- 目标：通过计算特征图上任意两个空间位置特征向量的相似性，来度量特征综合时的占比，再从空间维度更新特征（有点像聚类）
- 过程：
- 使用Query和Key计算相似性
- Scale且归一化
- 使用Value和Attention Map沿空间维度更新特征

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Multi-Head Self-Attention

- 能否引入multi-view ensemble的办法来进一步优化Attention？

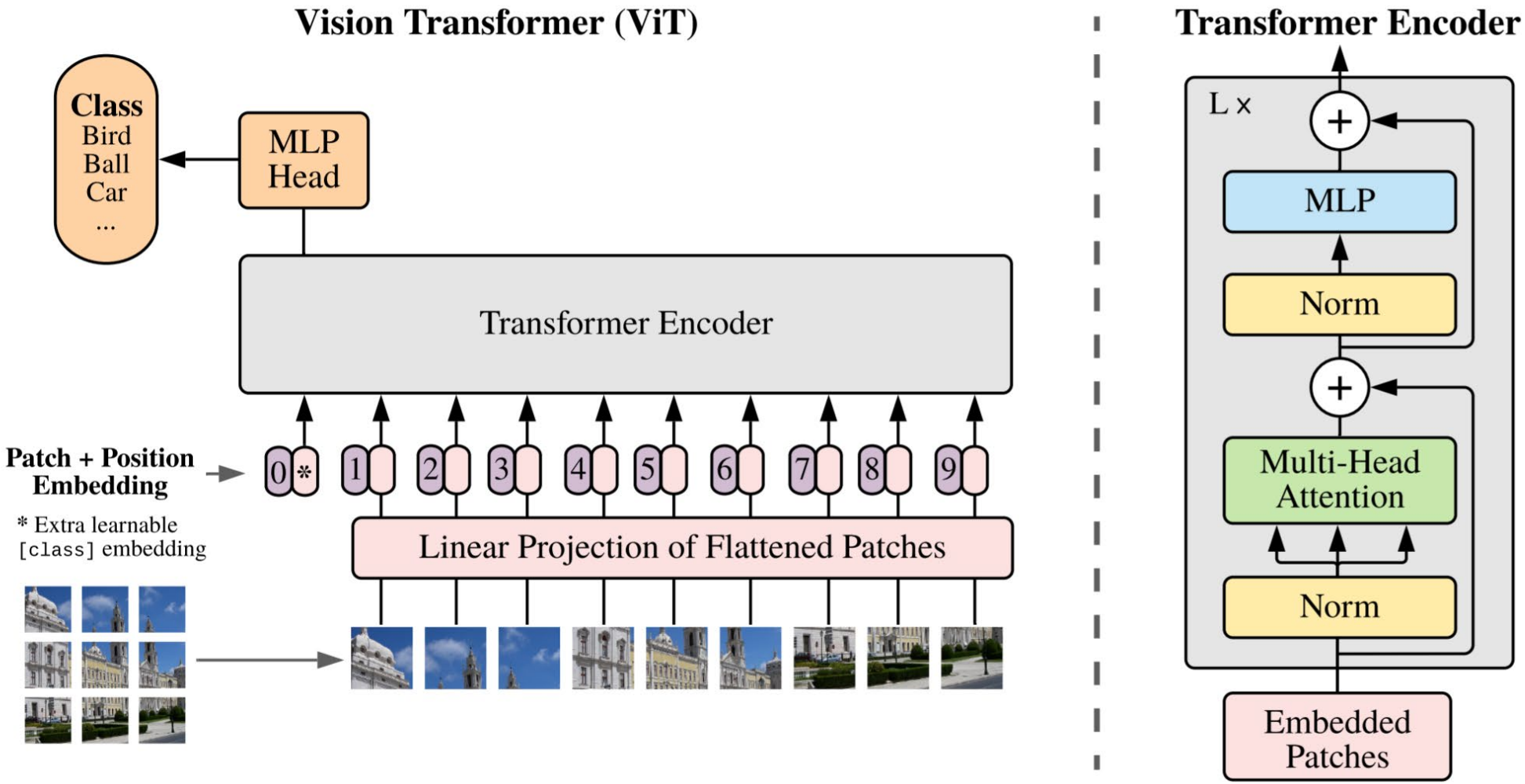
$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

- W^O 用来做channel间的特征合并
- 实验证明Multi-head确实是更容易收敛且表现稍好

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	25.4	

*注：single-head还是multi-head，可以看这篇的分析： https://arxiv.org/abs/2106.09650

Vision Transformer



*Tokens-to-Token ViT: Training Vision Transformers from Scratch on ImageNet, Arxiv 2020

DeiT

- Grid Search on Optimization Algorithms, Hyper-parameters, and Data Augmentation

Ablation on ↓	Pre-training	Fine-tuning	Rand-Augment	AutoAug	Mixup	CutMix	Erasing	Stoch. Depth	Repeated Aug.	Dropout	Exp. Moving Avg.	top-1 accuracy: pre-trained 224^2	top-1 accuracy: fine-tuned 384^2
none: DeiT-B	adamw	adamw	✓	✗	✓	✓	✓	✓	✓	✗	✗	81.8 ±0.2	83.1 ±0.1
optimizer	SGD	adamw	✓	✗	✓	✓	✓	✓	✓	✗	✗	74.5	77.3
	adamw	SGD	✓	✗	✓	✓	✓	✓	✓	✗	✗	81.8	83.1
data augmentation	adamw	adamw	✗	✗	✓	✓	✓	✓	✓	✗	✗	79.6	80.4
	adamw	adamw	✗	✓	✓	✓	✓	✓	✓	✗	✗	81.2	81.9
	adamw	adamw	✓	✗	✗	✓	✓	✓	✓	✗	✗	78.7	79.8
	adamw	adamw	✓	✗	✓	✗	✓	✓	✓	✗	✗	80.0	80.6
	adamw	adamw	✓	✗	✗	✗	✓	✓	✓	✗	✗	75.8	76.7
regularization	adamw	adamw	✓	✗	✓	✓	✗	✓	✓	✗	✗	4.3*	0.1
	adamw	adamw	✓	✗	✓	✓	✓	✗	✓	✗	✗	3.4*	0.1
	adamw	adamw	✓	✗	✓	✓	✓	✓	✗	✗	✗	76.5	77.4
	adamw	adamw	✓	✗	✓	✓	✓	✓	✓	✓	✗	81.3	83.1
	adamw	adamw	✓	✗	✓	✓	✓	✓	✓	✗	✓	81.9	83.1

Training data-efficient image transformers & distillation through attention, Arxiv 2020.

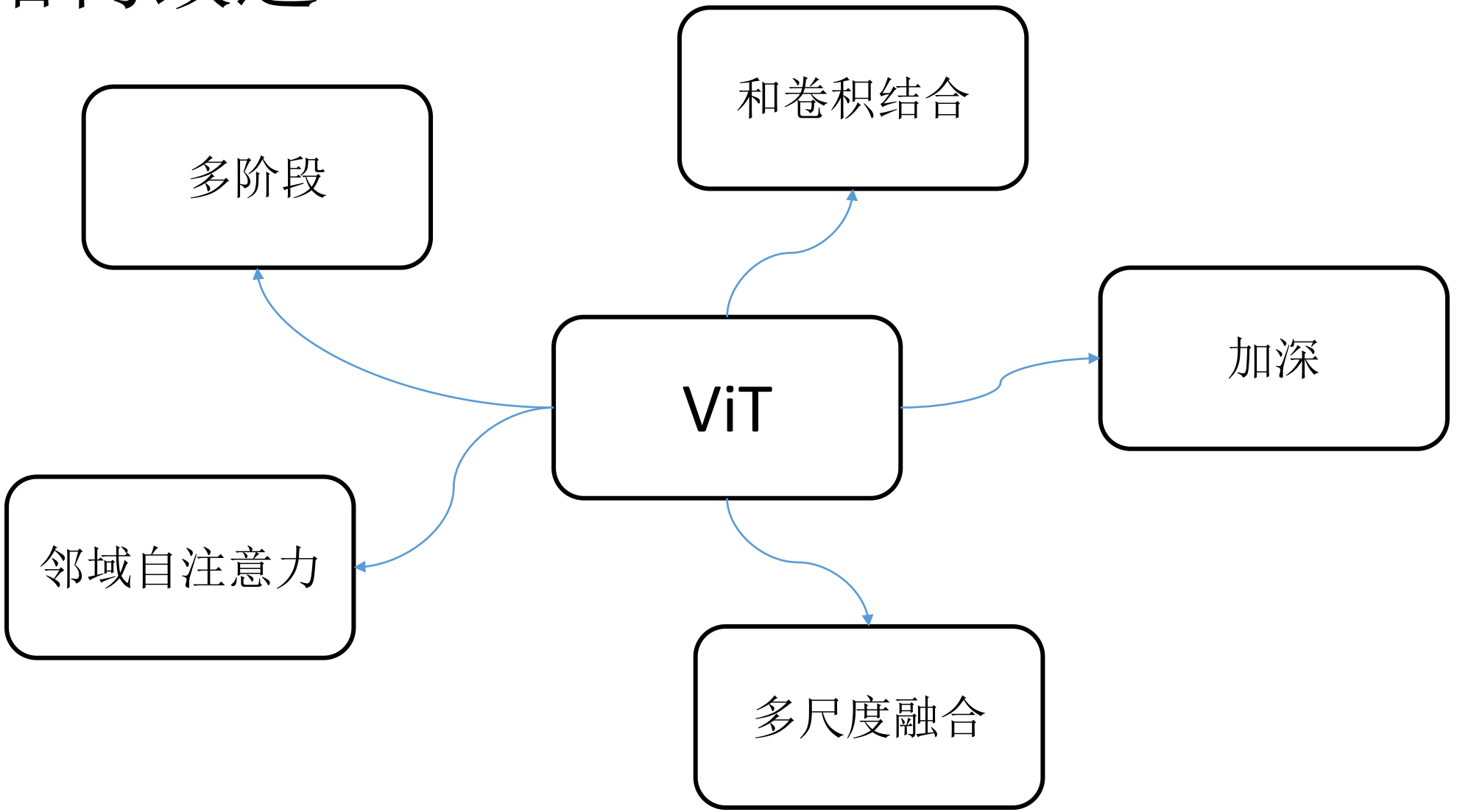
DeiT

- AdamW+strong regularization+ strong augmentation+more training epochs

Methods	ViT-B [15]	DeiT-B
Epochs	300	300
Batch size	4096	1024
Optimizer	AdamW	AdamW
learning rate	0.003	$0.0005 \times \frac{\text{batchsize}}{512}$
Learning rate decay	cosine	cosine
Weight decay	0.3	0.05
Warmup epochs	3.4	5
Label smoothing ε	✗	0.1
Dropout	0.1	✗
Stoch. Depth	✗	0.1
Repeated Aug	✗	✓
Gradient Clip.	✓	✗
Rand Augment	✗	9/0.5
Mixup prob.	✗	0.8
Cutmix prob.	✗	1.0
Erasing prob.	✗	0.25

Table 9: Ingredients and hyper-parameters for our method and Vit-B.

