



Лаборатория компьютерной
графики и мультимедиа
ВМК МГУ имени М.В. Ломоносова

Курс «Компьютерное зрение»

«Базовые свёрточные архитектуры»

Антон Конушин

2024 год



План лекции

- Визуализация работы нейросети
- Понятие о нейросетевых признаках и базовых моделях
- AlexNet, VGG, Inception
- ResNet-модели и SkipConnections, репараметризация
- Как обучить мощную модель?
- Архитектуры для мобильных моделей
- Насколько хороша ImageNet?

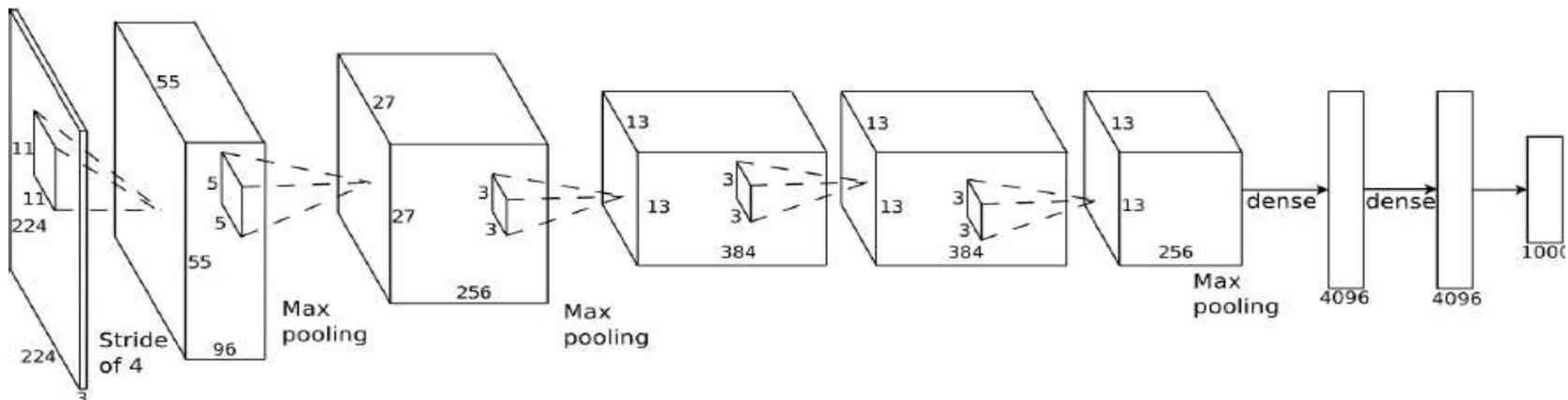


Визуализация работы нейросети

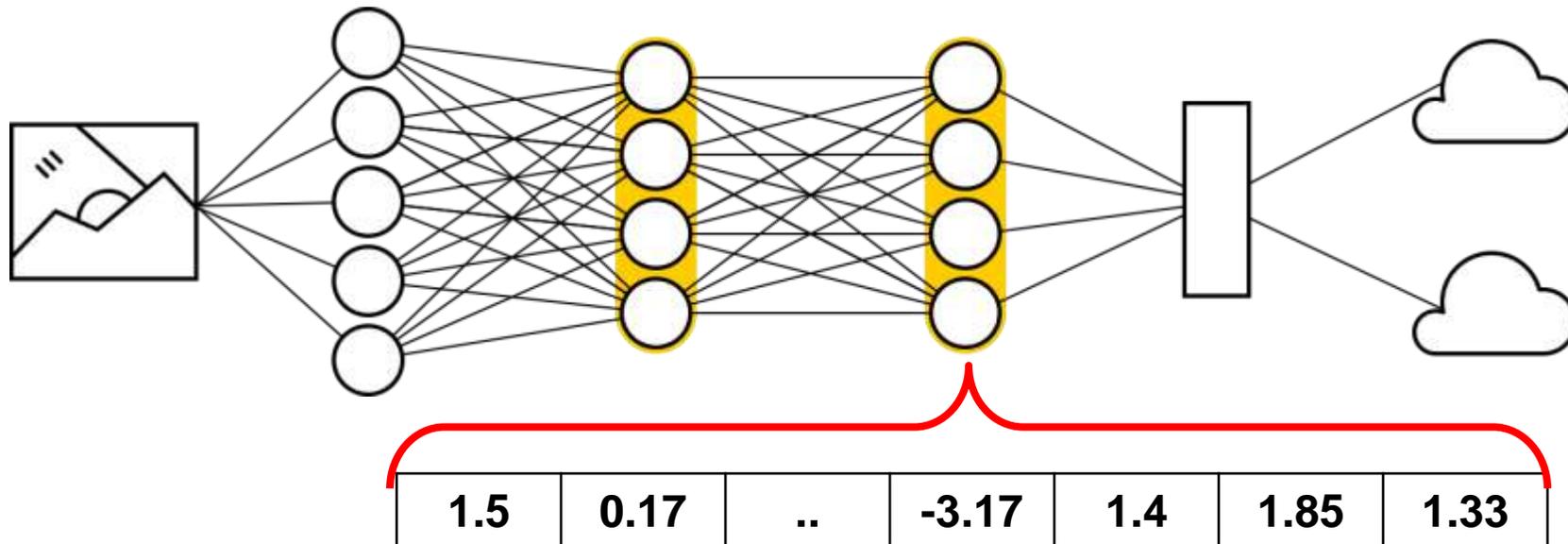


Что происходит внутри сети?

Мы умеем обучать нейросеть для классификации изображений



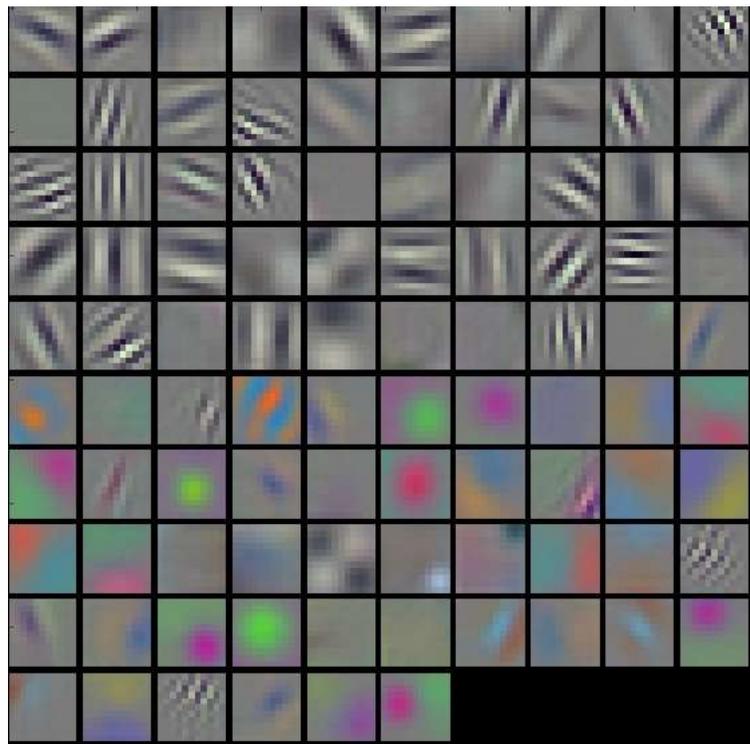
Что происходит внутри неё? Что будем визуализировать?



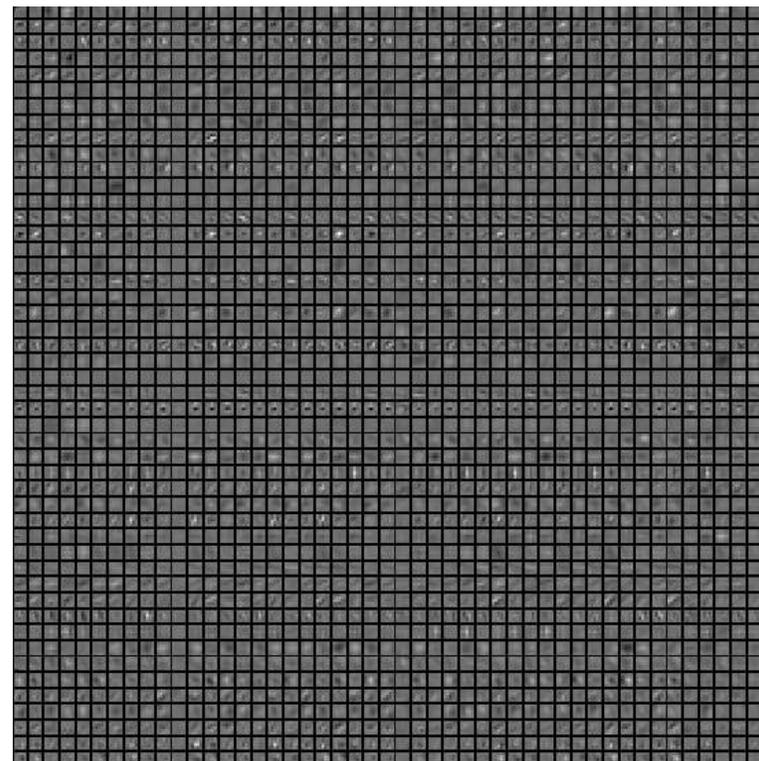
Визуализация работы нейросети



Визуализация фильтров

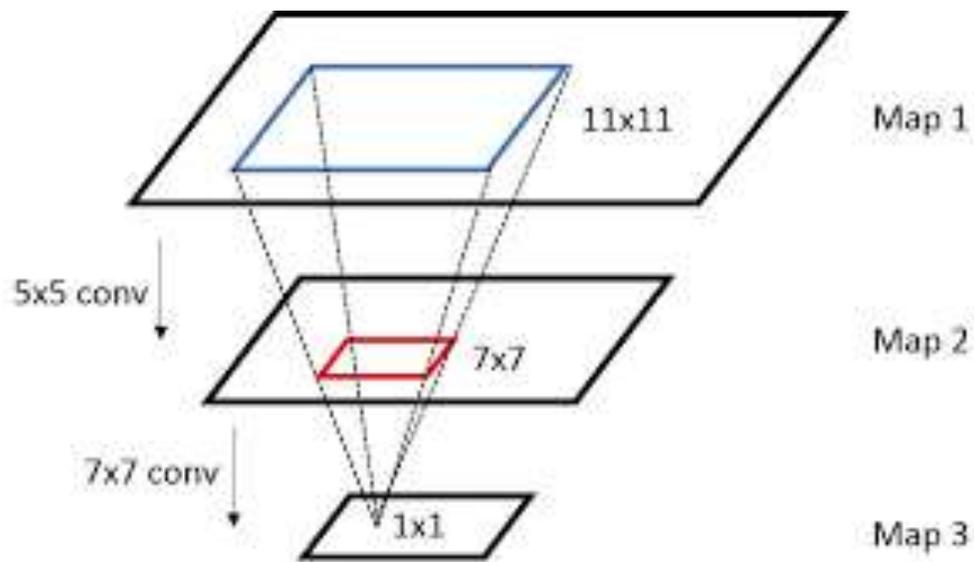


Слой conv1



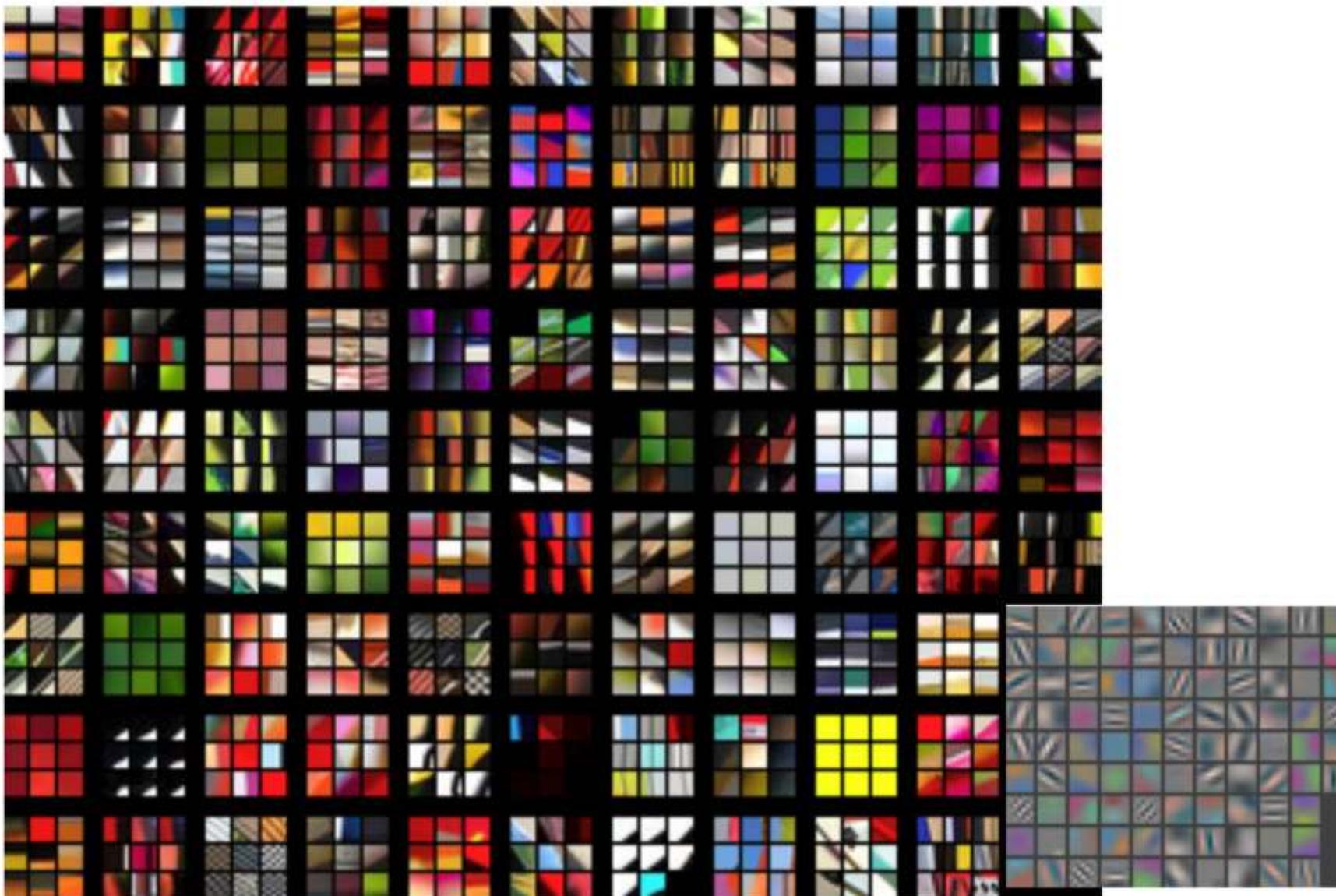
Слой conv2

Рецептивное поле и визуализация



- Напоминание – рецептивное поле, это область изображения, от которой зависит выход этого нейрона
- Запишем, какие фрагменты вызывают наиболее сильный отклик конкретного нейрона сети

