A ConvNet for the 2020s

Authors: Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, Saining Xie

Facebook AI Research (FAIR)

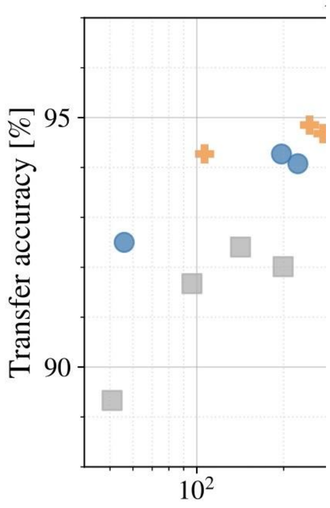
Submitted: Jan 10, 2022; Last Revised: March 2, 2022

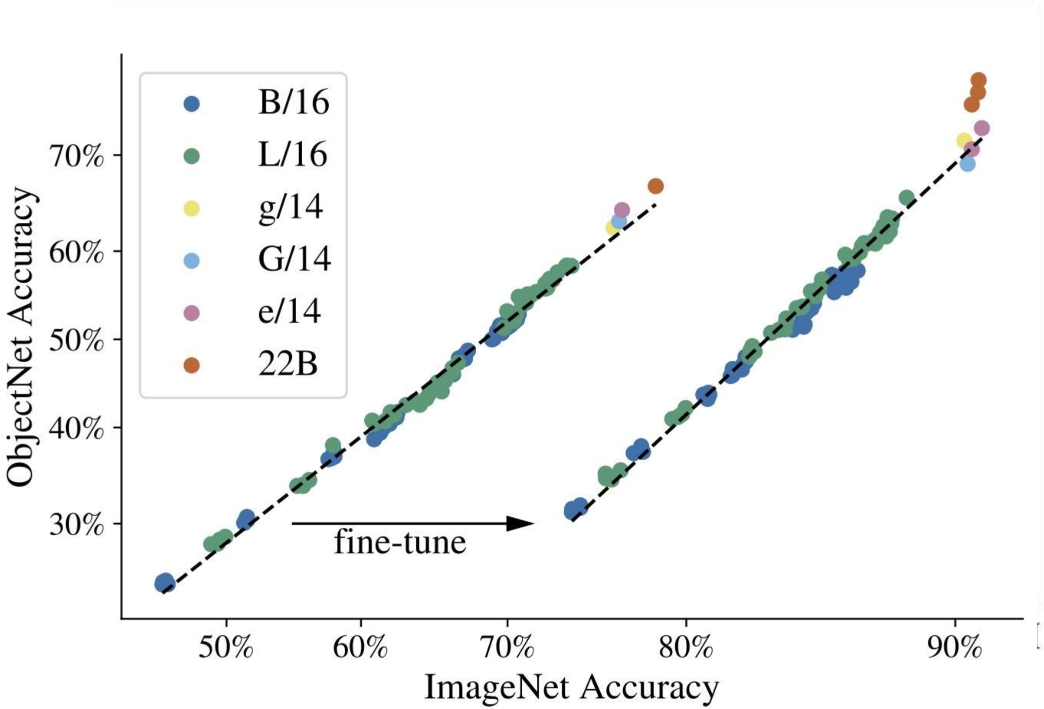
Presented By: Alex Georgiev and David Zhang

Recap

Compared to CNNs, transformers are widely considered to be more:

- Accurate[1]
- Efficient[1]
- Scalable[2]