结构改进



Pyramid Vision Transformer

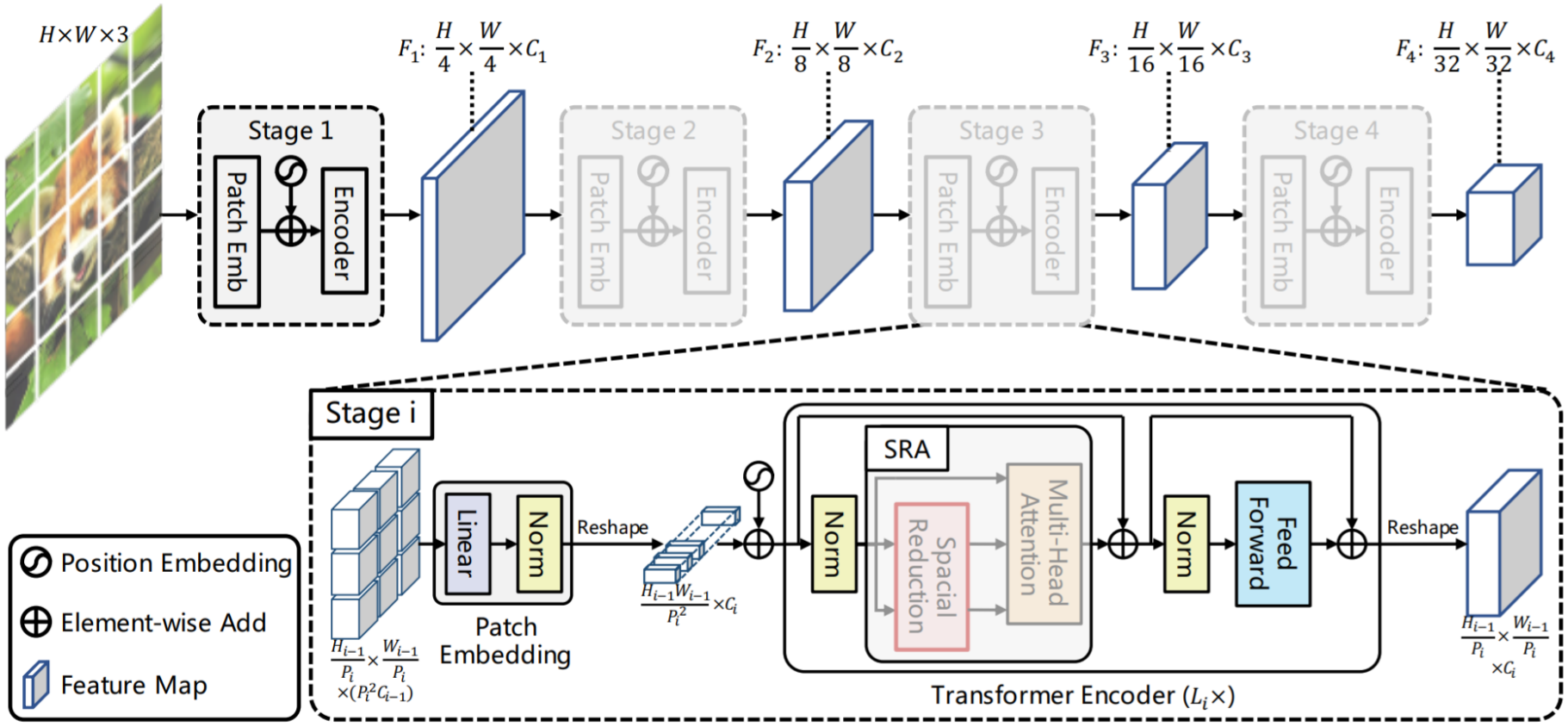


Figure 3: **Overall architecture of Pyramid Vision Transformer (PVT).** The entire model is divided into four stages, each of which is comprised of a patch embedding layer and a L_i-layer Transformer encoder. Following a pyramid structure, the output resolution of the four stages progressively shrinks from high (4-stride) to low (32-stride).

Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction Without Convolutions, ICCV 2021.

PVT V2

- 大致结构没有变，但是引入了三个很有用的组件
- 1.overlap patch embedding
- 2.用卷积模块构造的FFN
- 3.线性复杂度的Attention

Backbone	RetinaNet 1×							Mask R-CNN 1×						
	#P (M)	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	#P (M)	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
PVTv2-B0	**13.0**	**37.2**	**57.2**	**39.5**	**23.1**	**40.4**	**49.7**	**23.5**	**38.2**	**60.5**	**40.7**	**36.2**	**57.8**	**38.6**
ResNet18 [13]	**21.3**	31.8	49.6	33.6	16.3	34.3	43.2	**31.2**	34.0	54.0	36.7	31.2	51.0	32.7
PVTv1-Tiny [31]	23.0	36.7	56.9	38.9	22.6	38.8	50.0	32.9	36.7	59.2	39.3	35.1	56.7	37.3
PVTv2-B1 (ours)	23.8	**41.2**	**61.9**	**43.9**	**25.4**	**44.5**	**54.3**	33.7	**41.8**	**64.3**	**45.9**	**38.8**	**61.2**	**41.6**
ResNet50 [13]	37.7	36.3	55.3	38.6	19.3	40.0	48.8	44.2	38.0	58.6	41.4	34.4	55.1	36.7
PVTv1-Small [31]	34.2	40.4	61.3	43.0	25.0	42.9	55.7	44.1	40.4	62.9	43.8	37.8	60.1	40.3
PVTv2-B2-Li (ours)	**32.3**	43.6	64.7	46.8	28.3	47.6	57.4	**42.2**	44.1	66.3	48.4	40.5	63.2	43.6
PVTv2-B2 (ours)	35.1	**44.6**	**65.6**	**47.6**	**27.4**	**48.8**	**58.6**	45.0	**45.3**	**67.1**	**49.6**	**41.2**	**64.2**	**44.4**
ResNet101 [13]	56.7	38.5	57.8	41.2	21.4	42.6	51.1	63.2	40.4	61.1	44.2	36.4	57.7	38.8
ResNeXt101-32x4d [33]	56.4	39.9	59.6	42.7	22.3	44.2	52.5	**62.8**	41.9	62.5	45.9	37.5	59.4	40.2
PVTv1-Medium [31]	**53.9**	41.9	63.1	44.3	25.0	44.9	57.6	63.9	42.0	64.4	45.6	39.0	61.6	42.1
PVTv2-B3 (ours)	55.0	**45.9**	**66.8**	**49.3**	**28.6**	**49.8**	**61.4**	64.9	**47.0**	**68.1**	**51.7**	**42.5**	**65.7**	**45.7**
PVTv1-Large [31]	**71.1**	42.6	63.7	45.4	25.8	46.0	58.4	**81.0**	42.9	65.0	46.6	39.5	61.9	42.5
PVTv2-B4 (ours)	72.3	**46.1**	**66.9**	**49.2**	**28.4**	**50.0**	**62.2**	82.2	**47.5**	**68.7**	**52.0**	**42.7**	**66.1**	**46.1**
ResNeXt101-64x4d [33]	95.5	41.0	60.9	44.0	23.9	45.2	54.0	101.9	42.8	63.8	47.3	38.4	60.6	41.3
PVTv2-B5 (ours)	**91.7**	**46.2**	**67.1**	**49.5**	**28.5**	**50.0**	**62.5**	**101.6**	**47.4**	**68.6**	**51.9**	**42.5**	**65.7**	**46.0**

Table 3: **Object detection and instance segmentation on COCO `val2017`.** "-Li" denotes PVTv2 with linear SRA. "#P" refers to parameter number. AP^b and AP^m denote bounding box AP and mask AP, respectively.

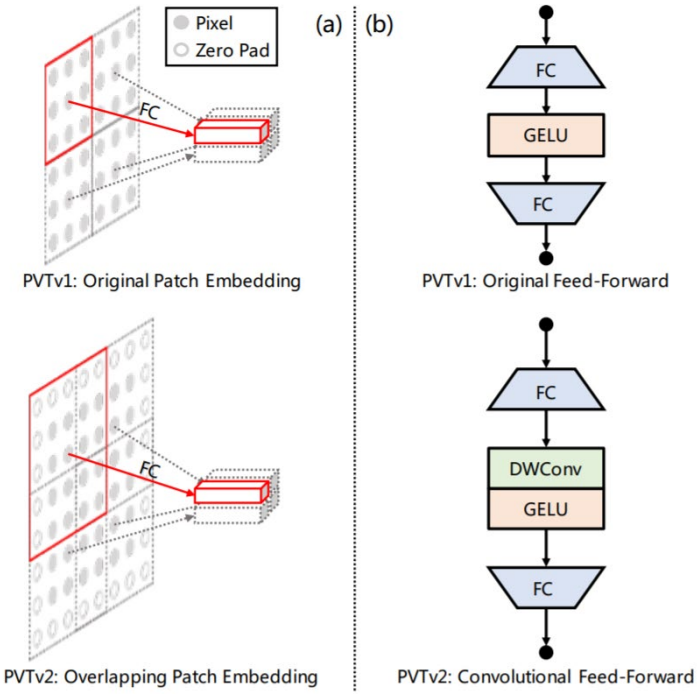
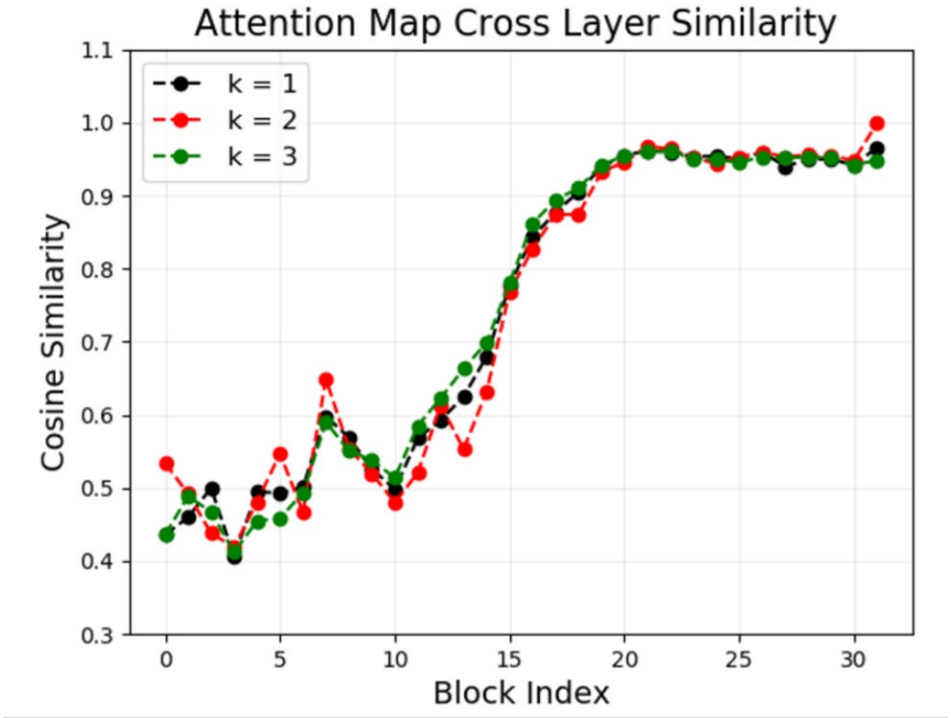


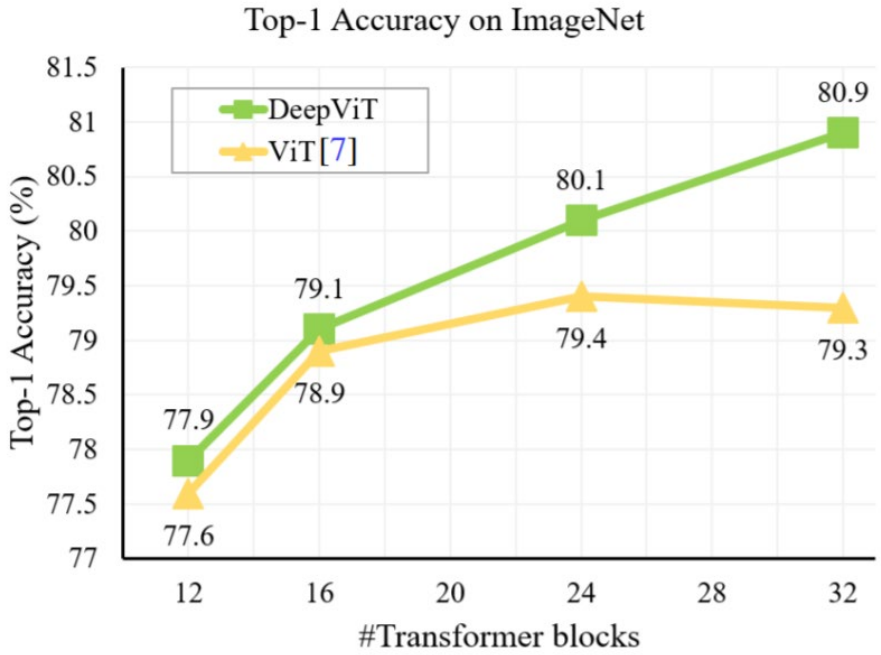
Figure 1: **Two improvements in PVTv2.** (1) Overlapping Patch Embedding. (2) Convolutional Feed Forward Network.

