



[논문 리뷰] ConvNeXt [2022]

Abstract

Visual Recognition은 ViTs(Vision Transformers)의 도입으로, 빠르게 대체되기 시작하였습니다.

한편으로는 Segmentation, Object Detection과 같은 일반적인 task에 사용하기에는 어려움이 있었습니다.

이후, Swin-Transformer가 등장한 후, 실제로 Application에 활용하는 것이 가능해졌으며, 일반적인 Vision의 Backbone으로 주목할 만한 것을 보여주었습니다.

그리고 Vision의 다양한 Task에서 성능을 발휘하였습니다.

본 논문에서는, Vision Transformer의 Design을 향해 ResNet을 점진적으로 “Mordernize”하고, 몇몇의 주요 구성 요소의 차이가 성능 차이를 나타낸다는 것을 발견하였습니다.

ConvNet의 단순함과 효율성을 유지하면서, ConvNeXts는 정확성 및 확장성의 측면에서 ImageNet1K에서 1등을 달성하였습니다. 또한, COCO에서 Swin Transformers를 능가하는 정확성 및 성능을 얻었습니다.

Introduction

2010년부터, Visual Feature Learning은 계속해서, 발전해왔습니다.

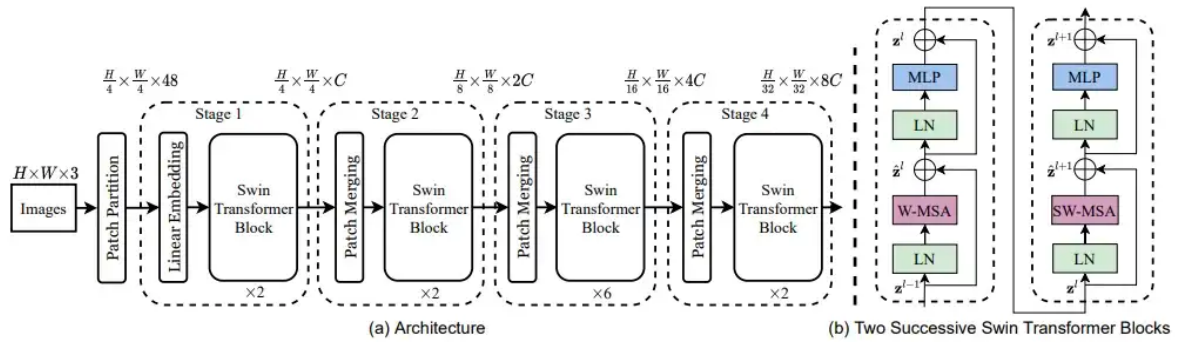
Computer Vision에서의 영역에서의 ConvNets의 지배는 우연이 아닙니다.

Sliding Window 전략은 Visual Processing에 영감을 주었습니다. 특히, 높은 해상도를 이미지를 가진 이미지는 더욱 그러합니다.

또한, ConvNets을 쌓음으로써, Visual Recognition System에서도 활용되게 되었습니다.

이후 NLP에서 Transformer가 대부분을 차지한 후, 2020년에 Vision 과 NLP가 Task가 다름에도 불구하고,

Transformer를 활용하는 것이 일반적인 것이 되었습니다.



Hierarchical Transformer는 Hybrid 접근법을 활용하여, 격차를 줄입니다. 특히 “**Sliding Window**” 전략은 Transformer에 다시 도입되었으며, ConvNet과 유사하게 사용되도록 되었습니다.

