Архитектуры CNN (I)

Даниил Лысухин, ML Team Lead @ Ozon

План

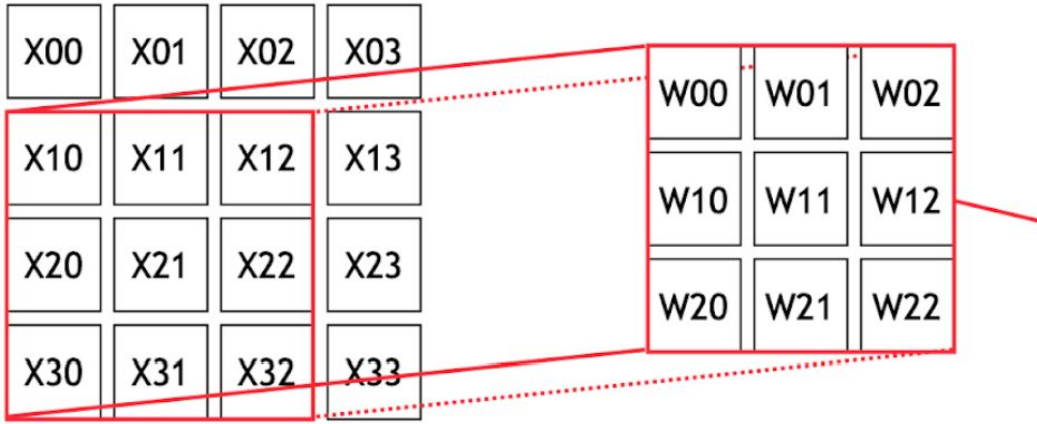
- Recap: CNN
- Соревнование ImageNet
- От классики к CNN
- Эра ResNet
- Итоги

Recap: CNN

Recap: свертка

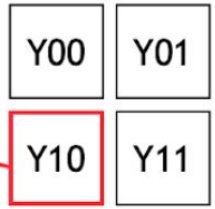
$$Y_{ij} = (X * W)_{ij} = \sum_{u=0}^{K-1} \sum_{v=0}^{K-1} X_{i+u,j+v} W_{uv}$$

Входной сигнал **X_ij**

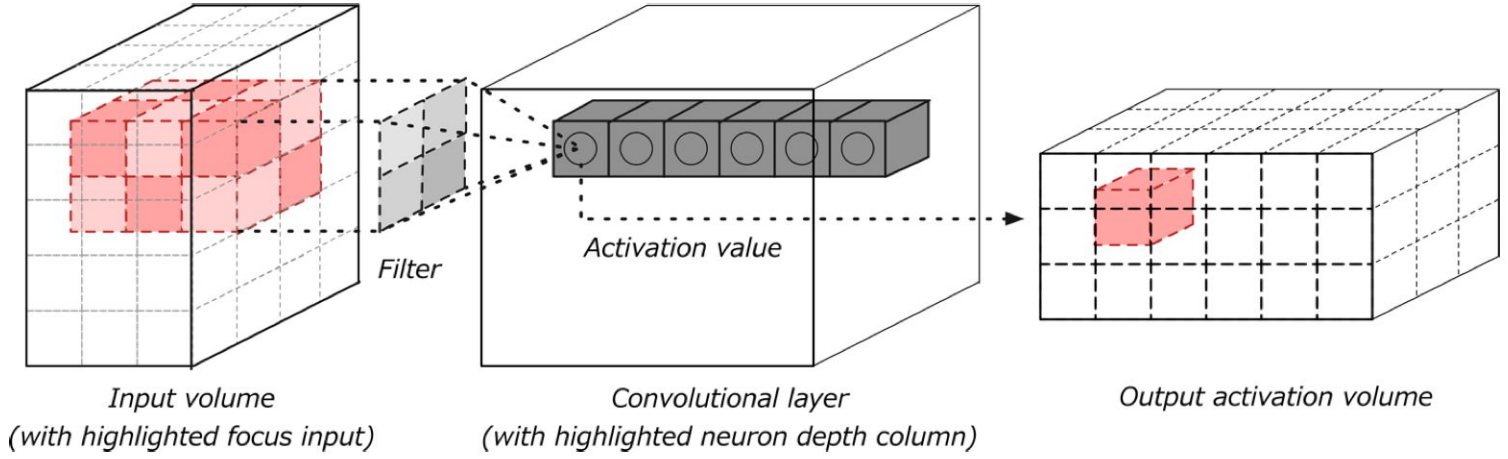


Ядро свертки **W**
Размер **3x3**
Веса W_ij

Результат свертки **Y_ij**



Recap: сверточный слой



Recap: сверточный слой - параметры

- Число ядер (оно же: "число фильтров", "ширина слоя")
- Размер ядра (3x3, 5x5, 3x5, ...)
- Padding (добавление значений по краям входного сигнала)
- Stride (величина шага скользящего окна, бывает и >1)
- Dilation (добавление "разреженности" в матрицу ядра свертки)

Recap: сверточный слой - параметры

- Число ядер (оно же: "число фильтров", "ширина слоя")
- Размер ядра (3x3, 5x5, 3x5, ...)
- Padding (добавление значений по краям входного сигнала)
- Stride (величина шага скользящего окна, бывает и >1)
- Dilation (добавление "разреженности" в матрицу ядра свертки)

Влияют на H/W выходного тензора

Определяют рецептивное поле ячеек выходного тензора (нейронов)

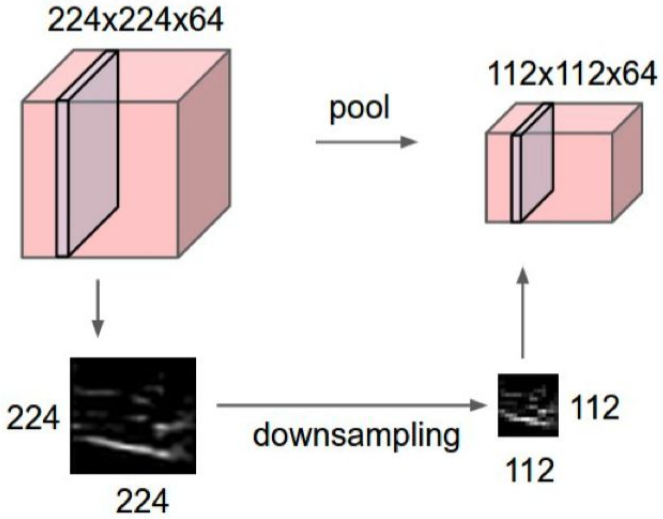
Recap: сверточный слой

- Подсчитать число обучаемых параметров сверточного слоя
  - Размер ядра свертки 3x3
    - bias = True
  - Число каналов (= "глубина") входного тензора 64
  - Число каналов выходного тензора 128

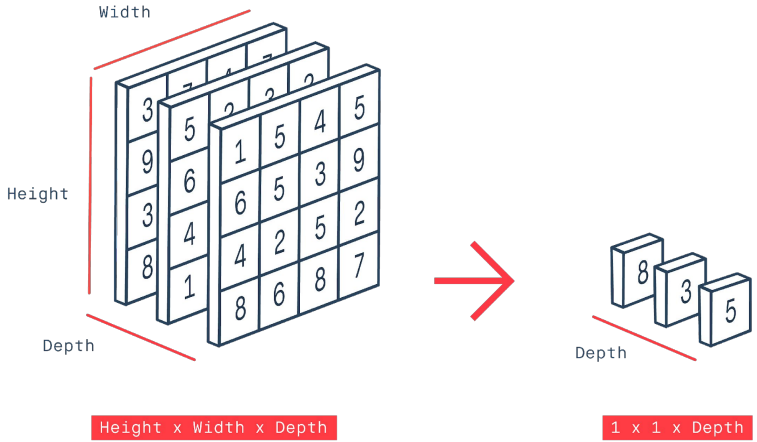
Recap: сверточный слой

- Подсчитать число обучаемых параметров сверточного слоя
  - Размер ядра свертки 3x3
    - bias = True
  - Число каналов (= "глубина") входного тензора 64
  - Число каналов выходного тензора 128
- Число параметров для 1 ядра свертки: 3 x 3 x 64 + 1 = 577
- Для 128 сверток: 577 x 128 = 73856

Recap: pooling



“Обычный” pooling



Global pooling

CNN, LeNet-5, 1989-1998

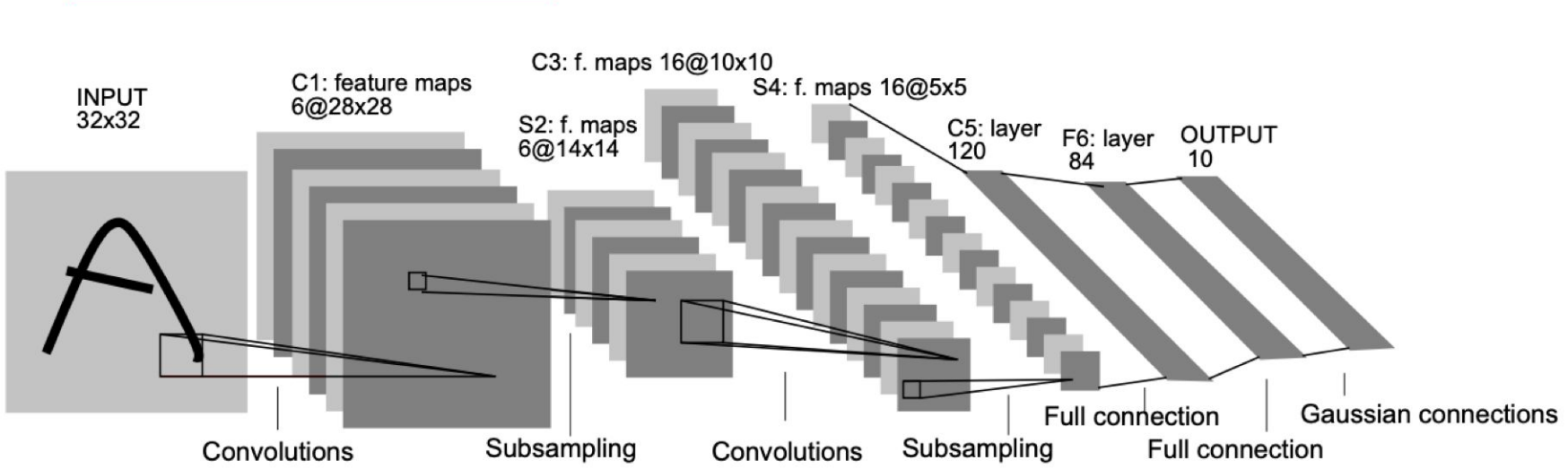


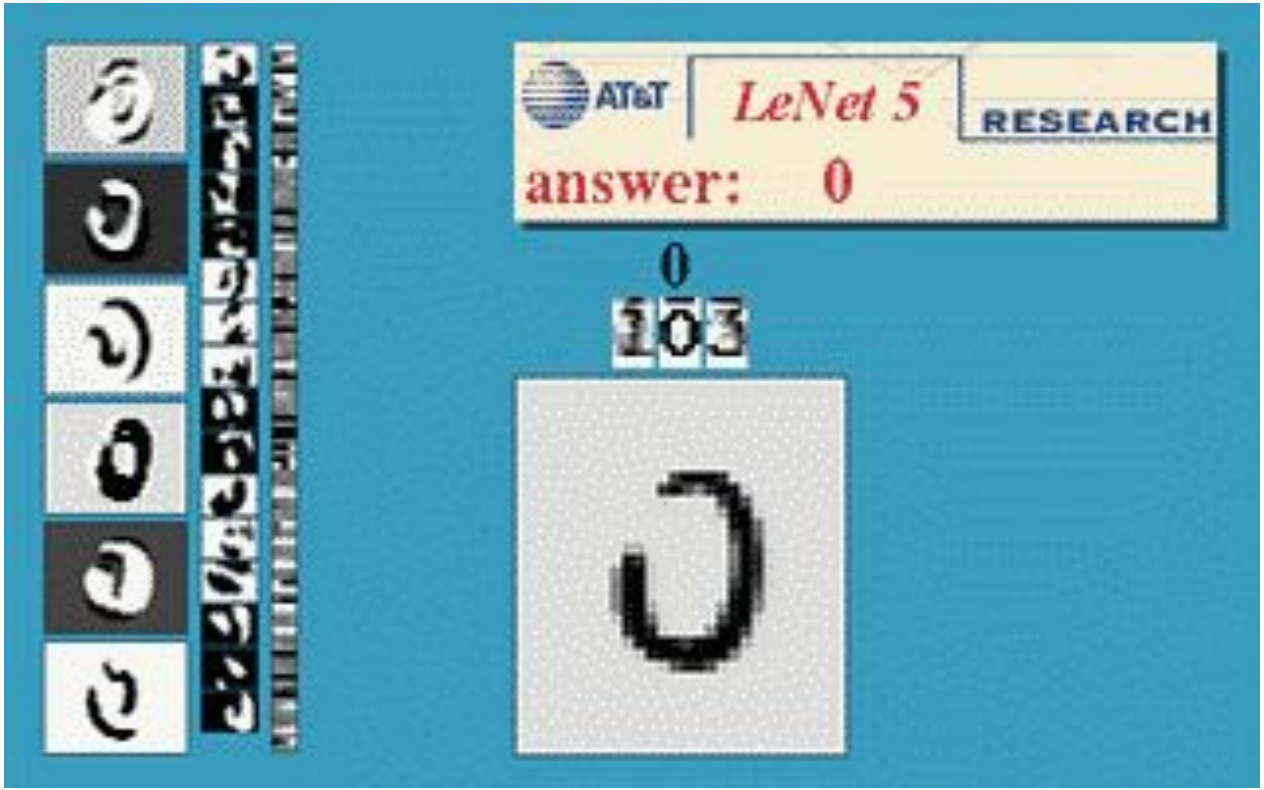
Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

LeNet

- Первая публикация - 1989, последняя - 1998
- Задача - распознавание рукописных символов
- Слои - сверточный, полносвязный, пулинг, активация (tanh)
  - В "версии" 1998 года - экзотический слой RBF
- Почитать на досуге:
  - Andrej Karpathy, Deep Neural Nets: 33 years ago and 33 years from now

LeNet Demo MNIST



http://yann.lecun.com/exdb/lenet/

ImageNet

ImageNet

- База изображений на основе иерархии WordNet
  - > 20k категорий
  - > 15M примеров
- Разметка = 1 категория + bounding box

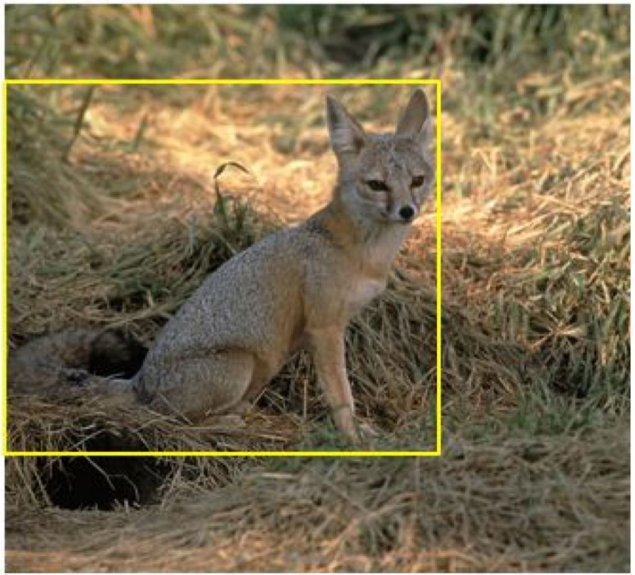
- Есть проблемы
  - 1 категория для фото независимо от числа объектов
  - Разнообразие классов (зачем столько пород собак?)