PVTv2: Improved Baselines with Pyramid Vision Transformer, Arxiv 2021.

加深的ViT

- 如果暴力增加ViT的深度，我们会发现效果并不好，会出现严重的Attention collapse现象。需要设计新结构来解决。





DeepViT

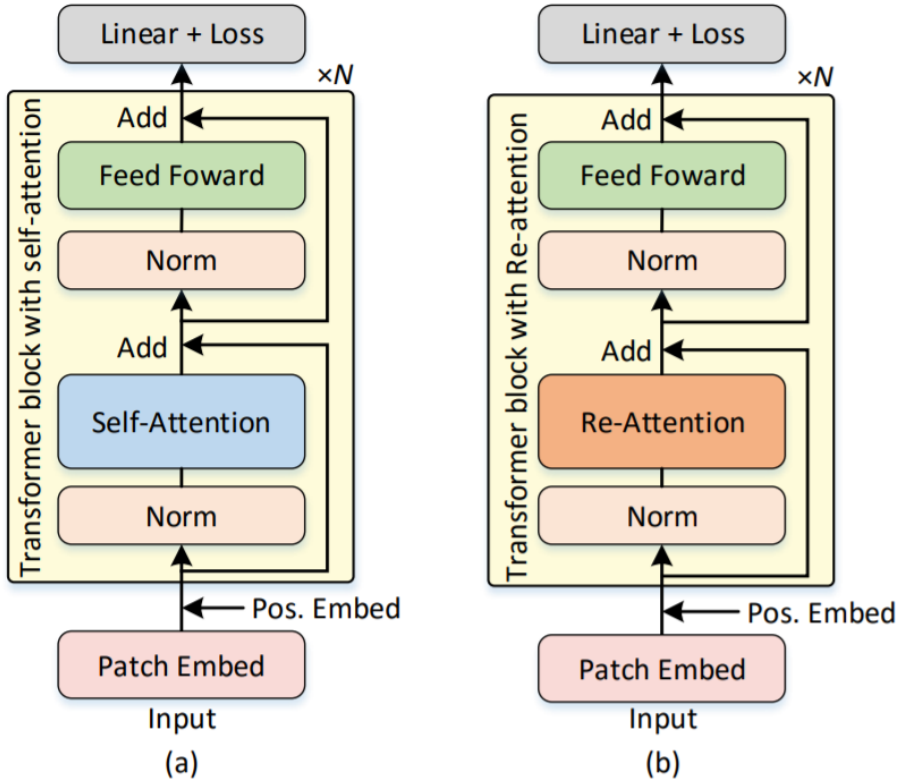


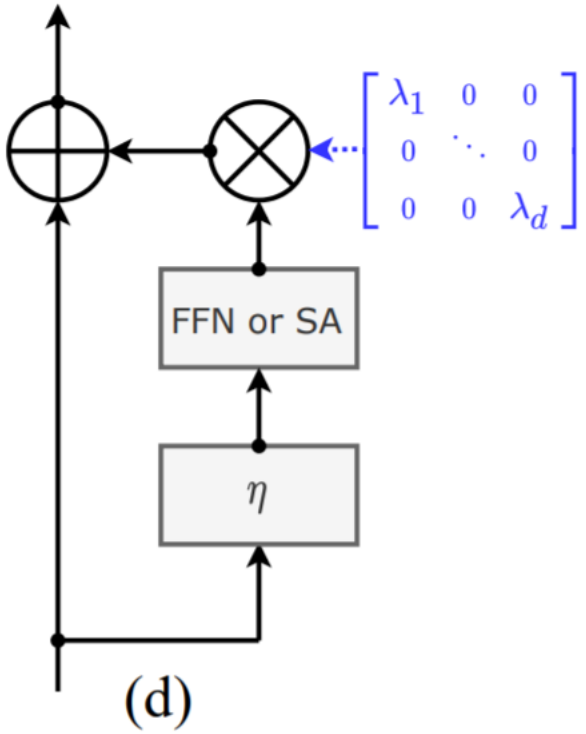
Figure 2: Comparison between the (a) original ViT with N transformer blocks and (b) our proposed DeepViT model. Different from ViT, DeepViT replaces the self-attention layer within the transformer block with the proposed Re-attention which effectively addresses the attention collapse issue and enables training deeper ViTs. More details are given in Sec. 4.2.

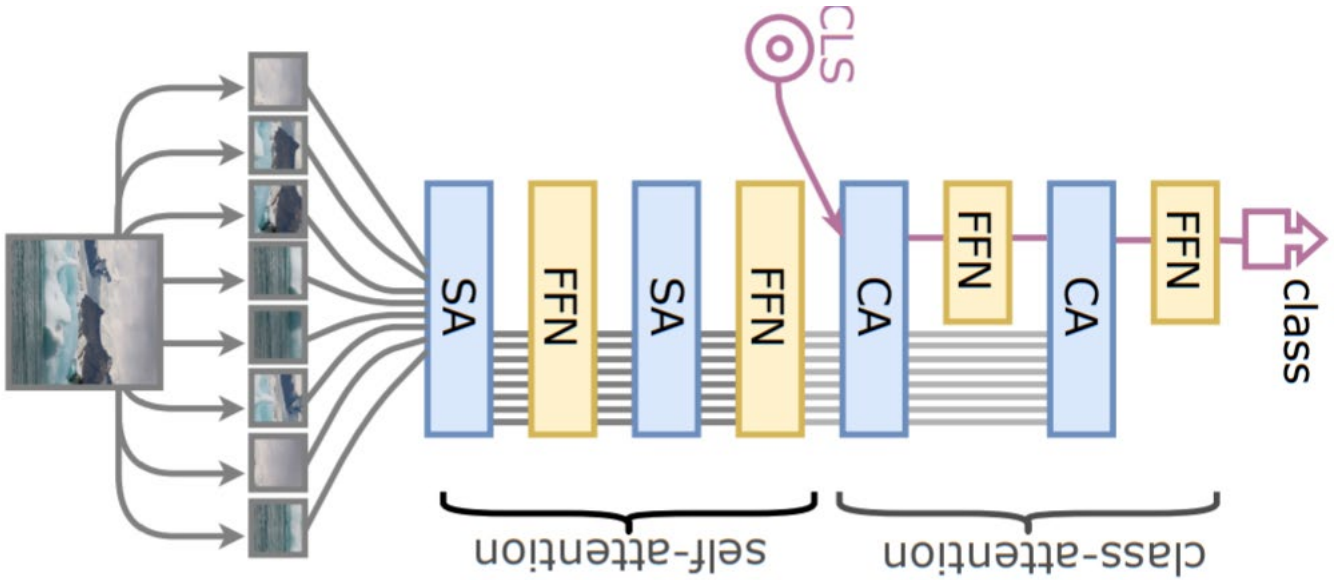
$$\text{Re-Attention}(Q, K, V) = \text{Norm}(\Theta^{\top}(\text{Softmax}(\frac{QK^{\top}}{\sqrt{d}})))V,$$

DeepViT: Towards Deeper Vision Transformer, Arxiv 2021.

CaiT

- 1.引入了一个类似于channel reweighting的方法
- 2.将class token的更新和patch token的更新分离开





Going deeper with Image Transformers, Arxiv 2021.

XCiT

- 不构造spatial self-attention
- 改成channel self-attention（空间信息交互用depthwise conv完成）

XCiT: Cross-Covariance Image Transformers, Arxiv 2021.

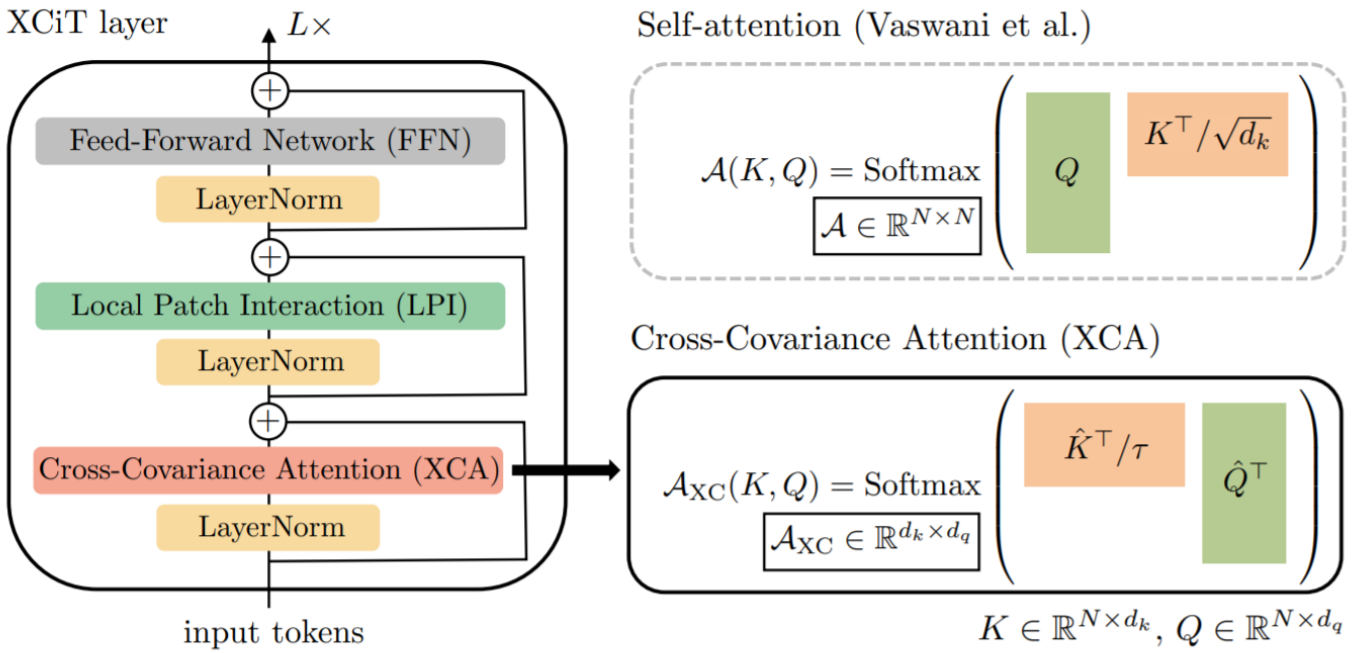


Figure 1: Our XCiT layer consists of three main blocks, each preceded by LayerNorm and followed by a residual connection: (i) the core cross-covariance attention (XCA) operation, (ii) the local patch interaction (LPI) module, and (iii) a feed-forward network (FFN). By transposing the query-key interaction, the computational complexity of XCA is linear in the number of data elements N, rather than quadratic as in conventional self-attention.