[1]An Image is Worth 16x16 Words

[2]Scaling Vision Transformers to 22 Billion Parameters

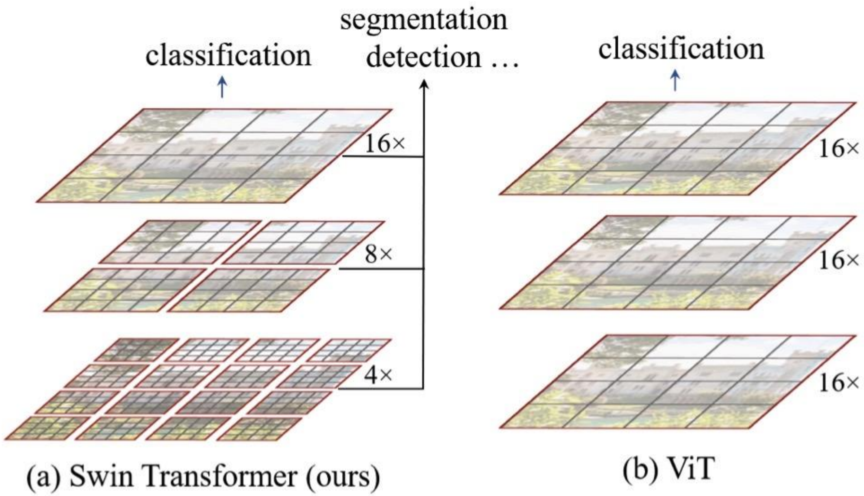
Motivation

- Prior work has attempted to **reintroduce inductive biases** to transformers
- Techniques that are applied to transformers can also be applied to CNNs
    - Training recipes
    - Layer layouts
    - Studied individually but not collectively
- Provide an architecture that can serve as a **general backbone** for image classification, object detection and segmentation
- **Scale a CNN** while keeping its computational overhead lower than a ViT

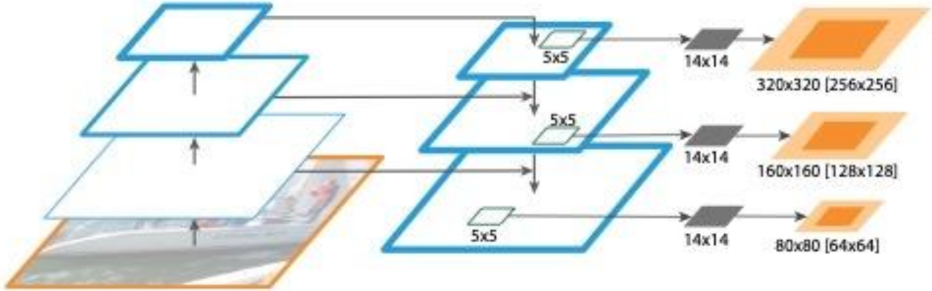
Hierarchical Learning

- Aggregating hierarchical feature maps captures coarse and fine-grained features, and makes the model more robust to scale

Swin Transformer



Feature Pyramid Network[1] (CNN)



[1]Feature Pyramid Networks for Object Detection

CNNs vs ViTs

CNN advantages

- Simple design
- Better on higher resolution inputs
- Inductive bias
- Easier to train
- Easier to apply quantization

Disadvantages:

- Struggles to capture global dependencies

Vision transformer advantages

- More scalable
- Higher accuracy
- Captures global dependencies

Disadvantages:

- Global attention has a quadratic complexity w.r.t. input size → intractable for higher resolution images

CNNs vs ViTs

CNN advantages

- Simple design
- Better on higher resolution inputs
- Inductive bias
- Easier to train
- Easier to apply quantization

Disadvantages:

- ~~Struggles to capture global dependencies~~

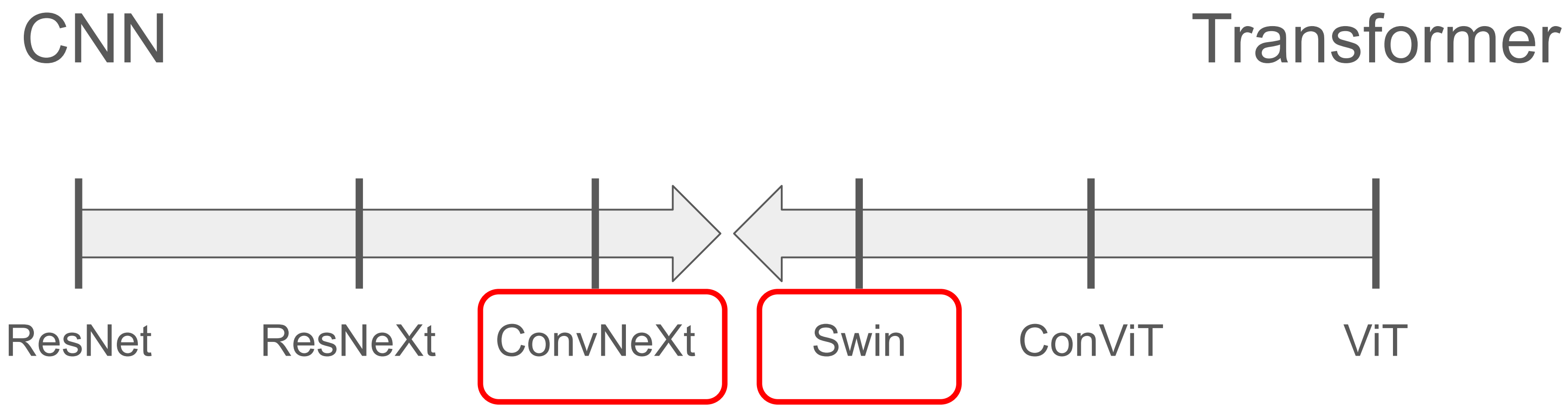
Vision transformer advantages

- More scalable
- Higher accuracy
- Captures global dependencies

Disadvantages:

- Global attention has a quadratic complexity w.r.t. input size → intractable for higher resolution images

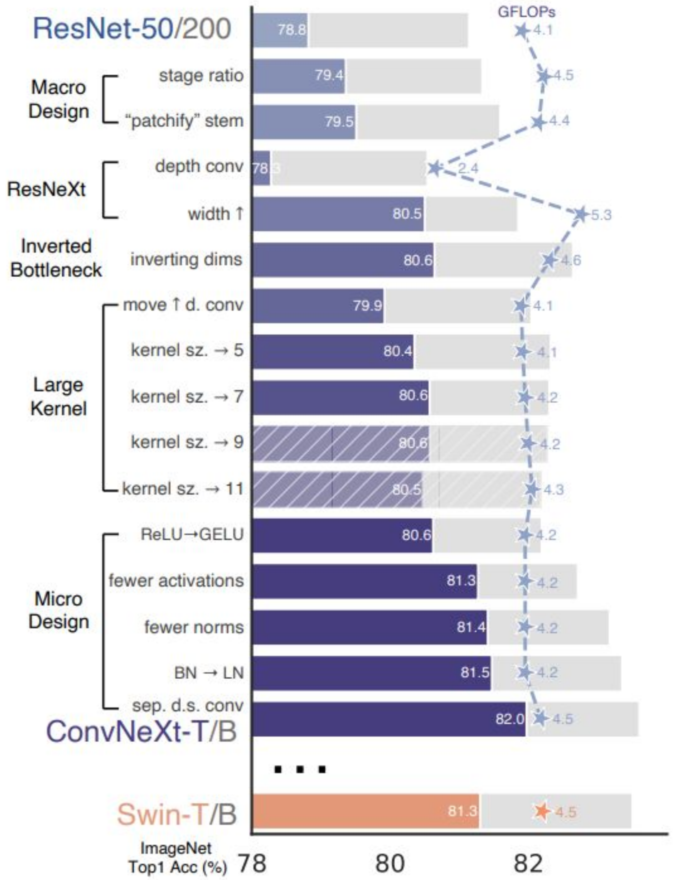
Convergence of Architectures over Time



Methods

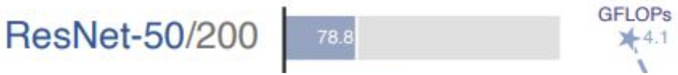
Roadmap

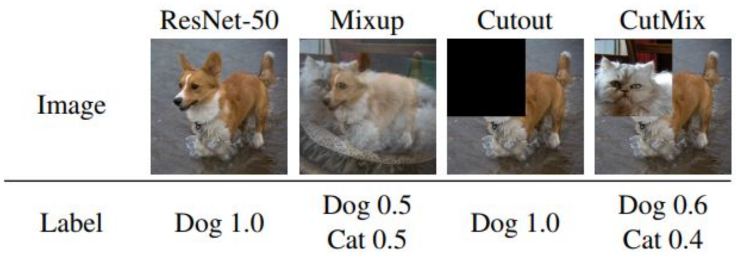
- ResNet → ConvNeXt
- Follow designs of Swin Transformer
    - While maintaining the simplicity of ConvNet

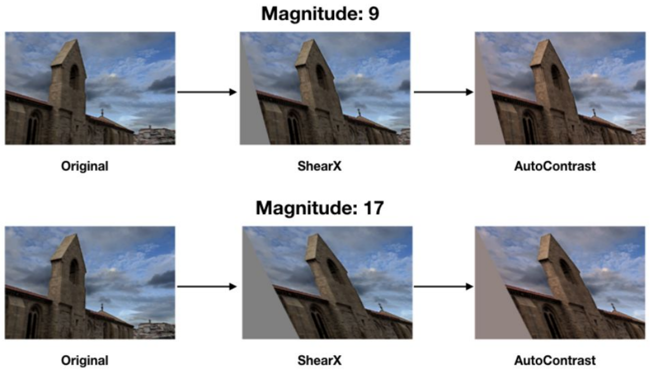


Training Techniques

- Baseline with the vision Transformer training procedure:
    - 90 → 300 epochs
    - AdamW optimizer
    - Mixup, Cutmix, RandAugment, Random Erasing
    - Stochastic Depth, Label Smoothing