ImageNet

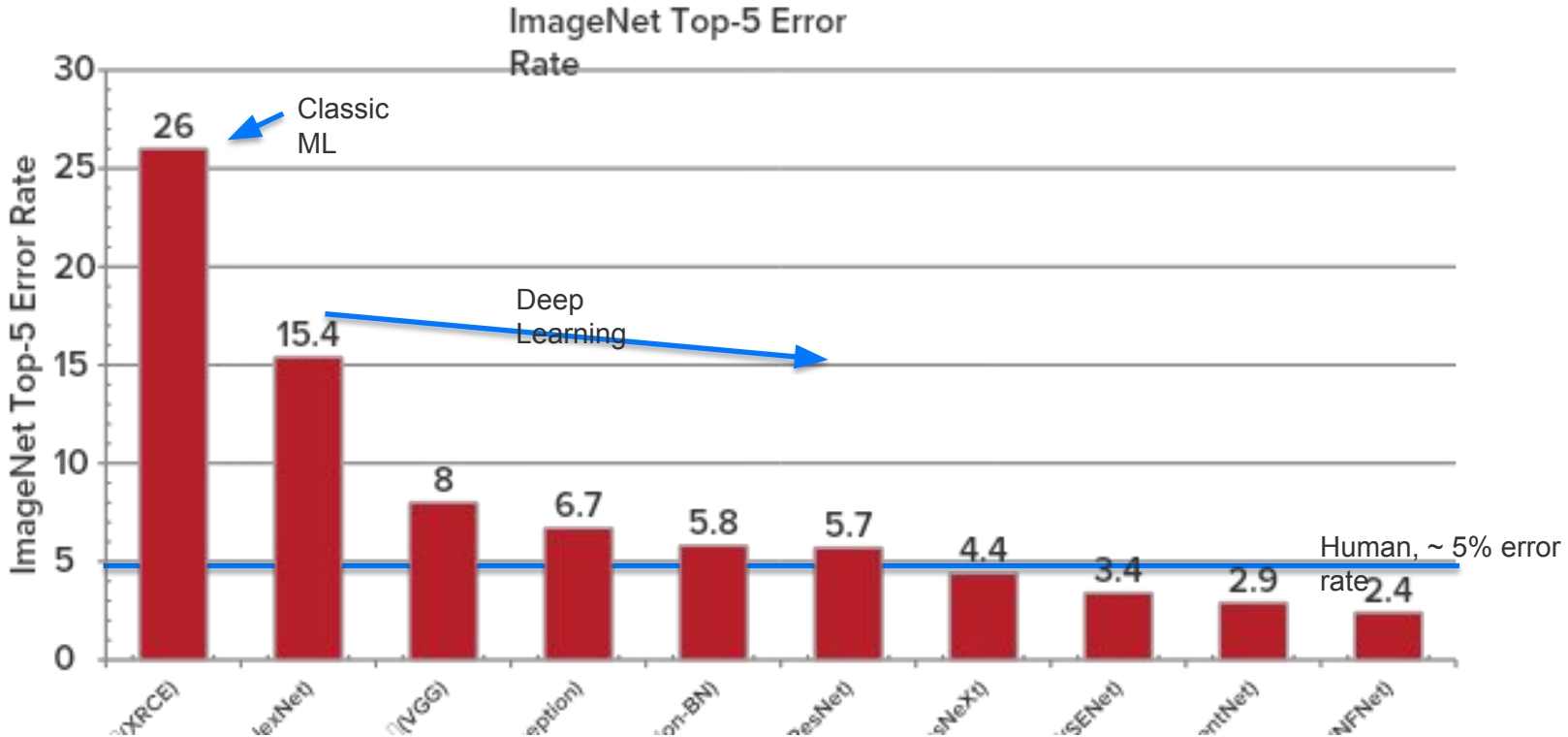

grille
mushroom
cherry
Madagascar cat
mite
container ship
motor scooter
leopard

ILSVRC

- ImageNet Large-scale Visual Recognition Challenge
  - Соревнование на данных ImageNet
- Подзадачи:
  - Классификация (1000 классов)
  - Локализация

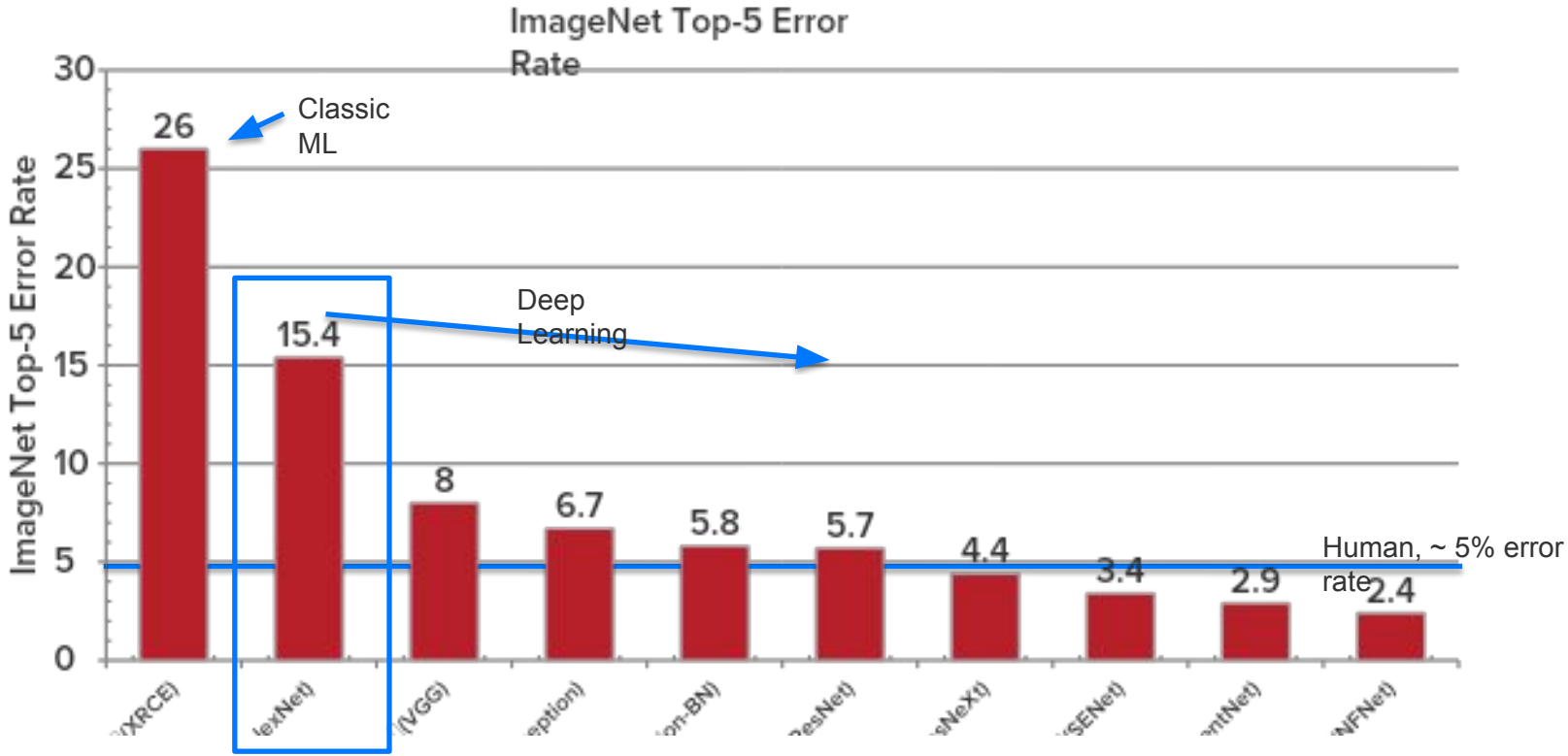


ILSVRC



https://paperswithcode.com/sota/image-classification-on-imagenet

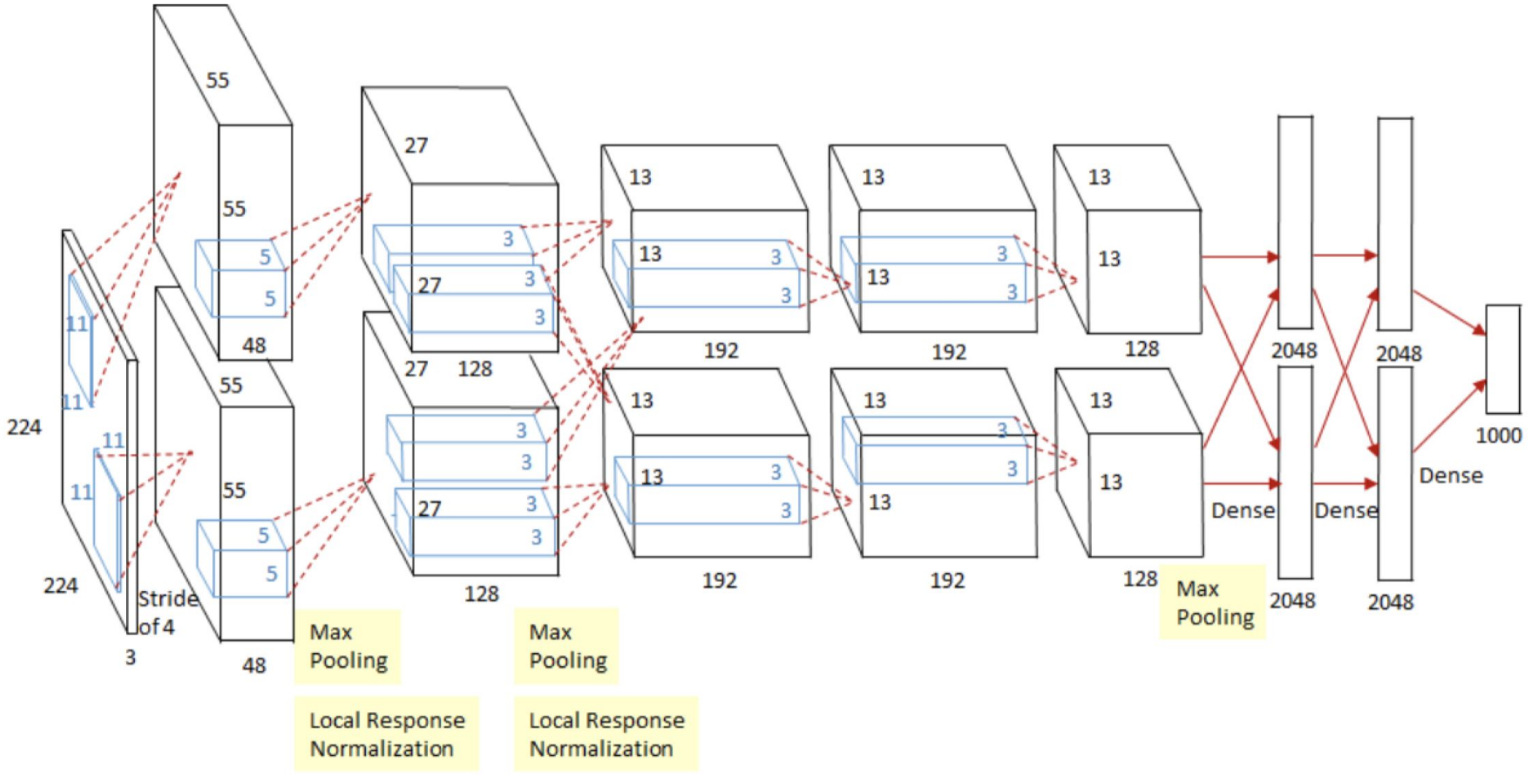
От классики к CNN


ImageNet Top-5 Error Rate
ImageNet Top-5 Error Rate
30
25
20
15
10
5
0
26
15.4
8
6.7
5.8
5.7
4.4
3.4
2.9
2.4
Classic ML
Deep Learning
Human, ~ 5% error rate