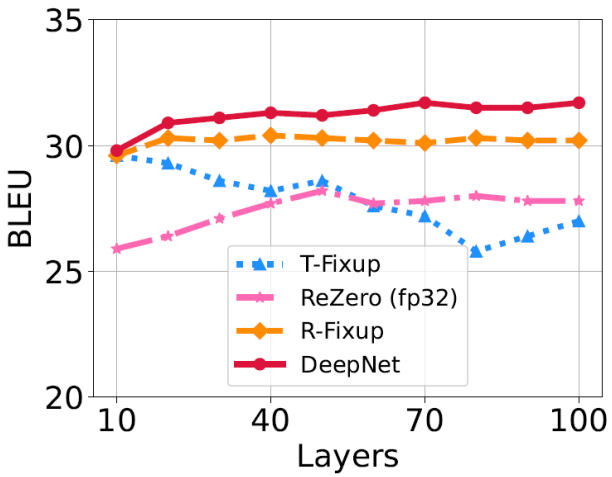
DeepNet

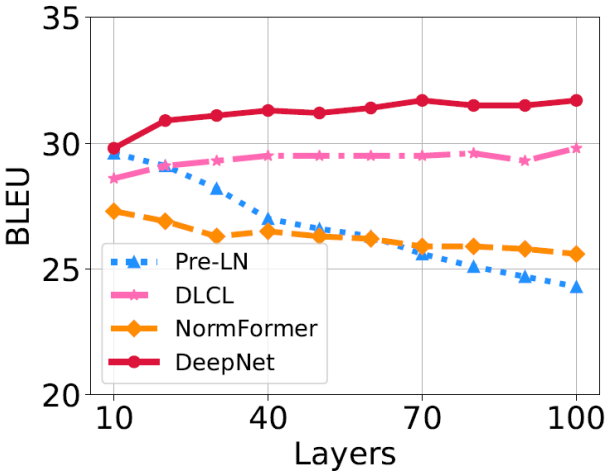
- PostNorm

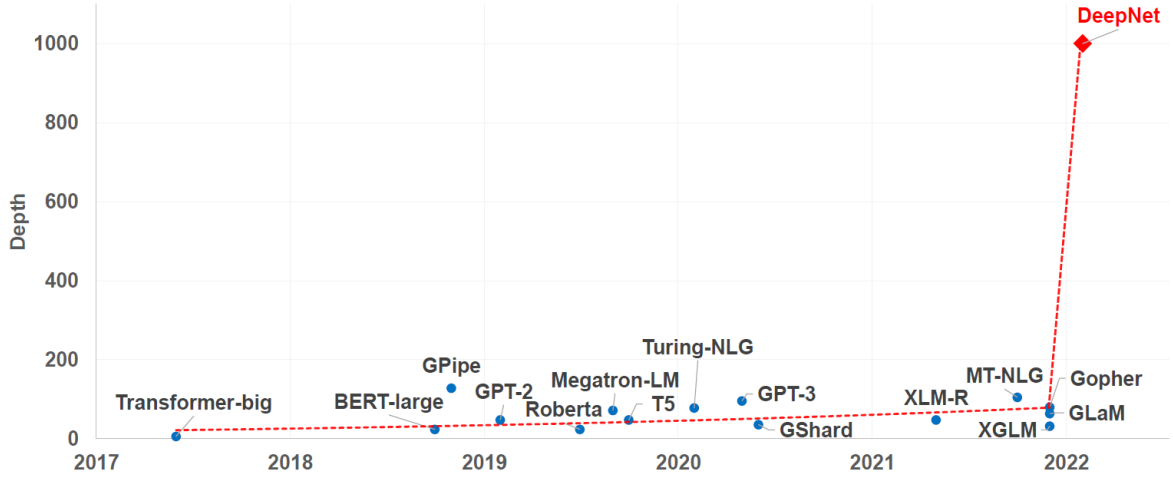
$$\boldsymbol{x}_{\ell+1} = \text{LayerNorm}(\boldsymbol{x}_\ell + F_\ell(\boldsymbol{x}_\ell))$$

- DeepNorm

$$x_{l+1} = LN(\alpha x_l + G_l(x_l, \theta_l))$$







Transformers without tears: Improving the normalization of self-attention, Arxiv 2019

DeepNet: Scaling Transformers to 1,000 Layers, Arxiv 2022.

卷积+Attention

- 传统的ResNet based结构，替换部分bottleneck block为带Attention的结构
- 目的还是强化全局信息交互

Early Convolutions Help Transformers See Better

model	flops (B)	params (M)	acts (M)	time (min)	batch size	epochs 100	epochs 200	epochs 400	IN 21k
ResNet-50	4.1	25.6	11.3	3.4	2048	22.5	21.2	20.7	21.6
ResNet-101	7.8	44.5	16.4	5.5	2048	20.3	19.1	18.5	19.2
ResNet-152	11.5	60.2	22.8	7.7	2048	19.5	18.4	17.7	18.2
ResNet-200	15.0	64.7	32.3	10.7	1024	19.5	18.3	17.6	17.7
RegNetY-1GF	1.0	9.6	6.2	3.1	2048	23.2	22.2	21.5	-
RegNetY-4GF	4.1	22.4	14.5	7.6	2048	19.4	18.3	17.9	18.4
RegNetY-16GF	15.5	72.3	30.7	17.9	1024	17.1	16.4	16.3	15.6
RegNetY-32GF	31.1	128.6	46.2	35.1	512	16.2	15.9	15.9	15.0
RegNetZ-1GF	1.0	11.0	8.8	4.2	2048	20.8	20.2	19.6	-
RegNetZ-4GF	4.0	28.1	24.3	12.9	1024	17.4	16.9	16.6	-
RegNetZ-16GF	16.0	95.3	51.3	32.0	512	16.0	15.9	15.9	-
RegNetZ-32GF	32.0	175.1	79.6	55.3	256	16.3	16.2	16.1	-

model	flops (B)	params (M)	acts (M)	time (min)	batch size	epochs 100	epochs 200	epochs 400	IN 21k
EffNet-B2	1.0	9.1	13.8	5.9	2048	21.4	20.5	19.9	-
EffNet-B4	4.4	19.3	49.5	19.4	512	18.5	17.8	17.5	-
EffNet-B5	10.3	30.4	98.9	41.7	256	17.3	17.0	17.0	-
ViT$_P$-1GF	1.1	4.8	5.5	2.6	2048	33.2	29.7	27.7	-
ViT$_P$-4GF	3.9	18.5	11.1	3.8	2048	23.3	20.8	19.6	20.6
ViT$_P$-18GF	17.5	86.6	24.0	11.5	1024	19.9	18.4	17.9	16.4
ViT$_P$-36GF	35.9	178.4	37.3	18.8	512	19.9	18.8	18.2	15.1
ViT$_C$-1GF	1.1	4.6	5.7	2.7	2048	28.6	26.1	24.7	-
ViT$_C$-4GF	4.0	17.8	11.3	3.9	2048	20.9	19.2	18.6	18.8
ViT$_C$-18GF	17.7	81.6	24.1	11.4	1024	18.4	17.5	17.0	15.1
ViT$_C$-36GF	35.0	167.8	36.7	18.6	512	18.3	17.6	16.8	14.2

Table 2: **Peak performance (grouped by model family)**: Model complexity and validation top-1 error at 100, 200, and 400 epoch schedules on ImageNet-1k, and the top-1 error after pretraining on ImageNet-21k (IN 21k) and fine-tuning on ImageNet-1k. This table serves as reference for the results shown in Figure 6. Blue numbers: best model trainable under 20 minutes per ImageNet-1k epoch. Batch sizes and training times are reported normalized to 8 32GB Volta GPUs (see Appendix).

Early Convolutions Help Transformers See Better, Arxiv 2021

BoTNet

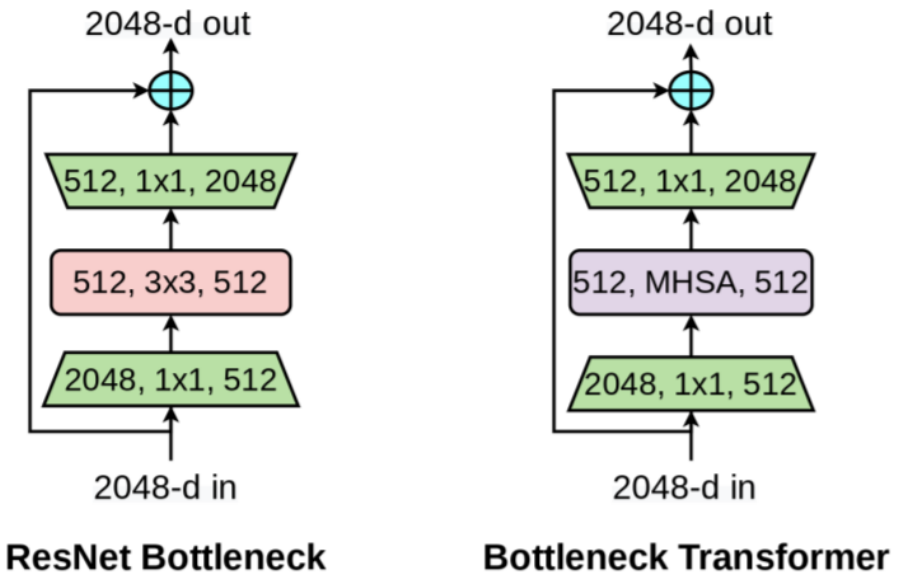


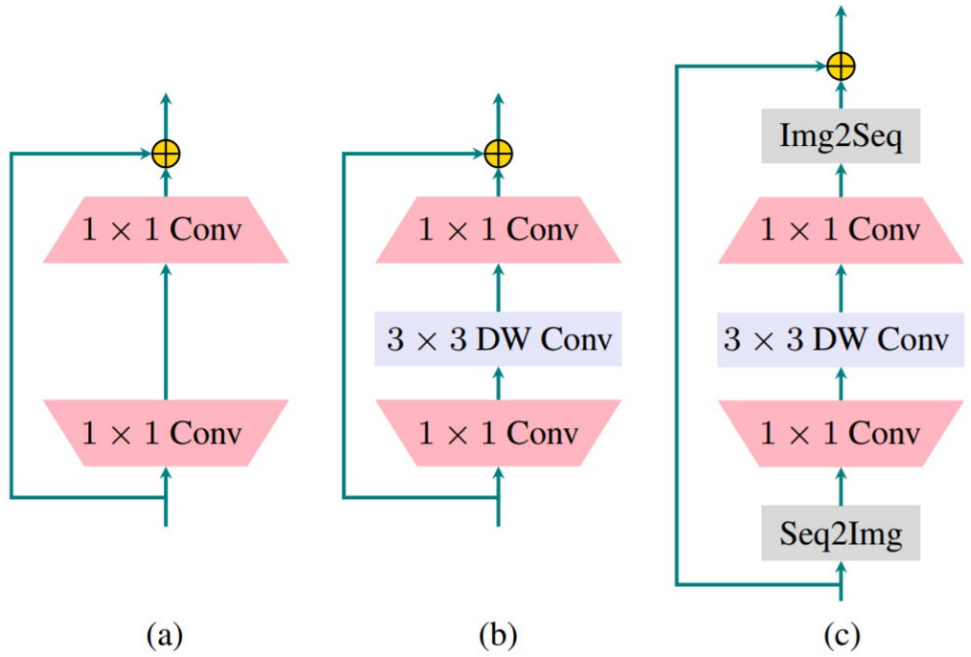
Figure 1: **Left:** A ResNet Bottleneck Block, **Right:** A Bottleneck Transformer (BoT) block. The only difference is the replacement of the spatial 3×3 convolution layer with Multi-Head Self-Attention (MHSA). The structure of the self-attention layer is described in Figure 4.

