A PAPER REVIEW OF

A CONVNET FOR THE 2020S

ZHUANG LIU,
HANZI MAO,
CHAO-YUAN WU,
CHRISTOPH FEICHTENHOFER,
TREVOR DARRELL,
SAINING XIE,
FACEBOOK AI RESEARCH (FAIR),
UC BERKELEY

Adam Klein

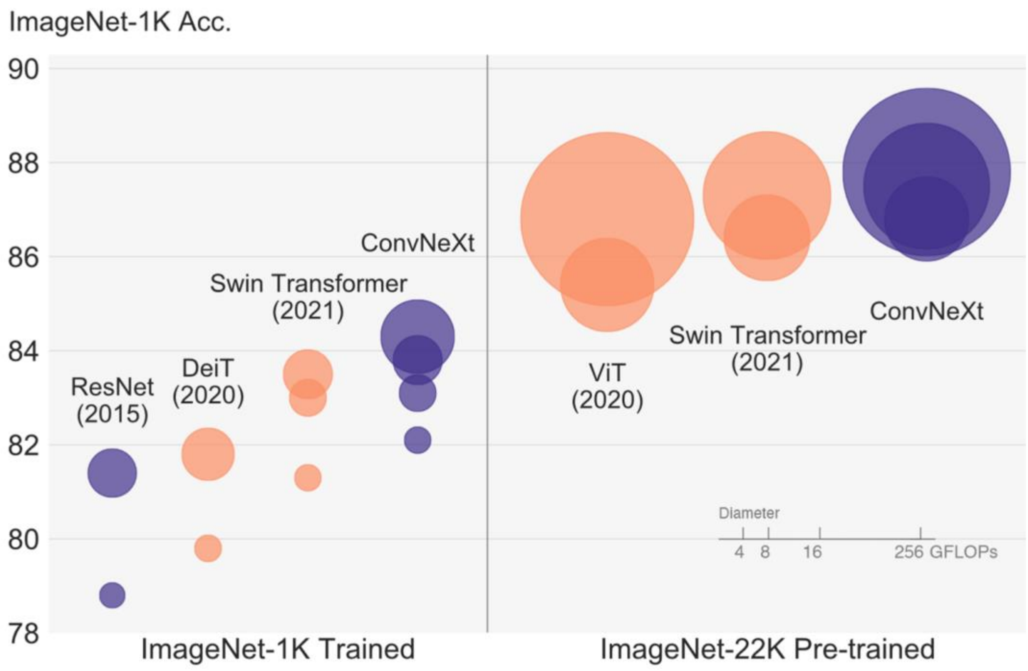
7/23/2022

- Application Engineer for Non-Volatile Memory at Synopsys, Inc.
- M.S. Electrical Computer Engineering
- Austin, TX
- Founder of Thinkcru.com
- Twitter: @thinkcru or @Adam___Klein
- LinkedIn: adam-klein
- Email: adam@thinkcru.com

A CONVNET FOR THE 2020S

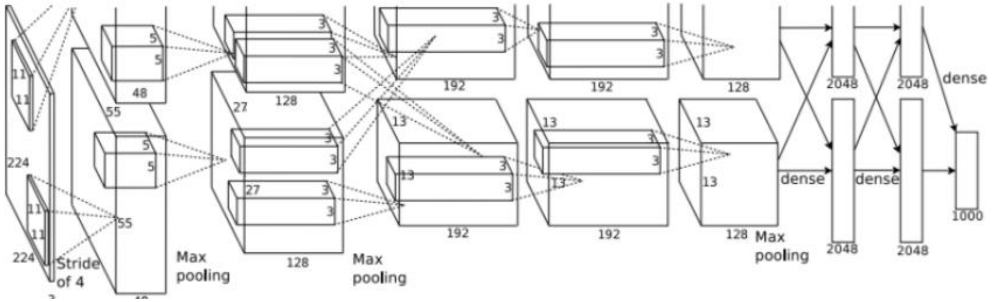
ConvNeXt

- Exploration of new family ConvNets dubbed **ConvNeXt**
- Compares Swin Transformers vs. ConvNeXt
- "Modernizes" standard ResNet using concepts from the Vision Transformers (ViTs)
- **Key Takeaway:**
  - Reexamines the design spaces vision Transformer, and picks key ingredients to build the ConvNeXt architecture
- Code: https://github.com/facebookresearch/ConvNeXt