Слой 1: Top-9 фрагментов



Слой 2: Топ-9 фрагментов

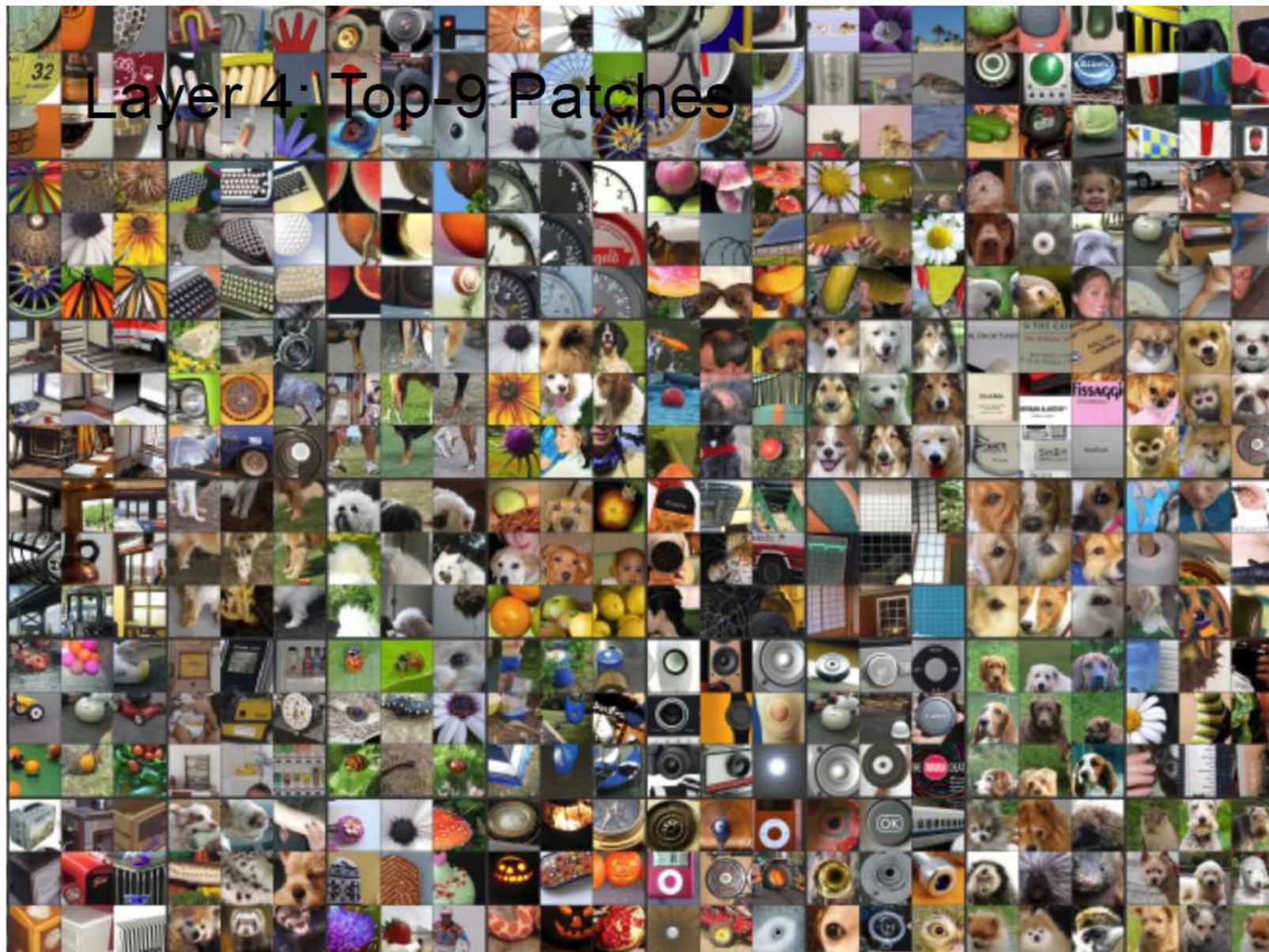


- Patches from validation images that give maximal activation of a given feature map

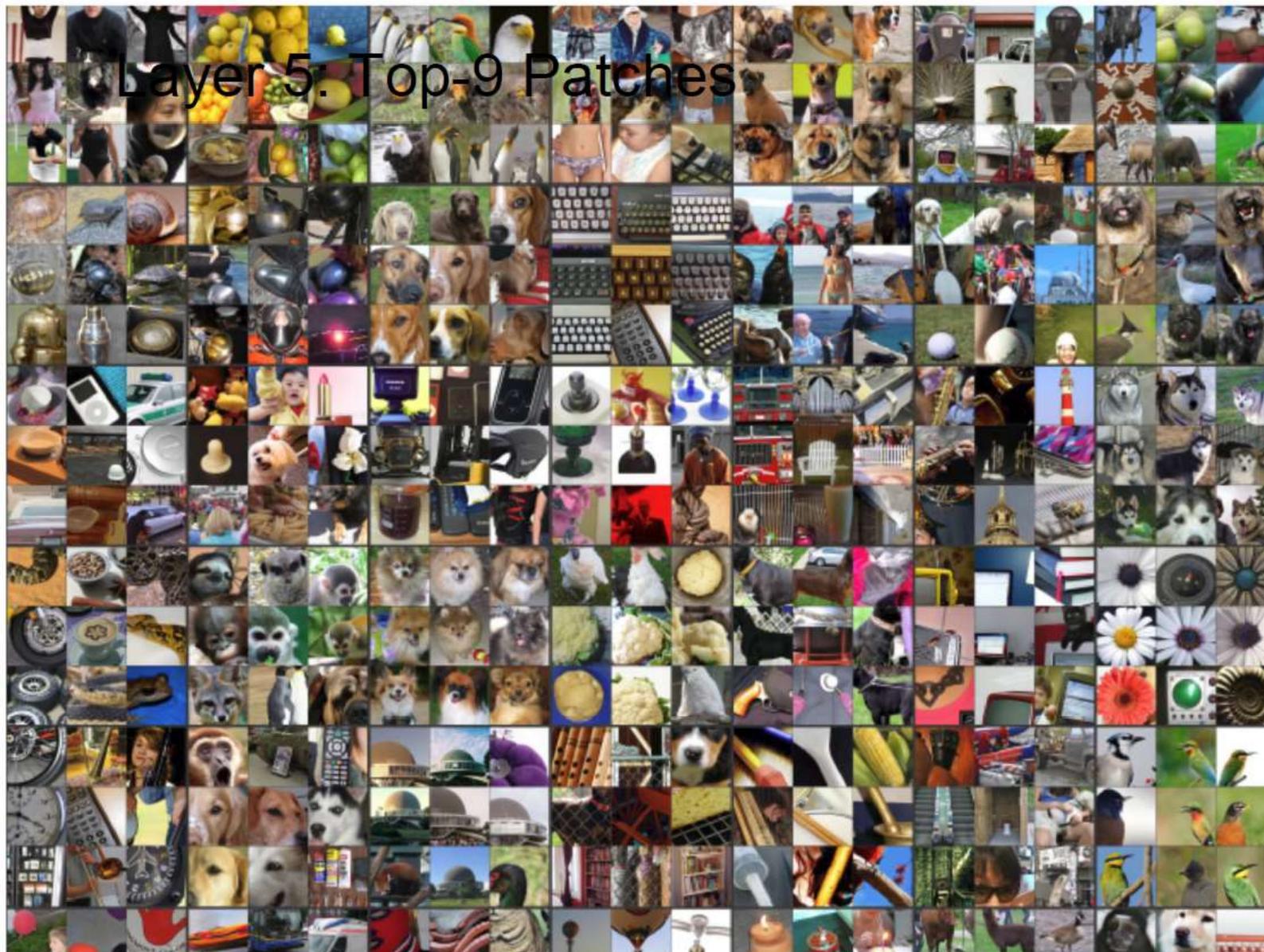
Слой 3: Топ-9 фрагментов



Слой 4: Топ-9 фрагментов



Слой 5: Топ-9 фрагментов



Визуализация работы нейросети

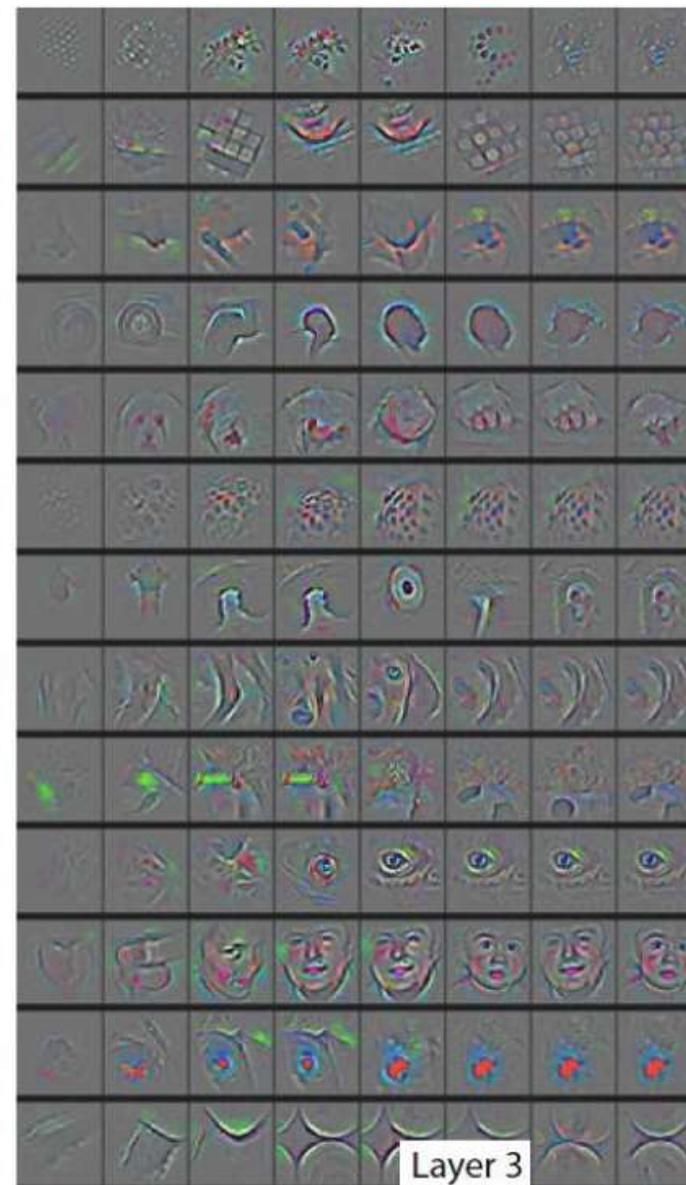
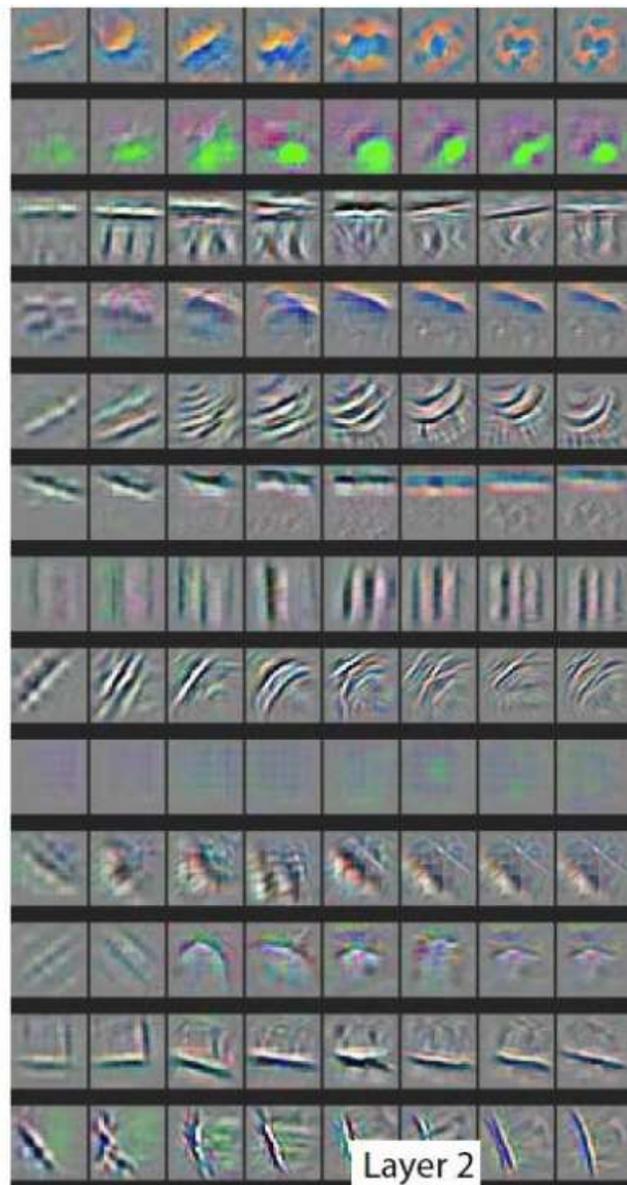
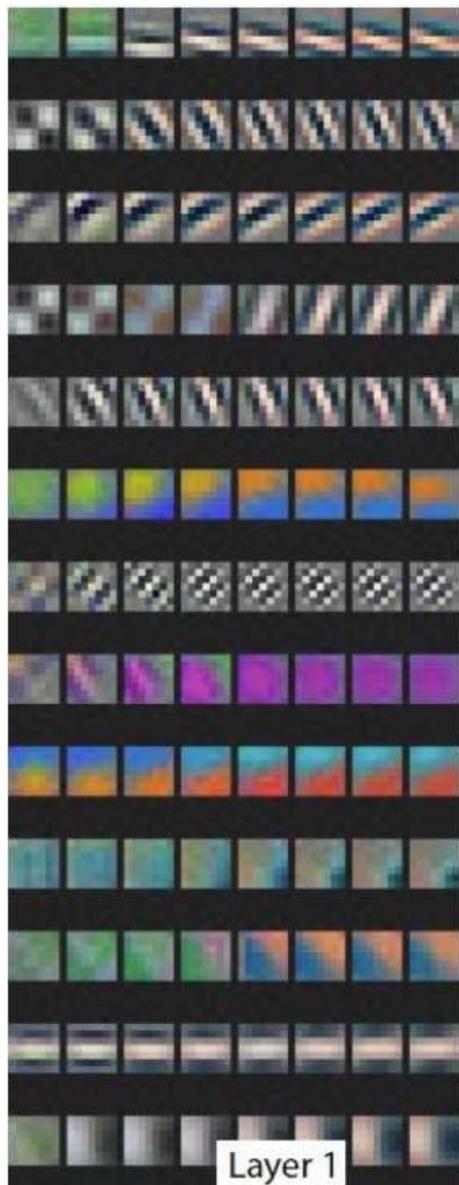


Изображения, на которых достигается максимальный отклик фильтра

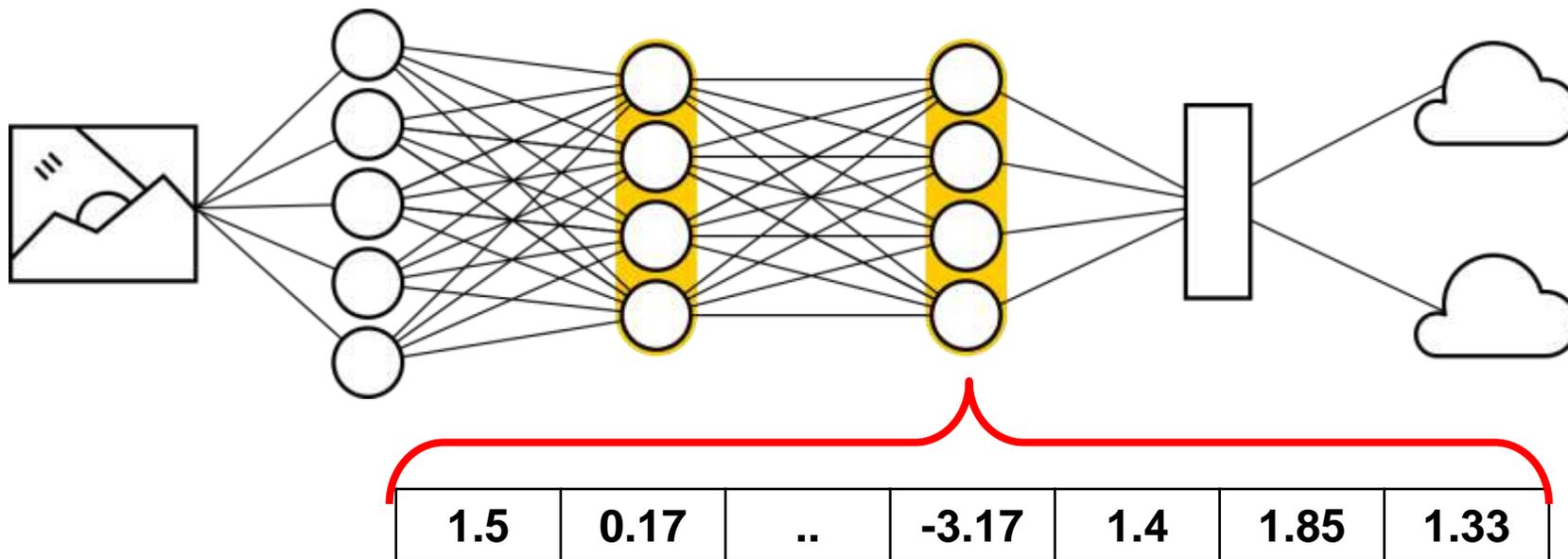


Фильтры слоя pool5

Эволюция признаков



Выход слоя как вектор-признак



- Мы увидели, что отдельные нейроны каждого слоя несут важное семантическое значение
- Совокупность выходов слоя можно рассматривать как вектор-признак изображения
- Есть способы анализа вектор-признаков в совокупности

