A ConvNet for the 2020s

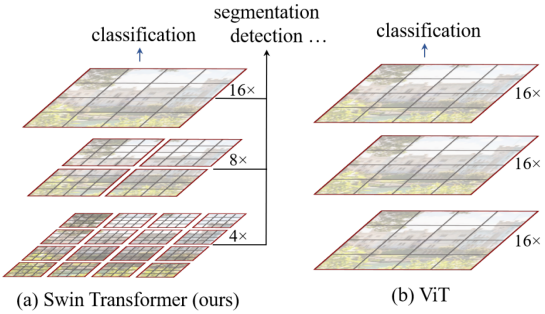
Шипилов Фома

ВШЭ ФКН ПМИ

5 октября 2022 г.

Swin Transformer: Рекап

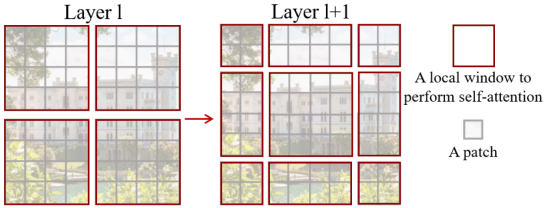
- Иерархическая архитектура;



(Liu et al.)

Swin Transformer: Рекап

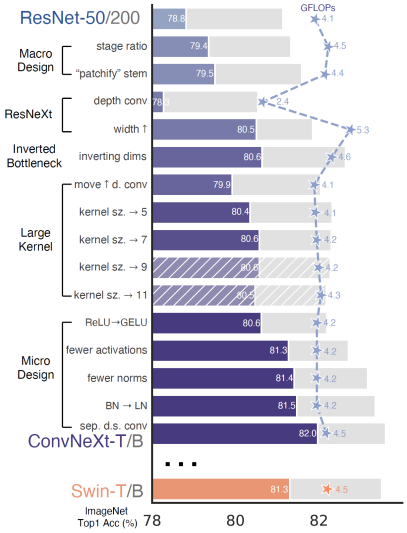
- Иерархическая архитектура;
- Замена глобального self-attention на локальный windowed attention.



(Liu et al.)

Roadmap

1. Бейзлайн: ResNet-50;
2. Пайплайн обучения;
3. Конфигурация блоков;
4. Патчизация;
5. ResNeXt;
6. Аналог windowed attention;
7. Функции активации;
8. Нормализации;
9. Даунсемплинг.



Пайплайн обучения

ImageNet Top1 76.1 → 78.8

Что?

- 90 эпох → 300 эпох;

Примеры аугментаций (Yun et al.)

Пайплайн обучения

ImageNet Top1 76.1 → 78.8

Что?

- 90 эпох → 300 эпох;
- Adam → AdamW;

Примеры аугментаций (Yun et al.)

Пайплайн обучения

ImageNet Top1 76.1 → 78.8

Что?

- 90 эпох → 300 эпох;
- Adam → AdamW;
- Аугментации: Mixup, Cutmix, . . . ;

Примеры аугментаций (Yun et al.)

Пайплайн обучения

ImageNet Top1 76.1 → 78.8

Что?

- 90 эпох → 300 эпох;
- Adam → AdamW;
- Аугментации: Mixup, Cutmix, . . . ;
- Регуляризации: Stochastic Depth, Label Smoothing.

Примеры аугментаций (Yun et al.)

Пайплайн обучения

ImageNet Top1 76.1 → 78.8

Что?

- 90 эпох → 300 эпох;
- Adam → AdamW;
- Аугментации: Mixup, Cutmix, . . . ;
- Регуляризации: Stochastic Depth, Label Smoothing.

Зачем?

- Пайлайн как у Swin Transformer и DeiT

Примеры аугментаций (Yun et al.)

Конфигурация блоков

ImageNet Top1 78.8 $\rightarrow$ 79.4

Что?

- Blocks(3, 4, 6, 3) $\rightarrow$ Blocks(3, 3, 9, 3)

Зачем?

- Swin-T: Blocks(1, 1, 3, 1);
- Объем вычислений на каждом уровне как у Swin-T.

Патчизация

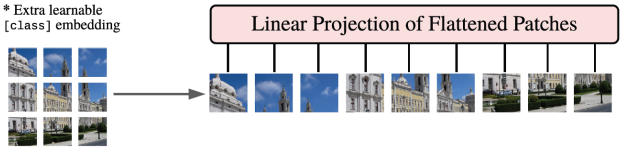
ImageNet Top1 79.4 → 79.5

Что?

- ResNet.in_conv: Conv2d(3, 64, kernel_size=7, stride=2) + MaxPool2d(2) →
  → Conv2d(3, 64, kernel_size=4, stride=4)

Зачем?