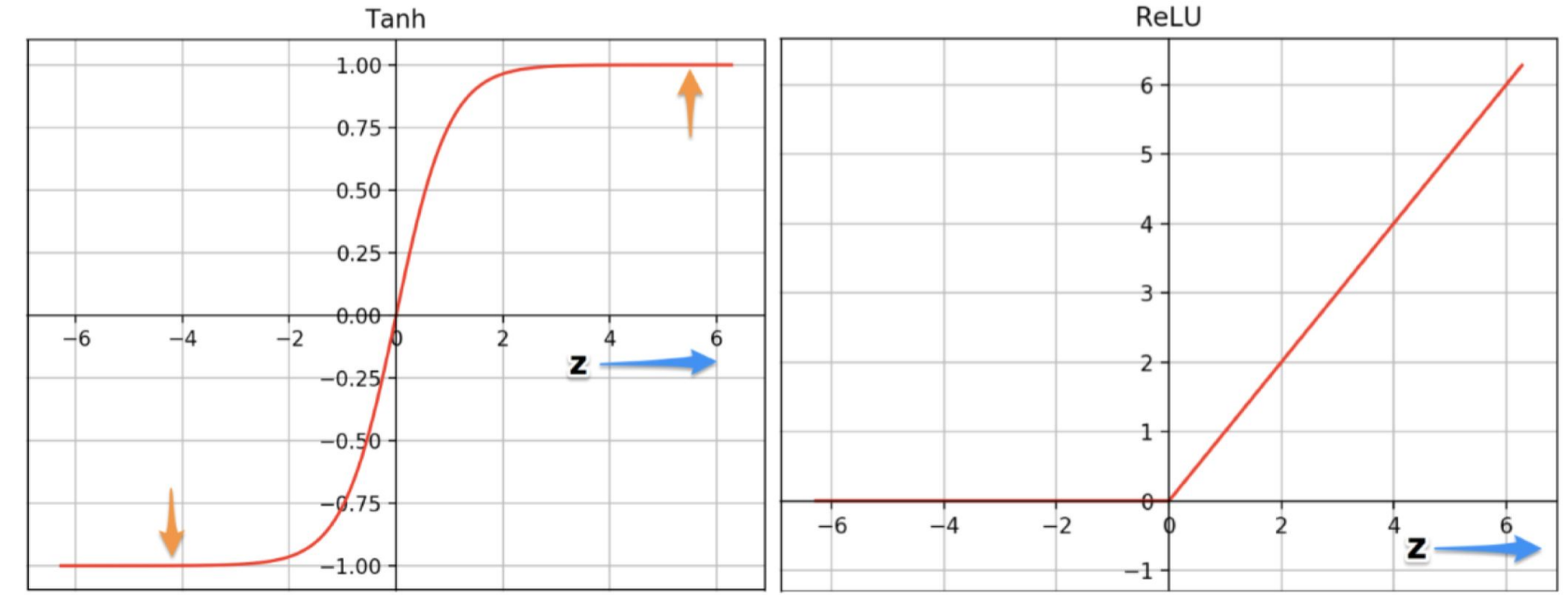
AlexNet (2012)

- ImageNet Classification with Deep Convolutional Neural Networks
- Первый победитель ILSVRC на основе нейросетей
- **60M** параметров
- **2** ветви на **2 GPU**
- Свертки от **3x3** до **11x11**
- ReLU, Dropout (в FC)
- **Test-time Augmentation**

AlexNet (2012)

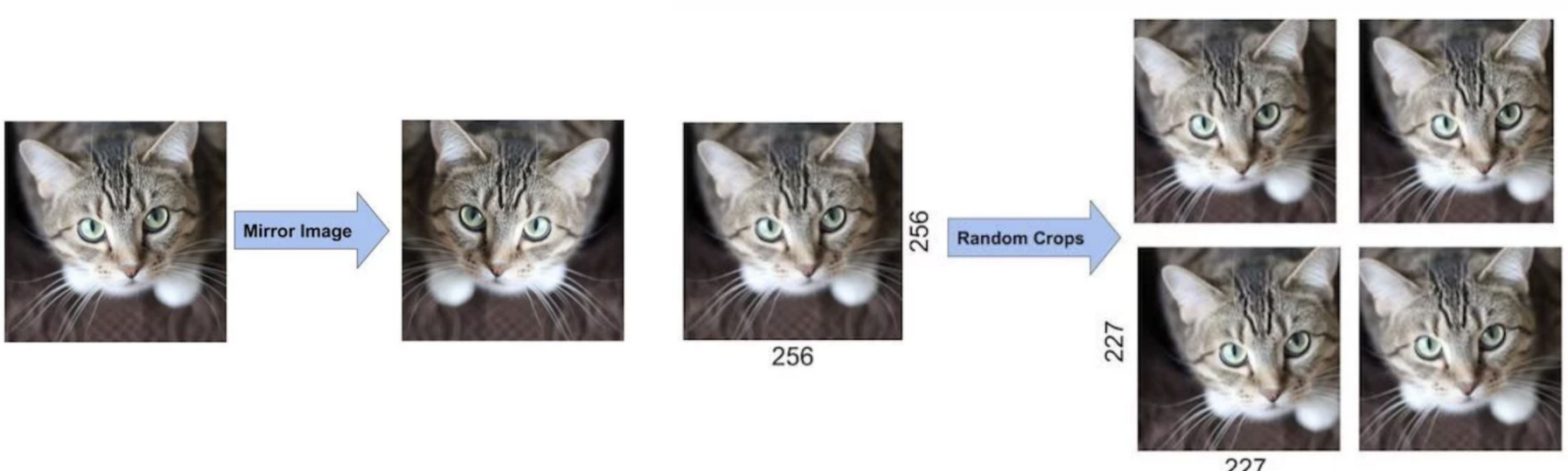


AlexNet (2012)



AlexNet (2012)

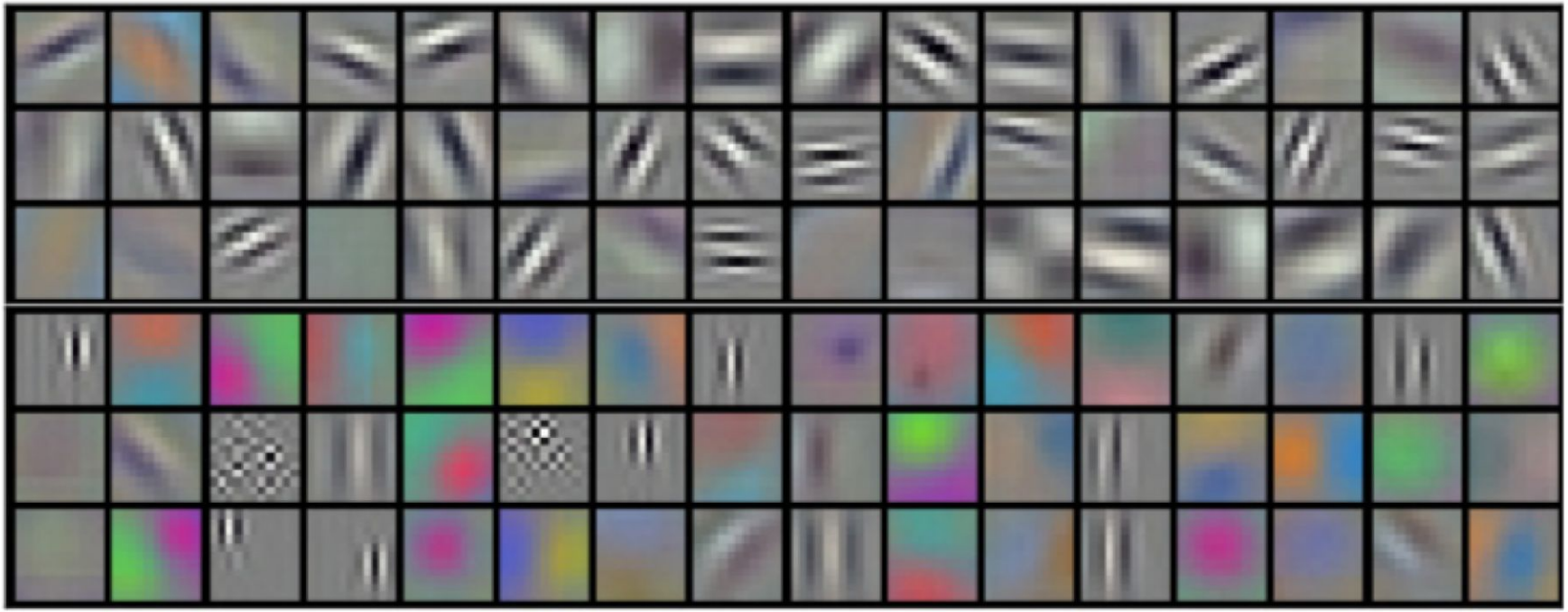
- Аугментации на обучении:
  - Случайные кропы 227x227 (из 256x256)
  - Зеркальное отражение



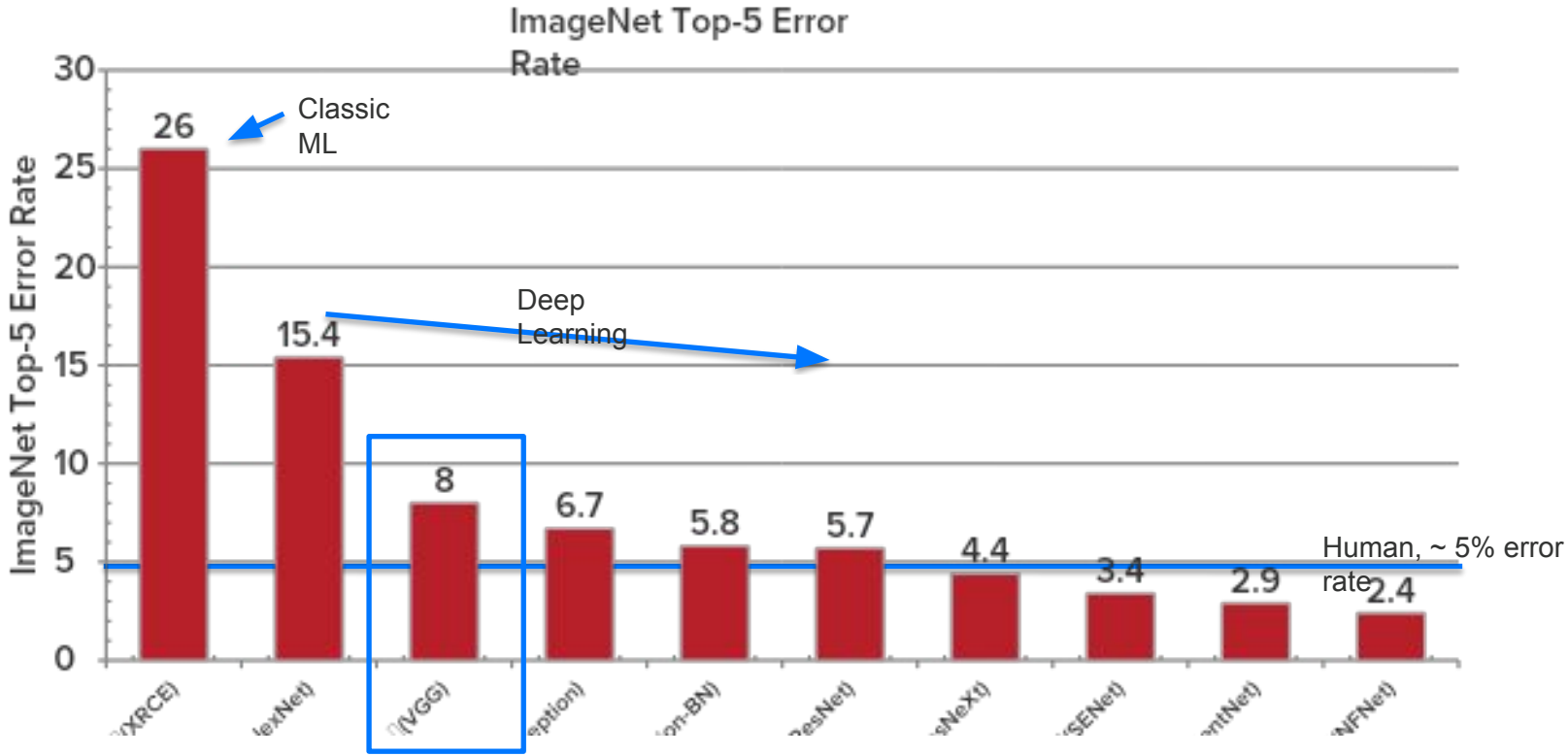
AlexNet (2012)

- Аугментации на инференсе (Test-time Augmentation, TTA):
  - Агрегирование предсказаний для 1 изображения по 10 его "вариациям":
    - 5 кропов (углы + центр)
    - x2 с учетом отраженных

AlexNet (2012)



Learnings of First convolution layer on image of size 224X224X3


ImageNet Top-5 Error Rate
ImageNet Top-5 Error Rate
30
25
20
15
10
5
0
26
15.4
8
6.7
5.8
5.7
4.4
3.4
2.9
2.4
Classic ML
Deep Learning
Human, ~ 5% error rate

VGG (2014)

- Very Deep Convolutional Networks for Large-Scale Image Recognition
  - Свертки 1x1 и 3x3
  - 138M параметров у самой большой вариации (VGG-19)

VGG (2014)

- Семейство моделей
- Отличаются глубиной
  - VGG-11 / 13 / 16 / 19
- Огромная часть параметров - в выходных слоях для классификации

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

VGG (2014)

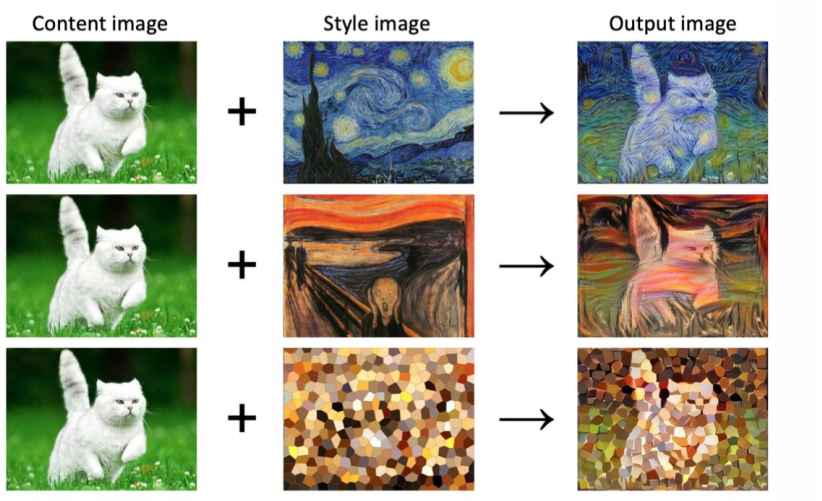
- Число параметров:
  - 3x3x512x512 ~ 2.4M