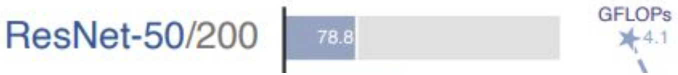


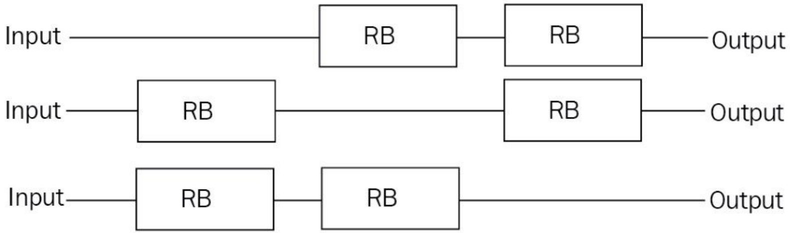


RandAugment

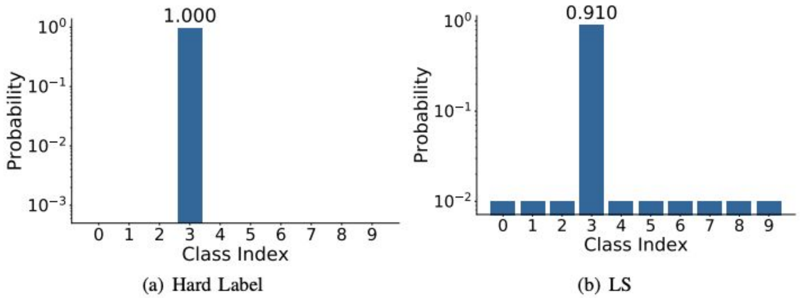
Training Techniques

- Baseline with the vision Transformer training procedure:
  - 90 → 300 epochs
  - AdamW optimizer
  - Mixup, Cutmix, RandAugment, Random Erasing
  - Stochastic Depth, Label Smoothing





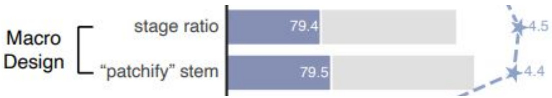
Stochastic Depth



Label Smoothing

Macro Design

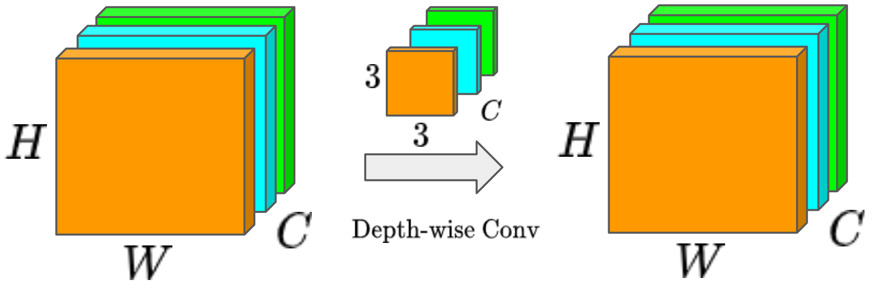
- Stage compute ratio → 1:1:3:1
- Stem: 4x4, stride 4 convolution
  - Simpler