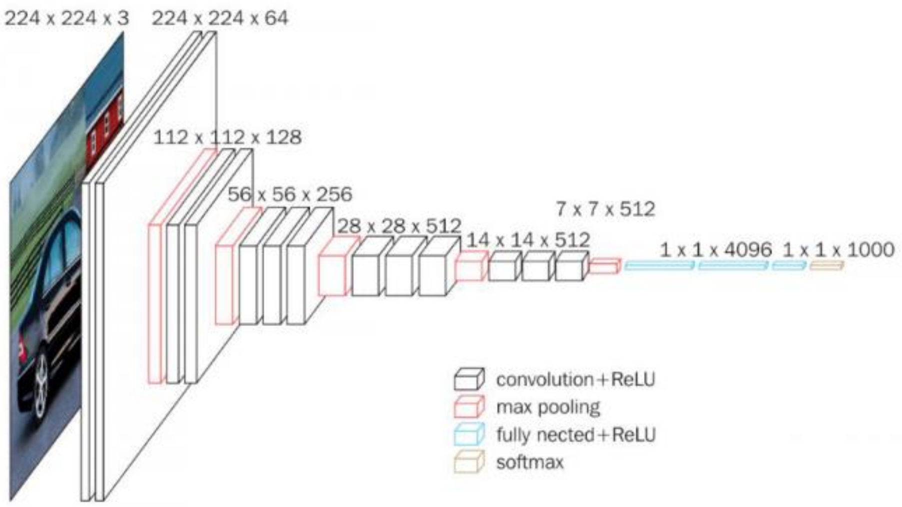


CONVOLUTIONS NEURAL NETWORKS

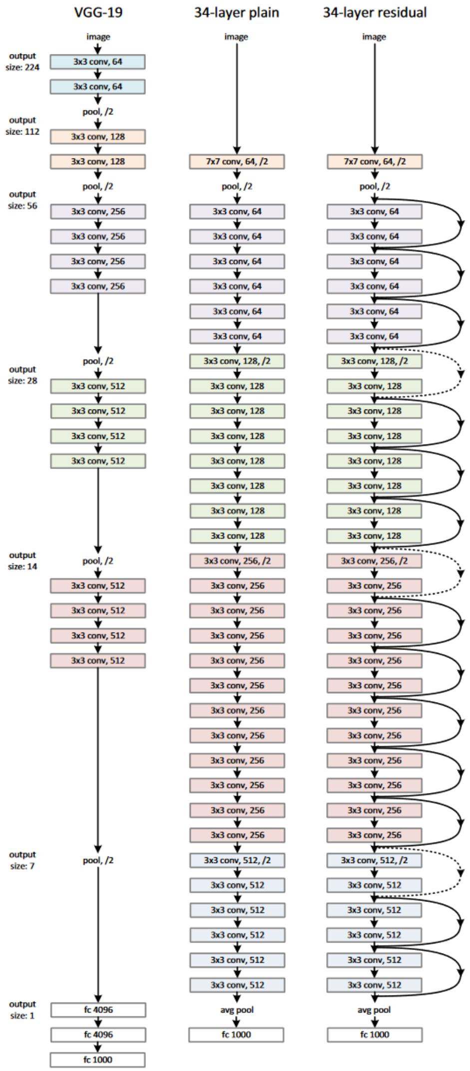
- Multiple CNN architectures
  - AlexNet (started here)
  - VGGNet
  - Inceptions
  - ResNet (residual or skip connections)
- CNN use "sliding window" strategy, computations are shared
- CNN have **inherent inductive** bias
- CNN have **translation equivariance**, good for object detection
- CNN used for image classification, segmentation, and object detection



AlexNet



VGGNet

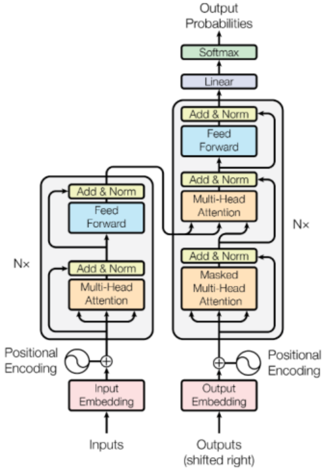


VGG Vanilla ResNet

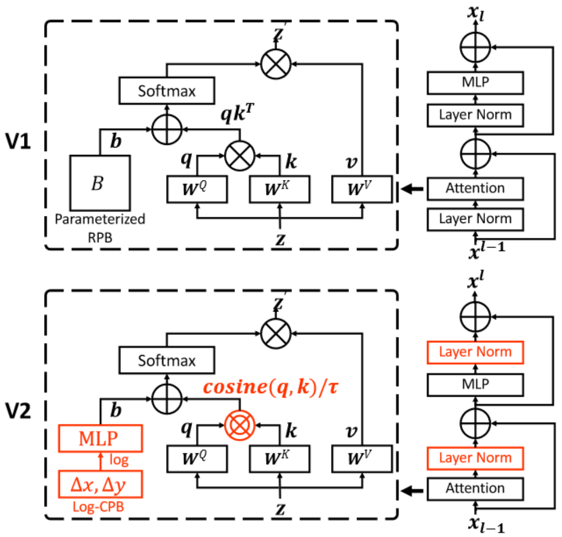
VISION TRANSFORMERS

ViTs

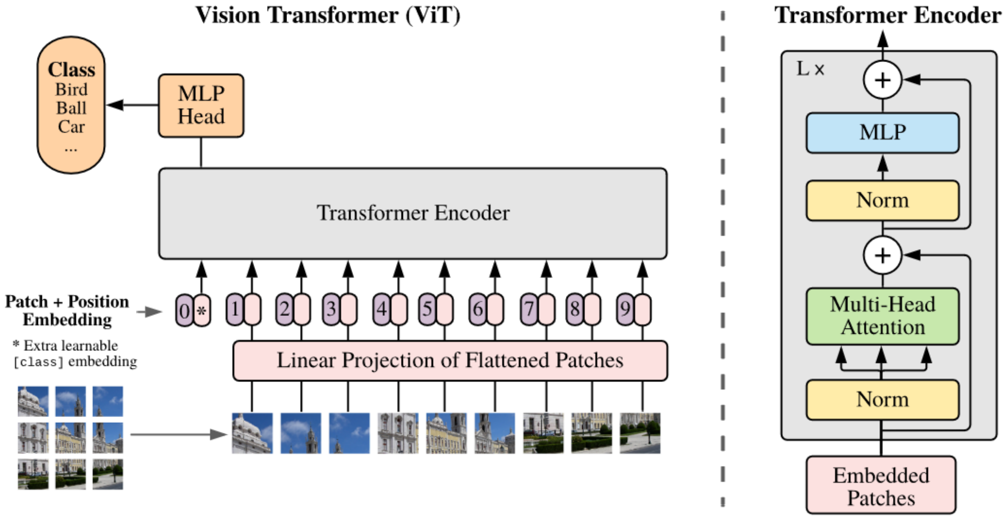
- Transformers replaced recurrent neural networks (RNN) to become the dominant backbone architecture
  - Much better long-range connections
  - Much easier to parallelize
  - Allows for deep layers compared to RNNs
- Introduces no **inductive bias**
- ViTs outperform standard ResNets by a large margin
- Swin Transformers:
  - Introduced the **Sliding Window,** which allows for "attention" in local window to be more similar to CNN
  - A.k.a. Hierarchy Transformer
- *"Swin Transformer's success and rapid adoption also revealed one thing: the essence of convolution is not becoming irrelevant; rather, it remains much desired and has never faded."*