Bottleneck Transformers for Visual Recognition, Arxiv 2021.

LocalViT

- 将FFN替换成depthwise conv

Network	γ	DW	Params (M)	FLOPs (G)	Top-1 Acc. (%)
DeiT-T [41]	4	No	5.7	1.3	72.2
LocalViT-T	4	No	5.7	1.3	72.5 (0.3↑)
LocalViT-T*	4	Yes	5.8	1.3	73.7 (1.5↑)
DeiT-T [41]	6	No	7.5	1.6	73.1†
LocalViT-T	6	No	7.5	1.6	74.3 (1.2↑)
LocalViT-T*	6	Yes	7.7	1.6	76.1 (3.0↑)



LocalViT: Bringing Locality to Vision Transformers, Arxiv 2021.

邻域信息上的Attention

- 我们强制transformer只关注图像每个邻域空间上的信息交互，充分利用图像的邻域相关性

- 相比卷积+Attention，空间信息的权重在不同的邻域空间上是动态的

Swin Transformer

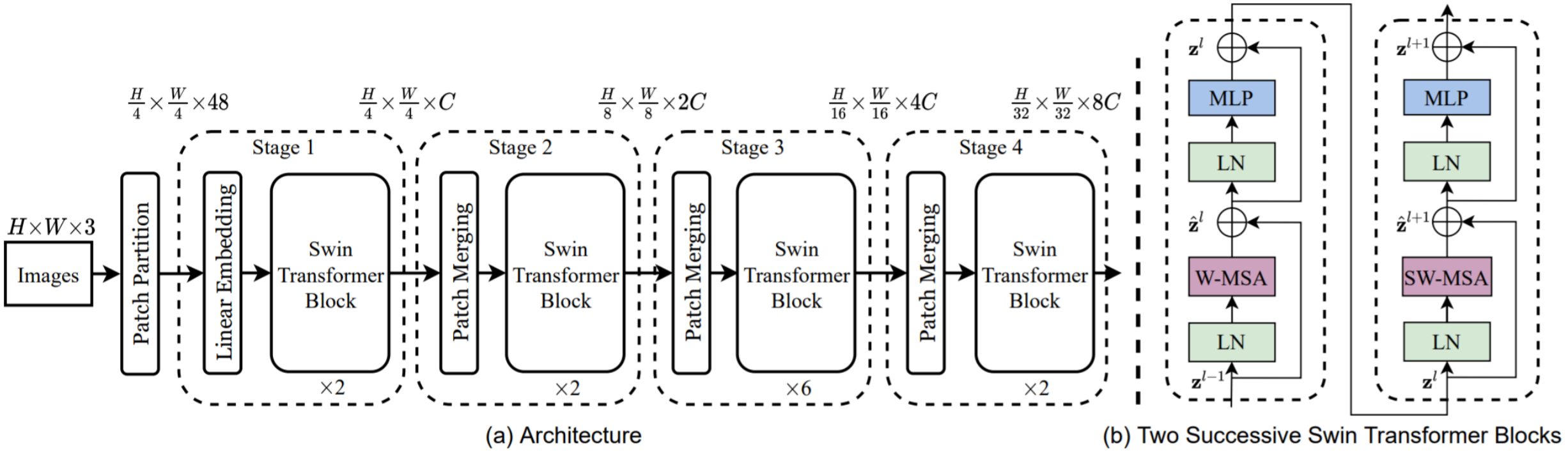


Figure 3. (a) The architecture of a Swin Transformer (Swin-T); (b) two successive Swin Transformer Blocks (notation presented with Eq. (3)). W-MSA and SW-MSA are multi-head self attention modules with regular and shifted windowing configurations, respectively.

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows, ICCV 2021

Swin Transformer

- Shifted Window Attention

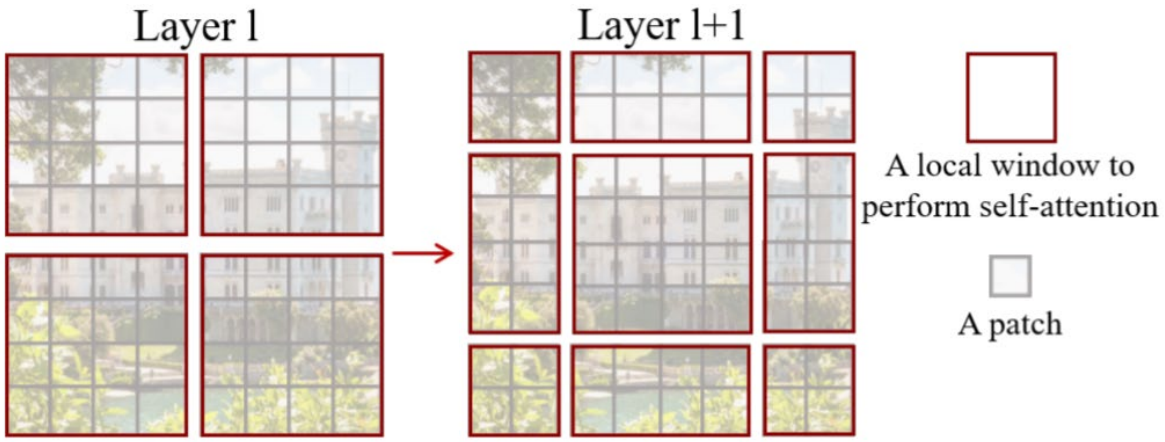


Figure 2. An illustration of the *shifted window* approach for computing self-attention in the proposed Swin Transformer architecture. In layer l (left), a regular window partitioning scheme is adopted, and self-attention is computed within each window. In the next layer $l + 1$ (right), the window partitioning is shifted, resulting in new windows. The self-attention computation in the new windows crosses the boundaries of the previous windows in layer l, providing connections among them.

Performance

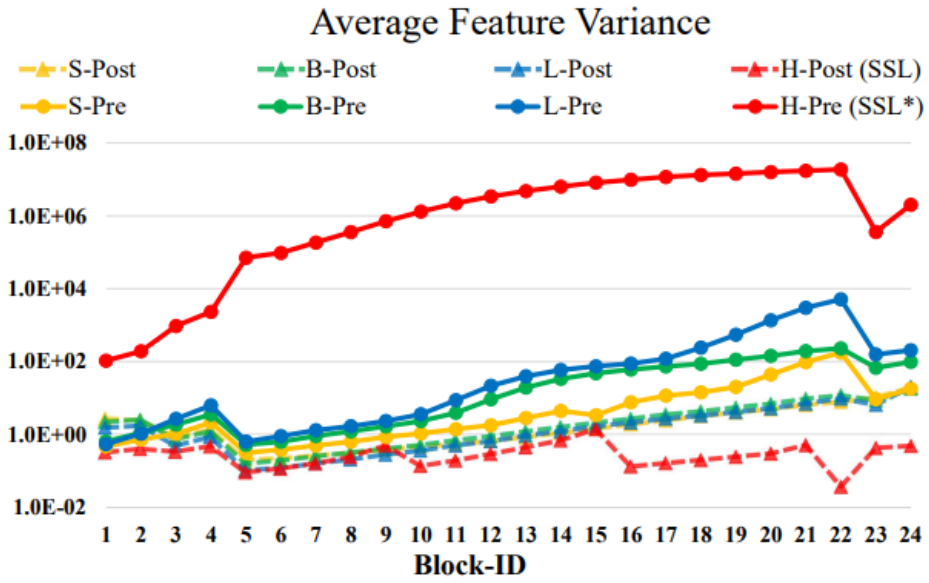
(a) Regular ImageNet-1K trained models					
method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
RegNetY-4G [48]	224^2	21M	4.0G	1156.7	80.0
RegNetY-8G [48]	224^2	39M	8.0G	591.6	81.7
RegNetY-16G [48]	224^2	84M	16.0G	334.7	82.9
EffNet-B3 [58]	300^2	12M	1.8G	732.1	81.6
EffNet-B4 [58]	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

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
MLP-Mixer-B/16 [61]	224^2	59M	12.7G	-	76.4
ResMLP-S24 [62]	224^2	30M	6.0G	715	79.4
ResMLP-B24 [62]	224^2	116M	23.0G	231	81.0
Swin-T/D24 (Transformer)	256^2	28M	5.9G	563	81.6
Swin-Mixer-T/D24	256^2	20M	4.0G	807	79.4
Swin-Mixer-T/D12	256^2	21M	4.0G	792	79.6
Swin-Mixer-T/D6	256^2	23M	4.0G	766	79.7
Swin-Mixer-B/D24 (no shift)	224^2	61M	10.4G	409	80.3
Swin-Mixer-B/D24	224^2	61M	10.4G	409	81.3

Table 10. Performance of Swin MLP-Mixer on ImageNet-1K classification. D indictes the number of channels per head. Throughput is measured using the GitHub repository of [68] and a V100 GPU, following [63].

Swin Transformer V2

- Architecture
- Post-Norm



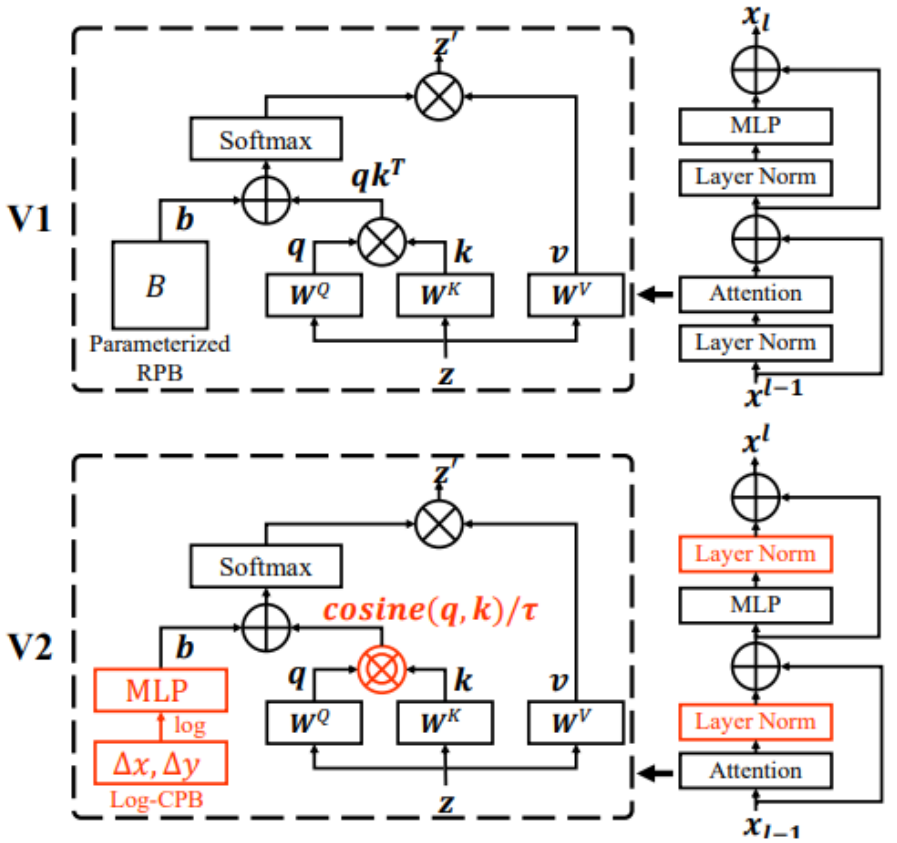


Figure 1. In order to better scale model capacity and window resolution, several adaptions are made on the original Swin Transformer architecture (V1): 1) a *post-norm* to replace the previous *pre-norm* configuration; 2) a *scaled cosine attention* to replace the original *dot product attention*; 3) a *log-spaced continuous* relative position bias approach to replace the previous *parameterized* approach. Adaptions 1) and 2) make the model easier to be scaled up in capacity. Adaption 3) makes the model more effectively transferred across window resolutions. The adapted architecture is named Swin Transformer V2.