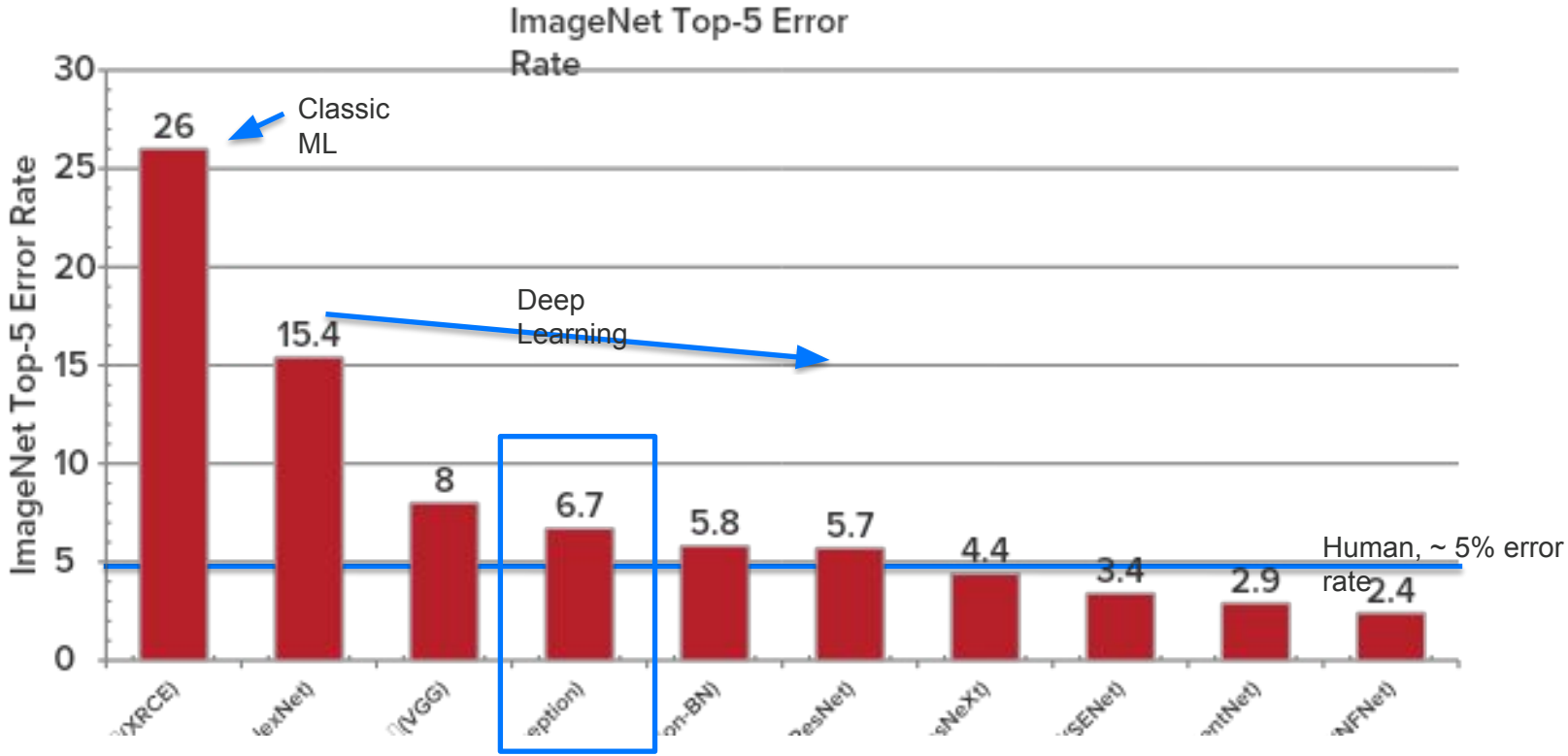
- Число параметров:
  - 7x7x512x4096 ~ 102.7M

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

VGG (2014)

- Оказалось, что предобученная VGG извлекает “хорошие”
- признаки, пригодные для переиспользования
  - Например, в задаче Style Transfer



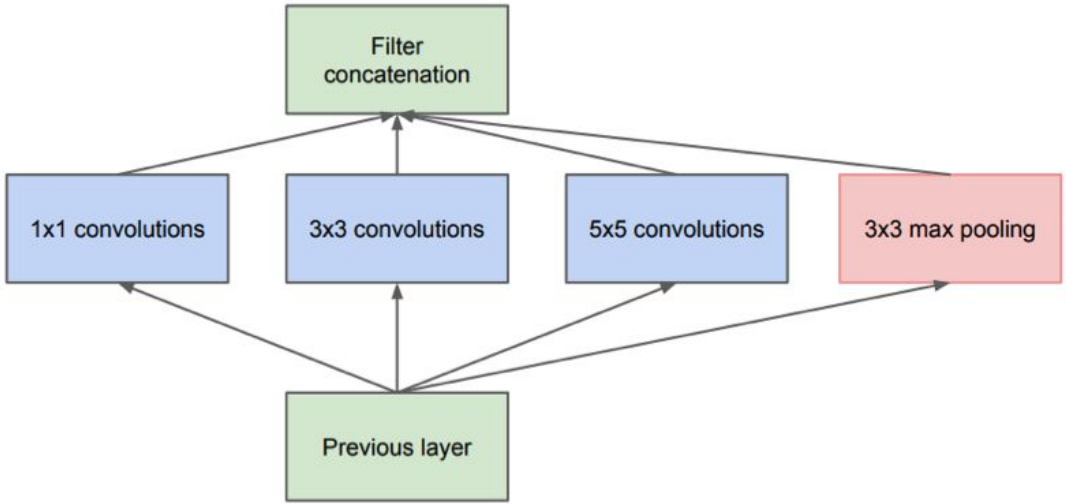

ImageNet Top-5 Error Rate
ImageNet Top-5 Error Rate
30
25
20
15
10
5
0
26
15.4
8
6.7
5.8
5.7
4.4
3.4
2.9
2.4
Classic ML
Deep Learning
Human, ~ 5% error rate

Inception / GoogLeNet (2014)

- Going Deeper with Convolutions
- Одновременное извлечение признаков **разного масштаба**
  - Новая структурная единица - InceptionBlock
- **Дополнительные выходы** с функцией потерь
- **Global Average Pooling** перед классификацией
- Мало параметров (**5M**)

Inception / GoogLeNet (2014)

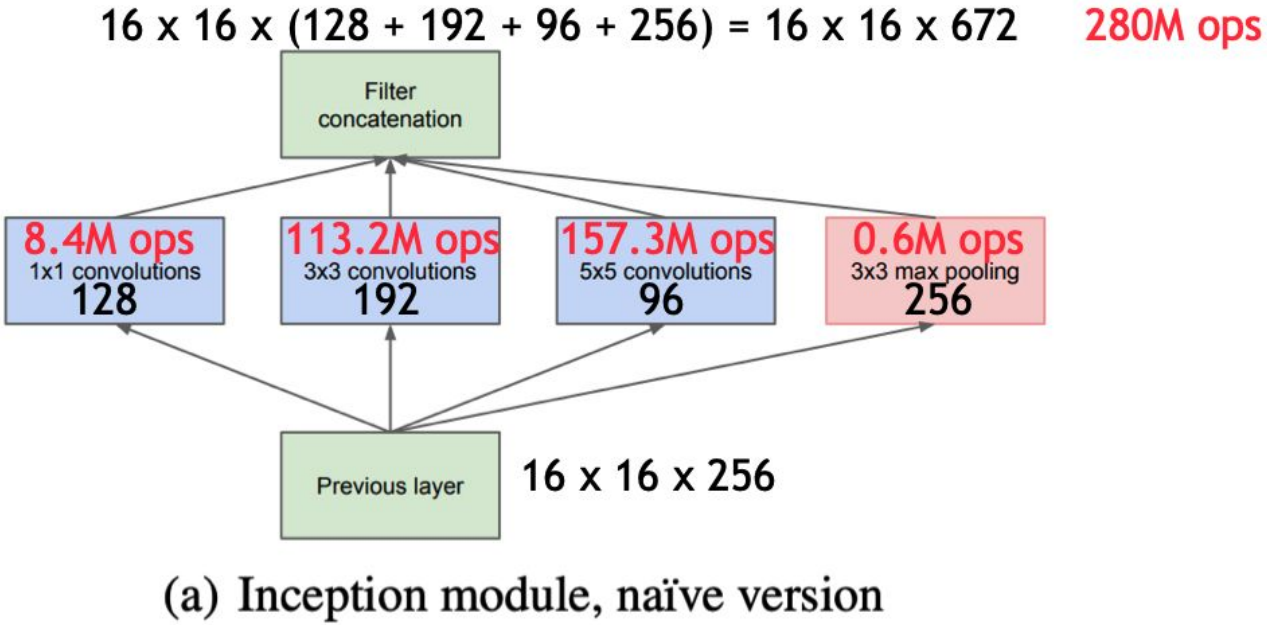
- InceptionBlock (“ванильный” вариант)



(a) Inception module, naïve version

Inception / GoogLeNet (2014)

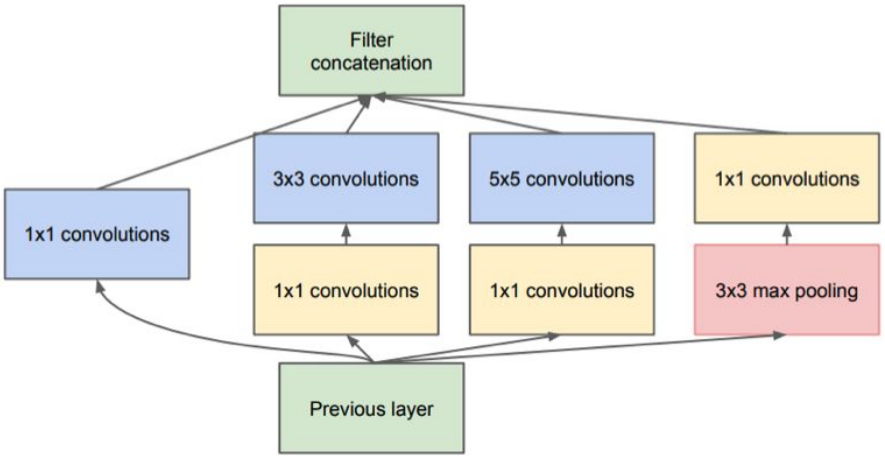
- InceptionBlock (“ванильный” вариант)



(a) Inception module, naïve version

Inception / GoogLeNet (2014)

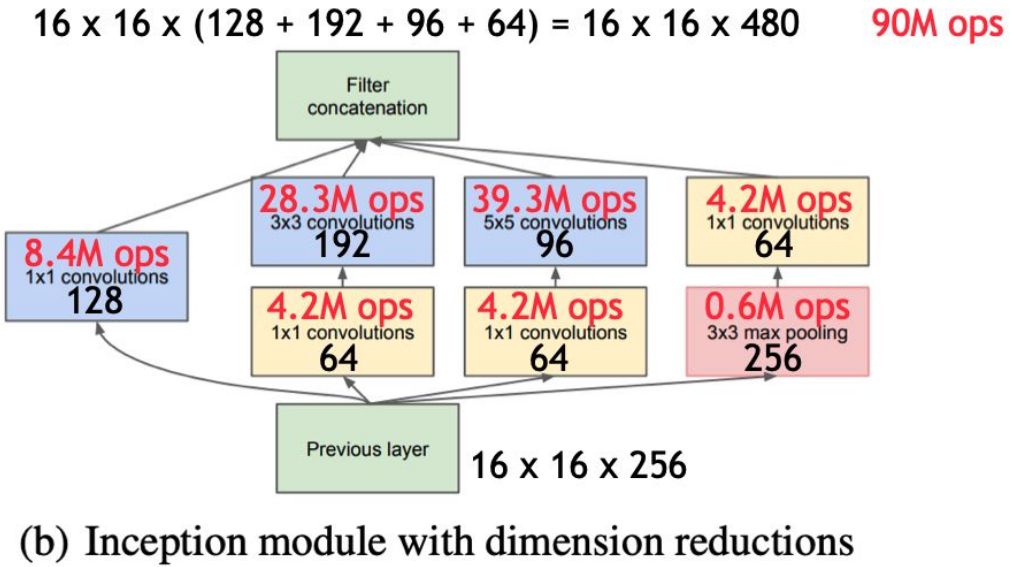
- InceptionBlock (bottleneck-вариант)