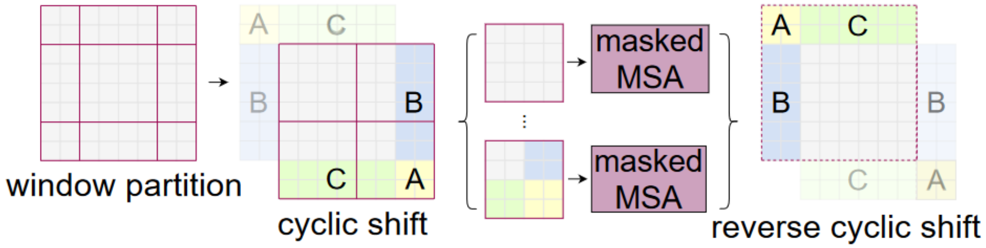
Original Transformer Architecture



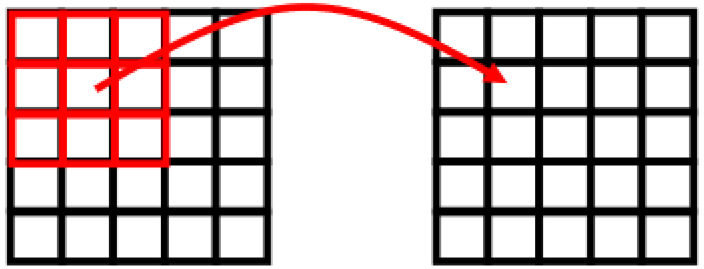
Swin Transformer



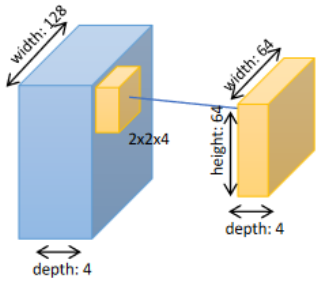
SLIDING WINDOWS

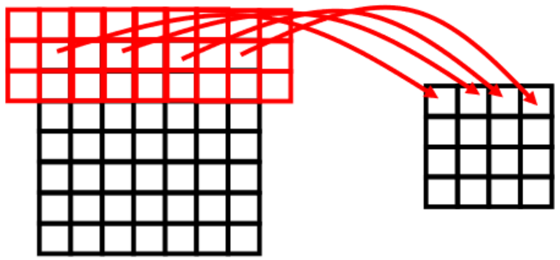


- Naïve sliding windows are computationally expensive
- Swin Transformers solved the problem with cyclic-shifting
- CNNs already have this built in with the learned weights of the filter and is strided
- Transformers were leading over CNN because:
  - they are hierarchical
  - **superior scaling** behavior with multi-head self-attention



CNN filter

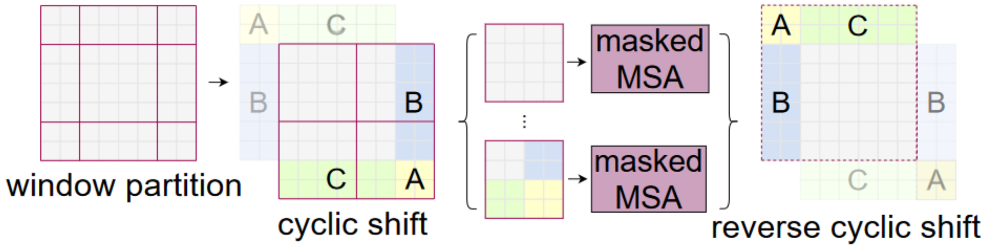




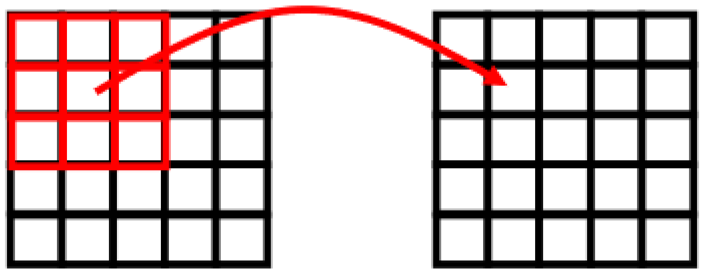
CNN strides

SLIDING WINDOWS

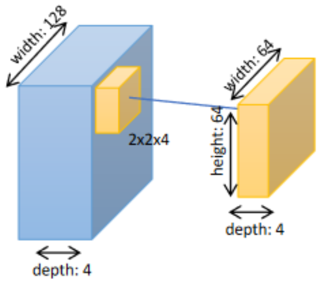
- Naïve sliding windows are computationally expensive
- Swin Transformers solved the problem with cyclic-shifting
- CNNs already have this built in with the learned weights of the filter and is strided
- Transformers were leading over CNN because:
  - they are hierarchical
  - **superior scaling** behavior with multi-head self-attention (MSA)