t-SNE

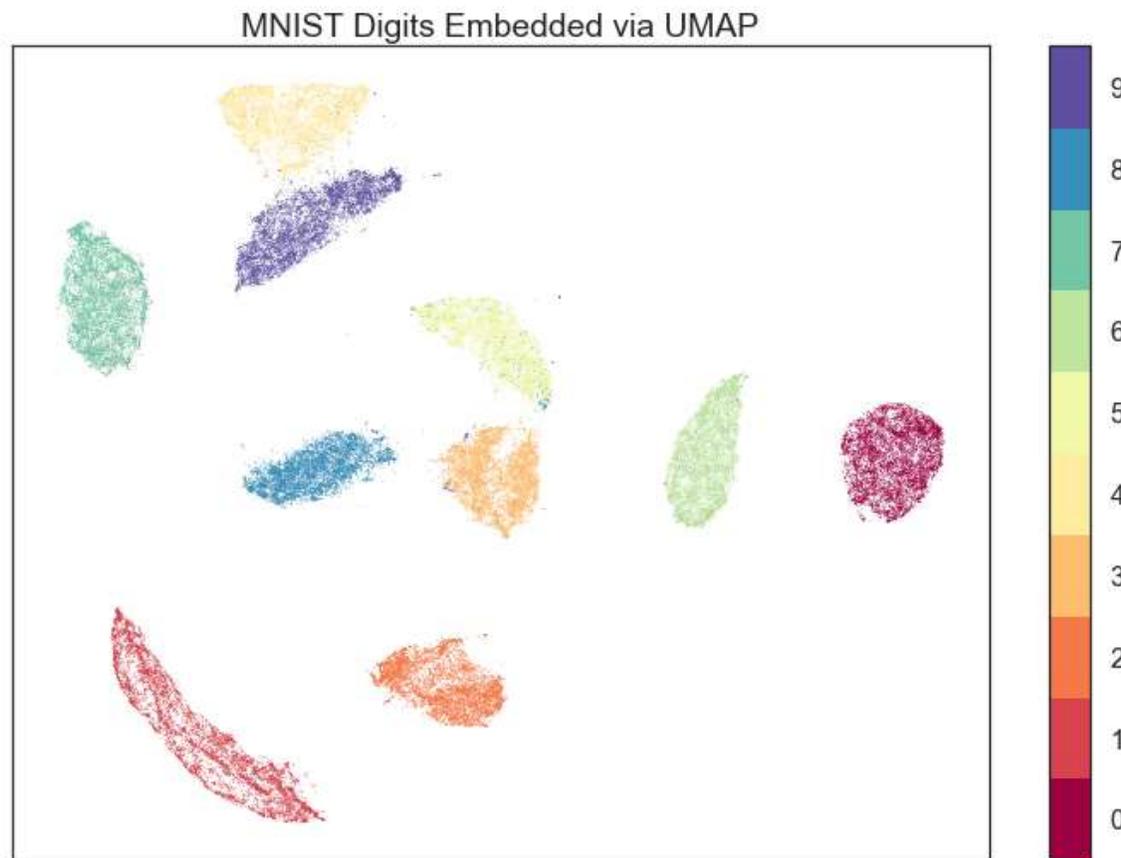


- Можем вычислить L2 расстояние между выходами full6 или full7 слоёв
- Воспользуемся отображением точек из 4096-мерного пространства на 2х мерное, сохраняющее L2 расстояния (приблизенно)
- Визуализируем изображения
- Видим, что близкие по смыслу изображения оказываются близки друг к другу





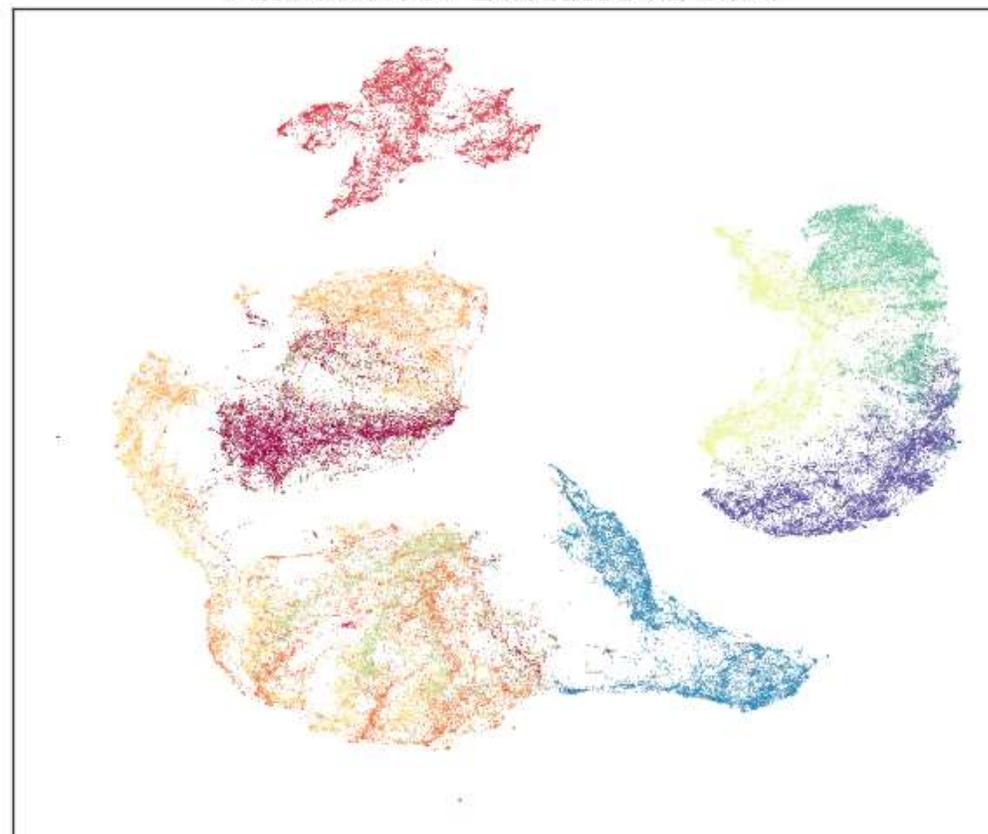
UMAP визуализация



UMAP визуализация

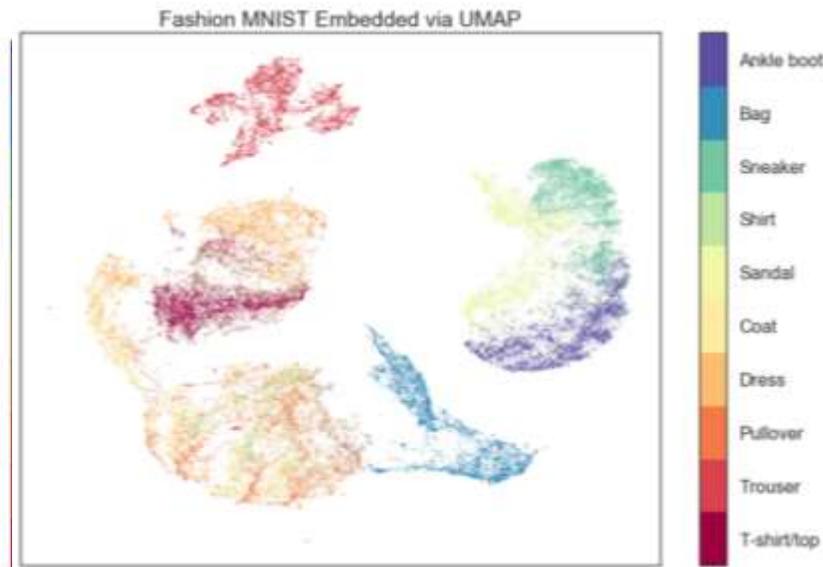
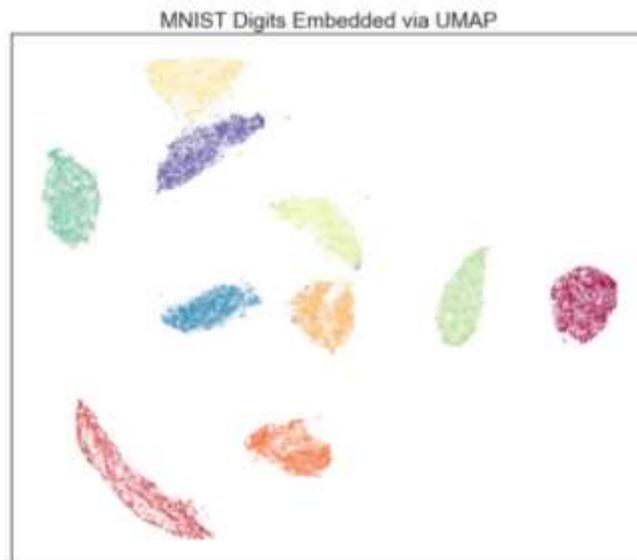


Fashion MNIST Embedded via UMAP



- Ankle boot
- Bag
- Sneaker
- Shirt
- Sandal
- Coat
- Dress
- Pullover
- Trouser
- T-shirt/top

Выводы

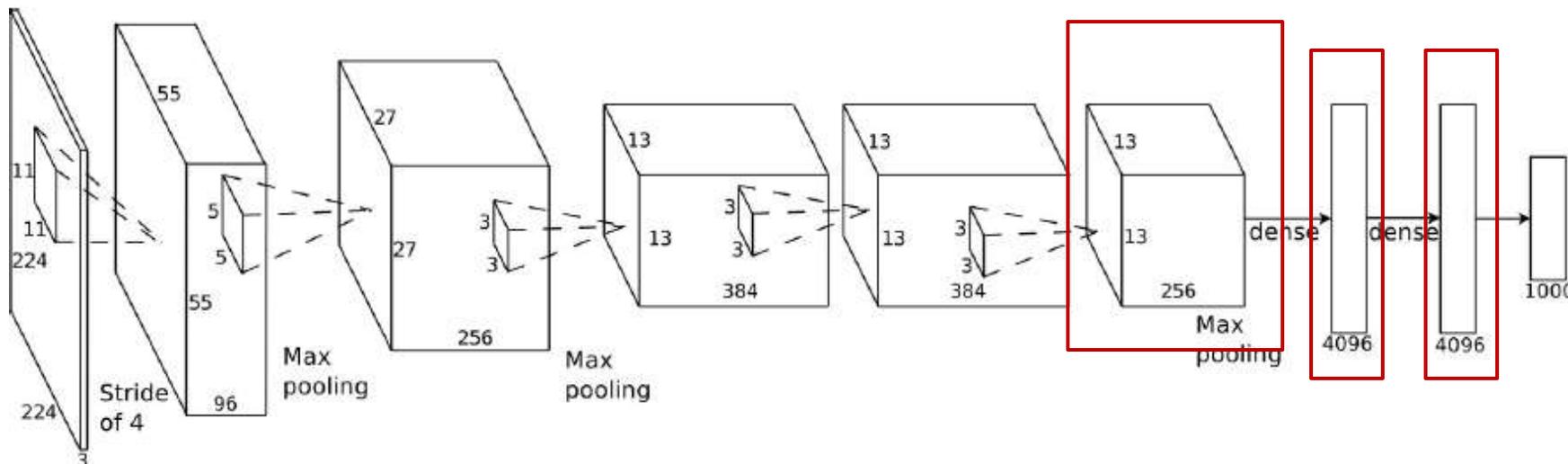


- Нейросеть обучается отображать изображения в вектор-признаки, кодирующие семантическую «похожесть» изображений.
- Близкие и визуально похожие друг на друга объекты оказываются близки по нейросетевым вектор-признакам.
- Вектор-признаки объектов разных классов далеки друг от друга



Нейросетевые признаки и базовые архитектуры

Нейросетевые признаки



- Можно использовать выходы выбранного слоя нейросети как вектор-признак
- Можно обучить сеть на одних данных (ImageNet) и применять её на других для вычисления признаков, обучая поверх классификатор



Перенос на другие датасеты

	DeCAF ₅	DeCAF ₆	DeCAF ₇
LogReg	63.29 ± 6.6	84.30 ± 1.6	84.87 ± 0.6
LogReg with Dropout	-	86.08 ± 0.8	85.68 ± 0.6
SVM	77.12 ± 1.1	84.77 ± 1.2	83.24 ± 1.2
SVM with Dropout	-	86.91 ± 0.7	85.51 ± 0.9
Yang et al. (2009)		84.3	
Jarrett et al. (2009)		65.5	

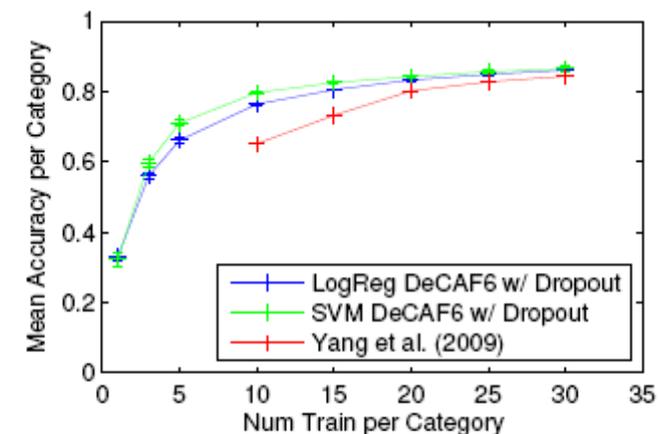


Figure 4. Left: average accuracy per class on Caltech-101 with 30 training samples per class across three hidden layers of the network and two classifiers. Our result from the training protocol/classifier combination with the best validation accuracy – SVM with Layer 6 (+ dropout) features – is shown in **bold**. Right: average accuracy per class on Caltech-101 at varying training set sizes.

- Обучили нейросеть для классификации изображений ImageNet
- Используем для извлечения признаков на других коллекциях, например, Caltech 101

Fine-graded classification



Method	Part info	mean Accuracy
Sift+Color+SVM[45]	✗	17.3
Pose pooling kernel[49]	✓	28.2
RF[47]	✓	19.2
DPD[50]	✓	51.0
Poof[5]	✓	56.8
CNN-SVM	✗	53.3
CNNaug-SVM	✗	61.8
DPD+CNN(DeCaf)+LogReg[10]	✓	65.0

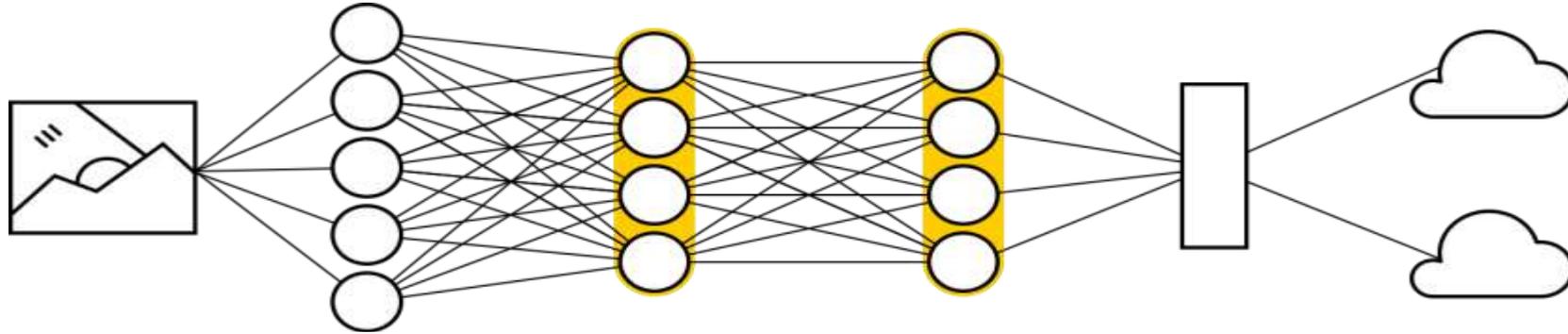
Table 3: Results on CUB 200-2011 Bird dataset. The table distinguishes between methods which use part annotations for training and sometimes for evaluation as well and those that do not. [10] generates a pose-normalized CNN representation using DPD [50] detectors which significantly boosts the results to 64.96.

- Возьмём нейросеть, обученную для классификации ImageNet
- Применим её для получения вектор-признаков изображений
- Обучаем классификатор поверх этих признаков
- Profit!