(b) Inception module with dimension reductions

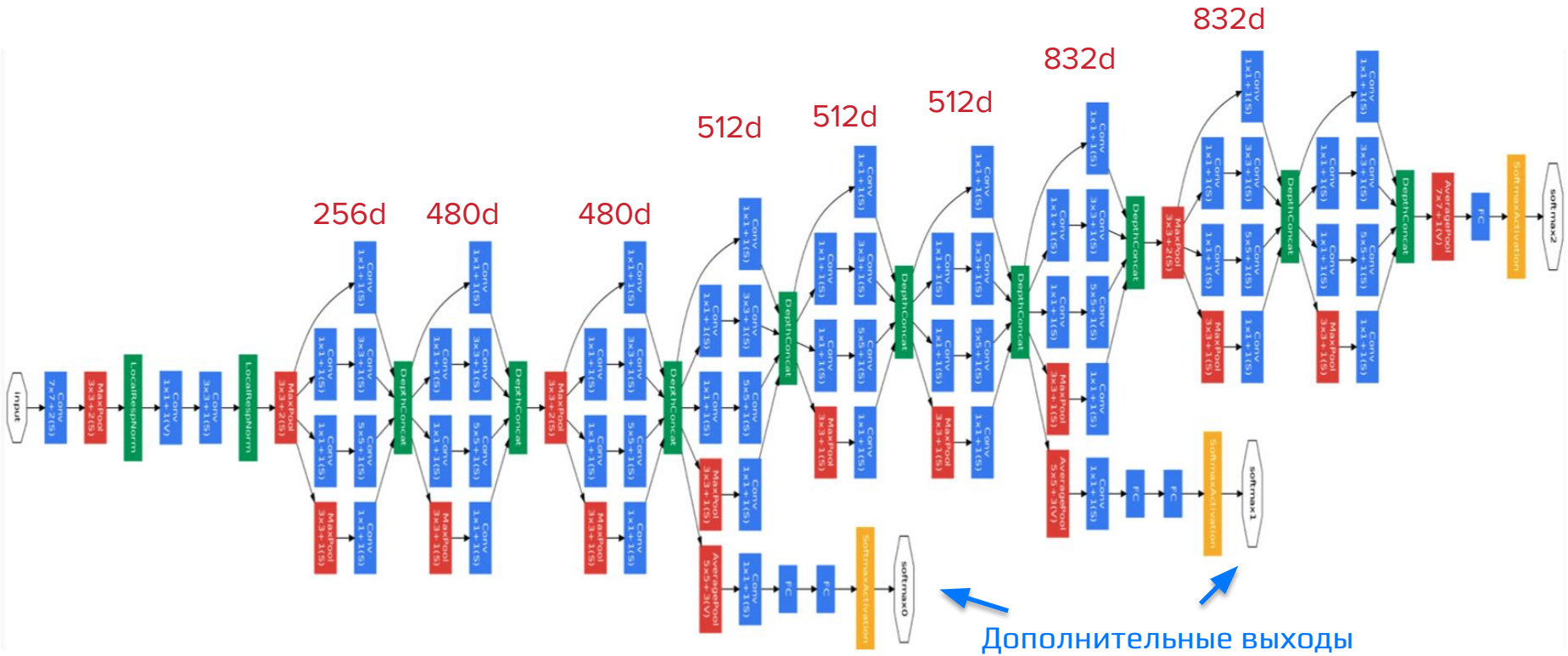
Inception / GoogLeNet (2014)

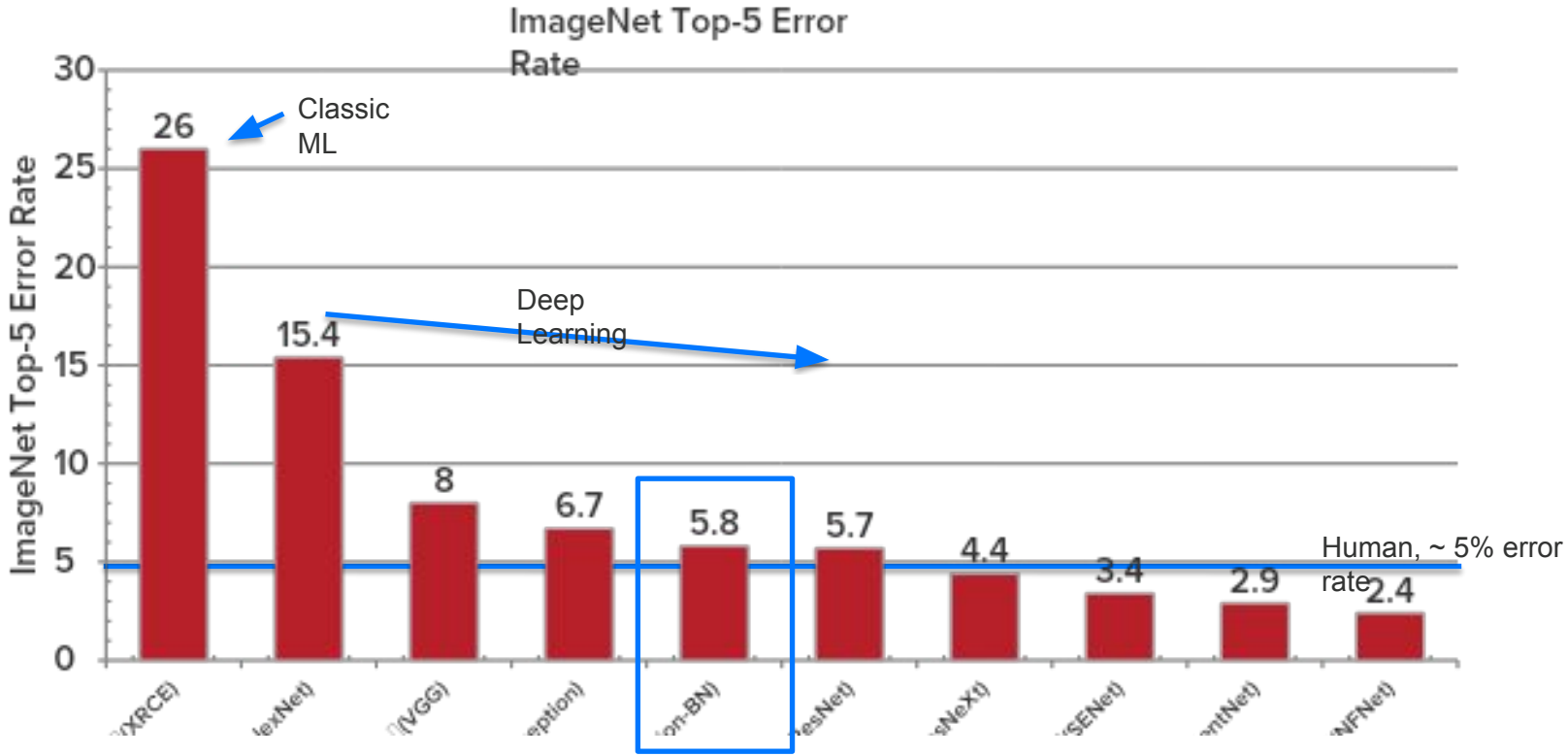
- InceptionBlock (bottleneck-вариант)



(b) Inception module with dimension reductions

Inception / GoogLeNet (2014)




ImageNet Top-5 Error Rate
ImageNet Top-5 Error Rate
30
25
20
15
10
5
0
26
15.4
8
6.7
5.8
5.7
4.4
3.4
2.9
2.4
Classic ML
Deep Learning
Human, ~ 5% error rate

Inception-BN / Inception-V2 (2015)

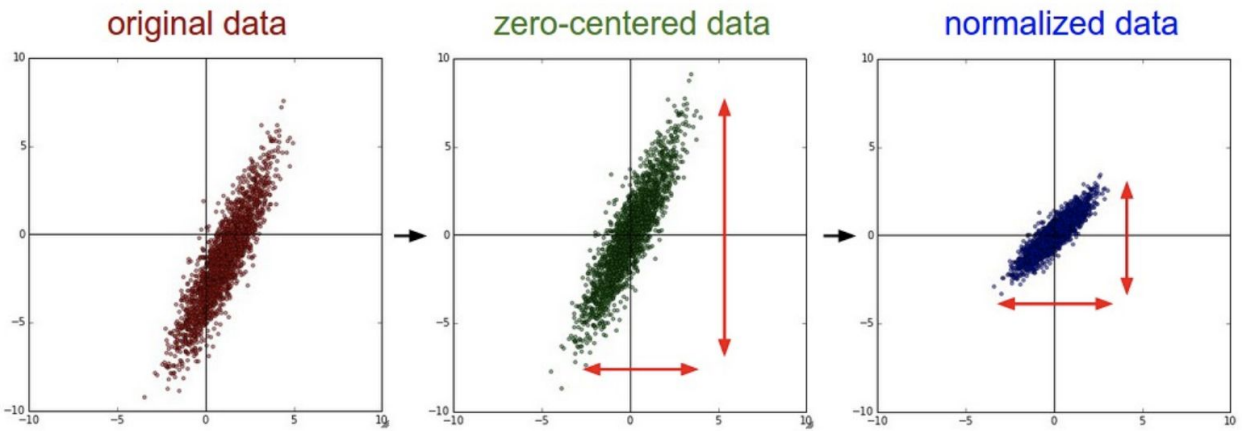
- Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift
- Новый слой нормализации **Batch Normalization**
  - Значительное **ускорение сходимости** при обучении

BatchNormalization

- Проблема: internal covariate shift
  - Обновление параметров слоя ведет к изменению распределения его выходных значений - эффект домино
- Некоторые функции активации“насыщаются” (sigmoid, tanh) и “хорошо” пропускают градиент только в окрестности 0

BatchNormalization

- Решение: делаем пере-нормировку активаций по ходу сети
  - Только при обучении по мини-батчам!



https://cs231n.github.io/neural-networks-2/

BatchNormalization

- Решение: делаем пере-нормировку активаций по ходу сети
  - Только при обучении по мини-батчам!

BatchNormalization

- Пусть $\boldsymbol{x}$ - вектор активаций после некоторого слоя сети для батча размера m

BatchNormalization

- Пусть ***x*** - вектор активаций после некоторого слоя сети для батча размера *m*
- Посчитаем среднее по батчу: $\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$

BatchNormalization

- Пусть ***x*** - вектор активаций после некоторого слоя сети для батча размера *m*
- Посчитаем среднее по батчу: $\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$
- Посчитаем дисперсию (по батчу): $\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$

BatchNormalization

- Пусть ***x*** - вектор активаций после некоторого слоя сети для батча размера *m*
- Посчитаем среднее по батчу: $\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$
- Посчитаем дисперсию (по батчу): $\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$
- Отнормируем: $\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$

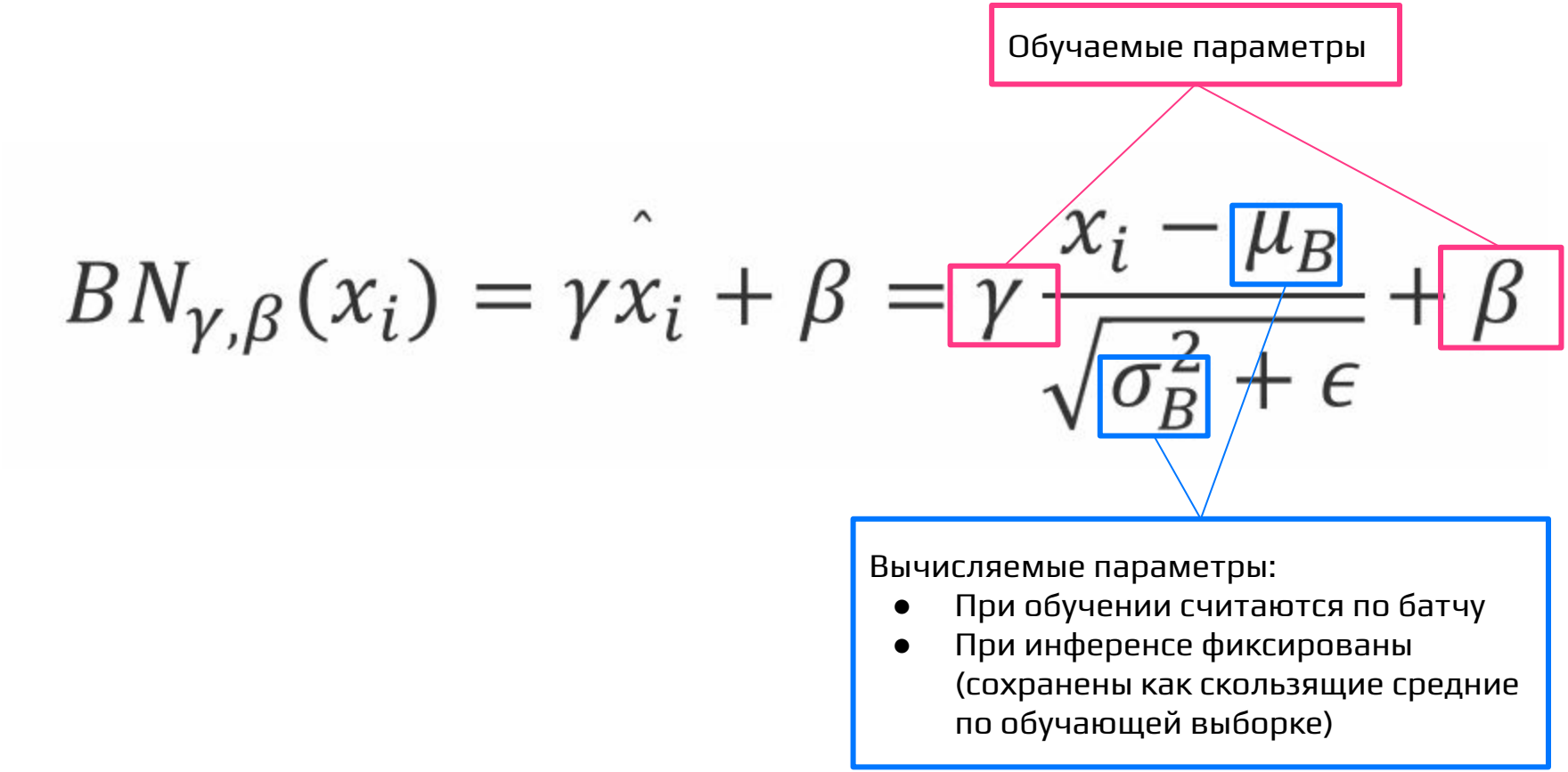
BatchNormalization

- Пусть ***x*** - вектор активаций после некоторого слоя сети для батча размера *m*
- Посчитаем среднее по батчу: $\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$
- Посчитаем дисперсию (по батчу): $\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$
- Отнормируем: $\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$
- Добавим обучаемое масштабирование: $BN_{\gamma,\beta}(x_i) = \gamma\hat{x}_i + \beta$