- Swin: токены — патчи 4×4;
- Более агрессивный даунсемплинг.



Патчизация ViT (Dosovitskiy et al.)

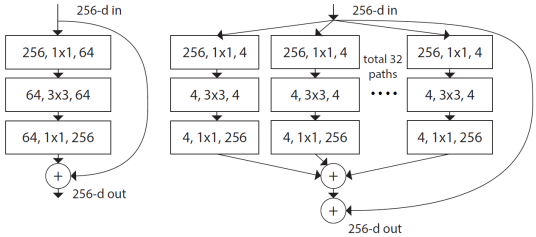
Что?

- Conv2d(64, 64, kernel_size=3) → 96×Conv2d(96, 1, kernel_size=1) + + 96×Conv2d(1, 1, kernel_size=3) + + 96×Conv2d(1, 96, kernel_size=1) + Sum()

  aka Depthwise(96, 96, hidden=96, kernel_size=3) свертка из MobileNet

Зачем?

- Разделение по-канальных и пространственных операций как в трансформерах;
- Количество каналов как в Swin-T.



Слева: блок ResNet, **справа**: блок ResNeXt (Xie et al.)

Inverted Bottleneck

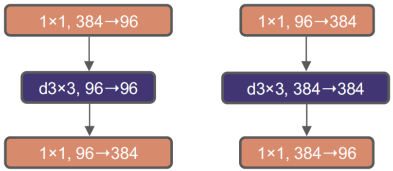
ImageNet Top1 80.5 → 80.6

Что?

- Depthwise(96, 96, hidden=96, kernel_size=3) → Depthwise(96, 96, hidden=384, kernel_size=3) aka inverted bottleneck из MobileNetV2

Зачем?

- В 4 раза больше параметров в скрытом слое как в трансформерах



Слева: блок ResNeXt, **справа**: блок с inverted bottleneck.

Порядок операций в depthwise свертке

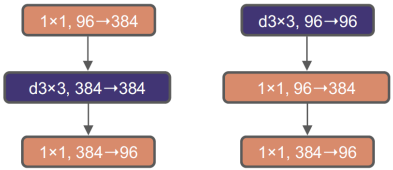
ImageNet Top1 80.6 → 79.9

Что?

- Depthwise(96, 96, hidden=384, kernel_size=3) →
  → 96×Conv2d(1, 1, kernel_size=3) + Sum() +
  + Conv2d(96, 384, kernel_size=1) +
  + Conv2d(384, 96, kernel_size=1)
  aka DepthwiseFirst(96, 96, hidden=384, kernel_size=3)

Зачем?

- В трансформерах attention применяется перед полносвязными слоями



Слева: блок с inverted bottleneck, **справа**: depthwise свертка сдвинута вверх.

Сверточные ядра больших размеров

ImageNet Top1 79.9 $\rightarrow$ 80.6

Что?

- DepthwiseFirst(96, 96, hidden=384, kernel_size=3) $\rightarrow$
  $\rightarrow$ DepthwiseFirst(96, 96, hidden=384, kernel_size=7)

Зачем?

- Depthwise свертка аналогична windowed attention;
- Размер окна в Swin 7×7.

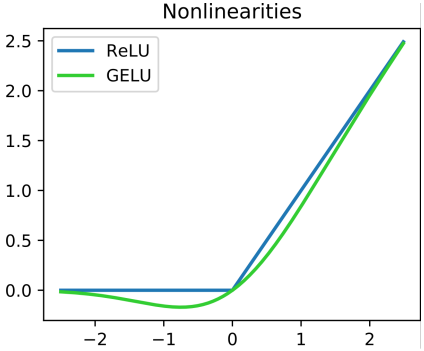
Функции активации

Что?

- ReLU → GELU

Зачем?

- GELU используется в BERT, GPT-2, ViT, Swin



Функции активации

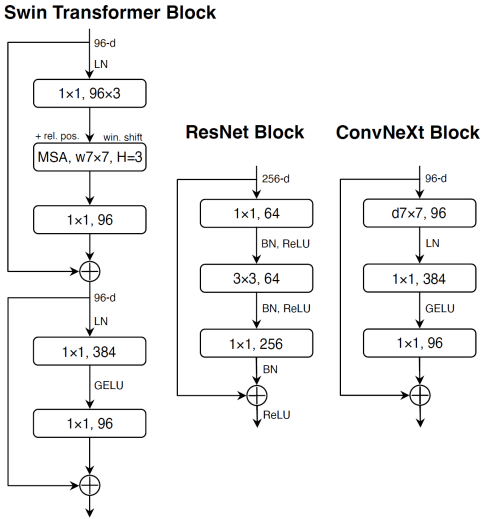
ImageNet Top1 80.6 → 81.3

Что?