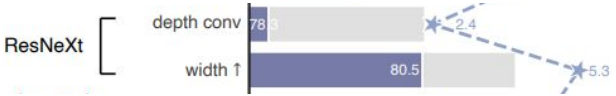


	output size	• ResNet-50	• ConvNeXt-T	○ Swin-T
stem	56×56	7×7, 64, stride 2 3×3 max pool, stride 2	4×4, 96, stride 4	4×4, 96, stride 4
res2	56×56	[1×1, 64 3×3, 64 1×1, 256] × 3	[d7×7, 96 1×1, 384 1×1, 96] × 3	[1×1, 96×3 MSA, w7×7, H=3, rel. pos. 1×1, 96 [1×1, 384 1×1, 96]] × 2
res3	28×28	[1×1, 128 3×3, 128 1×1, 512] × 4	[d7×7, 192 1×1, 768 1×1, 192] × 3	[1×1, 192×3 MSA, w7×7, H=6, rel. pos. 1×1, 192 [1×1, 768 1×1, 192]] × 2
res4	14×14	[1×1, 256 3×3, 256 1×1, 1024] × 6	[d7×7, 384 1×1, 1536 1×1, 384] × 9	[1×1, 384×3 MSA, w7×7, H=12, rel. pos. 1×1, 384 [1×1, 1536 1×1, 384]] × 6
res5	7×7	[1×1, 512 3×3, 512 1×1, 2048] × 3	[d7×7, 768 1×1, 3072 1×1, 768] × 3	[1×1, 768×3 MSA, w7×7, H=24, rel. pos. 1×1, 768 [1×1, 3072 1×1, 768]] × 2
FLOPs		4.1×10^9	4.5×10^9	4.5×10^9
# params.		25.6×10^6	28.6×10^6	28.3×10^6

ResNeXt-ify

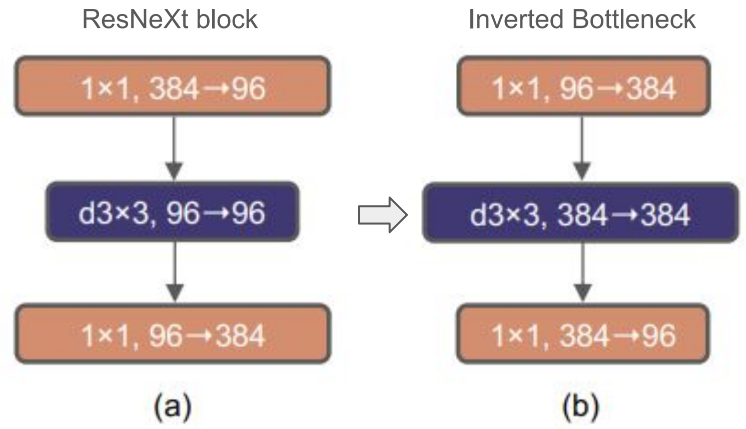
- ResNeXt - better FLOPs/accuracy trade-off
  - Depth convolution
  - Followed by 1x1 convolution
  - #Channel 64 → 96
- Separation of spatial and channel mixing
  - Similar to vision Transformers





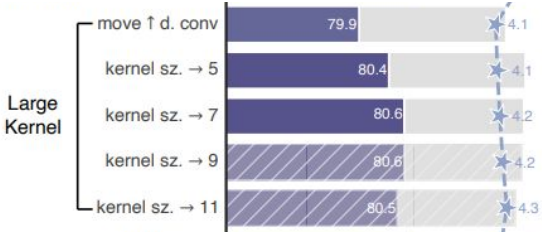
Inverted Bottleneck

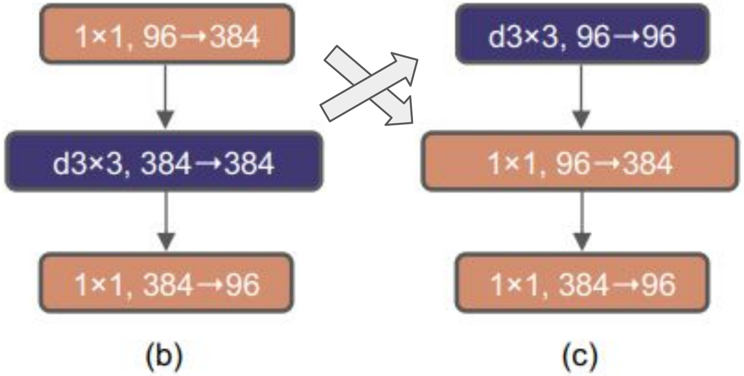
- Transformer block
  - Hidden dimension of the MLP block is four times wider than the input dimension
- FLOPs decreases due to smaller dimension in residual connection



Large Kernel Sizes

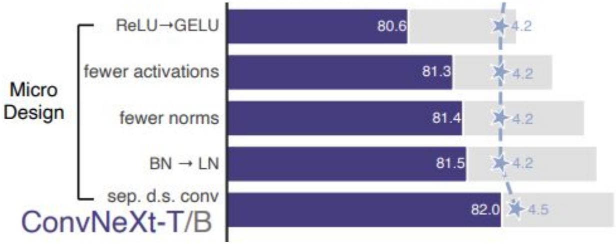
- Moving up depthwise conv layer
  - Similar to Transformer blocks
  - Reduce FLOPs
- Increase kernel sizes
  - Optimal at 7 x 7
  - Same as Swin Transformer