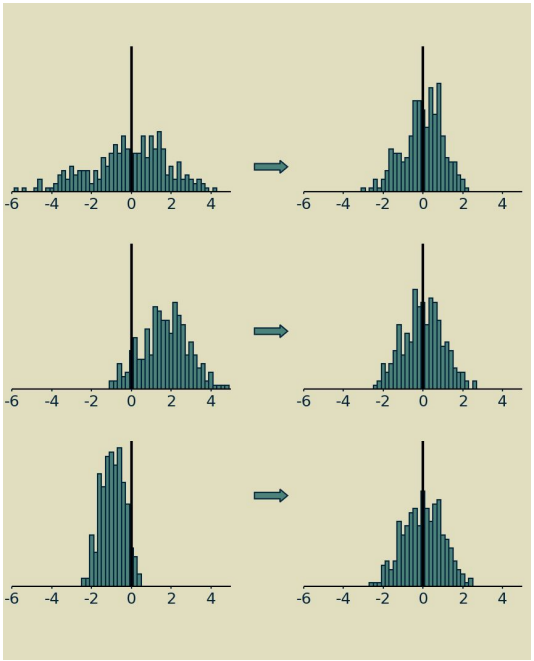
BatchNormalization

$$BN_{\gamma,\beta}(x_i) = \gamma \hat{x}_i + \beta = \gamma \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta$$

BatchNormalization



BatchNormalization



https://e2eml.school/batch_normalization.html

BatchNormalization

- Backprop - ?

$$BN_{\gamma,\beta}(x_i) = \gamma \hat{x}_i + \beta = \gamma \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta$$

BatchNormalization

- Backprop - ?
- Не забыть, что среднее и дисперсия - функции от ***x***
- Решение тут

$$BN_{\gamma,\beta}(x_i) = \gamma \hat{x}_i + \beta = \gamma \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta$$

BatchNormalization

- Ускорение сходимости

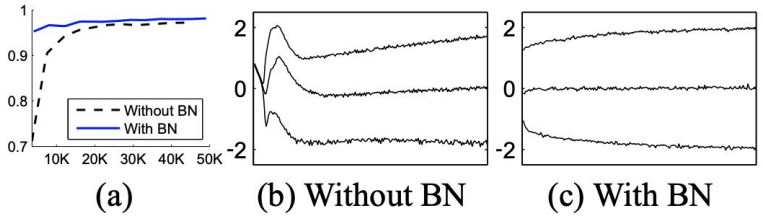


Figure 1: (a) *The test accuracy of the MNIST network trained with and without Batch Normalization, vs. the number of training steps. Batch Normalization helps the network train faster and achieve higher accuracy.* (b, c) *The evolution of input distributions to a typical sigmoid, over the course of training, shown as* $\{15, 50, 85\}$*th percentiles. Batch Normalization makes the distribution more stable and reduces the internal covariate shift.*

https://arxiv.org/pdf/1502.03167v3.pdf

BatchNormalization

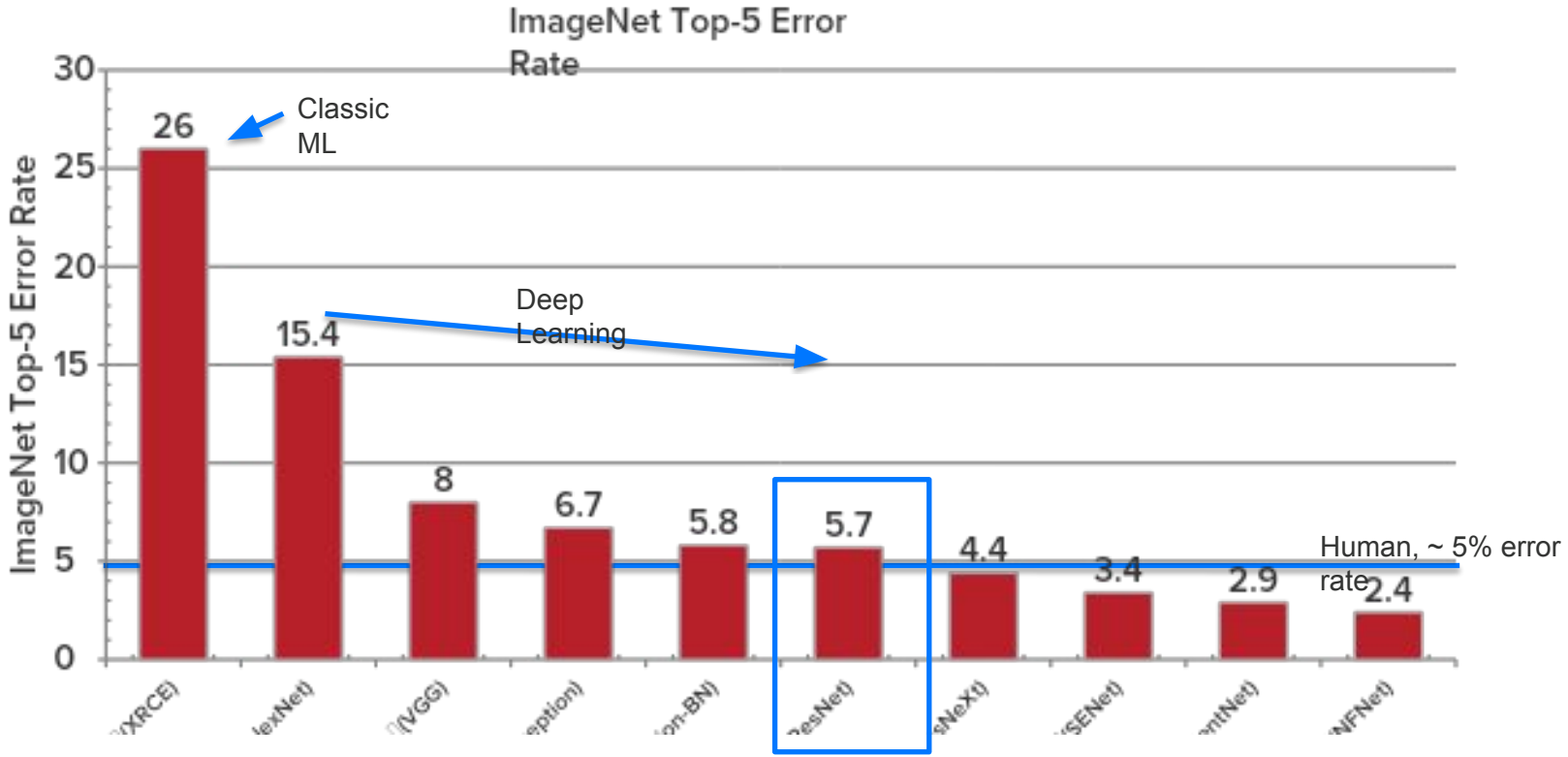
- Ускорение сходимости
- Добавление регуляризующего эффекта
- Повышение устойчивости к инициализации весов

- Требуется достаточно большой батч при обучении
- Отличается поведение не обучении и инференсе
- Требует дополнительных ресурсов (память, вычисления)

https://arxiv.org/pdf/1502.03167v3.pdf

BatchNormalization

- How Does Batch Normalization Help Optimization?



https://paperswithcode.com/sota/image-classification-on-image

Эра ResNet

ResNet (2015)

- Deep Residual Learning for Image Recognition
- Наращивание глубины сети с помощью Residual Connections
  - До 152 слоев

ResNet (2015)

- Наблюдение: увеличение глубины сети не обязательно приводит к улучшению качества

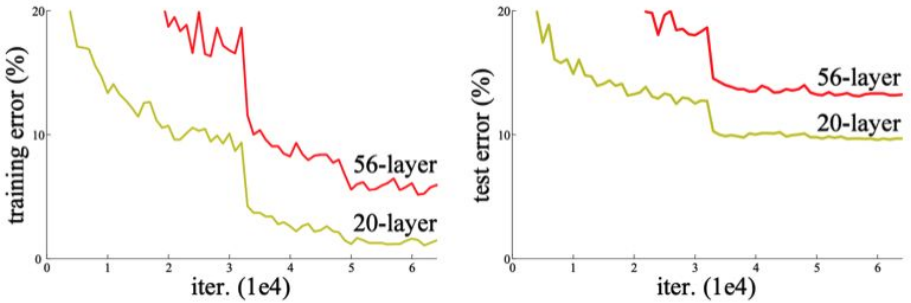


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

https://arxiv.org/pdf/1512.03385v1.pdf

ResNet (2015)

- Наблюдение: увеличение глубины сети не обязательно приводит к улучшению качества

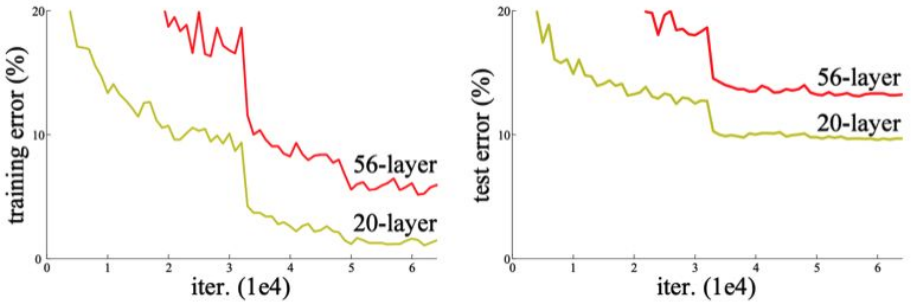


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

Почему?

https://arxiv.org/pdf/1512.03385v1.pdf

ResNet (2015)

- Идея: позволить сети “пропускать” слои, если они, например, не улучшают качество

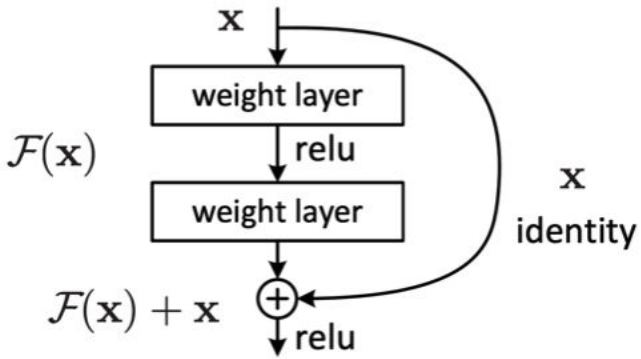


Figure 2. Residual learning: a building block

https://arxiv.org/pdf/1512.03385v1.pdf

ResNet (2015)

- Идея: позволить сети “пропускать” слои, если они, например, не улучшают качество

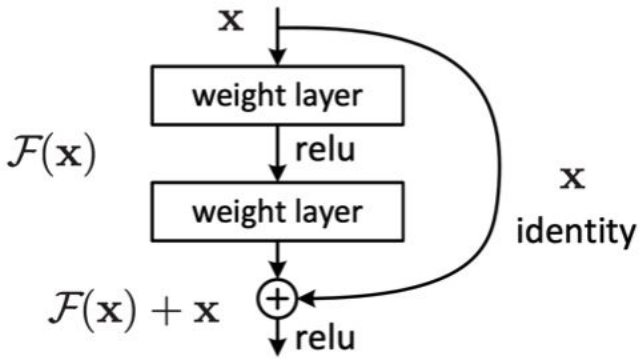


Figure 2. Residual learning: a building block.

Что с градиентом?

https://arxiv.org/pdf/1512.03385v1.pdf

ResNet (2015)

- Идея: позволить сети “пропускать” слои, если они, например, не улучшают качество

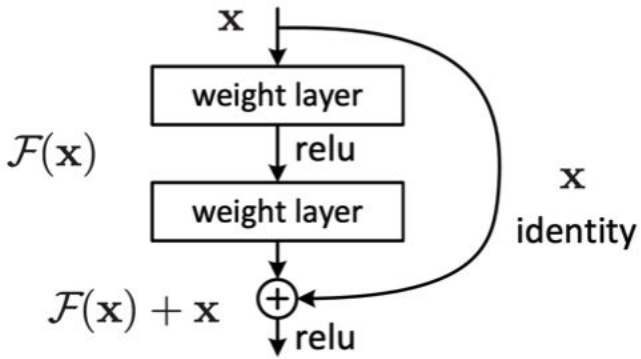


Figure 2. Residual learning: a building block.

Что с градиентом?

$$y = F(x) + x$$

https://arxiv.org/pdf/1512.03385v1.pdf

ResNet (2015)

- Идея: позволить сети “пропускать” слои, если они, например, не улучшают качество

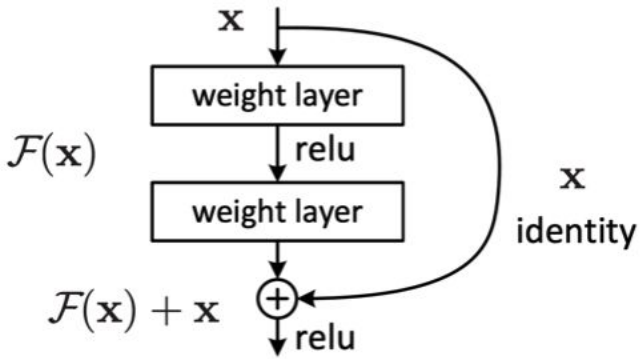


Figure 2. Residual learning: a building block.

Что с градиентом?