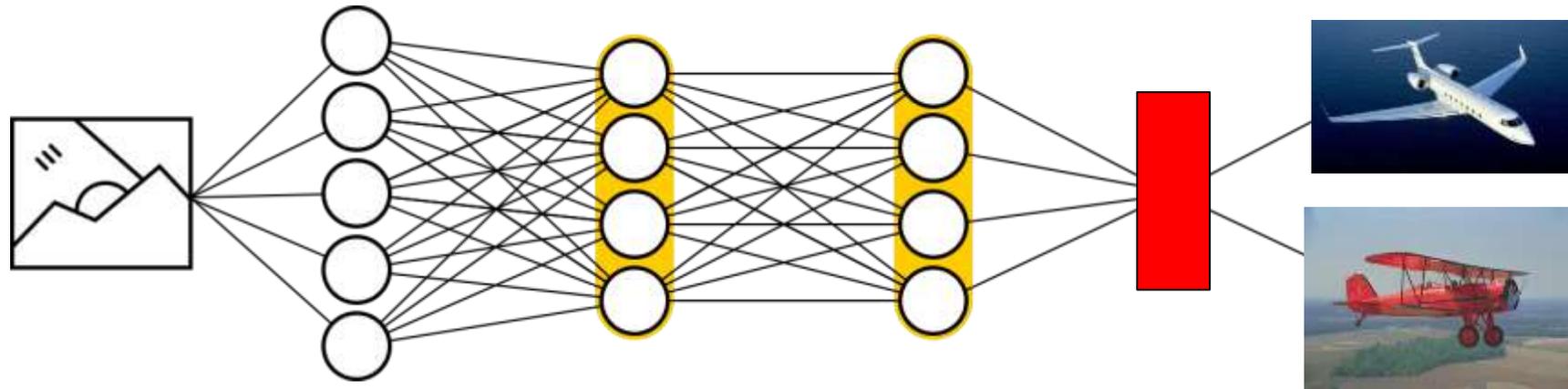
Дообучение (fine-tuning)



Возьмём сеть A, обученную на одной коллекции (например, ImageNet)

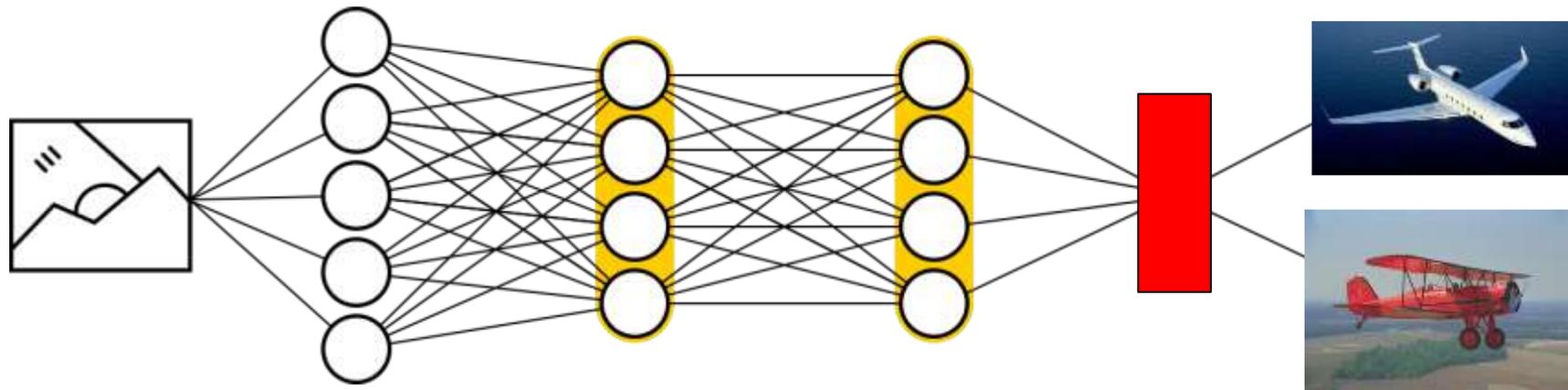


Заменим в ней последний слой (классификатор)



Обучим («дообучим») эту сеть на новой коллекции для решения новой задачи

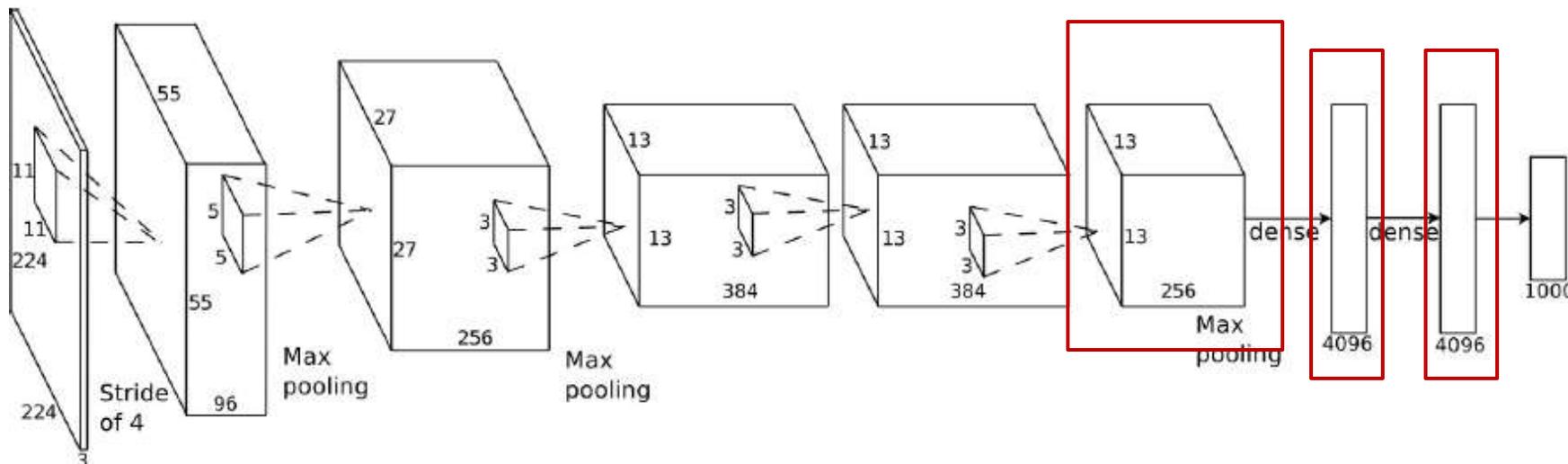
Плюсы дообучения



- Вторая коллекция может быть недостаточного размера для обучения нейросети «с нуля» (from scratch)
- Предобучение на большой и разнообразной коллекции может «хорошо» инициализировать новую сеть, и дообучится она быстрее и лучше



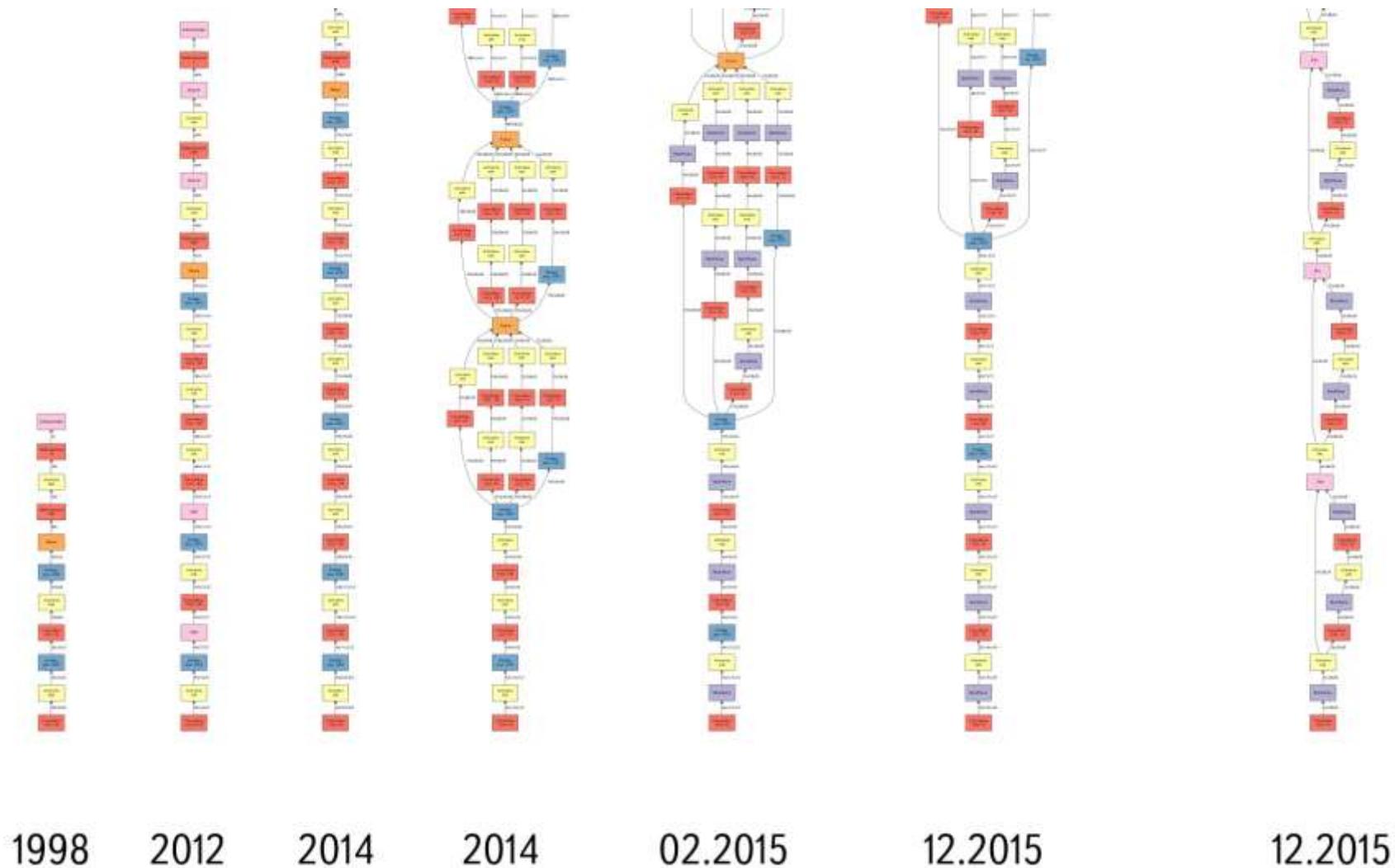
Идея «базовой» архитектуры



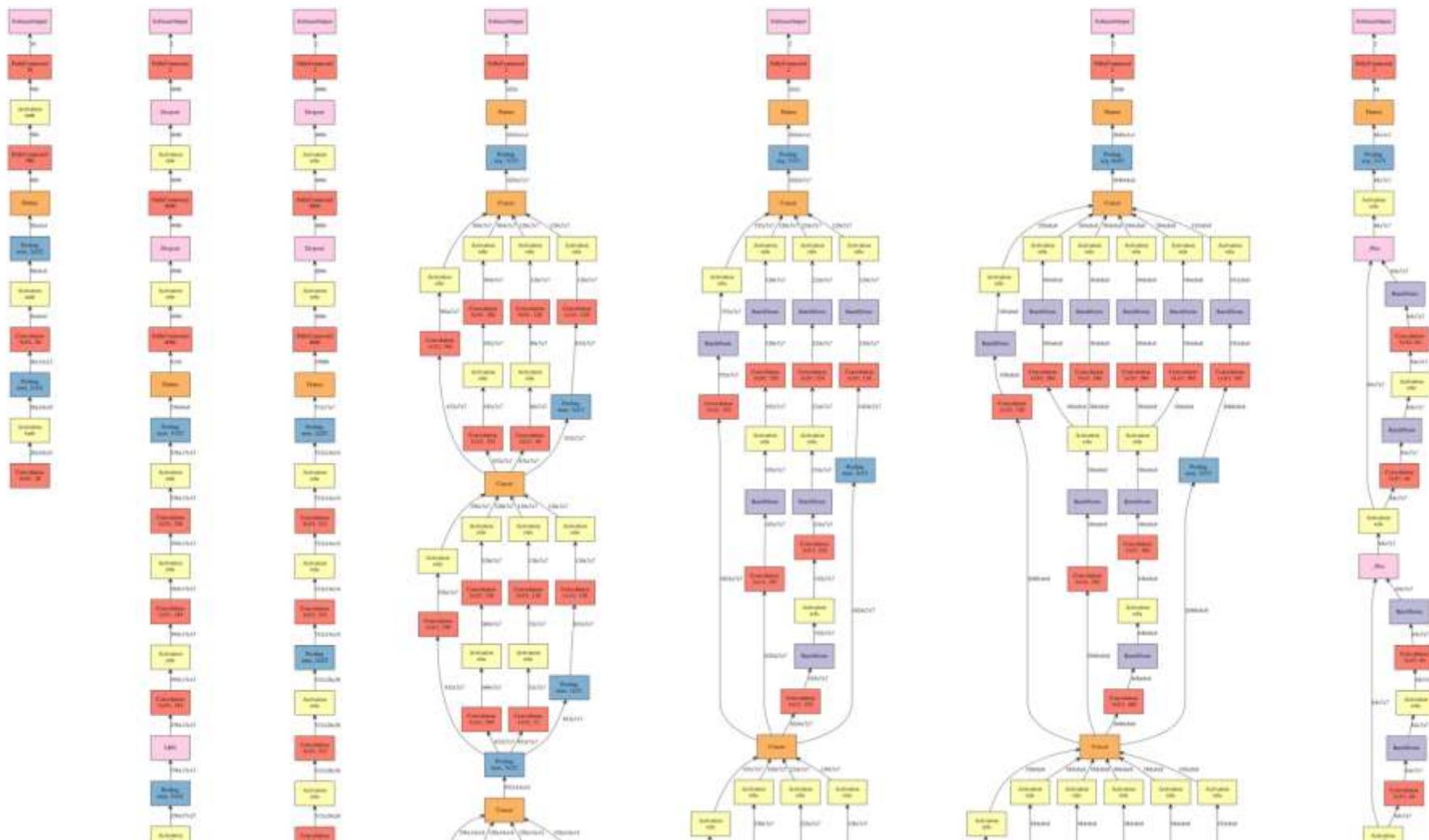
- Можем обучать нейросети-классификаторы на больших коллекциях («базовые сети»), использовать их как основу для решения новых задач (дообучением)
- Создали коллекции моделей (zoo – зоопарки)
- Пока в основном обучают на ImageNet



Развитие базовых архитектур



Развитие базовых архитектур



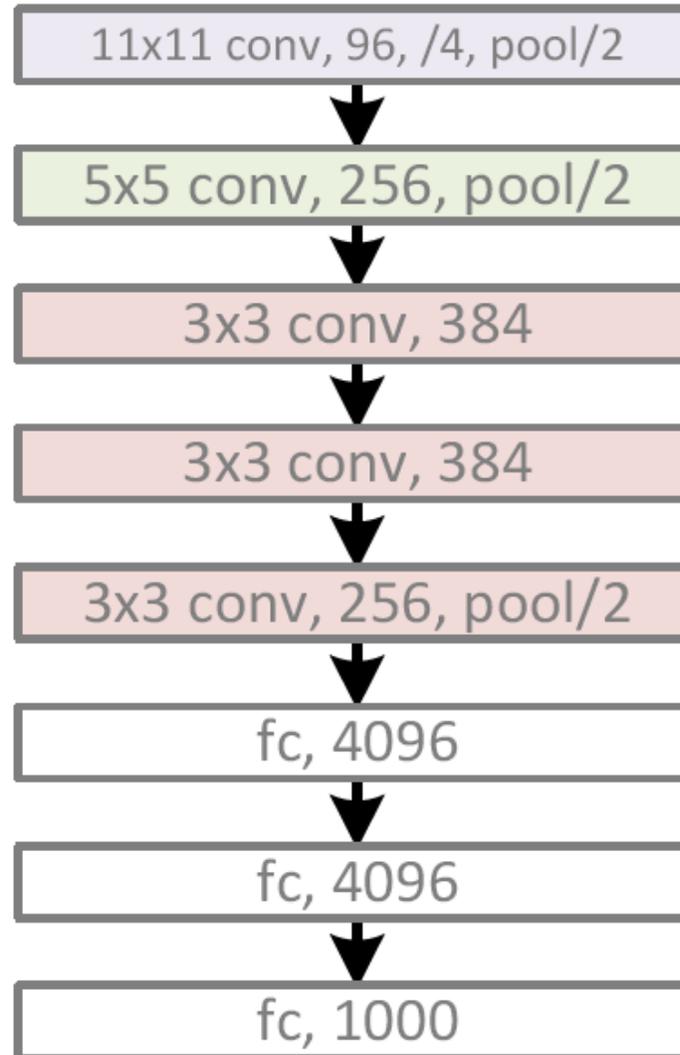


Первые базовые модели:
AlexNet, VGG, Inception

AlexNet



AlexNet, 8 layers
(ILSVRC 2012)

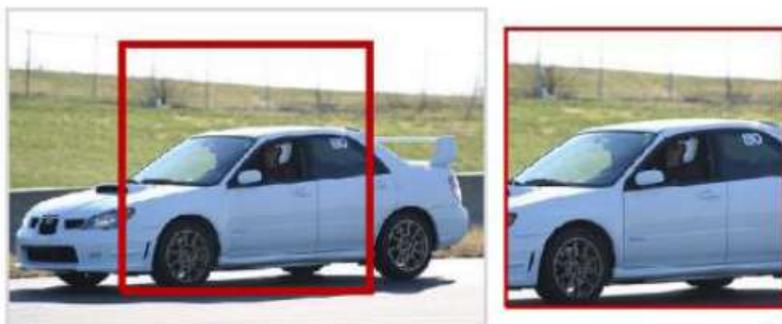


Krizhevsky A., Sutskever I., Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks // NIPS 2012



Применение AlexNet

- На вход нейросеть принимает изображение фиксированного размера 224x224 пиксела. Можем ли мы подать другое?
- Как быть с изображениями других размеров?
- 2 варианта действий:

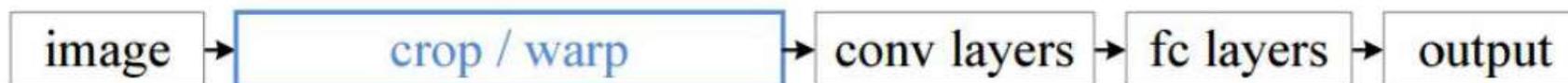


Масштабировать к разным разрешениям, пр. [256;512]xN случайно и вырезать фиксированные (image crop)