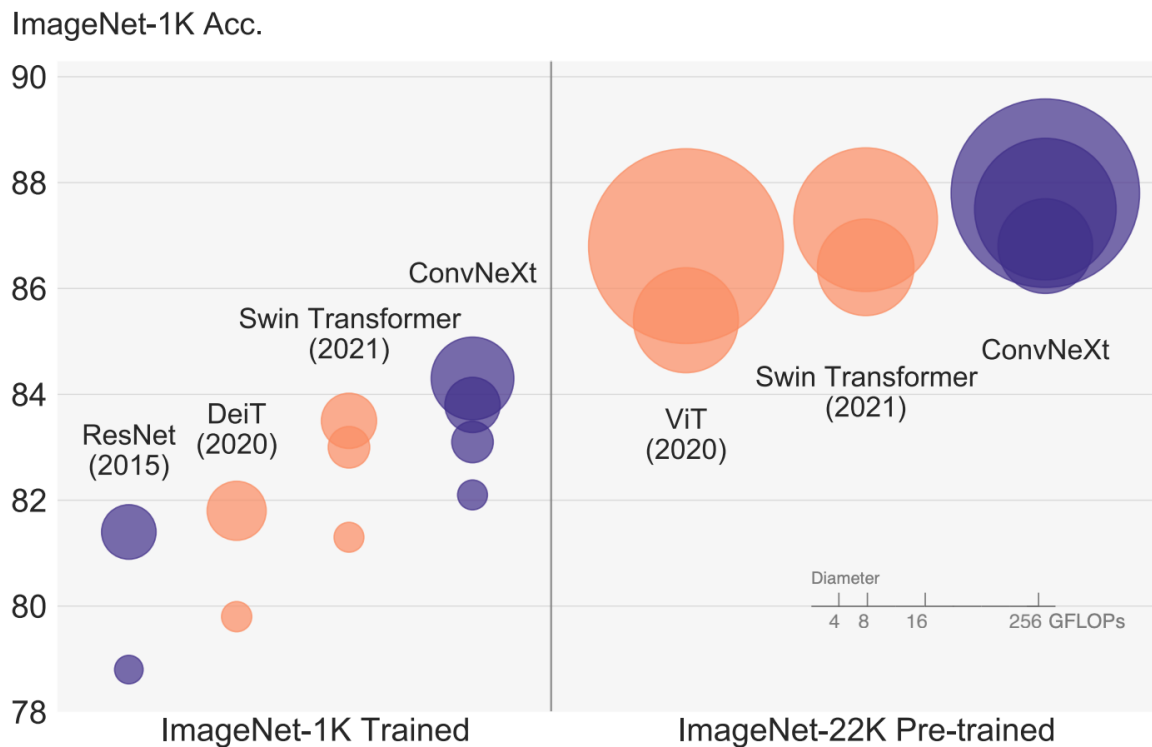
ConvNet이 활력을 잃고 있는 이유는, Transformer기반의 모델들이 많은 Vision task에서 우월한 성능을 가지기 때문입니다.

연구에서는, ConvNet과 Transformer사이의 구조적 차이를 조사하고, 성능을 비교할 때, 교란 변수를 식별하려고 합니다.

Several Key Components들이 성능의 차이에 기여한다는 것을 발견하였습니다.

ConvNeXt라고 명칭을 정하였습니다.

ConvNeXt는 ConvNets의 효율성을 유지할 뿐만 아니라, 구현도 간단하며, Transformer와 견줄만한 Accuracy, Scalability, Robustness을 가지고 있습니다.