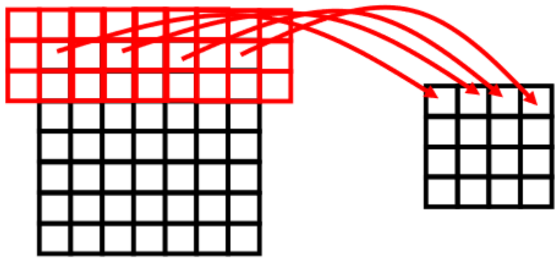


Swin Transformer Cyclic-Shifting



CNN filter



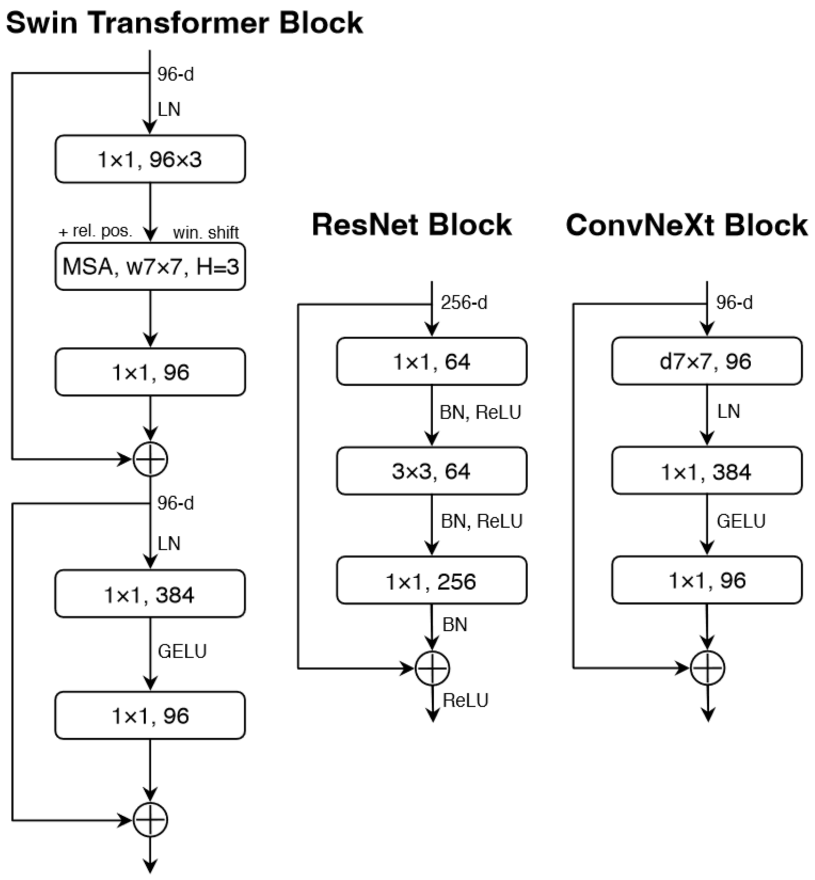


CNN strides

BUILDING THE CONVNEXT

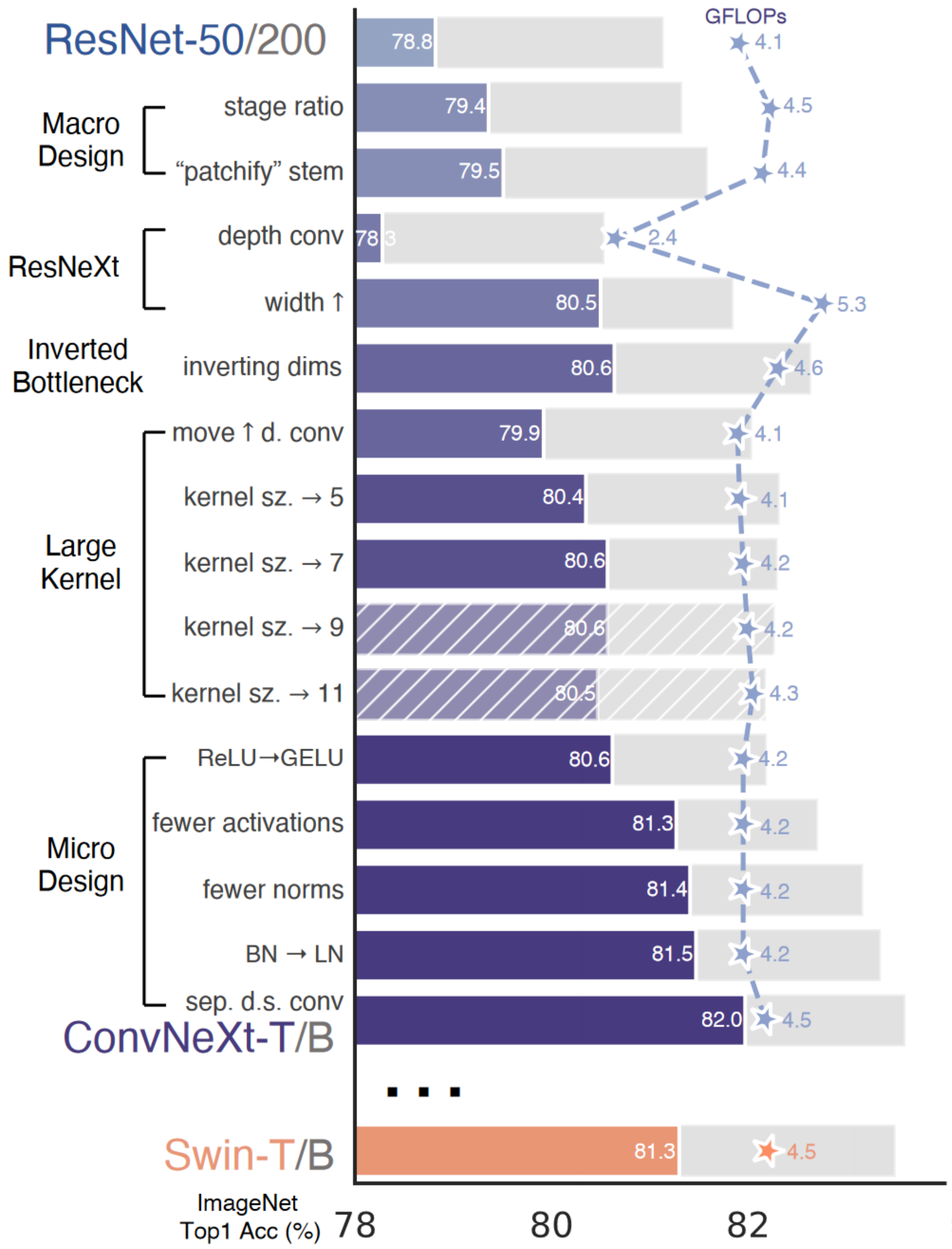
THE PURPOSE OF ANALYSIS

- This research is intended to find the strengths of the ViTs and apply to ConvNets and test the limits of the new ConvNet – ConvNeXt
- This is a hybrid approach:
  - Hierarchical ViT (e.g., Swin Transformer) + ResNet = ConvNeXt
- **Key Question:** *How do design decisions in Transformers impact ConvNets' performance?*
- *Evaluation Metrics:*
  - *Image Classification: ImageNet1K and ImageNet22K*
  - *Object Detection/Segmentation: COCO*
  - *Semantic Segmentation: ADE20K*