Приводить к фиксированному разрешению (image warp)

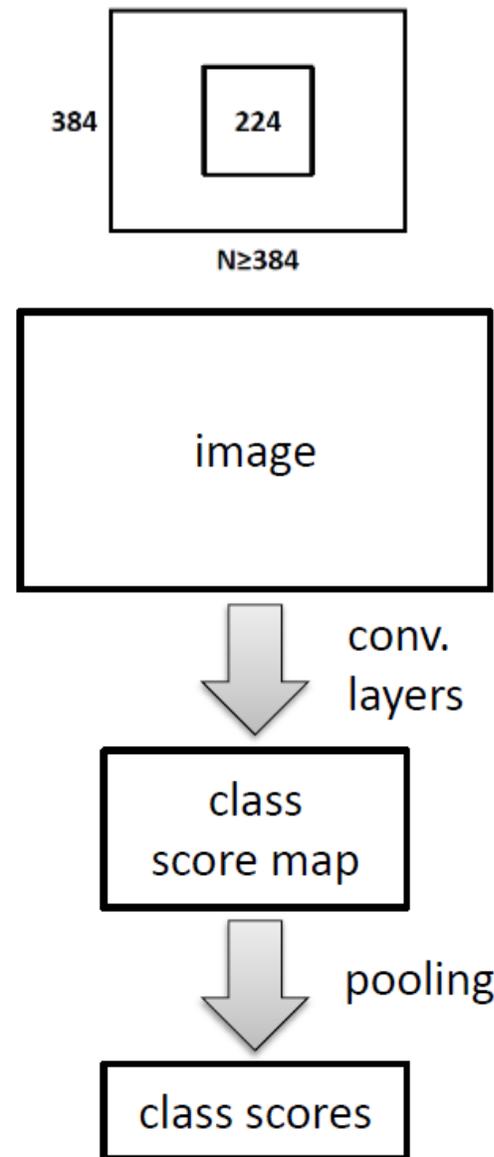
- Схема применения



Применение моделей



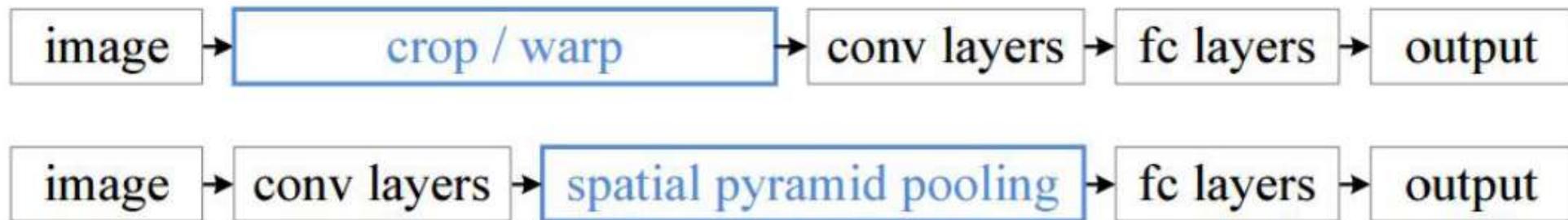
- Вариант 1
 - Применить к фиксированному изображению, как сказали раньше
- Вариант 2
 - Применить несколько раз к разным версиям изображения
- Вариант 3
 - Применить ко всем фрагментам заданного разрешения, т.е. «просканировать» изображение



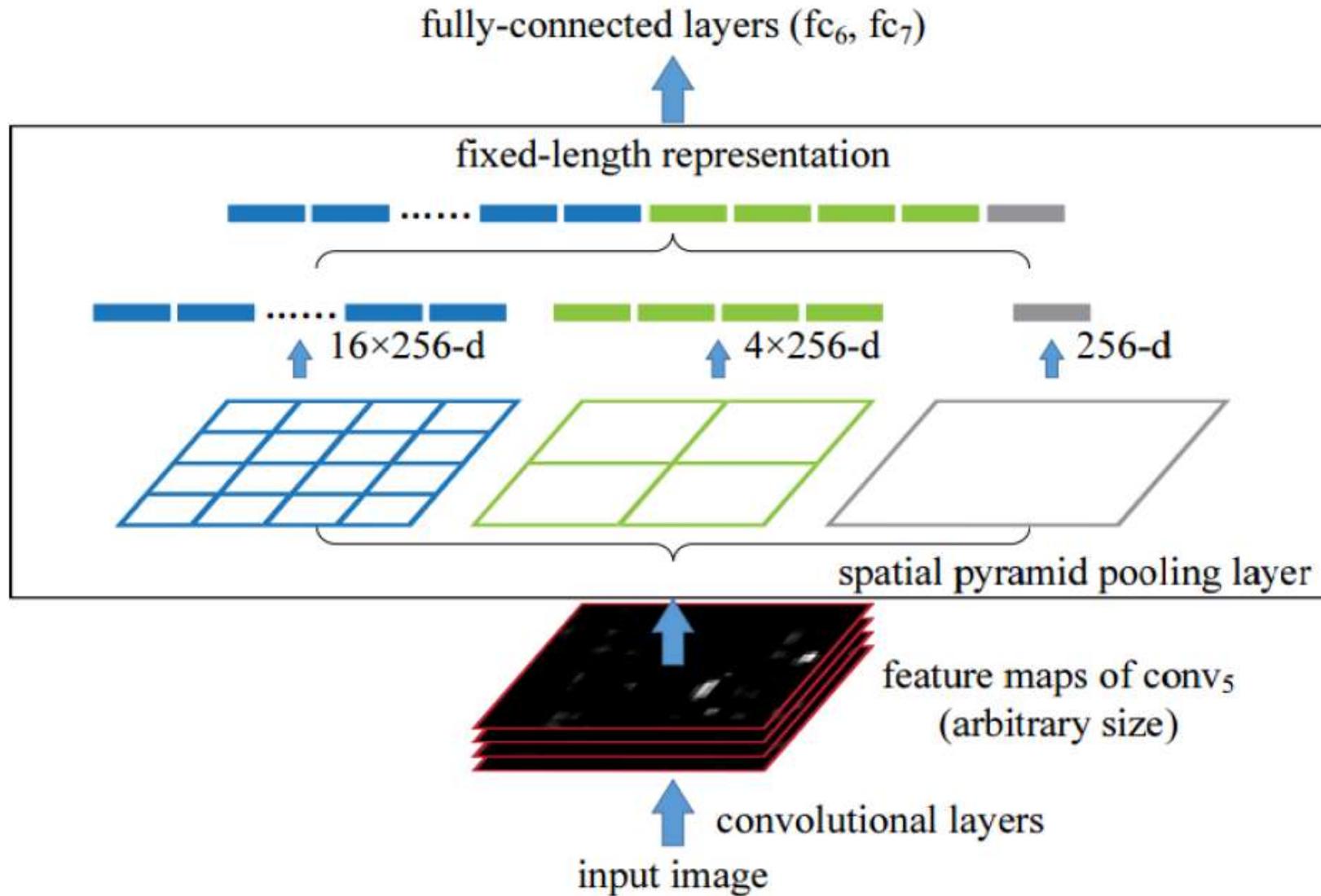
Spatial Pyramid Pooling (SPP)



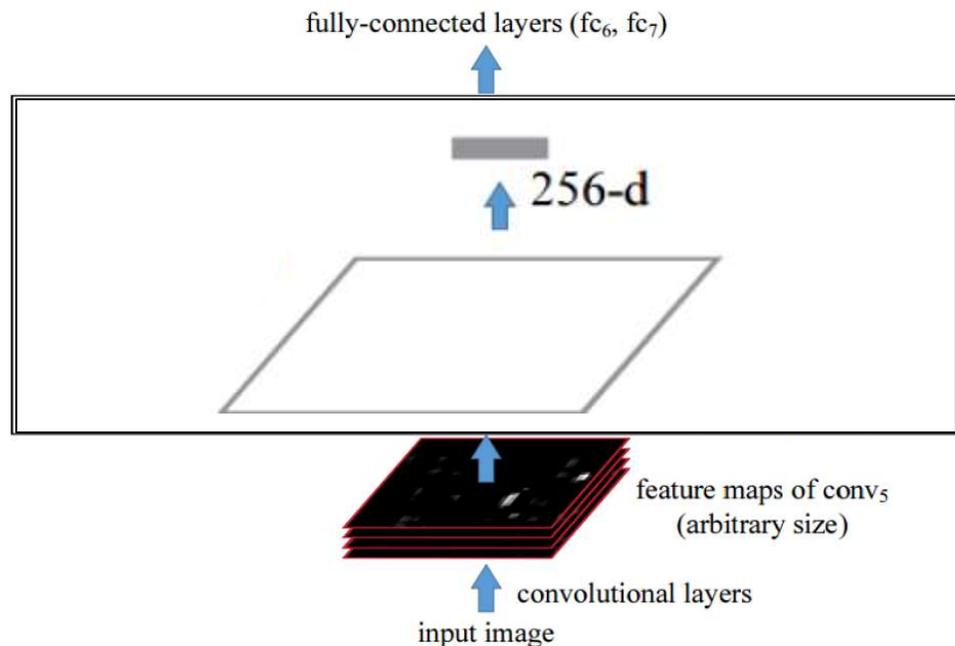
Идея: преобразовать выходы свёрточного слоя (признаки) произвольного разрешения к вектор-признаку фиксированной длины



Spatial Pyramid Pooling (SPP)



Average Pooling



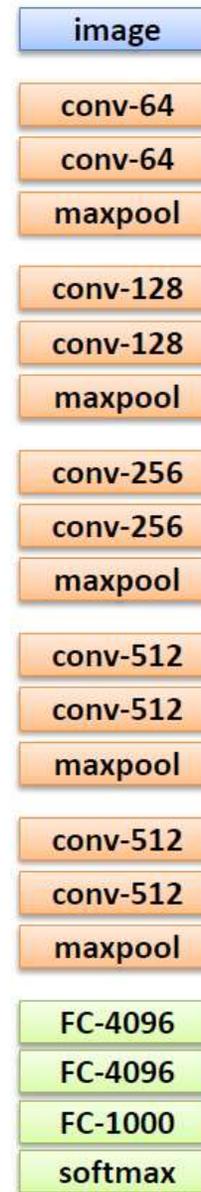
- Можно обойтись без пирамиды, ограничившись только Average Pooling по всему слою
- Тогда у нас получится вектор признаков длиной равной числу свёрток на последнем уровне
- Для ряда задач этого оказывается достаточно

Модель VGG (Visual Geometry Group)



Идеи:

- Исследовать рост качества за счёт увеличения глубины нейросети
- Использовать только маленькие 3x3 свёртки
- Stride 1 в свёртках чтобы не терять информацию
- ReLU активация
- Нет нормализации
- Уменьшение разрешения через maxpooling
- Число фильтров x2 при уменьшении разрешения в 2 раза

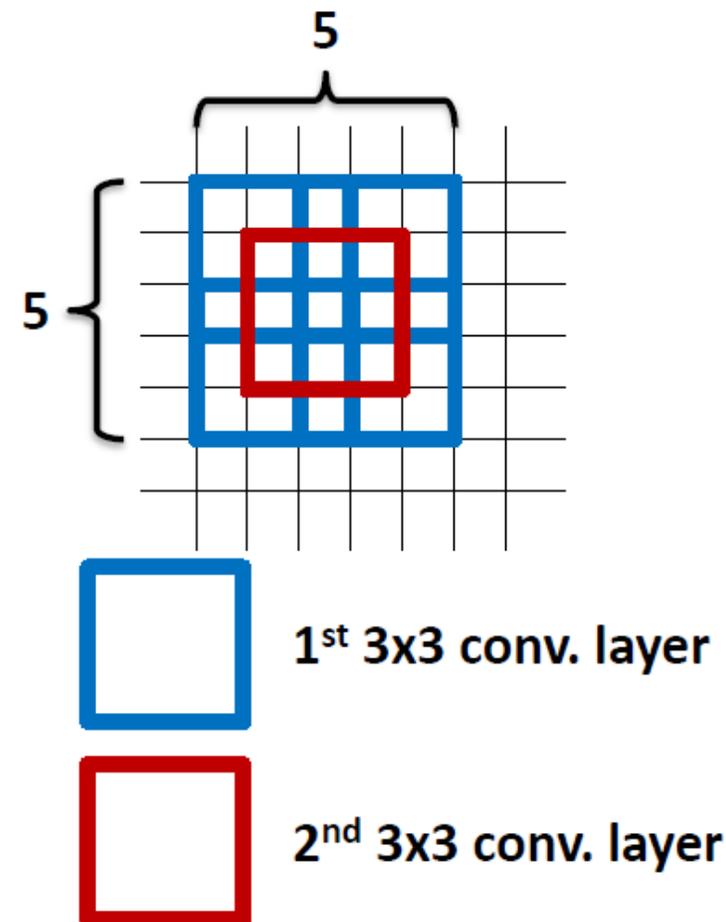


K. Simonyan, A. Zisserman [Very Deep Convolutional Networks for Large-Scale Image Recognition](#). ICLR 2015

Свёртки 3x3



- Стек свёрток позволяет обеспечить бОльшее рецептивное поле (reception field)
 - 5x5 для 2-х свёрток
 - 7x7 для 3-х свёрток
- БОльшая нелинейность за счёт ReLU активаций
- Меньше параметров
 - 18x (2 3x3) vs 25x (5x5)
 - 27x (3 3x3) vs 49x (7x7)



Исследование вариантов



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 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 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 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

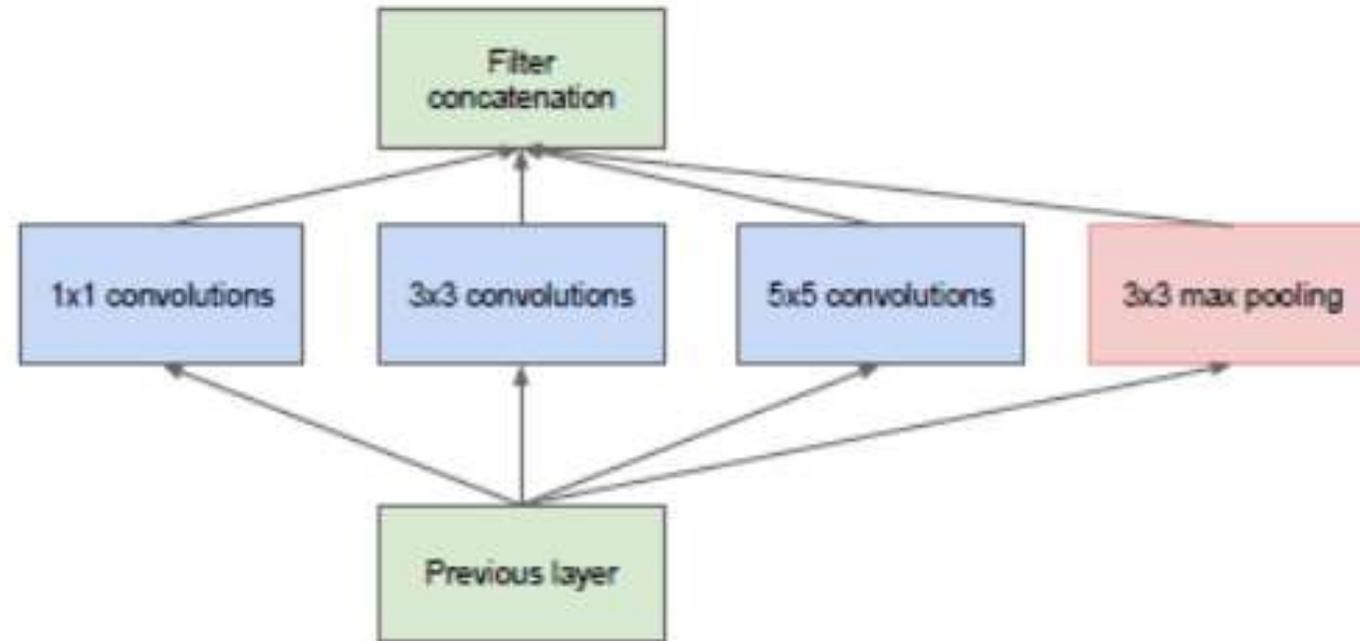
D - VGG-16

E - VGG-19



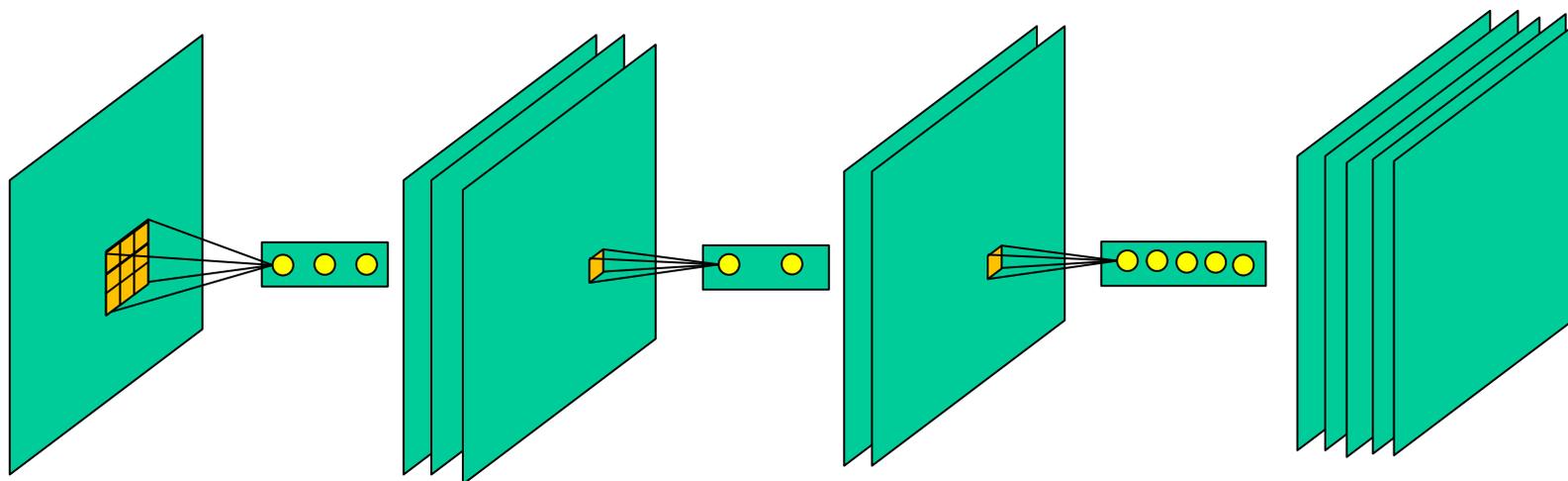
Архитектура Inception и идея ветвей

Построим нейросеть из модулей более сложной структуры, чем просто набор свёрточных слоёв VGG



В чём смысл применять одновременно свёртки разных размеров?