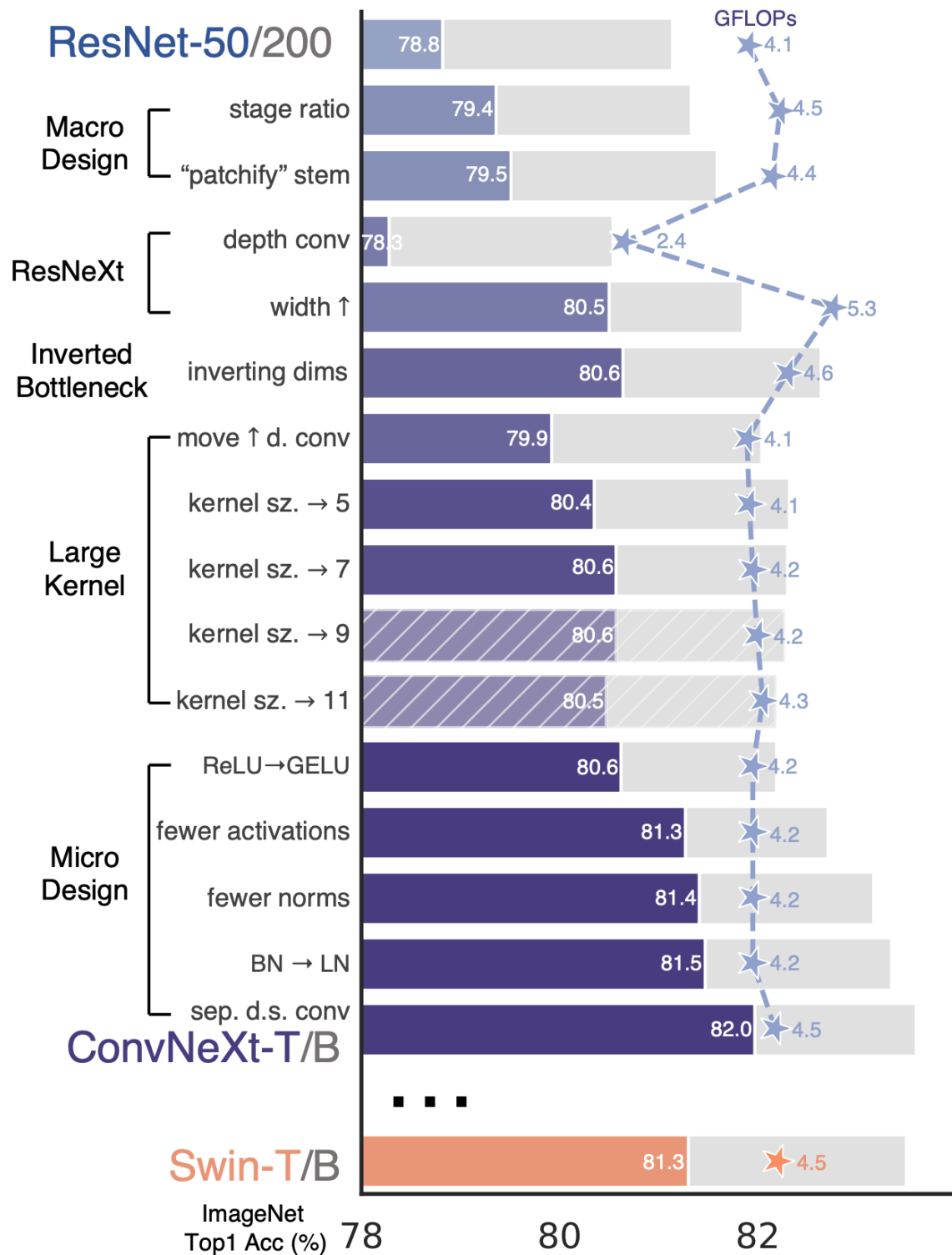


Modernizing a ConvNet: a Roadmap

Transformer와 유사한 ConvNet에 ResNet을 연결합니다.

Vision Transformer가 학습하는 방법과 유사하게 학습을 합니다.

이러한 방법은 ResNet-50보다 높은 성능을 보여줍니다.



Training Techniques

Training procedure는 성능에 상당한 영향을 끼칩니다. Vision Transformer에서 새로운 module을 가져올 뿐만 아니라, AdamW Optimizer를 사용합니다.

training을 하는 방식으로는 DeiT's와 Swin Transformer's와 유사한 방식으로 학습을 진행하였습니다.

Epochs의 횟수는 90에서 300으로 연장하였습니다.

Optimizer로는 AdamW를 활용하였으며,

Data Augmentation 방식으로는 MixUp, CutMix, RandAugment, RandErasing, Regularization schemes, Depth, Label Smoothing을 활용하였습니다.

| (pre-)training config | ConvNeXt-T/S/B/L | ConvNeXt-T/S/B/L/XL |
|-----------------------------|---------------------------------|----------------------------------|
| | ImageNet-1K 224 ² | ImageNet-22K 224 ² |
| weight init | trunc. normal (0.2) | trunc. normal (0.2) |
| optimizer | AdamW | AdamW |
| base learning rate | 4e-3 | 4e-3 |
| weight decay | 0.05 | 0.05 |
| optimizer momentum | $\beta_1, \beta_2=0.9, 0.999$ | $\beta_1, \beta_2=0.9, 0.999$ |
| 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, 12] | None | None |
| randaugment [14] | (9, 0.5) | (9, 0.5) |
| mixup [90] | 0.8 | 0.8 |
| cutmix [89] | 1.0 | 1.0 |
| random erasing [91] | 0.25 | 0.25 |
| label smoothing [69] | 0.1 | 0.1 |
| stochastic depth [37] | 0.1/0.4/0.5/0.5 | 0.0/0.0/0.1/0.1/0.2 |
| layer scale [74] | 1e-6 | 1e-6 |
| head init scale [74] | None | None |
| gradient clip | None | None |
| exp. mov. avg. (EMA) [51] | 0.9999 | None |