多尺度融合

- CoaT

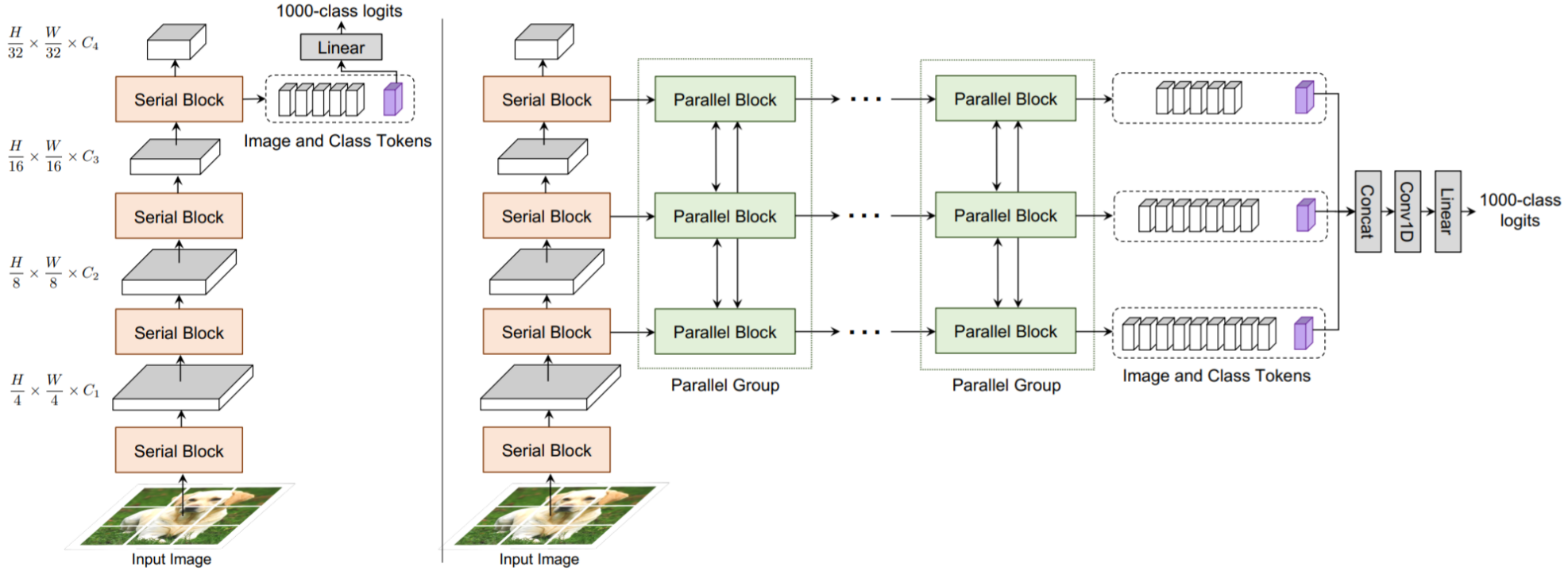
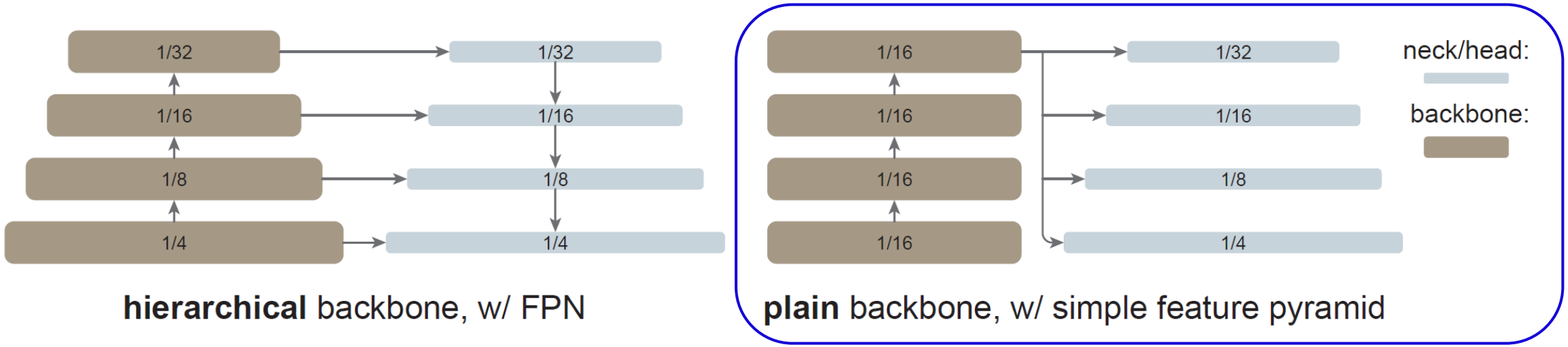


Figure 3. **CoaT model architecture.** (Left) The overall network architecture of **CoaT-Lite**. CoaT-Lite consists of serial blocks only, where image features are down-sampled and processed in a sequential order. (Right) The overall network architecture of **CoaT**. CoaT consists of serial blocks and parallel blocks. Both blocks enable the co-scale mechanism.

Co-Scale Conv-Attentional Image Transformers, Arxiv 2021.

Feature Pyramid from A Single-Scale Map



pyramid design	ViT-B AP^{box}	ViT-B AP^{mask}	ViT-L AP^{box}	ViT-L AP^{mask}
no feature pyramid	47.8	42.5	51.2	45.4
(a) FPN, 4-stage	50.3 (+2.5)	44.9 (+2.4)	54.4 (+3.2)	48.4 (+3.0)
(b) FPN, last-map	50.9 (+3.1)	45.3 (+2.8)	**54.6** (+3.4)	48.5 (+3.1)
(c) simple feature pyramid	**51.2** (+3.4)	**45.5** (+3.0)	**54.6** (+3.4)	**48.6** (+3.2)

Exploring Plain Vision Transformer Backbones for Object Detection, Arxiv 2022.

Other Network Architecture

- MLP-Mixer

MLP-Mixer: An all-MLP Architecture for Vision, Arxiv 2021.

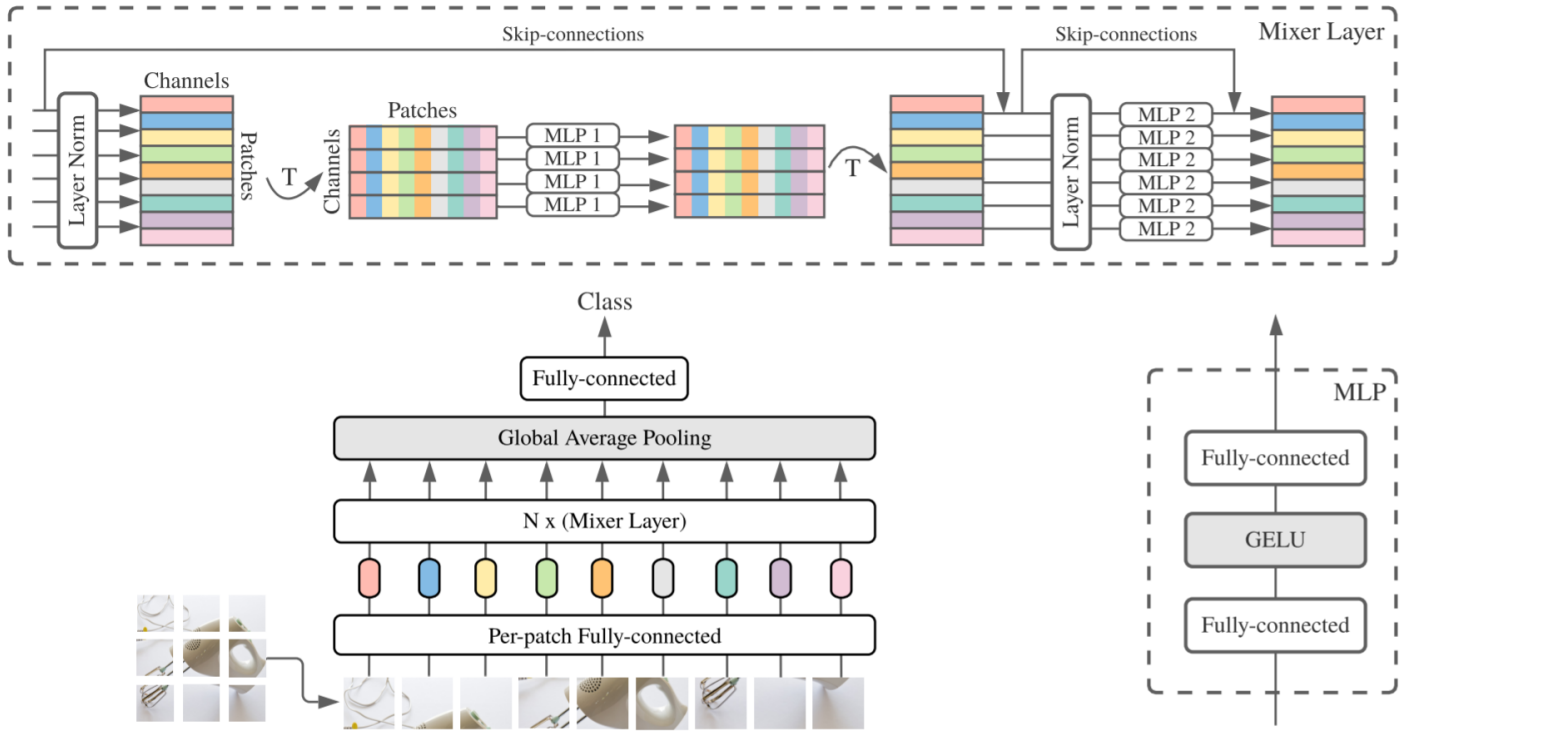


Figure 1: MLP-Mixer consists of per-patch linear embeddings, Mixer layers, and a classifier head. Mixer layers contain one token-mixing MLP and one channel-mixing MLP, each consisting of two fully-connected layers and a GELU nonlinearity. Other components include: skip-connections, dropout, and layer norm on the channels.