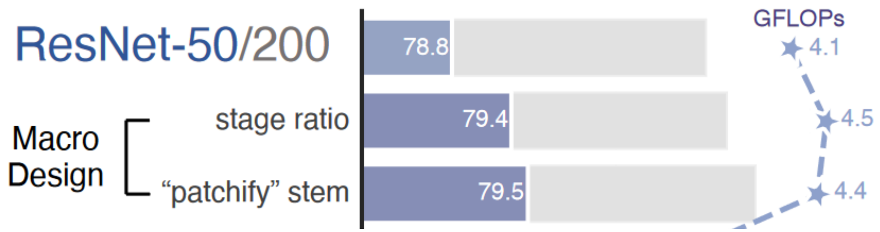
STARTING POINT

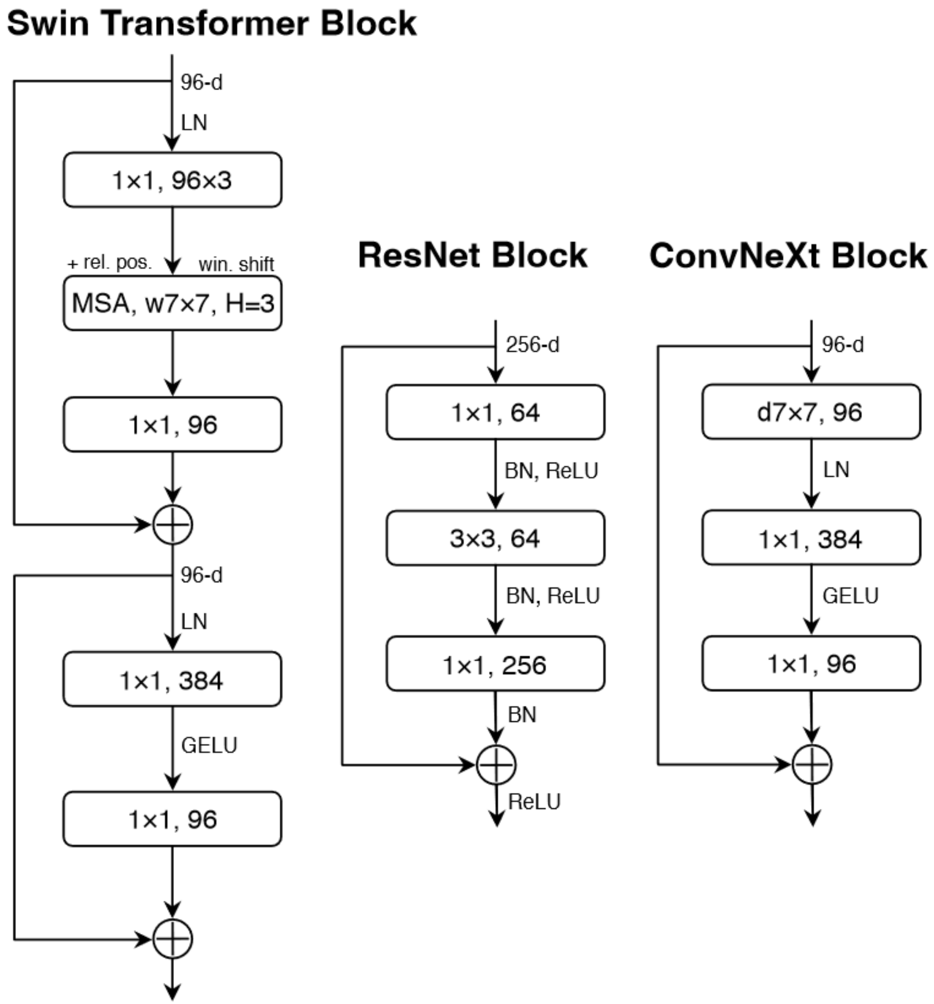
- Baseline: Standard ResNet-50 and building upon this
- Training techniques are taken from Transformers:
  - AdamW optimization
  - 300 epochs
  - Data augmentation (e.g., Mixup, Cutmix, RandAugment, RandomErasing, Stochastic Depth, Label Smoothing)
  - Improved +2.7% acc just from replicating ViT training techniques



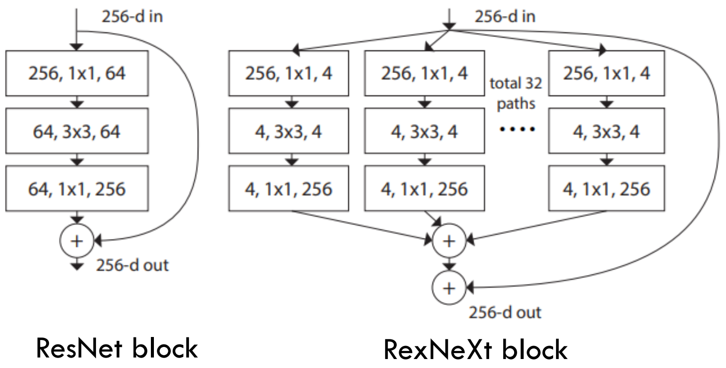
MACRO DESIGN

- Follows multi-stage design from ResNets
- Changing compute ratio:
  - (3,4,5,3) → (3,3,9,3)
  - Accuracy: 78.8% → 79.4%
- Changing step to "patchify":
  - Images have inherent redundancy
  - Common stem cell will downsample input images to feature maps
  - Replaced ResNet-style stem cell with patchify layer
    - 4x4 non-overlapping convolution
  - 79.4% → 79.5%

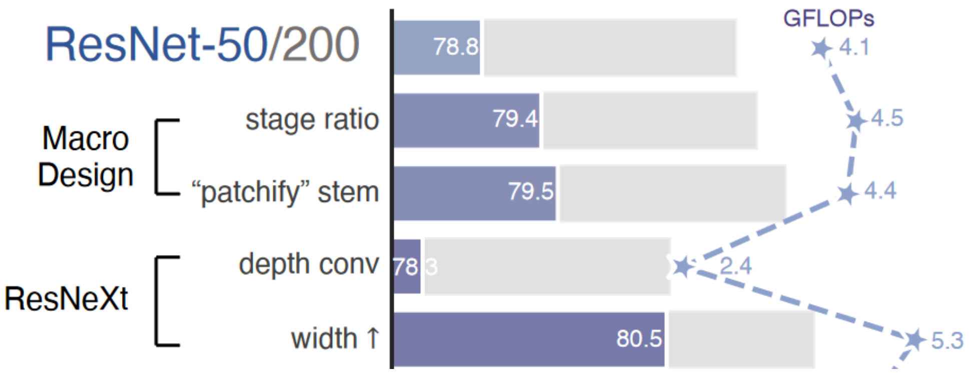




RESNEXT-IFY



A layer is shown as (# in channels, filter size, # out channels).