Macro Design

Swin Transformer의 macro network design을 분석하였습니다. Swin-Transformer는 ConvNets의 multi-stage Design을 따르며, 각각의 stage는 다른 Feature map의 resolution을 가지고 있습니다.

Change stage compute ratio:

ResNet의 여러 단계에 걸친 계산 분포의 일반적인 설계는 대체로 경험적입니다. “res4”는 객체 감지와 같은 downstream tasks와 같은, 작업과 호환되도록 의도었습니다.

ResNet의 각 단계의 블록 수를 (3, 4, 6, 3)에서, (3, 3, 9, 3)으로 조정하였습니다.

연구진들은, Distribution of Computation을 조사하였으며, 최적의 디자인 설계를 위해 연구하였습니다.

Changin stem to Patchify:

일반적으로, Stem cell 디자인은 Input Image가 네트워크의 시작에서 어떻게 처리되는지와 연관되어 있습니다.

Swin Transformer의 예시를 들어, 4x4와 stride 4를 활용하여 patchify layer를 구현합니다.

Inverted Bottleneck

Transformer Block의 가장 중요한 디자인은,

Input Dimension보다 4배 큰 inverted bottleneck을 생성하는 Invertible Block을 사용한다.

이러한 방식을 **ConvNeXt에도 downsampling residual block으로 활용**하고자 한다.

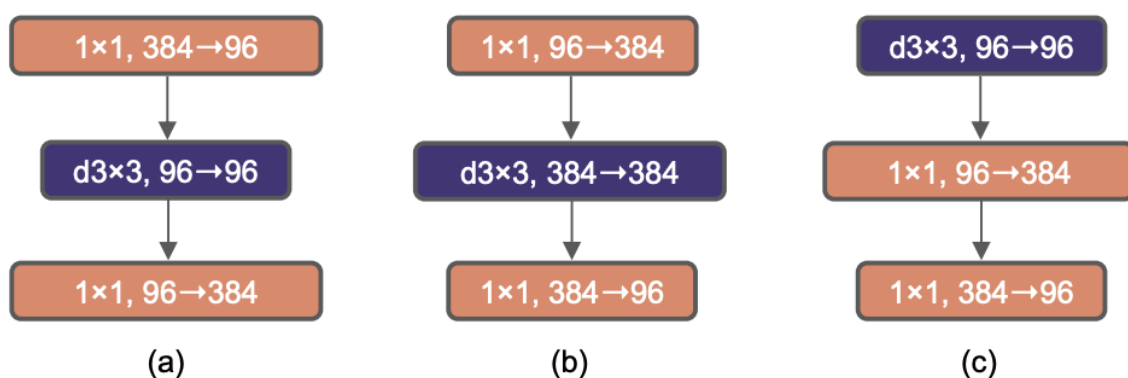


Figure 3. Block modifications and resulted specifications. (a) is a ResNeXt block; in (b) we create an inverted bottleneck block and in (c) the position of the spatial depthwise conv layer is moved up.

Large Kernel Sizes

large Convolutional Kernel을 사용하는 것에 초점을 두었다. Vision Transformer와 가장 구별되는 측면은, non-local self-attention을 사용하지 않는다는 것이다.

Moving up depthwise conv layer

Swin-T와 비교해보자면, MLP Block안에 MSA를 수행하는 것이 된다. 그렇기 때문에 저자들은, inverted block을 Swin-T의 구조를 적용하여 MSA를 먼저 수행하고, MLP를 수행하는 구조로 변환하였다.