Content

- CNN -> Vision Transformers

- Vision Transformers -> CNN

Why Deep CNNs （vs Transformer）

- Large Kernel
- Layer Norm
- Nonlinear Activations
- Optimization: AdamW、LAMB

- ConvNeXts compete favorably with Transformers in terms of accuracy and scalability

A ConvNet for the 2020s, Arxiv 2022 (Facebook)
Scaling Up Your Kernels to 31x31: Revisiting Large Kernel Design in CNNs , Arxiv 2022 (旷视)
Simple Baselines for Image Restoration, Arxiv 2022 (旷视)
Visual Attention Network, Arxiv 2022 (清华)

ConvNeXt: Training Techniques

- Begin with ResNet-50/200
- 90 -> 300 epochs
- AdamW
- Data augmentation
- Regularization schemes

- 76.1% -> 78.8% (ResNet-50)

(pre-)training config	ConvNeXt-T/S/B/L ImageNet-1K 224^2	ConvNeXt-B/L/XL ImageNet-22K 224^2
optimizer	AdamW	AdamW
base learning rate	4e-3	4e-3
weight decay	0.05	0.05
optimizer momentum	β_1, β_2=0.9, 0.999	β_1, β_2=0.9, 0.999
batch size	4096	4096
training epochs	300	90
learning rate schedule	cosine decay	cosine decay
warmup epochs	20	5
warmup schedule	linear	linear
layer-wise lr decay [6, 10]	None	None
randaugment [12]	(9, 0.5)	(9, 0.5)
label smoothing [65]	0.1	0.1
mixup [85]	0.8	0.8
cutmix [84]	1.0	1.0
stochastic depth [34]	0.1/0.4/0.5/0.5	0.1/0.1/0.2
layer scale [69]	1e-6	1e-6
gradient clip	None	None
exp. mov. avg. (EMA) [48]	0.9999	None

A ConvNet for the 2020s, Arxiv 2022

ConvNeXt: Macro Design

- Number of blocks
  - (3, 4, 6, 3) -> (3, 3, 9, s3) (ResNet-50)
  - 78.8% -> 79.4%
- Patchify
  - 7x7 convolution with stride 2 following by max pooling -> 4x4, stride 4 convolution
  - 79.4% -> 79.5%

ConvNeXt: ResNeXtify and Inverted Bottleneck

- ResNeXtify
  - Depthwise conv
  - Increase the network width
  - -> 80.5%
- Inverted Bottleneck
  - 80.5% to 80.6%

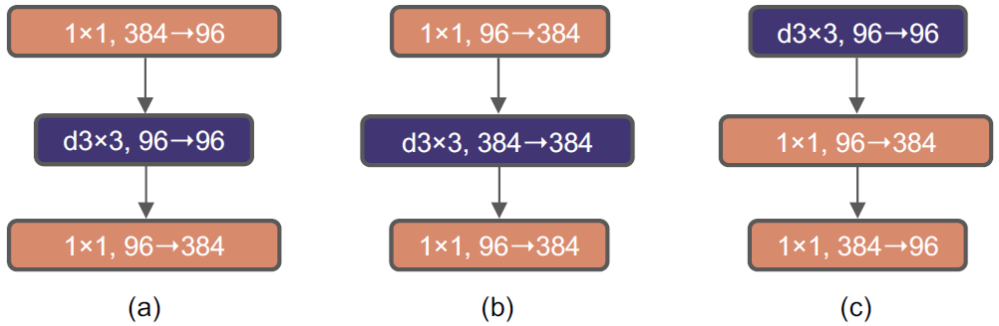
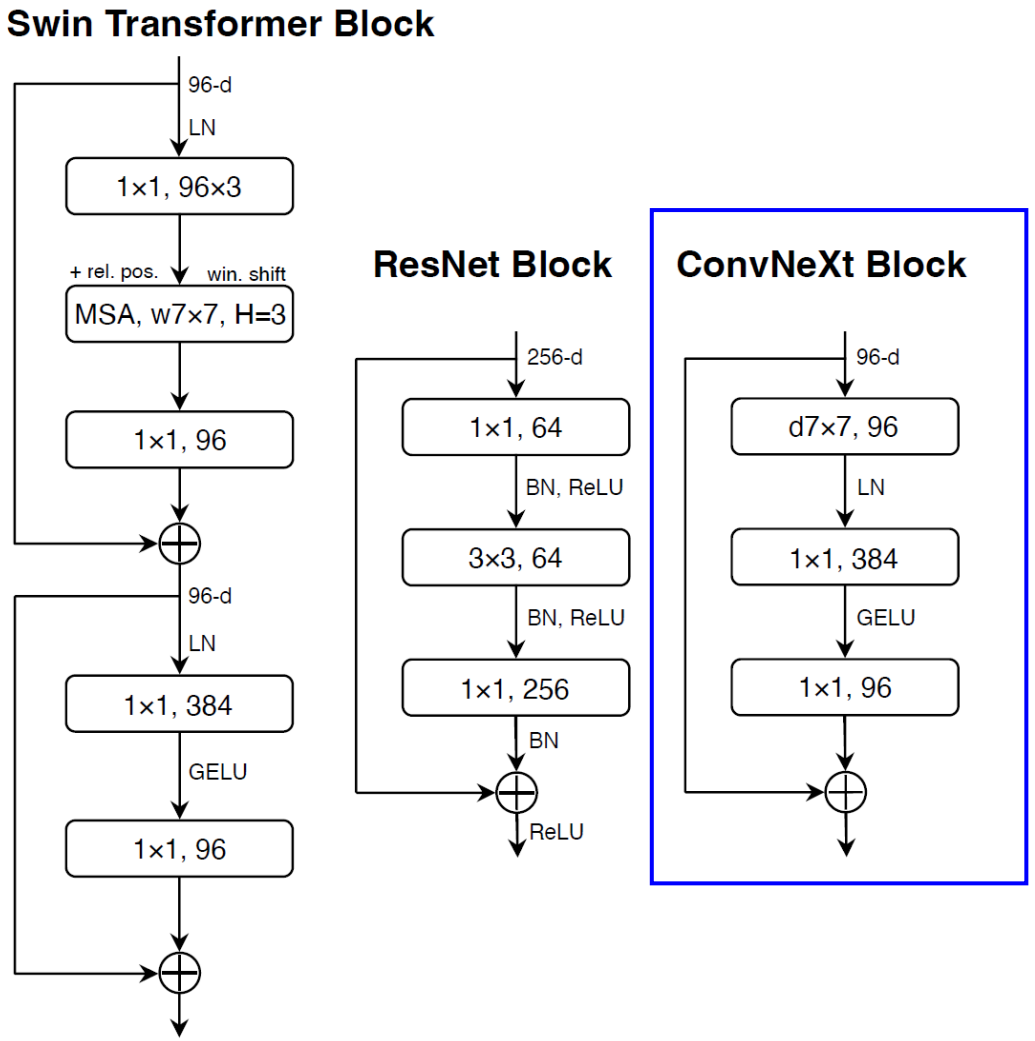


Figure 3. **Block modifications and resulted specifications. (a)** is a ResNeXt block; in **(b)** we create an inverted bottleneck block and in **(c)** the position of the spatial depthwise conv layer is moved up.

ConvNeXt: Large Kernel Sizes and Micro Design

- Large Kernel Sizes
  - 3x3 -> 7x7
  - -> 80.6%
- Micro Design
  - Replacing ReLU with GELU: 80.6%
  - A single GELU activation in each block: 81.3%
  - Remove two BN layers, leaving only one BN layer before 1 x 1 conv layers: 81.4%
  - Substituting BN with LN: 81.5%
  - Separate downsampling layers: 82.0%

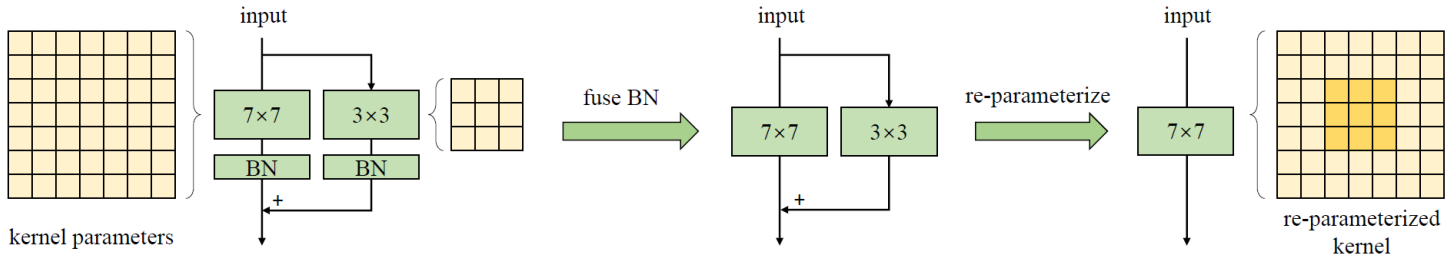


ConvNeXt

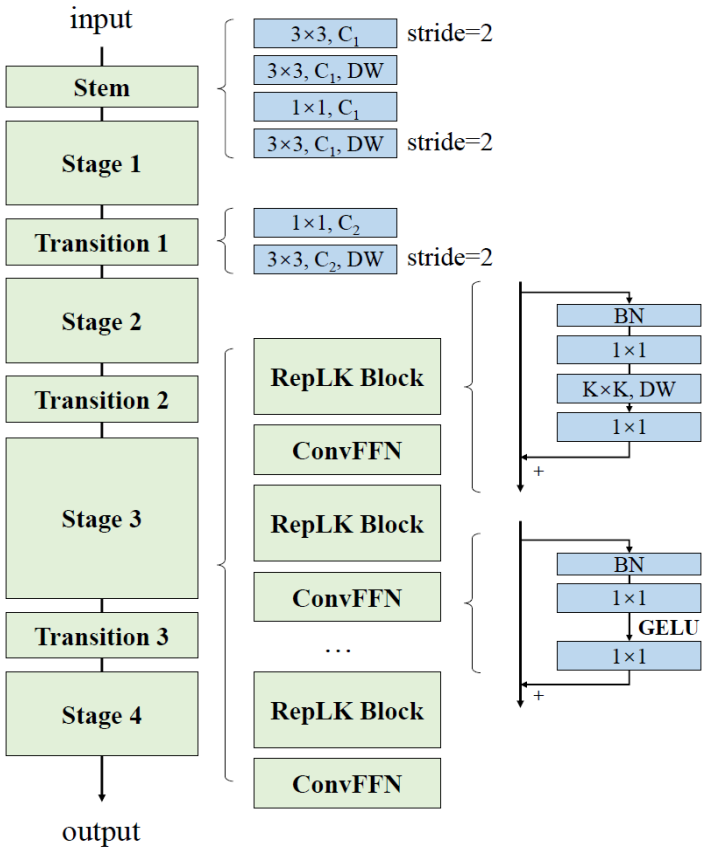
model	image size	#param.	FLOPs	throughput (image / s)	IN-1K top-1 acc.
ImageNet-1K trained models					
● RegNetY-4G [51]	224^2	21M	4.0G	1156.7	80.0
● RegNetY-8G [51]	224^2	39M	8.0G	591.6	81.7
● RegNetY-16G [51]	224^2	84M	16.0G	334.7	82.9
● EffNet-B3 [67]	300^2	12M	1.8G	732.1	81.6
● EffNet-B4 [67]	380^2	19M	4.2G	349.4	82.9
● EffNet-B5 [67]	456^2	30M	9.9G	169.1	83.6
● EffNet-B6 [67]	528^2	43M	19.0G	96.9	84.0
● EffNet-B7 [67]	600^2	66M	37.0G	55.1	84.3
○ DeiT-S [68]	224^2	22M	4.6G	978.5	79.8
○ DeiT-B [68]	224^2	87M	17.6G	302.1	81.8
○ Swin-T	224^2	28M	4.5G	757.9	81.3
● ConvNeXt-T	224^2	29M	4.5G	774.7	**82.1**
○ Swin-S	224^2	50M	8.7G	436.7	83.0
● ConvNeXt-S	224^2	50M	8.7G	447.1	**83.1**
○ Swin-B	224^2	88M	15.4G	286.6	83.5
● ConvNeXt-B	224^2	89M	15.4G	292.1	**83.8**
○ Swin-B	384^2	88M	47.1G	85.1	84.5
● ConvNeXt-B	384^2	89M	45.0G	95.7	**85.1**
● ConvNeXt-L	224^2	198M	34.4G	146.8	**84.3**
● ConvNeXt-L	384^2	198M	101.0G	50.4	**85.5**