- ResNeXt core components are grouped convolutions where filters are separated in specific groups
- Signifcantly reduces FLOPs
- Uses **depthwise** convolution (# groups = # channels)
  - 3 x 3 conv layer bottleneck
- Depthwise convolutions is similar to weighted sum of self-attention (taken from MobileNet, Xception)
- Combining depthwise conv and 1 x 1 conv leads to separation of special and channel mixing (ViTs have this too!)
- Accuracy 79.5% → 80.5%

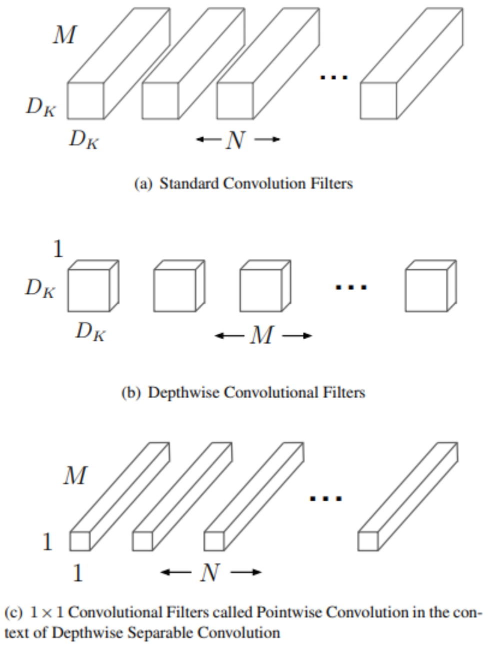
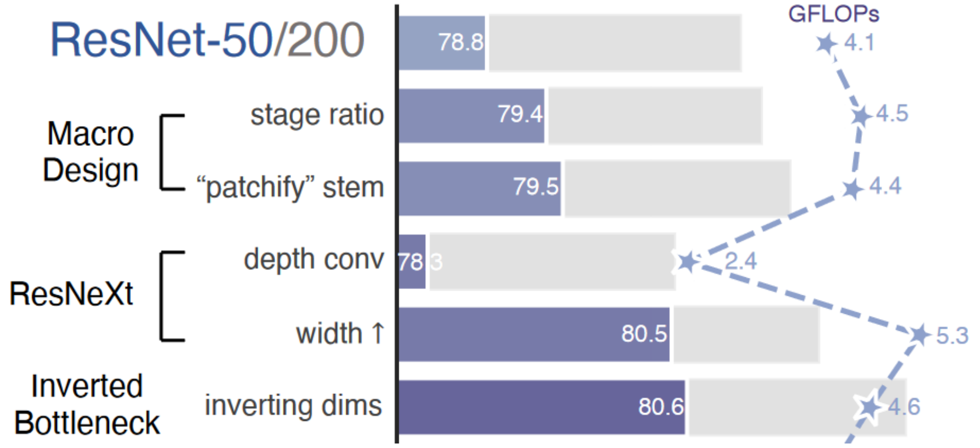
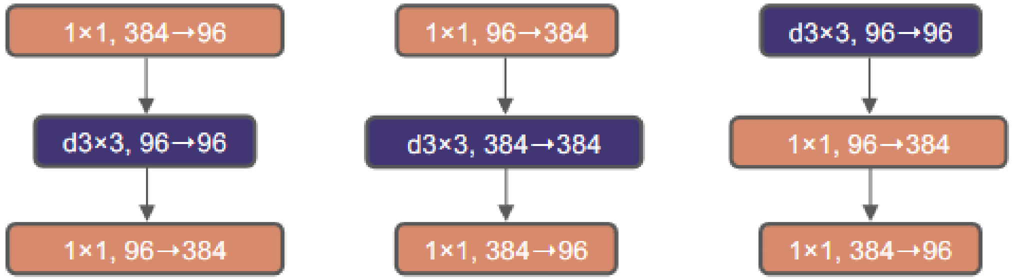


Figure 2. The standard convolutional filters in (a) are replaced by two layers: depthwise convolution in (b) and pointwise convolution in (c) to build a depthwise separable filter.

INVERTED BOTTLENECK

- All transformers have inverted bottleneck, hidden dimension of the MLP is 4 x input dimension
- Reduces overall FLOPs
- Accuracy 80.5% → 80.6%