Использование свёрток 1x1

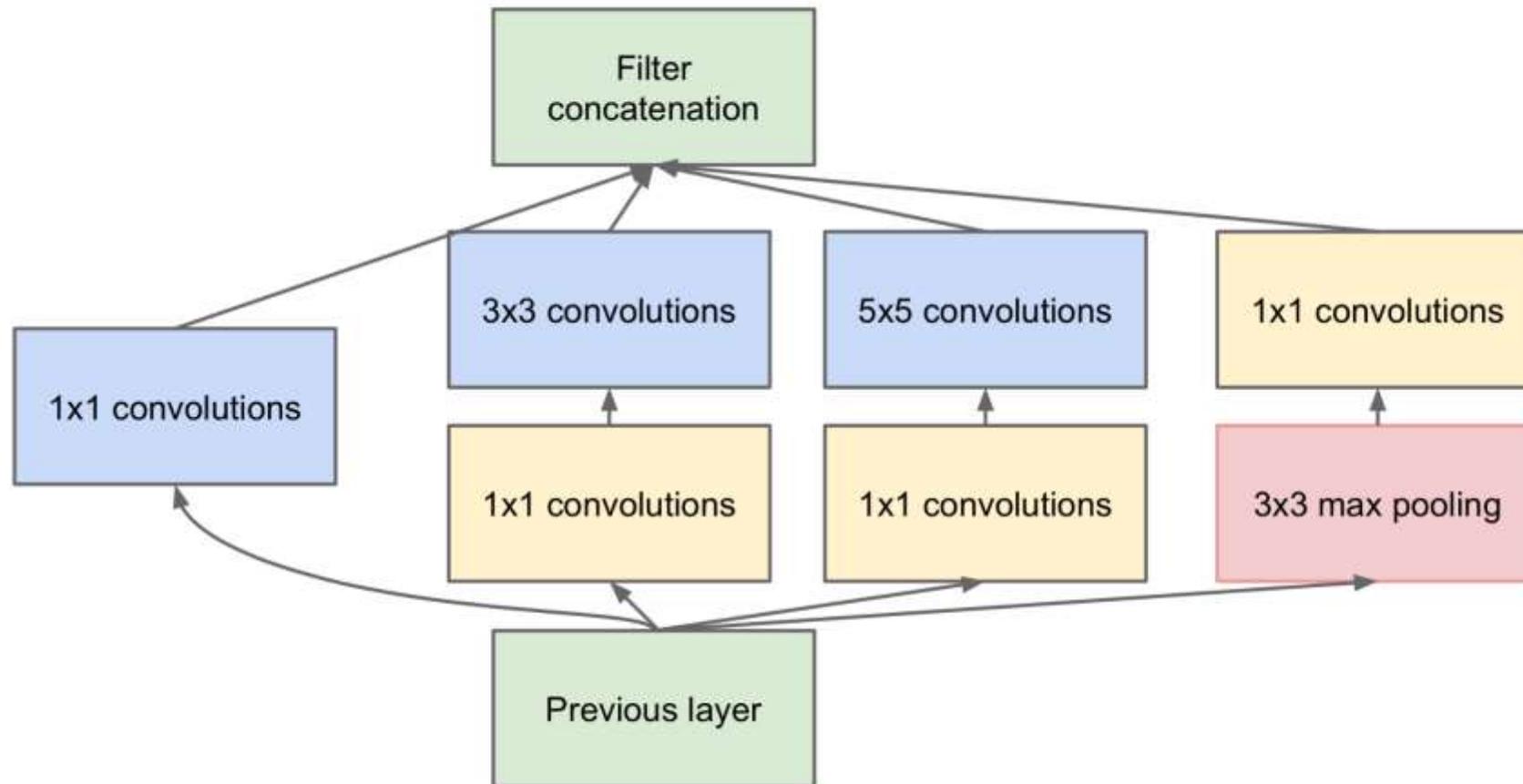


- Как трактовать свёртку 1x1?
 - Отображение вектора длины n (толщина входного слоя = число свёрток предыдущего слоя) на вектор длины k (число свёрток 1x1)
 - Или как набор «локальных классификаторов»
- Можем управлять «глубиной» тензора, регулируя k - число свёрток 1x1, по сравнению с n – глубиной предыдущего тензора
 - $K < N$, значит мы уменьшили до k глубину тензора (сжали)
 - $K > N$, значит мы увеличили глубину тензора

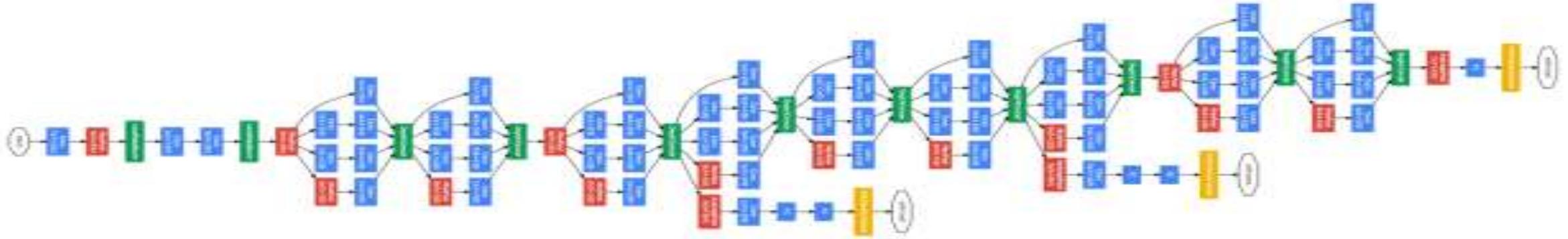
Модуль Inception



Добавим 1x1 свёртки для повышения производительности



Архитектура Inception



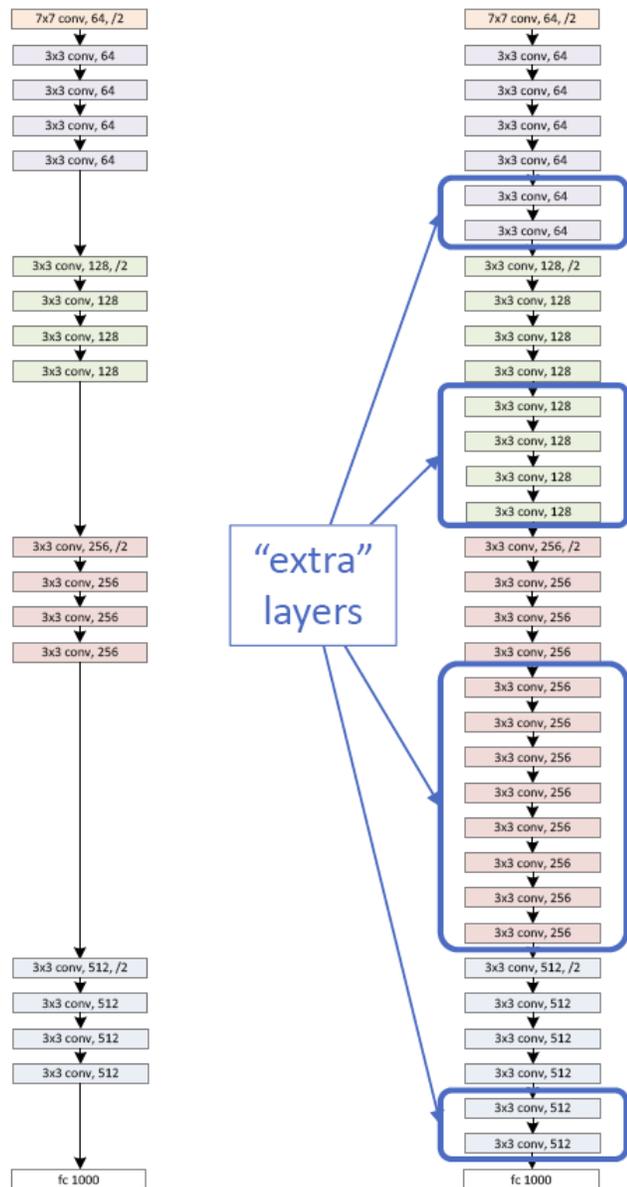
- Глубокая сеть
- Inception-модули
- Несколько уровней supervision



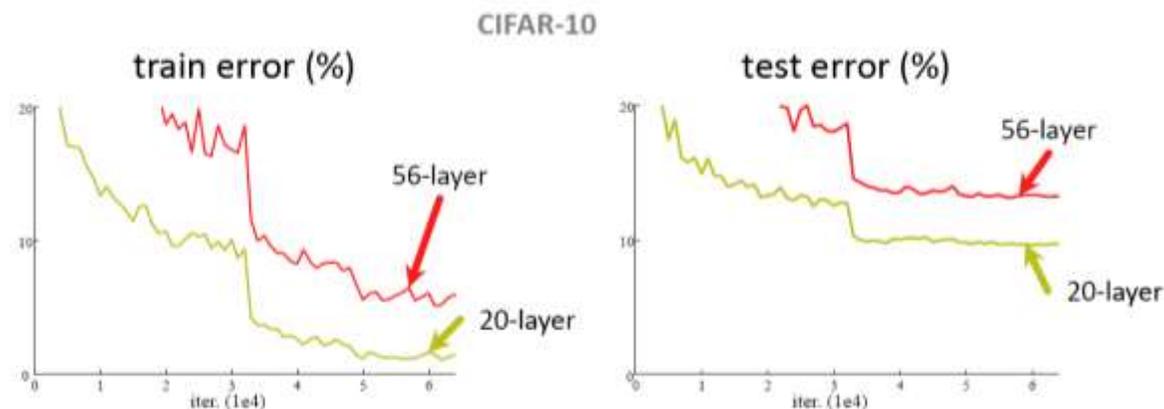
ResNet, Skip Connection и репараметризация



Проблема глубины

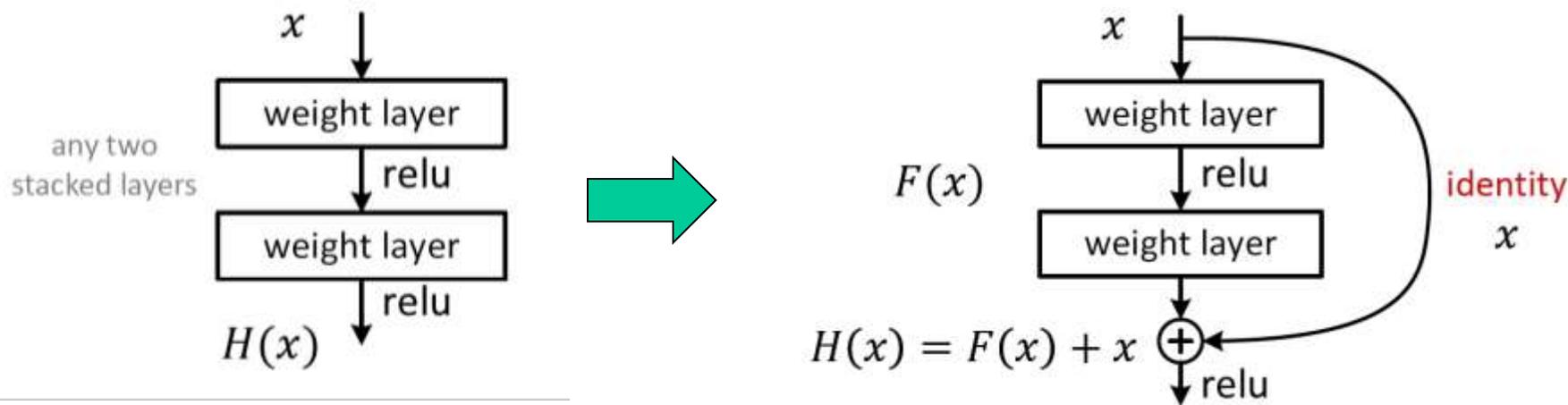


- Дальнейшее увеличение числа слоёв приводило к падению качества итоговой модели



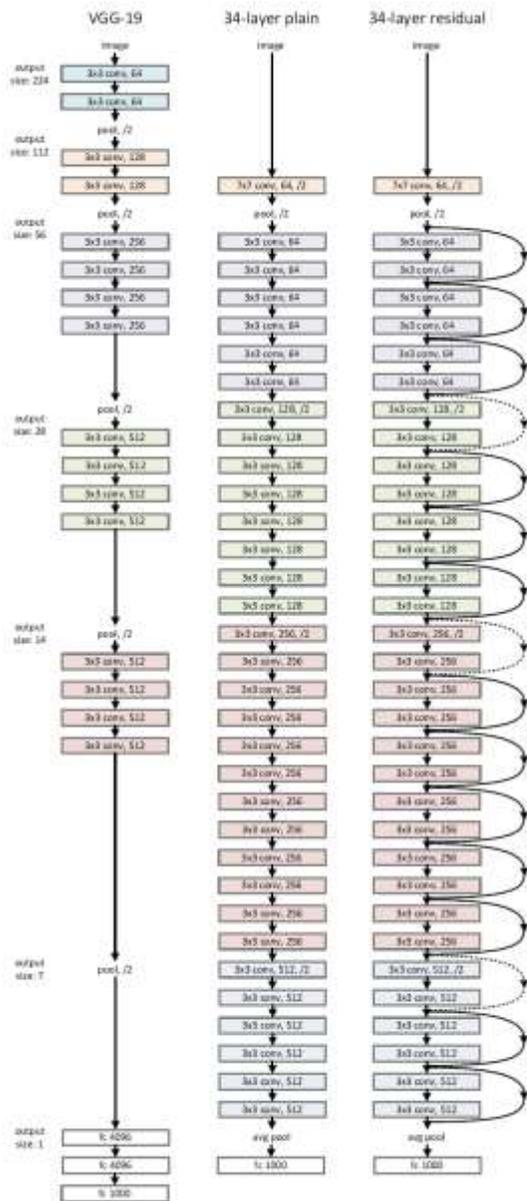
- При этом можно добавить «единичные» слои к сети, оставив ту же самую функцию и качество
- Значит, проблема в обучении сети

Residual net и skip-connections



- Добавим обходной путь (skip-connection) к блоку
- Будем учить не преобразование, а добавку (пертурбацию) к тождественному преобразования
 - Если единичное преобразование оптимально, тогда мы его сохраняем
 - Небольшие флуктуации оказывается обучать проще
- Также градиенты будут «обходить» блок и передаваться на предыдущий напрямую