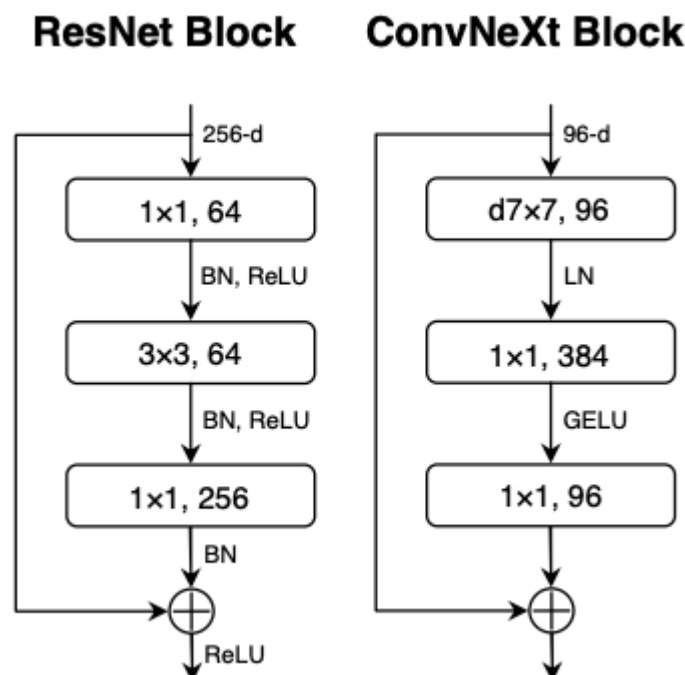
Increasing the kernel size

저자들은 3, 5, 7, 9, 11 kernel size 모두 실험을 통하여, 7x7 kernel size가 saturation이 되었다고 한다.

FLOPs는 거의 유지된 상태로 정확도가 상승되었습니다.

Micro Design

일반적으로, CNN에서는 ReLU를 사용하는 것이 일반적이었지만, ReLU를 GeLU로 변경하여, 모델에 적용하였습니다.



Transformer는 ResNet block과 다르게, 적은 activation function을 사용하고 있습니다. MSA부분에는 activation이 없고, MLP block에 한 개의 activation function만 포함합니다. 이러한 Transformer의 전략을 활용하기 위해, ConvNeXt에서는 두개의 1x1 사이에 하나의 GELU를 사용하여 성능을 올립니다.

Normalization의 개수를 줄임으로써, Transformer와 비율을 비슷하게 갑니다.

Batch Normalization은 Layer Normalization과 다르게, batch size에 따라, 성능 변화의 편차가 심하다는 문제가 있습니다. 반면에, LN은 지금까지 만든 ConvNext에 적용한다면, 오히려 성능을 향상시킵니다.

기존 ResNet은 각 Block이 끝난 뒤 높은 channel로 feature extraction을 하기 위해, block의 첫 번째 layer를 3X3 conv, stride 2를 사용하여, width와 height를 절반으로 줄입니다.

하지만, Swin Transformer의 경우 DownSampling layer가 따로 존재하기에, ConvNeXt에 DownSampling을 위한 layer를 추가하였다.

Result

| 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

Conclusion

2020년대, hierarchical Transformer가 유행이 되기 시작하였다. Swin Transformer는 일반적으로 ConvNet의 backbone으로 활용하고 있습니다. 여기에서 이뤄진 연구는 지난 연구들에서 활용된 것들을 조합하여, 만든 것이고, ComptuerVision에서 Convolutional의 중요성을 일깨우고 있습니다.

Reference

A ConvNet for the 2020s


The "Roaring 20s" of visual recognition began with the introduction of Vision Transformers (ViTs), which quickly superseded ConvNets as the state-of-the-art image

 <https://arxiv.org/abs/2201.03545>



[논문리뷰] A ConvNet for the 2020s

이번 포스팅에는 2022년 1월 FAIR에서 발표한 'A ConvNet for the 2020s' 라는 논문을 리뷰하려고 한다. 해당 논문은 2020년에 ViT(Vision Transformer)가 발표된 이후 Vision task에서

 <https://blog.kubwa.co.kr/%EB%85%BC%EB%AC%B8%EB%A6%AC%EB%B7%B0-a-convnet-for-the-2020s-9b45ac666d04>

04