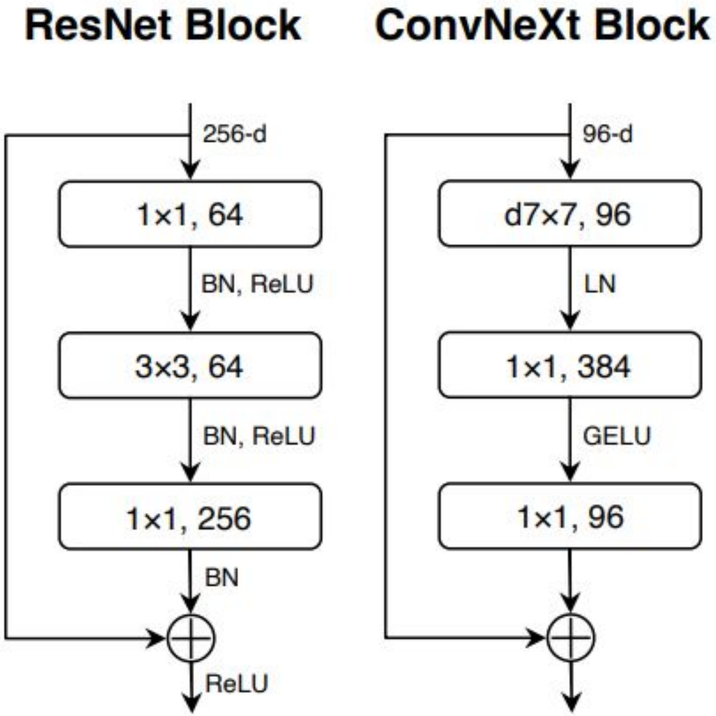




Micro Design

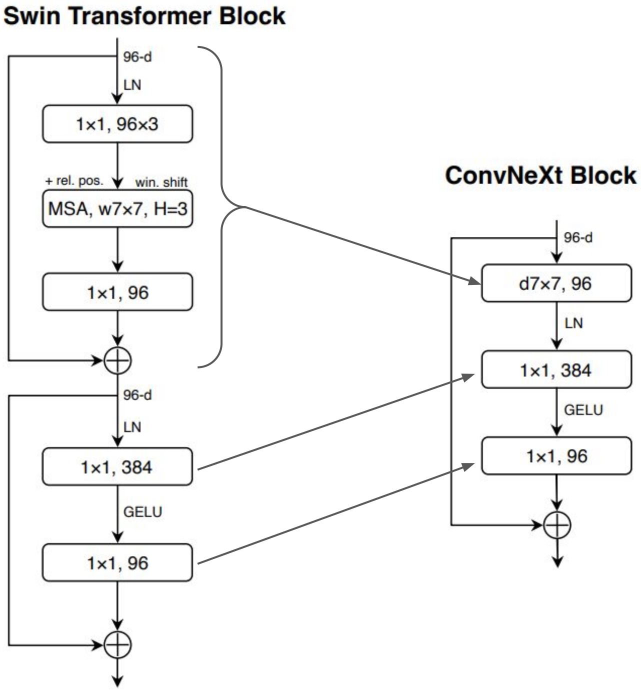
- ReLU → GELU
- A single GELU activation in each block
- A single BN in each block
- BN → LN
- Separate downsampling layers
    - 2 x 2 conv, stride 2