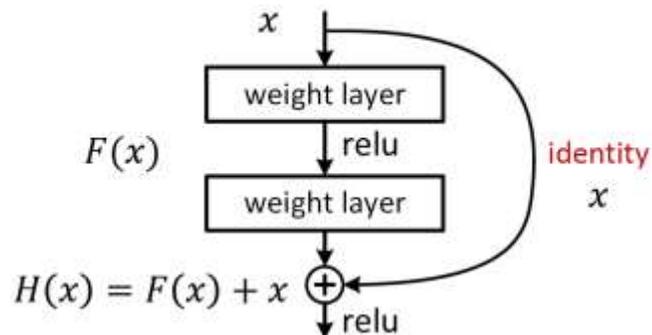
ResNet



Базовая модель

- Свёртки 3x3
- Subsampling через свёртку с шагом 2

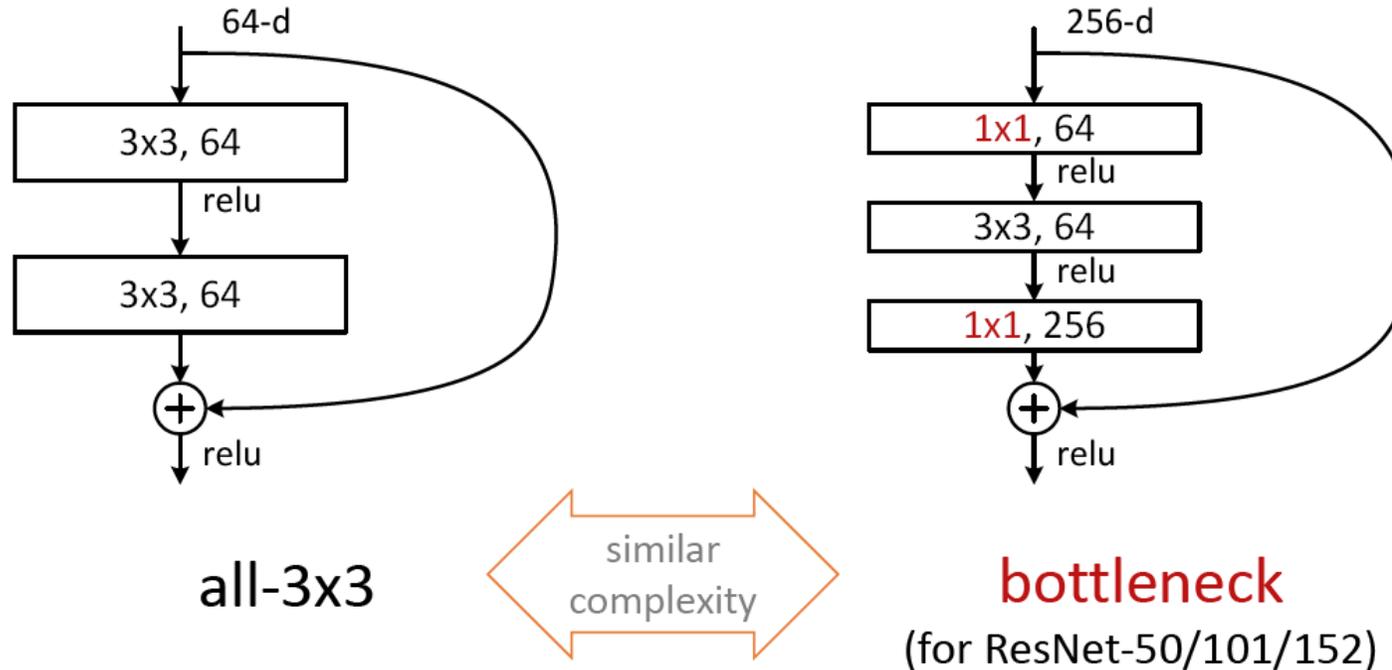
Residual net



При изменении размеров тензоров пробуют варианты:

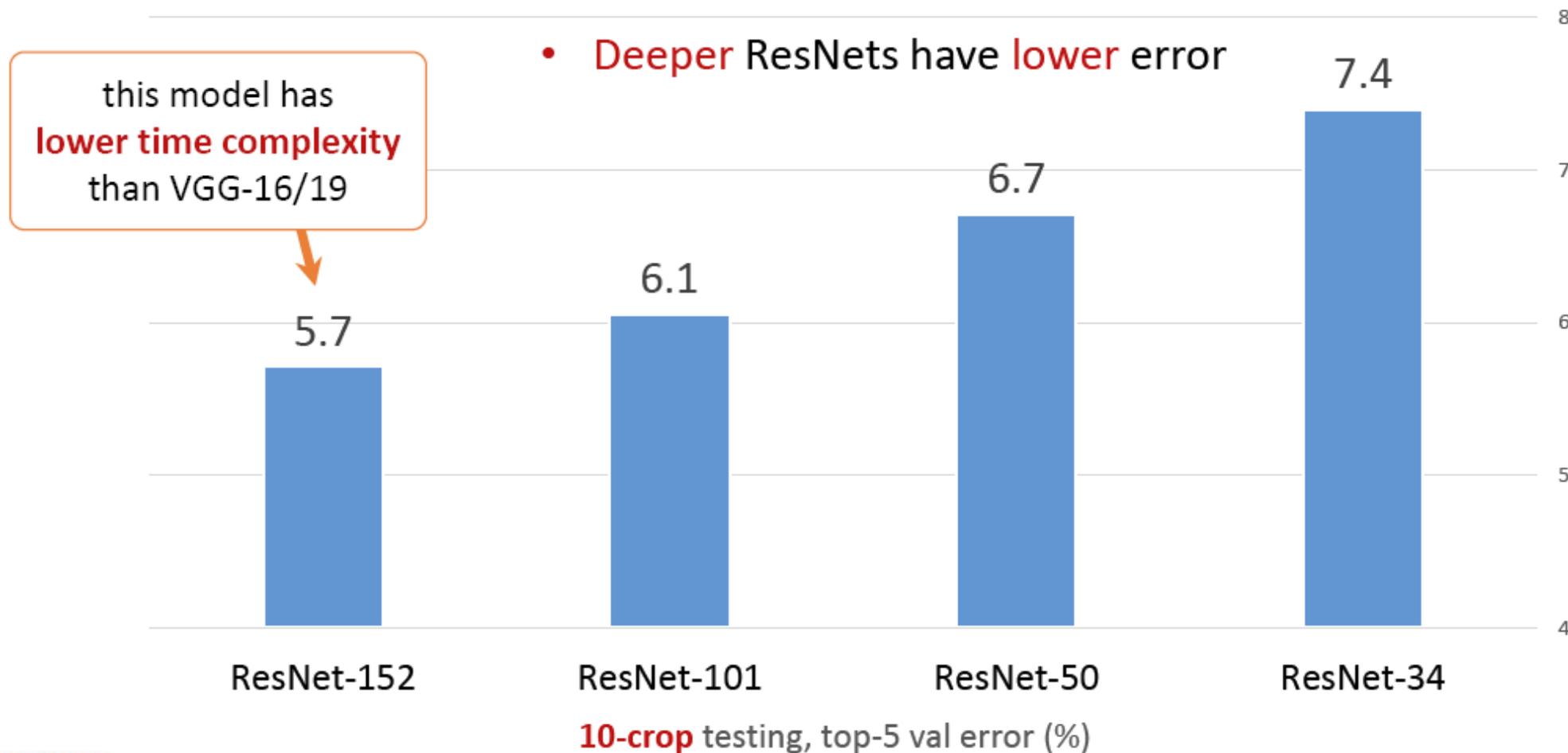
- Добавление нулями
- Линейная проекция
- Шаг 2

Повышение производительности



- Понижение размерности (256->64) за счёт 1x1 свёрток
- Свёртка 3x3 на тензоре меньшей глубины
- Повышение размерности свёртками 1x1

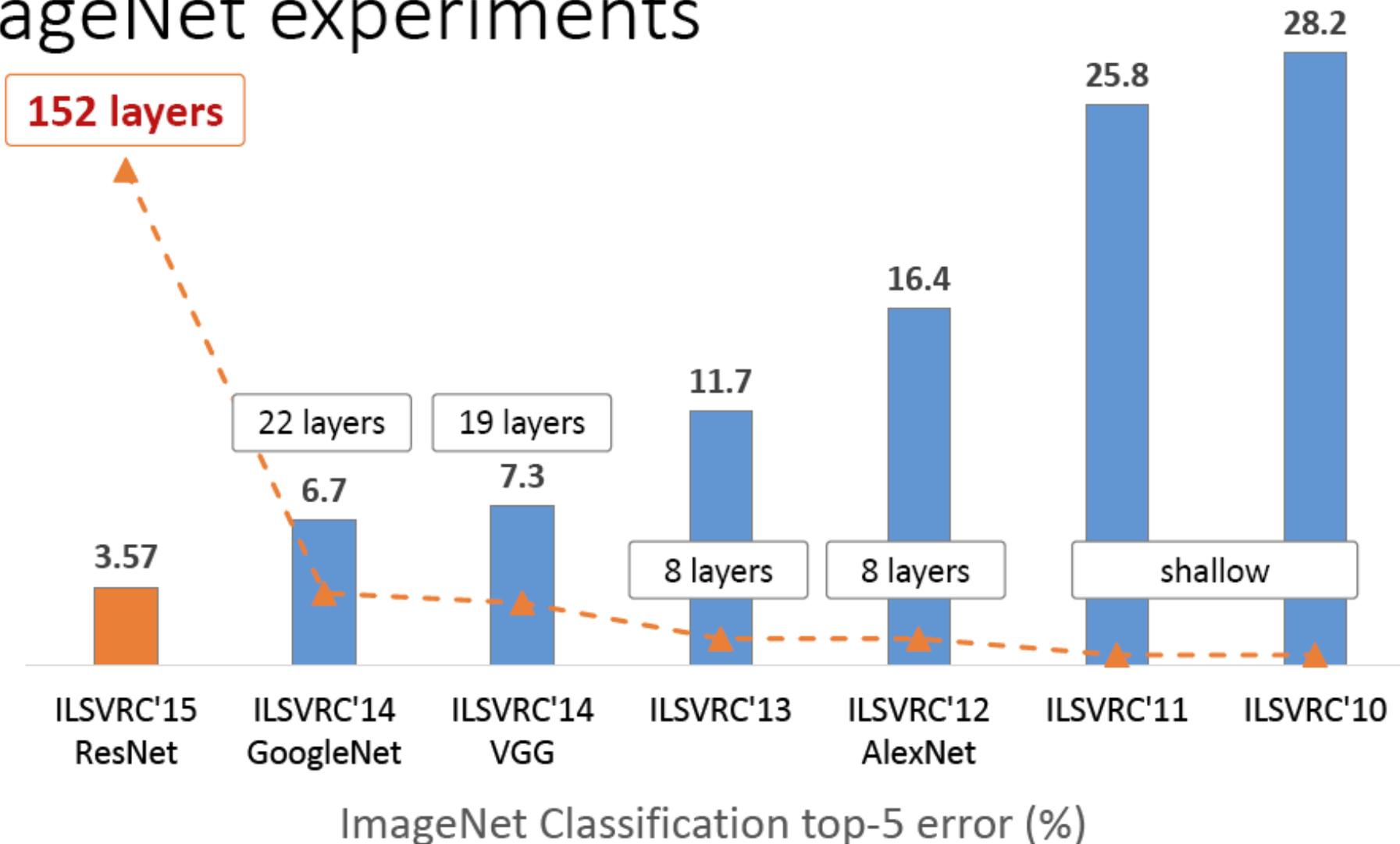
Результаты на ImageNet



Результаты на ImageNet



ImageNet experiments



ResNeXt

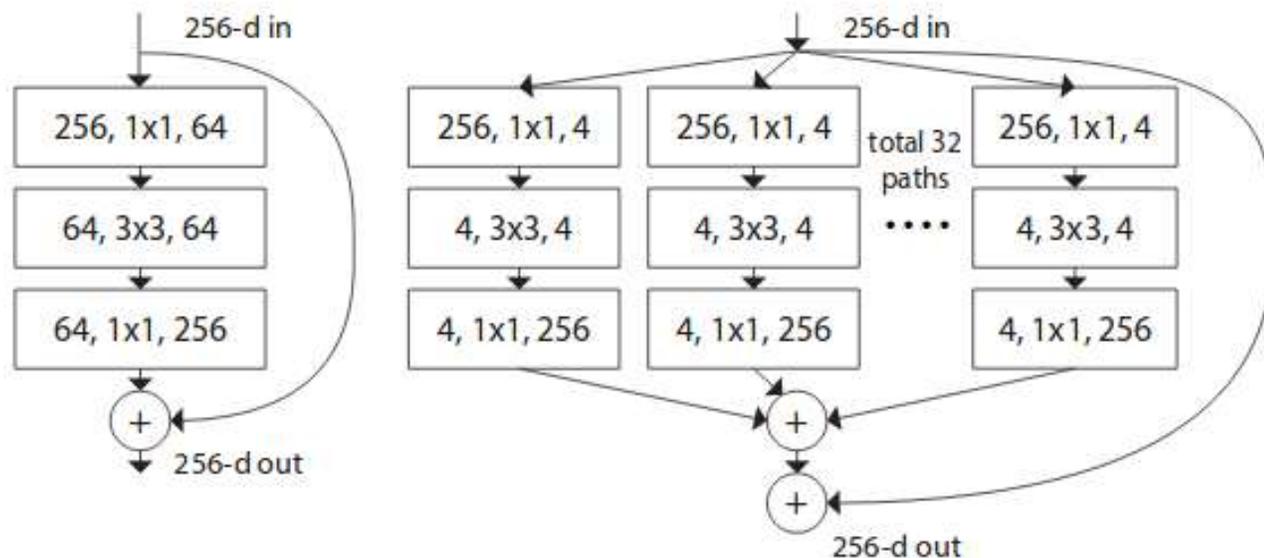


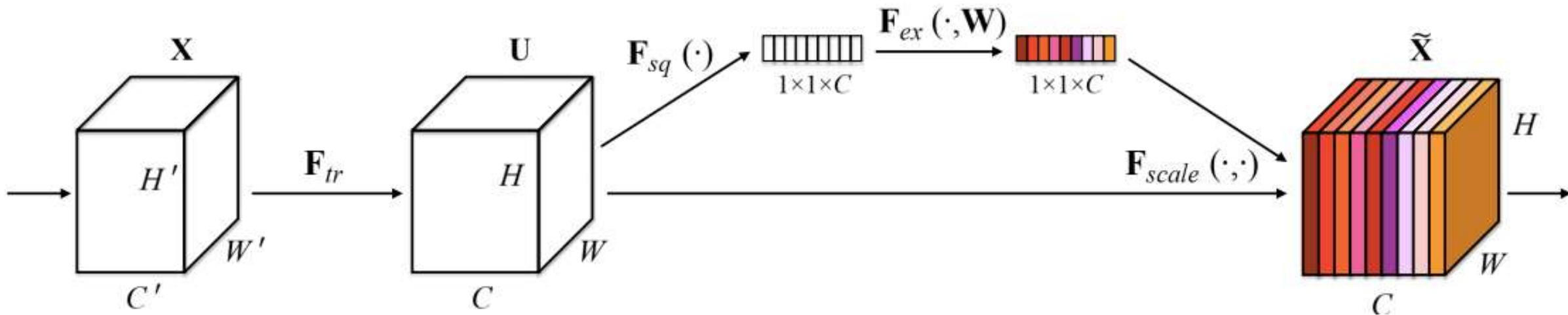
Figure 1. **Left:** A block of ResNet [14]. **Right:** A block of ResNeXt with cardinality = 32, with roughly the same complexity. A layer is shown as (# in channels, filter size, # out channels).

stage	output	ResNet-50	ResNeXt-50 (32×4d)
conv1	112×112	7×7, 64, stride 2	7×7, 64, stride 2
conv2	56×56	3×3 max pool, stride 2	3×3 max pool, stride 2
		$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128, C=32 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3	28×28	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256, C=32 \\ 1 \times 1, 512 \end{bmatrix} \times 4$
		$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512, C=32 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$
conv5	7×7	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 1024 \\ 3 \times 3, 1024, C=32 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
		1×1	global average pool 1000-d fc, softmax
# params.		25.5 ×10 ⁶	25.0 ×10 ⁶
FLOPs		4.1 ×10 ⁹	4.2 ×10 ⁹

Table 1. **(Left)** ResNet-50. **(Right)** ResNeXt-50 with a 32×4d template (using the reformulation in Fig. 3(c)). Inside the brackets are the shape of a residual block, and outside the brackets is the number of stacked blocks on a stage. “C=32” suggests grouped convolutions [24] with 32 groups. *The numbers of parameters and FLOPs are similar between these two models.*