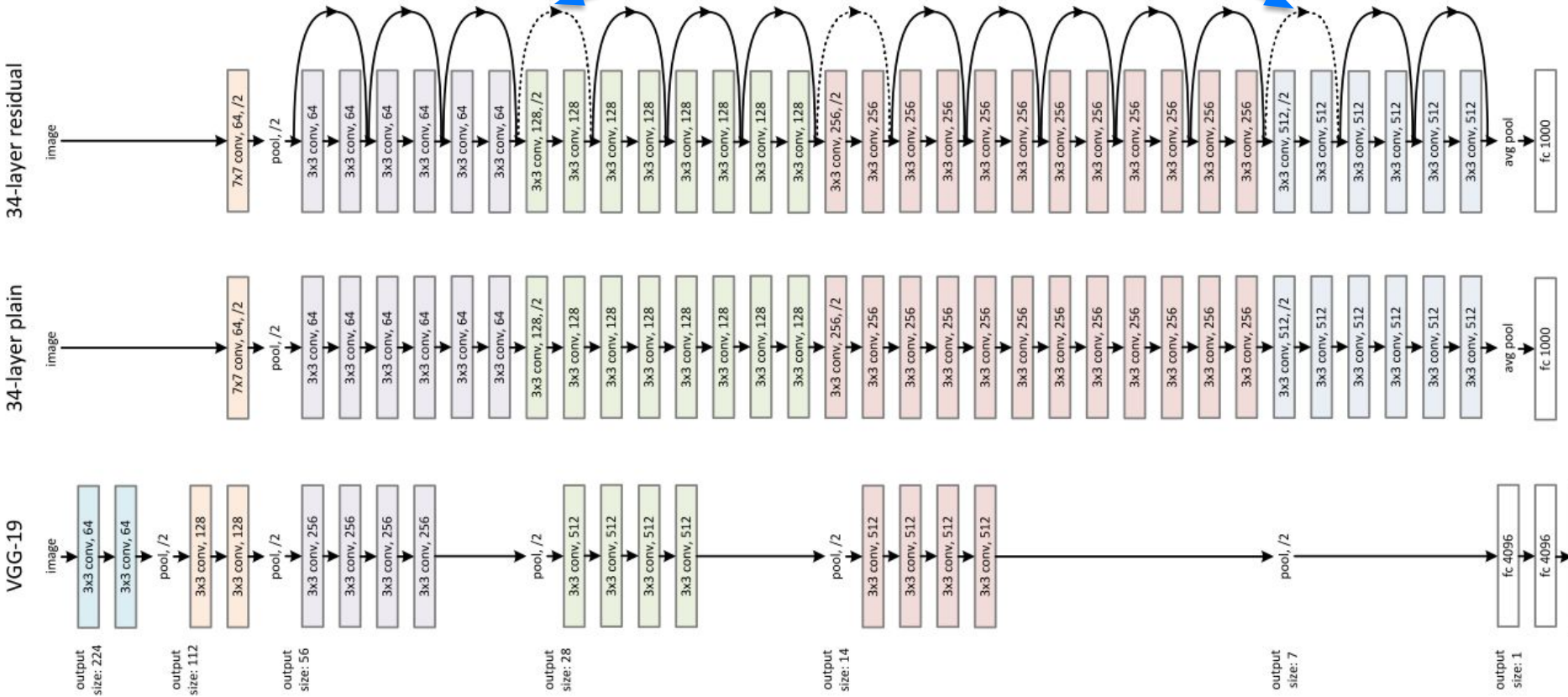
$$y = F(x) + x$$

$$dy/dx = F'(x) + 1$$

https://arxiv.org/pdf/1512.03385v1.pdf

ResNet (2015)

Рост ширины блоков
+ пулинг



ResNet (2015)

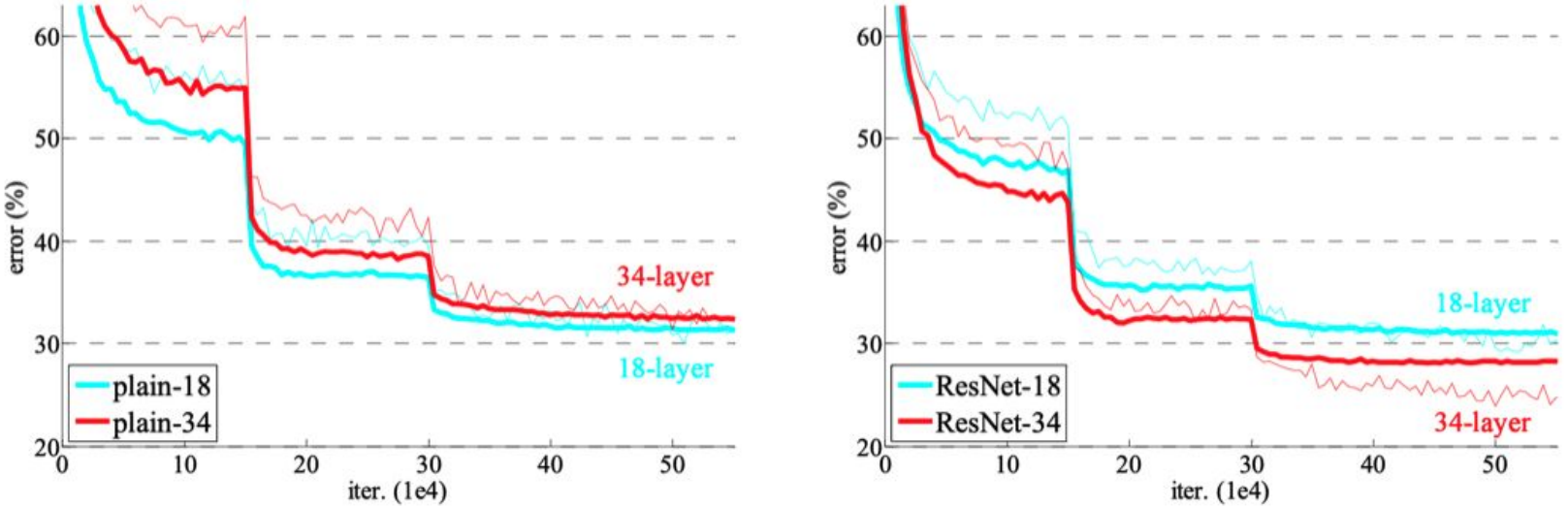


Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

https://arxiv.org/pdf/1512.03385v1.pdf

ResNet (2015)

- 2 типа базовых блоков
  - ResNet-18/34: “обычный” блок (слева)
  - ResNet-50/101/152: bottleneck-блок (справа)

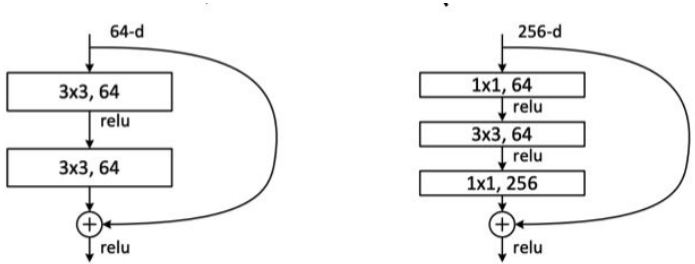
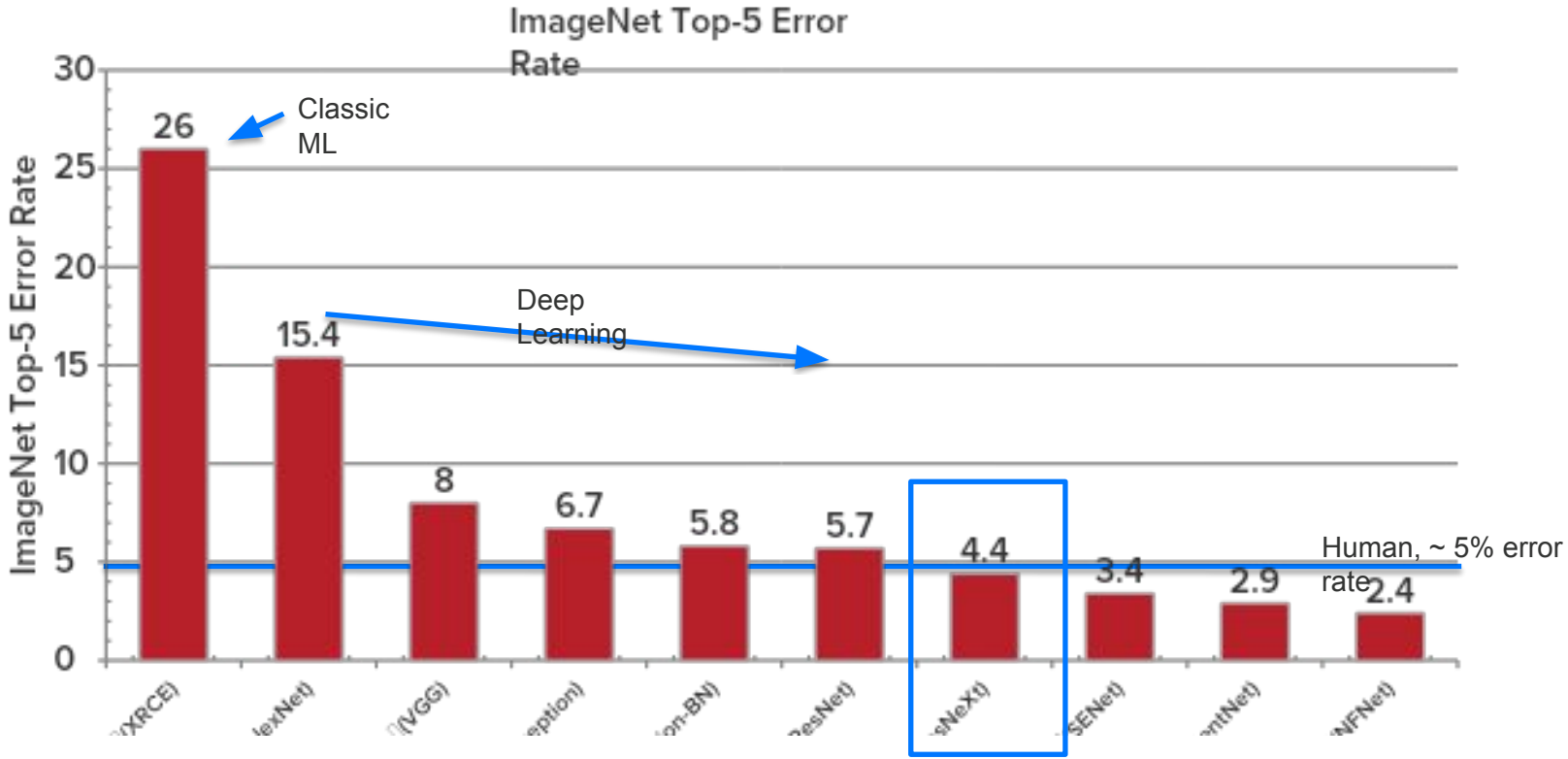


Figure 5. A deeper residual function $\mathcal{F}$ for ImageNet. Left: a building block (on 56×56 feature maps) as in Fig. 3 for ResNet-34. Right: a “bottleneck” building block for ResNet-50/101/152.


ImageNet Top-5 Error Rate
ImageNet Top-5 Error Rate
30
25
20
15
10
5
0
26
15.4
8
6.7
5.8
5.7
4.4
3.4
2.9
2.4
Classic ML
Deep Learning
Human, ~ 5% error rate

ResNet (2015)

- Aggregated Residual Transformations for Deep Neural Networks
- Совмещение идей о параллельных вычислениях в рамках одного блока (Inception) и Residual Connections (ResNet)

ResNeXt (2017)

- Aggregated Residual Transformations for Deep Neural Networks
- Совмещение идей о параллельных вычислениях в рамках одного блока (Inception) и Residual Connections (ResNet)

ResNeXt (2017)

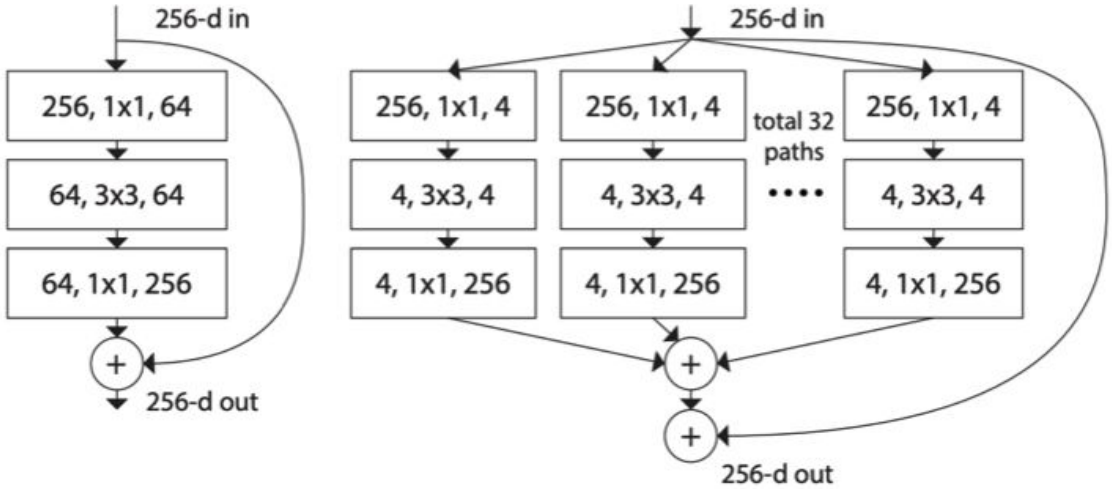


Figure 1. **Left**: A block of ResNet [14]. **Right**: A block of ResNeXt with cardinality = 32, with roughly the same complexity. A layer is shown as (# in channels, filter size, # out channels).