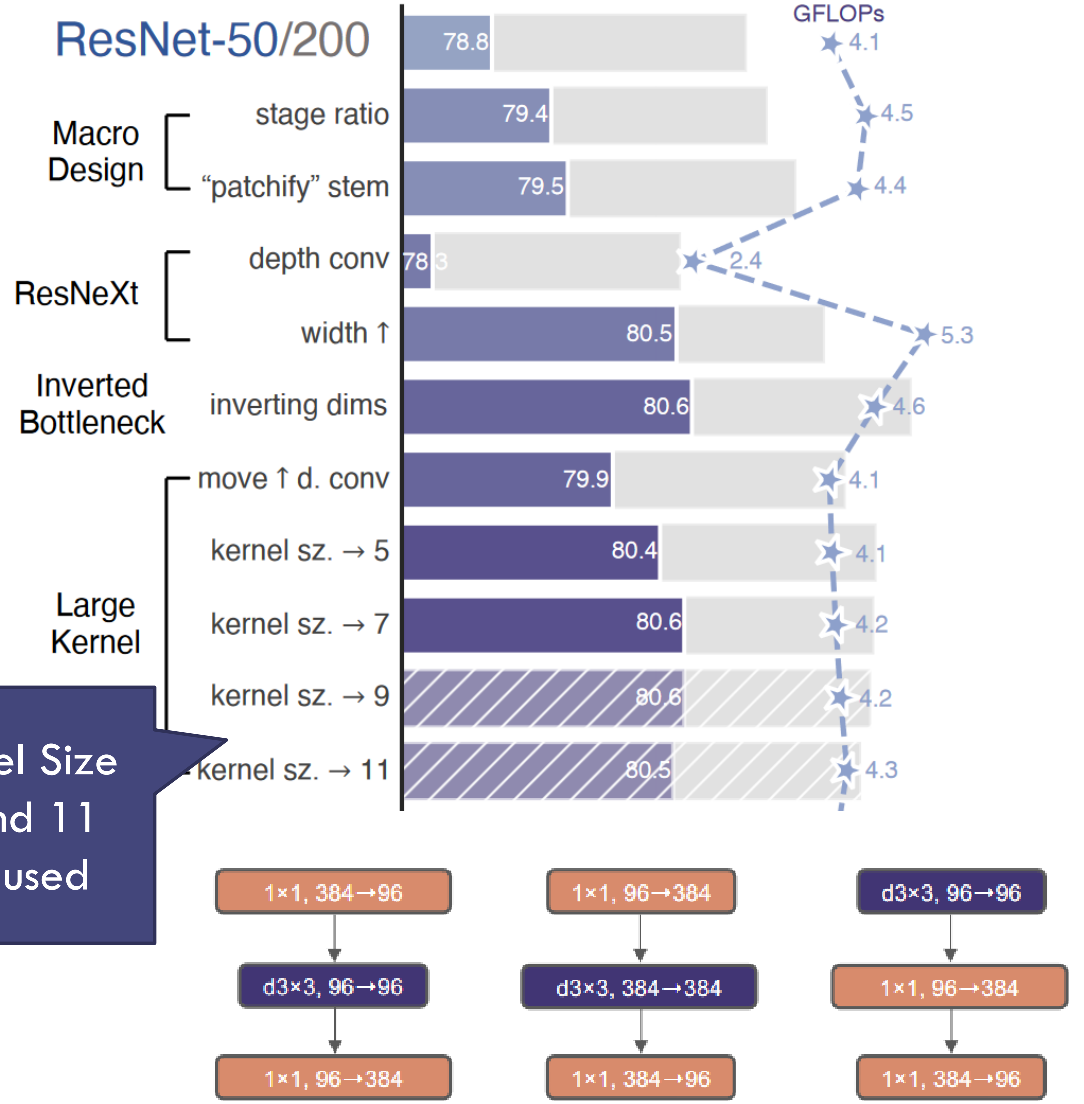
RexNeXt block

ConvNeXt inverted bottleneck block

ConvNeXt position of the spatial depthwise conv layer moved up

LARGE KERNEL SIZES

- Gold standard (e.g., VGGNet) for ConvNets is 3x3 kernel size
- Swin Transformers use 7x7 local window to the self-attention block
- Benefits of larger kernel size saturates, final used 7x7
- Also Moved depthwise conv layer up
  - Transformers have MSA block prior to MLP layers
- Accuracy 80.6% (stays same) reduced overall FLOPs



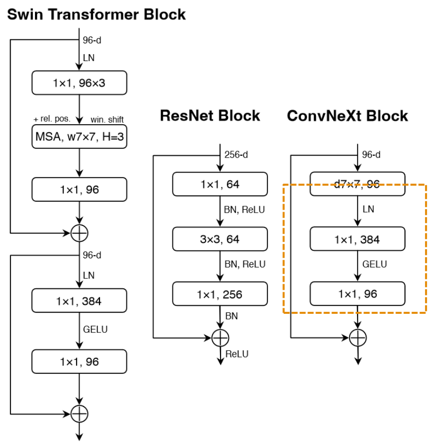
Kernel Size 9 and 11 not used

RexNeXt block

ConvNeXt inverted bottleneck block

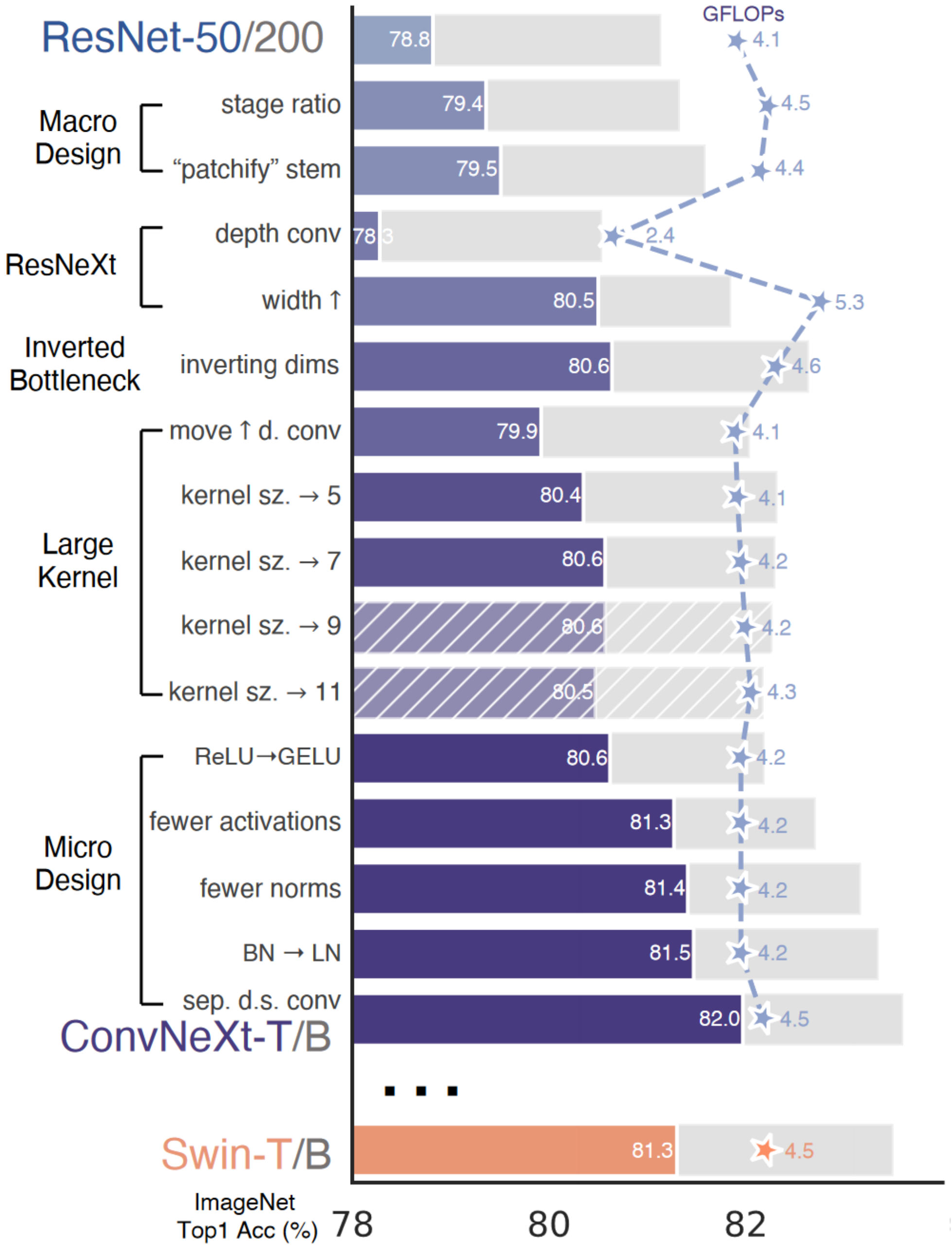
ConvNeXt position of the spatial depthwise conv layer moved up