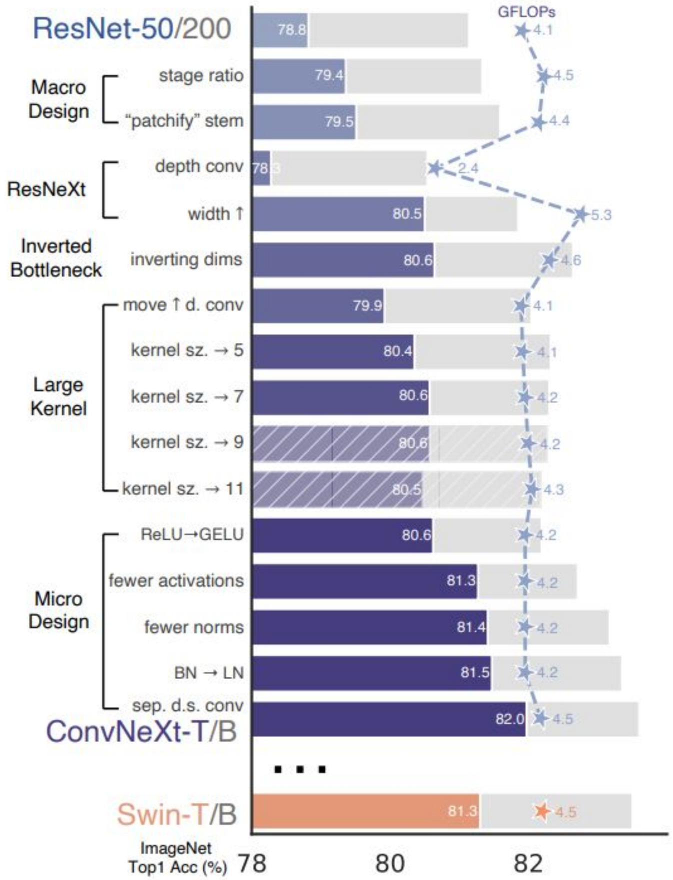




ConvNeXt

- ConvNeXt v.s. Swin Transformer





Results

Image Classification

model	image size	#param.	FLOPs	throughput (image / s)	IN-1K top-1 acc.
ImageNet-1K trained models					
• RegNetY-16G [54]	224^2	84M	16.0G	334.7	82.9
• EffNet-B7 [71]	600^2	66M	37.0G	55.1	84.3
• EffNetV2-L [72]	480^2	120M	53.0G	83.7	85.7
○ DeiT-S [73]	224^2	22M	4.6G	978.5	79.8
○ DeiT-B [73]	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**

model	image size	#param.	FLOPs	throughput (image / s)	IN-1K top-1 acc.
ImageNet-22K pre-trained models					
• R-101x3 [39]	384^2	388M	204.6G	-	84.4
• R-152x4 [39]	480^2	937M	840.5G	-	85.4
• EffNetV2-L [72]	480^2	120M	53.0G	83.7	86.8
• EffNetV2-XL [72]	480^2	208M	94.0G	56.5	87.3
○ ViT-B/16 (☎) [67]	384^2	87M	55.5G	93.1	85.4
○ ViT-L/16 (☎) [67]	384^2	305M	191.1G	28.5	86.8
• ConvNeXt-T	224^2	29M	4.5G	774.7	**82.9**
• ConvNeXt-T	384^2	29M	13.1G	282.8	**84.1**
• ConvNeXt-S	224^2	50M	8.7G	447.1	**84.6**
• ConvNeXt-S	384^2	50M	25.5G	163.5	**85.8**
○ 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**

Object Detection

backbone	FLOPs	FPS	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
Mask-RCNN 3× schedule								
○ Swin-T	267G	23.1	46.0	68.1	50.3	41.6	65.1	44.9
● ConvNeXt-T	262G	25.6	**46.2**	67.9	50.8	**41.7**	65.0	44.9
Cascade Mask-RCNN 3× schedule								
● ResNet-50	739G	16.2	46.3	64.3	50.5	40.1	61.7	43.4
● X101-32	819G	13.8	48.1	66.5	52.4	41.6	63.9	45.2
● X101-64	972G	12.6	48.3	66.4	52.3	41.7	64.0	45.1
○ Swin-T	745G	12.2	50.4	69.2	54.7	43.7	66.6	47.3
● ConvNeXt-T	741G	13.5	**50.4**	69.1	54.8	**43.7**	66.5	47.3
○ Swin-S	838G	11.4	51.9	70.7	56.3	45.0	68.2	48.8
● ConvNeXt-S	827G	12.0	**51.9**	70.8	56.5	**45.0**	68.4	49.1
○ Swin-B	982G	10.7	51.9	70.5	56.4	45.0	68.1	48.9
● ConvNeXt-B	964G	11.4	**52.7**	71.3	57.2	**45.6**	68.9	49.5
○ Swin-B‡	982G	10.7	53.0	71.8	57.5	45.8	69.4	49.7
● ConvNeXt-B‡	964G	11.5	**54.0**	73.1	58.8	**46.9**	70.6	51.3
○ Swin-L‡	1382G	9.2	53.9	72.4	58.8	46.7	70.1	50.8
● ConvNeXt-L‡	1354G	10.0	**54.8**	73.8	59.8	**47.6**	71.3	51.7
● ConvNeXt-XL‡	1898G	8.6	**55.2**	74.2	59.9	**47.7**	71.6	52.2