ResNeXt (2017)

ResNeXt-50_32x4d

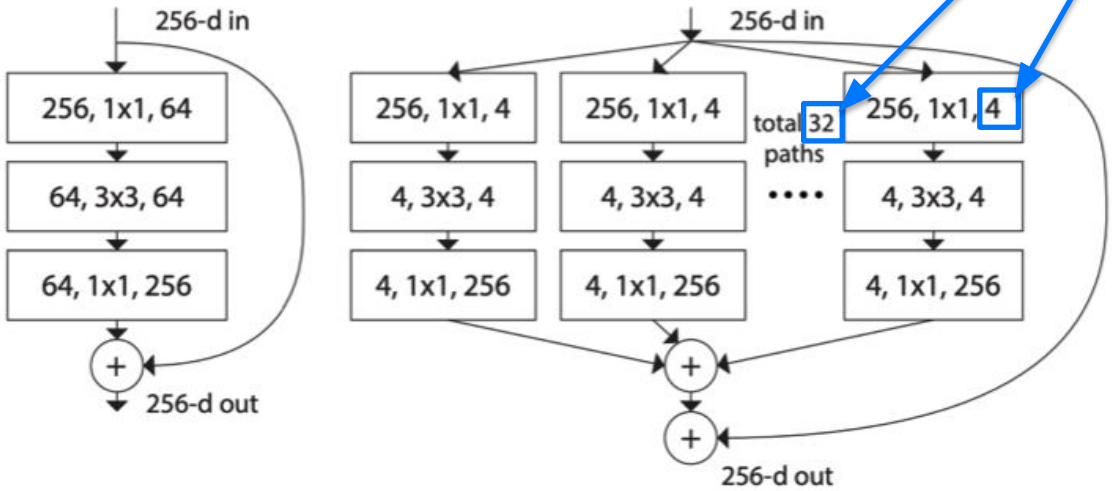
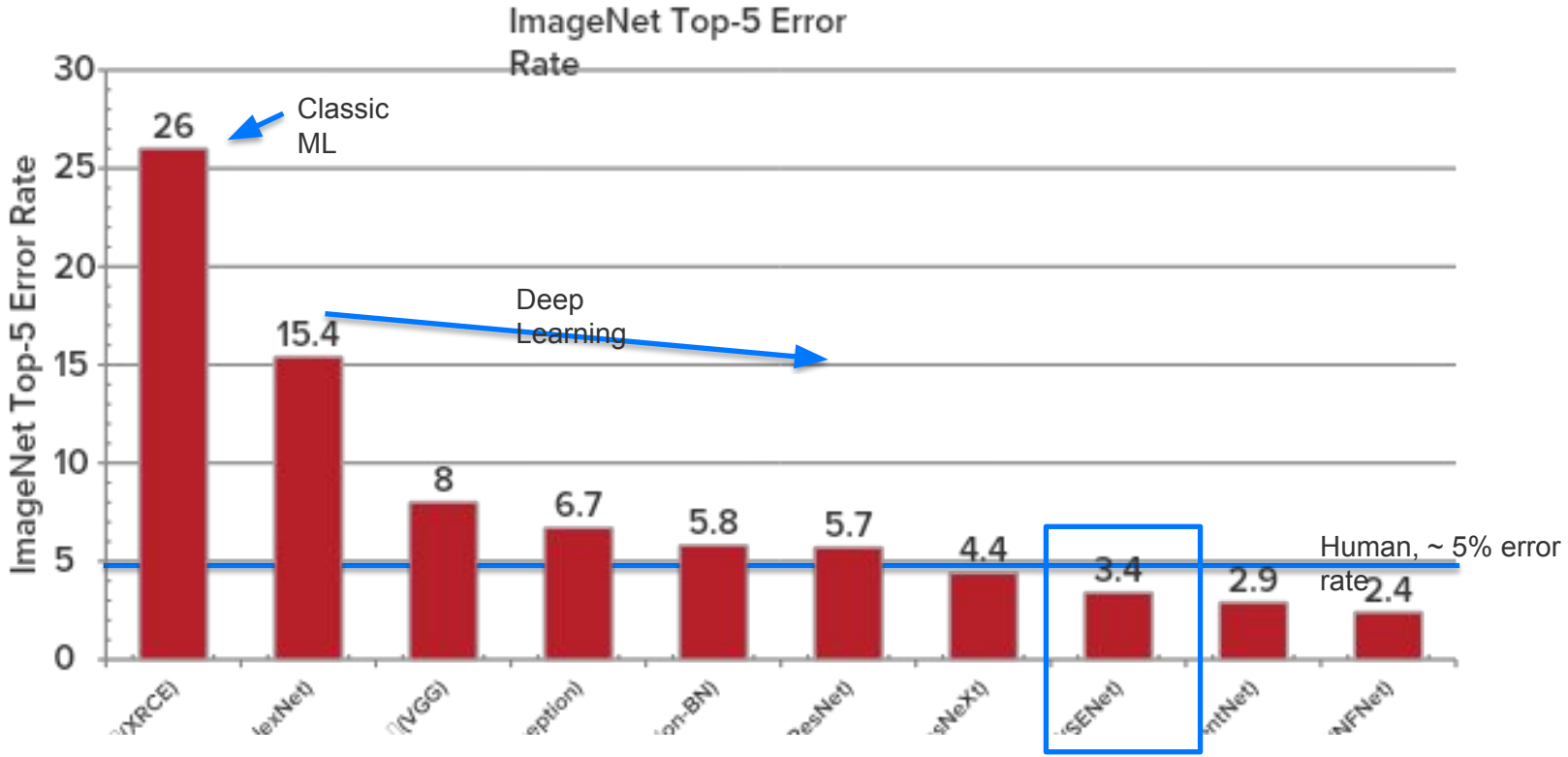


Figure 1. **Left**: A block of ResNet [14]. **Right**: A block of ResNeXt with cardinality = 32, with roughly the same complexity. A layer is shown as (# in channels, filter size, # out channels).


ImageNet Top-5 Error Rate
ImageNet Top-5 Error Rate
30
25
20
15
10
5
0
26
15.4
8
6.7
5.8
5.7
4.4
3.4
2.9
2.4
Classic ML
Deep Learning
Human, ~ 5% error rate

Squeeze-n-Excitation (SENet) (2017)

- Squeeze-and-Excitation Networks
- Идея о перевзвешивании карт активаций
- Не все признаки одинаково полезны

Squeeze-n-Excitation (SENet) (2017)

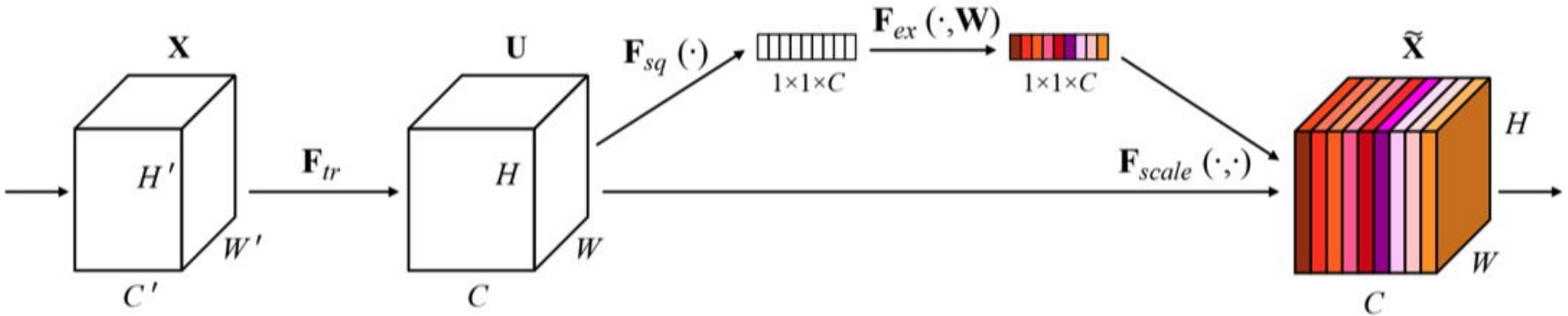
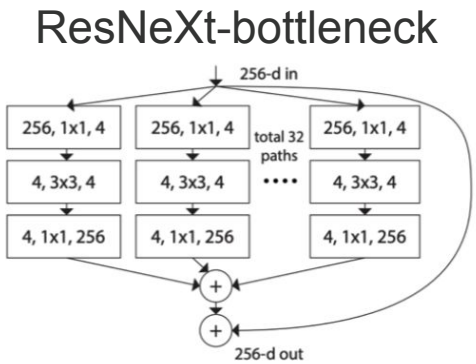


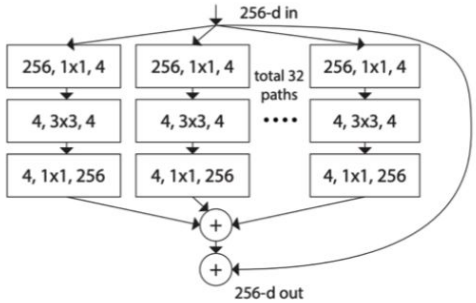
Fig. 1. A Squeeze-and-Excitation block.

ResNeXt

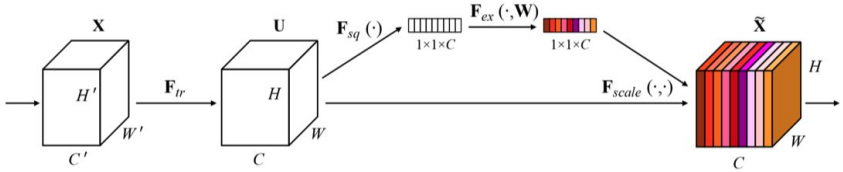

ResNeXt-bottleneck
256-d in
256, 1x1, 4
256, 1x1, 4
total 32 paths
256, 1x1, 4
4, 3x3, 4
4, 3x3, 4
4, 3x3, 4
4, 1x1, 256
4, 1x1, 256
4, 1x1, 256
+
+
256-d out

ResNeXt + SE-ResNet =

ResNeXt-bottleneck

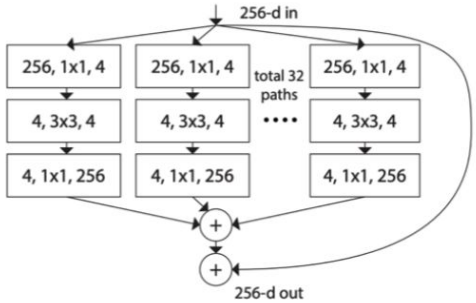


SE-block

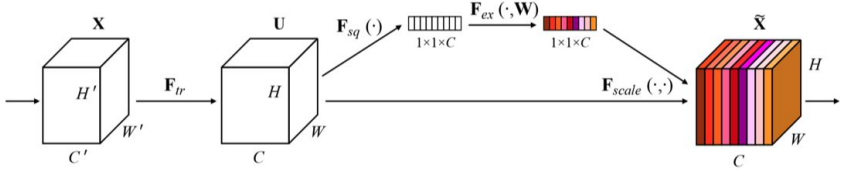


ResNeXt + SE-ResNet = SE-ResNeXt

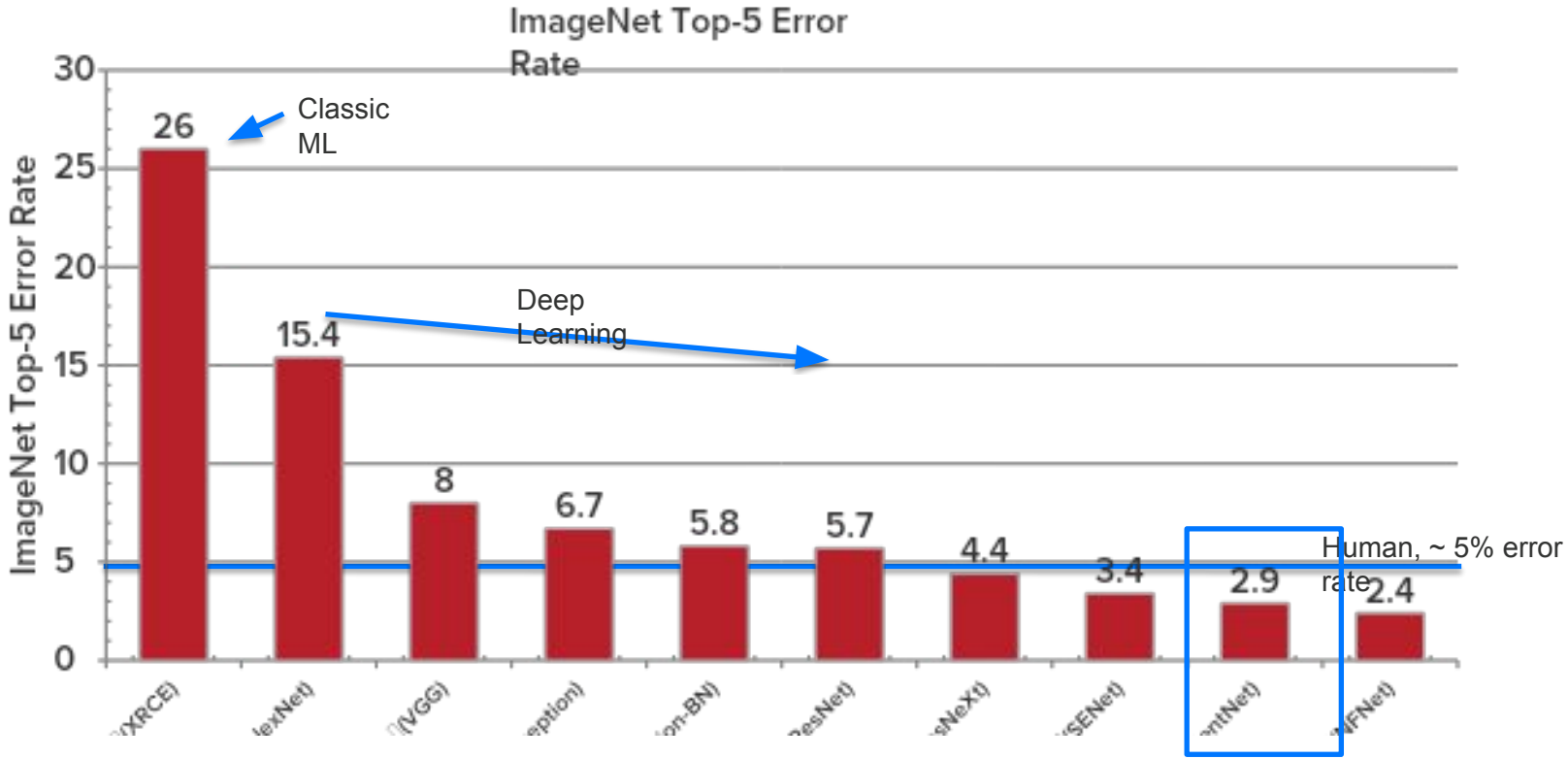
ResNeXt-bottleneck



SE-block



https://paperswithcode.com/model/seresnext?variant=seresnext50-32x4d


ImageNet Top-5 Error Rate
ImageNet Top-5 Error Rate
30
25
20
15
10
5
0
26
15.4
8
6.7
5.8
5.7
4.4
3.4
2.9
2.4
Classic ML
Deep Learning
Human, ~ 5% error rate

EfficientNet, RegNet, NFNet, ...

- В следующий раз!
- А также Transfer Learning

Итоги

Итоги

- Нейросети > классики для задач CV
- Многие эффективные идеи в DL **очень** просты
- **ResNet - база**

Вопросы для самопроверки

- Можно ли использовать обученную модель, содержащую слои BatchNormalization, с батчом = 1?
- Откуда в названии "ResNet34" взялось "34"?
- Во сколько раз оригинальный ResNet50 уменьшает ширину входного изображения (перед последним GAP)?