- Удаляем из блока все активации кроме одной

Зачем?

- Одна активация на блок как в трансформерах



Нормализация

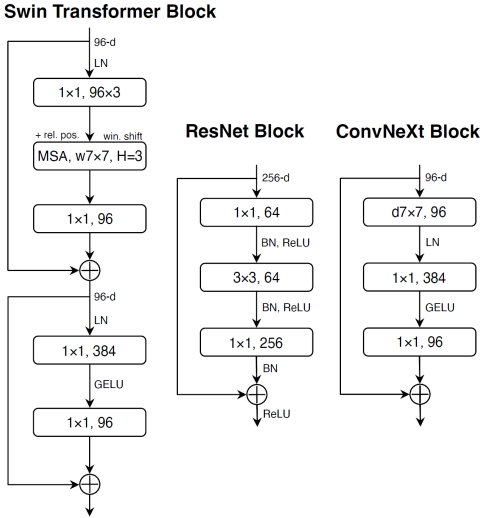
ImageNet Top1 81.3 → 81.4

Что?

- Удаляем из блока все батч-нормализации кроме одной

Зачем?

- Меньше нормализаций на блок как в трансформерах



Нормализация

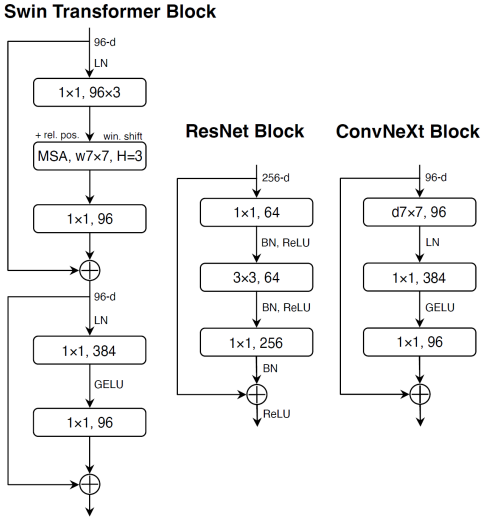
ImageNet Top1 81.3 → 81.4

Что?

- Заменяем батч-нормализацию на LayerNorm

Зачем?

- LayerNorm используется в трансформерах



Даунсемплинг

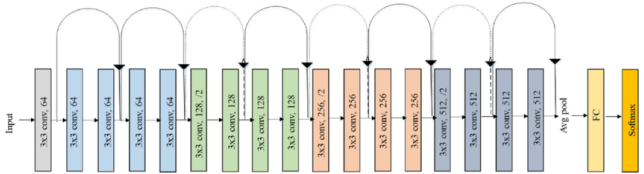
ImageNet Top1 81.4 → 82.0

Что?

- ResNet.downsample: Conv2d(64, 128, kernel_size=3, stride=2) →
  → LayerNorm(96) +
  + Conv2d(96, 192, kernel_size=2, stride=2) +
  + LayerNorm(192) +
  + DepthwiseFirst(192, 192, hidden=768, kernel_size=7);
- Не делаем skip-connection через даунсемплинг.

Зачем?

- Стратегия даунсемплинга как в Swin



(He et al.)

Список источников

1. A ConvNet for the 2020s
2. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows
3. CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features
4. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale
5. Aggregated Residual Transformations for Deep Neural Networks
6. Deep Residual Learning for Image Recognition

ConvNeXt

от ResNet до SwinTransformer

Настя Городилова

БПМИ191

Факультет Компьютерных Наук
НИУ ВШЭ

4 октября 2022 г.

Когда и кем выпущена статья?

- ArXiv – январь 2022
- CVPR – июнь 2022

Авторы - Facebook AI Research

Откуда взялась идея: архитектура ResNeXt (CVPR 2017)

Базовые статьи

1. Aggregated residual transformations for deep neural networks. CVPR, 2017 (ResNeXt)
2. Revisiting resnets: Improved training and scaling strategies. NeurIPS, 2021
3. Resnet strikes back: An improved training procedure in timm. CVPR, 2021

Прямые продолжения

1. VidConv: A modernized 2D ConvNet for Efficient Video Recognition, 2022
2. More ConvNets in the 2020s: Scaling up Kernels Beyond 51×51 using Sparsity, 2022

Плюсы статьи

- Конкурентоспособность ConvNeXt по отношению к Transformer-ам
- "Playthrough"

Минусы статьи

- Нет ветвления в поставленных экспериментах