MICRO DESIGN



- Replaced ReLU with GELU
  - Most advanced Transformers (e.g., BERT, GPT-2) use Gaussian Error Linear Unit (GELU)
  - Accuracy did not change
- Fewer activation functions
  - Transformers have few activation functions
    - Only 1 activation function in the MLP block
  - ConvNets need activation appended to each Conv layer and linear layers
- Fewer normalization layers and replace BN with LN
  - Removed Batch Normalization (BN) with Layer Normalization (LN)
  - Boosted performance to Accuracy 81.5%
- Separate down sampling layers
  - ResNet spatial sampling is achieved with residual blocks at start of each stage
  - Swin Transformers use downsampling layer between stages
  - Adding normalizing layers stabilized training

- **Final Accuracy:** 82.0% vs. Swin-T/B 81.3%
- **Final FLOPs:** 4.5 GFLOPs == Swin-T/B 4.5 GFLOPs



MODEL COMPARISONS

- ConvNeXt-B (B=Baseline)
- ConvNeXt-XL – is larger network to test scalability
- Number of *channels* doubles each new stage:
  - ConvNeXt-T: C = (96, 192, 384, 768), B = (3, 3, 9, 3)
  - ConvNeXt-S: C = (96, 192, 384, 768), B = (3, 3, 27, 3)
  - ConvNeXt-B: C = (128, 256, 512, 1024), B = (3, 3, 27, 3)
  - ConvNeXt-L: C = (192, 384, 768, 1536), B = (3, 3, 27, 3)
  - ConvNeXt-XL: C = (256, 512, 1024, 2048), B = (3, 3, 27, 3)
- ImageNet-1K: 1000 classes (pre-trained 300 epochs)
- ImageNet-22K: 21,841 classes (pre-trained 90 epochs)
- Interesting training tip: Exponential Moving Average (EMA) elevates large model overfitting
- V100 GPU

Inference

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**
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**

DOWNSTREAM TASKS

- Object Detection and Segmentation - COCO dataset
  - Fine tuned Mask R-CNN and Cascade Mas R-CNN using ConvNeXt as the backbone
  - Performance equivalent to Swin Transformers
- Semantic segmentation ADE20K dataset
  - Used UperNet
  - Seems to beat out Swin models
- Model efficiency:
  - ConvNeXt models required less memory when training compared to Swin-T
  - Claim: Improved efficiency is a result of the ConvNet inductive bias, not directly related to self-attention mechanism in ViTs

Object Detection and Segmentation

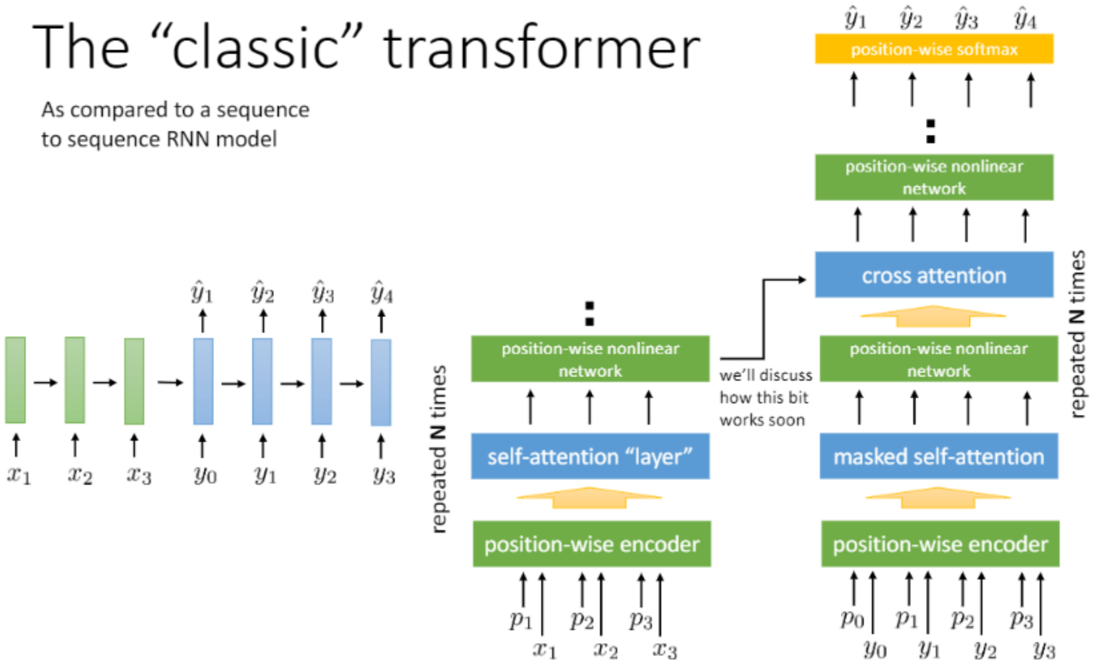
backbone	FLOPs	FPS	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
Mask-RCNN 3× schedule								
○ Swin-T	267G	23.1	46.0	68.1	50.3	41.6	65.1	44.9
● ConvNeXt-T	262G	25.6	**46.2**	67.9	50.8	**41.7**	65.0	44.9
Cascade Mask-RCNN 3× schedule								
● ResNet-50	739G	16.2	46.3	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

LIMITATIONS

- Transformers may be more flexible for certain tasks
  - Multi-modal learning – requires cross-attention module to model feature interactions across many modalities
  - Can handle tasks requiring discretized, sparse, or structured outputs



Sergey Levine UC Berkeley CS182 slides

THANK YOU

Q&A