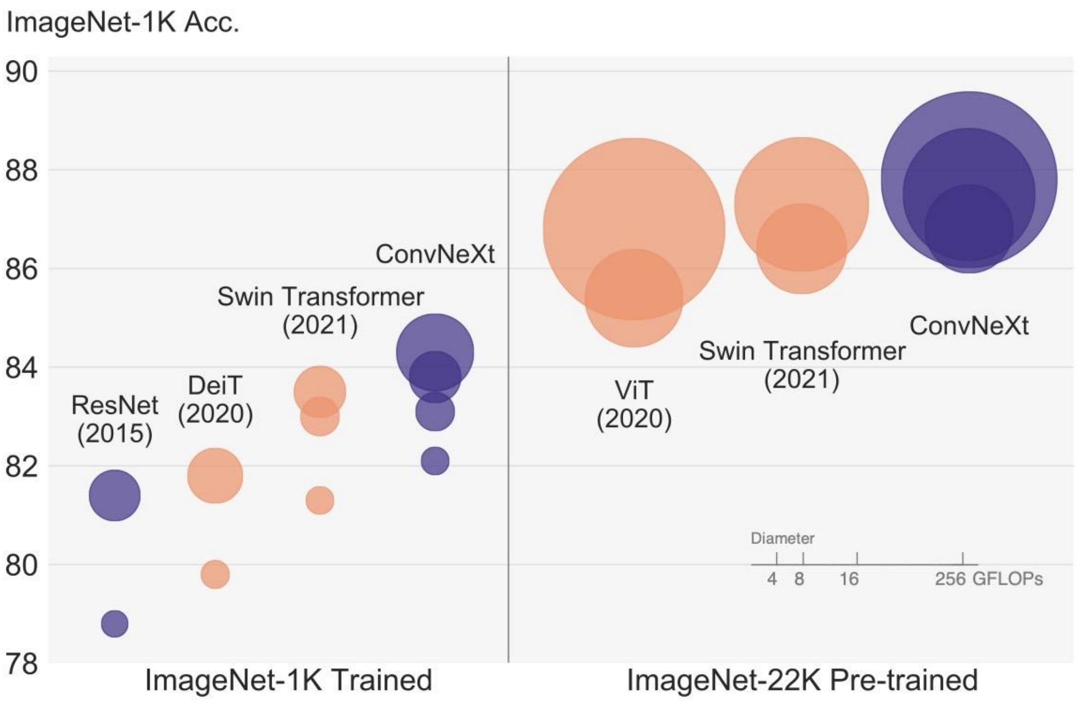
Semantic Segmentation

backbone	input crop.	mIoU	#param.	FLOPs
ImageNet-1K pre-trained				
○ Swin-T	512^2	45.8	60M	945G
● ConvNeXt-T	512^2	**46.7**	60M	939G
○ Swin-S	512^2	49.5	81M	1038G
● ConvNeXt-S	512^2	**49.6**	82M	1027G
○ Swin-B	512^2	49.7	121M	1188G
● ConvNeXt-B	512^2	**49.9**	122M	1170G
ImageNet-22K pre-trained				
○ Swin-B‡	640^2	51.7	121M	1841G
● ConvNeXt-B‡	640^2	**53.1**	122M	1828G
○ Swin-L‡	640^2	53.5	234M	2468G
● ConvNeXt-L‡	640^2	**53.7**	235M	2458G
● ConvNeXt-XL‡	640^2	**54.0**	391M	3335G

Computational Comparison with ViT

model	#param.	FLOPs	throughput (image / s)	training mem. (GB)	IN-1K acc.
○ ViT-S	22M	4.6G	978.5	4.9	79.8
● ConvNeXt-S (*iso.*)	22M	4.3G	1038.7	4.2	79.7
○ ViT-B	87M	17.6G	302.1	9.1	81.8
● ConvNeXt-B (*iso.*)	87M	16.9G	320.1	7.7	82.0
○ ViT-L	304M	61.6G	93.1	22.5	82.6
● ConvNeXt-L (*iso.*)	306M	59.7G	94.4	20.4	82.6

Scalability



What could a ViT be potentially better for?

- Multi-modal learning (attention can be applied across modalities)
- More flexible for tasks that require discretized, sparse, or structured outputs
- Capturing temporal dependencies

Takeaways

- A CNN can still achieve the same scalability as a ViT
- ConvNeXt matched the performance of transformer models yet had a lower computational overhead
  - Easier to train because it uses less GPU memory
- CNNs are still relevant for computer vision tasks