ImageNet-22K pre-trained models					
● R-101x3 [36]	384^2	388M	204.6G	-	84.4
● R-152x4 [36]	480^2	937M	840.5G	-	85.4
○ ViT-B/16 [18]	384^2	87M	55.5G	93.1	84.0
○ ViT-L/16 [18]	384^2	305M	191.1G	28.5	85.2
○ Swin-B	224^2	88M	15.4G	286.6	85.2
● ConvNeXt-B	224^2	89M	15.4G	292.1	**85.8**
○ Swin-B	384^2	88M	47.0G	85.1	86.4
● ConvNeXt-B	384^2	89M	45.1G	95.7	**86.8**
○ Swin-L	224^2	197M	34.5G	145.0	86.3
● ConvNeXt-L	224^2	198M	34.4G	146.8	**86.6**
○ Swin-L	384^2	197M	103.9G	46.0	87.3
● ConvNeXt-L	384^2	198M	101.0G	50.4	**87.5**
● ConvNeXt-XL	224^2	350M	60.9G	89.3	**87.0**
● ConvNeXt-XL	384^2	350M	179.0G	30.2	**87.8**

RepLKNet



- Re-parameterized large depth-wise convolutions
  - large depth-wise convolutions
  - identity shortcut is vital
  - re-parameterizing
  - large convolutions boost downstream tasks much more
  - large kernel is useful even on small feature maps



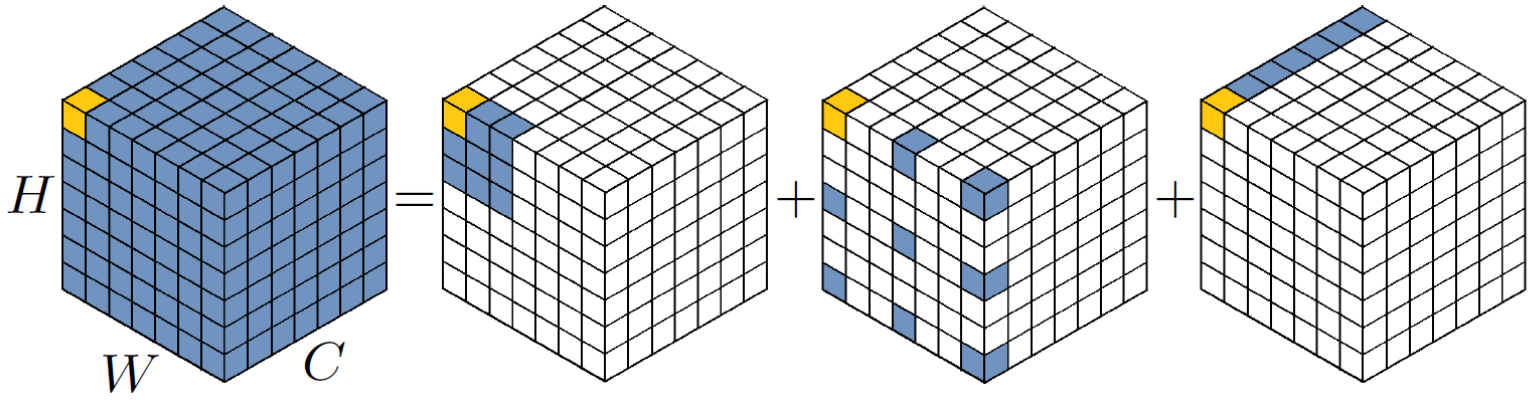
Scaling Up Your Kernels to 31x31: Revisiting Large Kernel Design in CNNs, Arxiv 2022

RepLKNet

Model	Input resolution	Top-1 acc	Params (M)	FLOPs (G)	Throughput examples/s
RepLKNet-31B	224×224	83.5	79	15.3	295.5
Swin-B	224×224	83.5	88	15.4	226.2
RepLKNet-31B	384×384	84.8	79	45.1	97.0
Swin-B	384×384	84.5	88	47.0	67.9
RepLKNet-31B ‡	224×224	85.2	-	-	-
Swin-B ‡	224×224	85.2	-	-	-
RepLKNet-31B ‡	384×384	86.0	-	-	-
Swin-B ‡	384×384	86.4	-	-	-
RepLKNet-31L ‡	384×384	86.6	172	96.0	50.2
Swin-L ‡	384×384	87.3	197	103.9	36.2
RepLKNet-XL ◇	320×320	87.8	335	128.7	39.1

VAN: Visual Attention Network

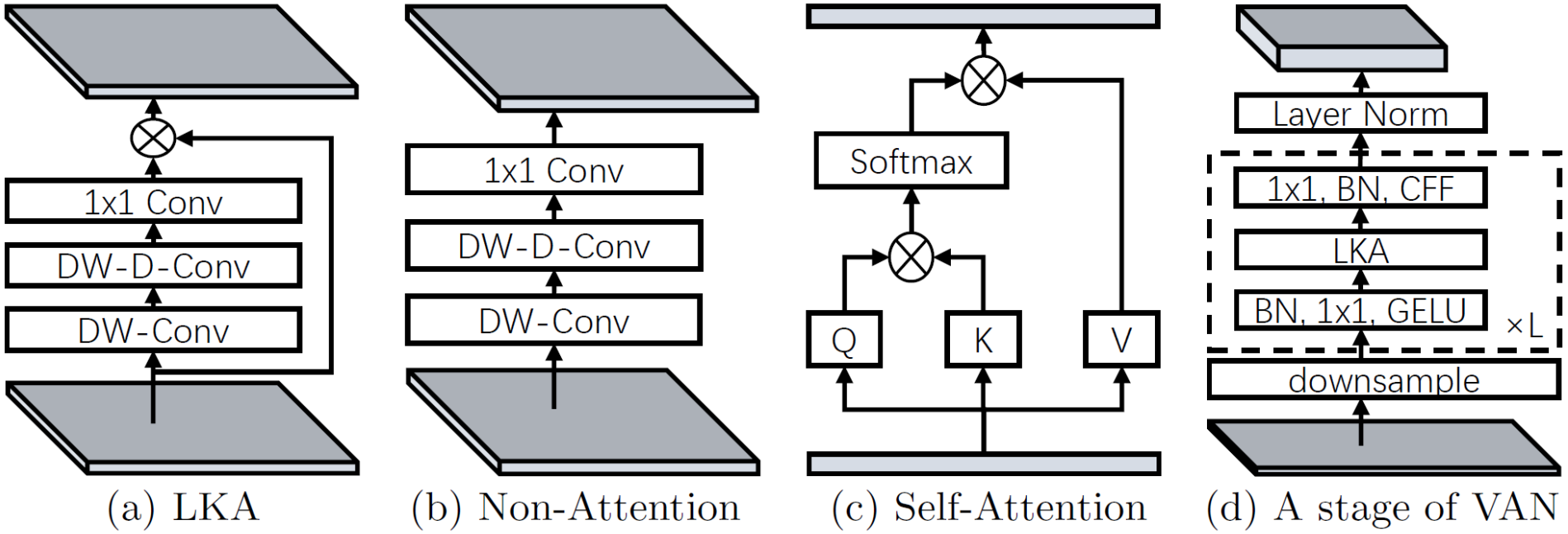
- depth-wise convolution (DW-Conv)
- Depthwise dilation convolution (DW-D-Conv)
- pointwise convolution



Visual Attention Network, Arxiv 2022

VAN: Visual Attention Network

- CFF : convolutional feed-forward network



(a) LKA (b) Non-Attention (c) Self-Attention (d) A stage of VAN

VAN: Visual Attention Network

Method	Params. (M)	GFLOPs	Top-1 Acc (%)
PVTv2-B0 [84]	3.4	0.6	70.5
T2T-ViT-7 [97]	4.3	1.1	71.7
DeiT-Tiny/16 [76]	5.7	1.3	72.2
TNT-Ti [28]	6.1	1.4	73.9
VAN-Tiny	4.1	0.9	**75.4**
ResNet18 [31]	11.7	1.8	69.8
PVT-Tiny [85]	13.2	1.9	75.1
PoolFormer-S12 [96]	11.9	2.0	77.2
PVTv2-B1 [84]	13.1	2.1	78.7
VAN-Small	13.9	2.5	**81.1**
ResNet50 [31]	25.6	4.1	76.5
ResNeXt50-32x4d [92]	25.0	4.3	77.6
RegNetY-4G [63]	21.0	4.0	80.0
DeiT-Small/16 [76]	22.1	4.6	79.8
T2T-ViT$_t$-14 [97]	21.5	6.1	81.7
PVT-Small [85]	24.5	3.8	79.8
TNT-S [28]	23.8	5.2	81.3
ResMLP-24 [75]	30.0	6.0	79.4
gMLP-S [48]	20.0	4.5	79.6
Swin-T [54]	28.3	4.5	81.3
PoolFormer-S24 [96]	21.4	3.6	80.3
Twins-SVT-S [14]	24.0	2.8	81.7
PVTv2-B2 [84]	25.4	4.0	82.0
Focal-T [94]	29.1	4.9	82.2
ConvNeXt-T [55]	28.6	4.5	82.1
VAN-Base	26.6	5.0	**82.8**
ResNet101 [31]	44.7	7.9	77.4
ResNeXt101-32x4d [92]	44.2	8.0	78.8
Mixer-B/16 [74]	59.0	11.6	76.4
T2T-ViT$_t$-19 [97]	39.2	9.8	82.4
PVT-Medium [85]	44.2	6.7	81.2
Swin-S [54]	49.6	8.7	83.0
ConvNeXt-S [54]	50.1	8.7	83.1
PVTv2-B3 [84]	45.2	6.9	83.2
Focal-S [94]	51.1	9.1	83.5
VAN-Large	44.8	9.0	**83.9**
ResNet152 [31]	60.2	11.6	78.3
T2T-ViT$_t$-24 [97]	64.0	15.0	82.3
PVT-Large [85]	61.4	9.8	81.7
TNT-B [28]	66.0	14.1	82.8
PVTv2-B4 [84]	62.6	10.1	83.6
VAN-Huge	60.3	12.2	**84.2**

Summary

- CNN -> Transformer -> ?
- Transformer -> CNN?
- Network Design, Training Scheme, …
- What are you needed? Chip, Task, …

- 掌握近年来的主流卷积操作和网络结构
- 结合实际问题和任务灵活使用
- 持续了解和跟进研究进展
- 自己能有一些思考、拓展和突破
- 深度学习的可解释性 (可信赖性)