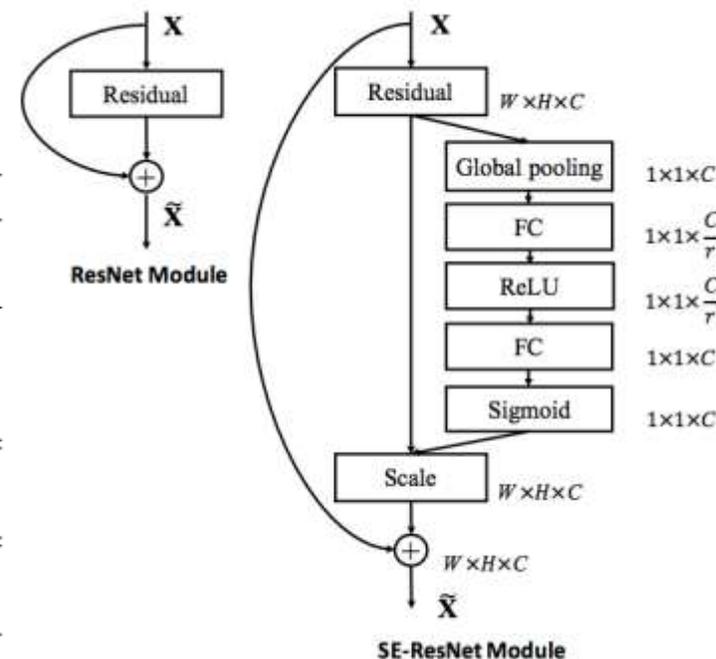


Squeeze-and-excitation blocks



- Вариант «attention»
- Подчёркиваем определённые признаки (каналы)

	original		re-implementation			SENet		
	top-1 err.	top-5 err.	top-1 err.	top-5 err.	GFLOPs	top-1 err.	top-5 err.	GFLOPs
ResNet-50 [9]	24.7	7.8	24.80	7.48	3.86	23.29 _(1.51)	6.62 _(0.86)	3.87
ResNet-101 [9]	23.6	7.1	23.17	6.52	7.58	22.38 _(0.79)	6.07 _(0.45)	7.60
ResNet-152 [9]	23.0	6.7	22.42	6.34	11.30	21.57 _(0.85)	5.73 _(0.61)	11.32
ResNeXt-50 [43]	22.2	-	22.11	5.90	4.24	21.10 _(1.01)	5.49 _(0.41)	4.25
ResNeXt-101 [43]	21.2	5.6	21.18	5.57	7.99	20.70 _(0.48)	5.01 _(0.56)	8.00
BN-Inception [14]	25.2	7.82	25.38	7.89	2.03	24.23 _(1.15)	7.14 _(0.75)	2.04
Inception-ResNet-v2 [38]	19.9 [†]	4.9 [†]	20.37	5.21	11.75	19.80 _(0.57)	4.79 _(0.42)	11.76



Проблема теоретической сложности



- Теоретическая вычислительная сложность плохо коррелирует с реальной производительностью модели
- Например, VGG-16 требует в 8.4x раз больше FLOPS, чем EfficientNet-B3 (о ней далее), но работает в 1.8 раз быстрее на GPU
- Поэтому у VGG-16 практическая «вычислительная плотность» (computational density) в 15 раз выше, чем у EfficientNet
- Почему так происходит?

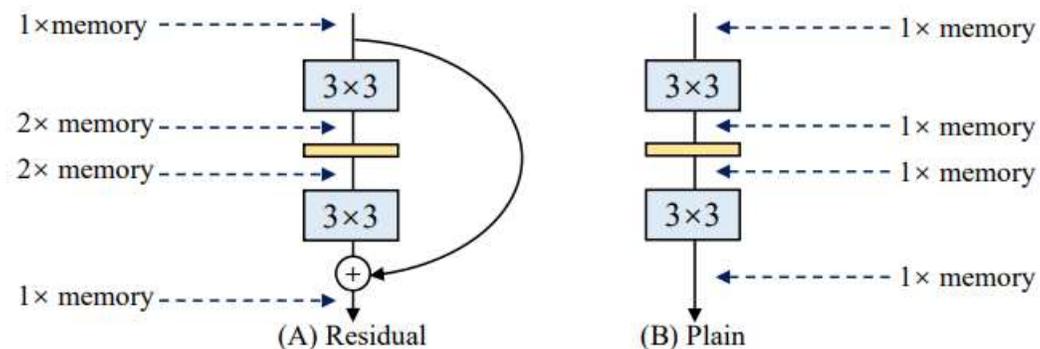


Практическая производительность

- Разные блоки с разной эффективностью реализуются
 - Winograd Convolution обеспечивает высокую эффективность для 3x3 свёрток
- Разная степень параллелизма
 - Каждый новый тип операции в блоке влечёт дополнительные расходы (overhead) и повышает фрагментацию
- Доступ к памяти (MAC) очень затратен по времени
- Дополнительные затраты памяти в процессе работы также снижают практическую производительность

Kernel size	Theoretical FLOPs (B)	Time usage (ms)	Theoretical TFLOPS
1 × 1	420.9	84.5	9.96
3 × 3	3788.1	198.8	38.10
5 × 5	10522.6	2092.5	10.57
7 × 7	20624.4	4394.3	9.38

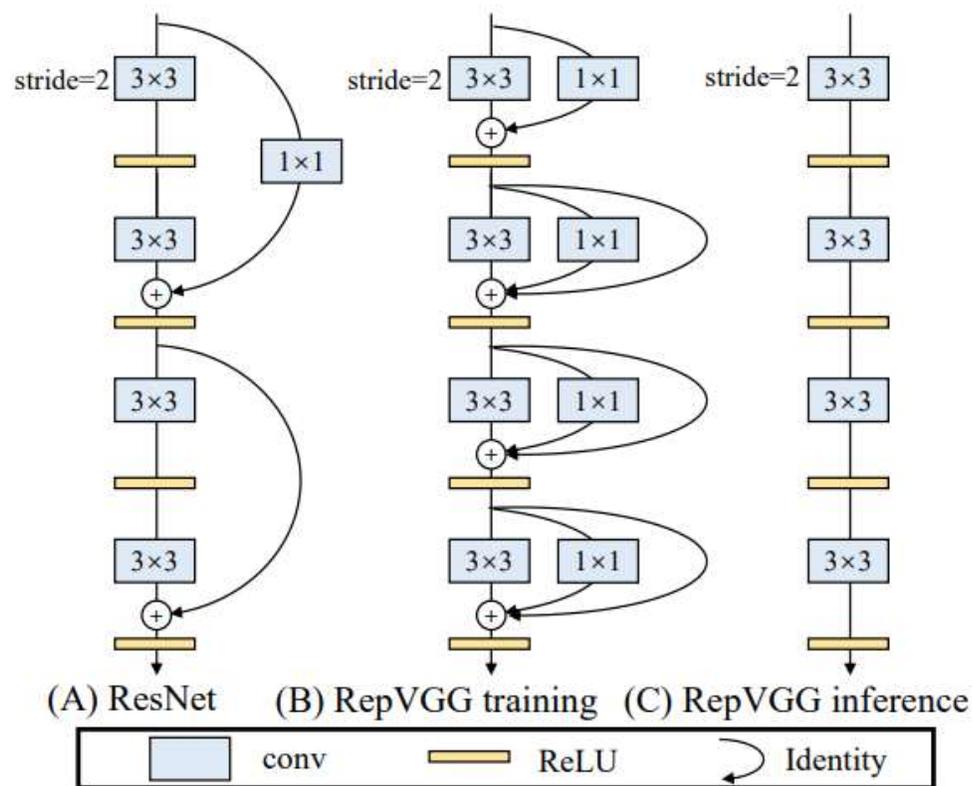
Вычислительная плотность разных свёрток



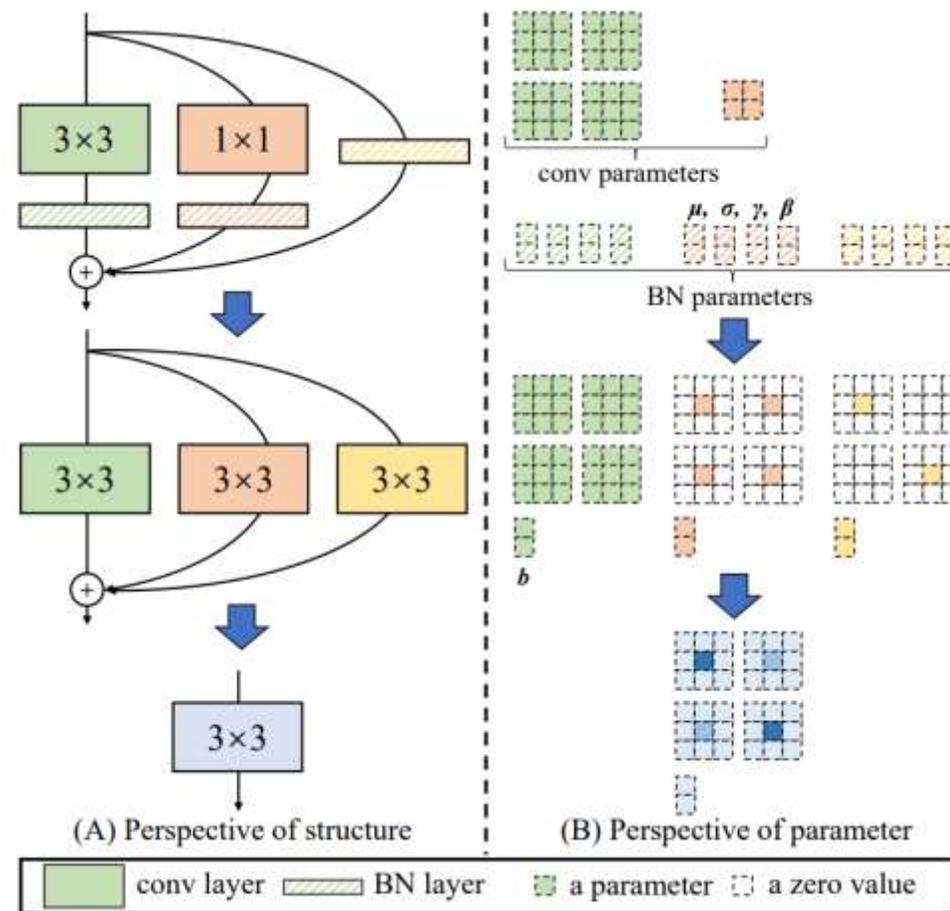
Затраты памяти при вычислении обходных путей



Репараметризация сетей



Учим модель с ветками, а затем объединяем в одну метку для инференса



Визуализация репараметризации набора веток в одну свёртку



Как обучить мощную модель?



Процесс обучения играет большую роль

Модификация процесса обучения может повысить точность ResNet-50 с ~76% до ~82% на ImageNet

Основные идеи:

- Продвинутая аугментация изображений
- Трюки в процессе обучения
 - Warmup
 - Cosine/linear decay learning rate
 - Exponential Moving Average (EMA)
- Дистилляция нейросети

CutMix



	ResNet-50	Mixup [48]	Cutout [3]	CutMix
Image				
Label	Dog 1.0	Dog 0.5 Cat 0.5	Dog 1.0	Dog 0.6 Cat 0.4
ImageNet Cls (%)	76.3 (+0.0)	77.4 (+1.1)	77.1 (+0.8)	78.6 (+2.3)
ImageNet Loc (%)	46.3 (+0.0)	45.8 (-0.5)	46.7 (+0.4)	47.3 (+1.0)
Pascal VOC Det (mAP)	75.6 (+0.0)	73.9 (-1.7)	75.1 (-0.5)	76.7 (+1.1)

Yun et.al. CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features.

<https://arxiv.org/abs/1905.04899> (Naver)

Дистилляция

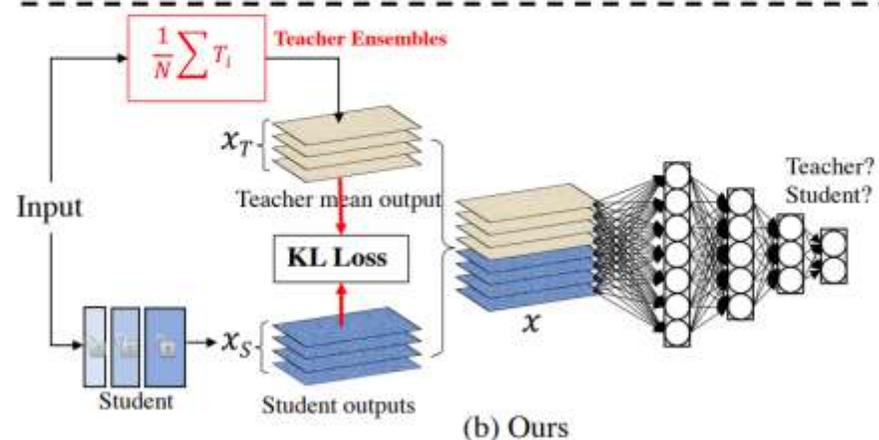
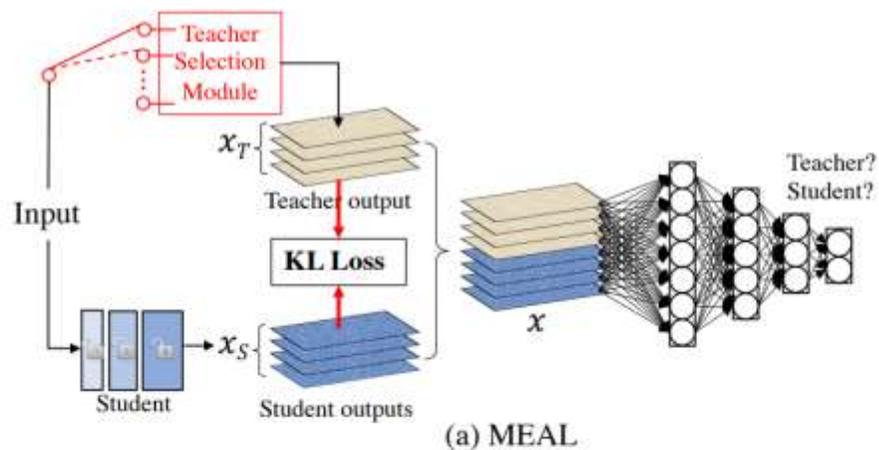


Figure 1. An illustration of the comparison between MEAL [18] and our method. We use an ensemble of all teacher networks instead of the teacher selection module as adopted in MEAL.

- Дистилляция – обучения сети студента так, чтобы его выходы были похожи на сеть-учителя
- Есть множество методов дистилляции
- Пример – MEAL V2
 - Берём комитет мощных учителей
 - Усредняем их выходы
 - Требуем от ученика похожести выходов на учителя
 - Добавляем классификатор учитель/ученик по признакам, градиент

Результаты

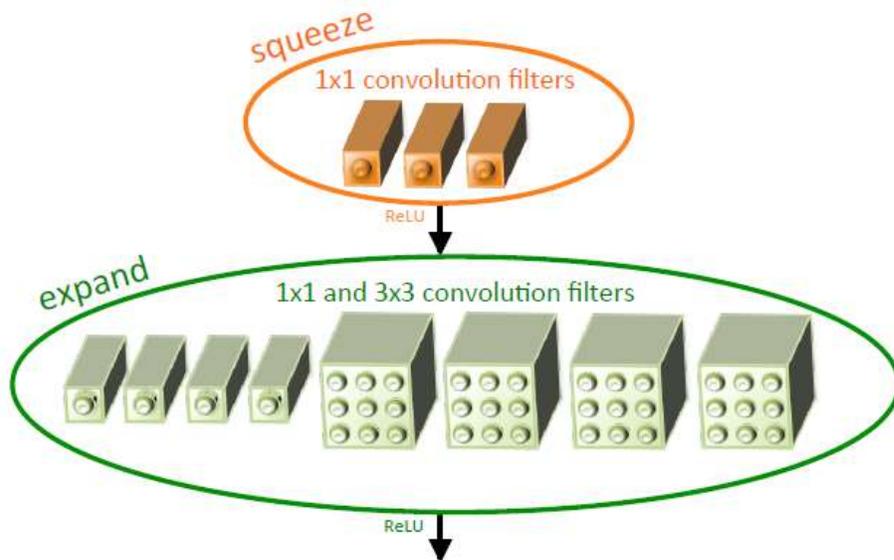


Network	Resolution	#Params	Top-1 (%)	Top-5 (%)
ResNet-50	224	25.6M	76.51	93.20
ResNet-50 + DropBlock, (kp=0.9) [5]	224	25.6M	78.13	94.02
ResNet-50 + DropBlock (kp=0.9) [5] + label smoothing (0.1)	224	25.6M	78.35	94.15
ResNet-50 + MEAL [18]	224	25.6M	78.21	94.01
ResNet-50 + Ours (MEAL V2)	224	25.6M	80.67	95.09
ResNet-50 + FixRes [22]	384	25.6M	79.0	94.6
ResNet-50 + FixRes (*) [22]	384	25.6M	79.1	94.6
ResNet-50 + Ours (MEAL V2)	380	25.6M	81.72	95.81
ResNet-50 + FixRes [22] + CutMix	320	25.6M	79.7	94.9
ResNet-50 + FixRes [22] + CutMix (*)	320	25.6M	79.8	94.9
ResNet-50 + Ours (MEAL V2) + CutMix	224	25.6M	80.98	95.35

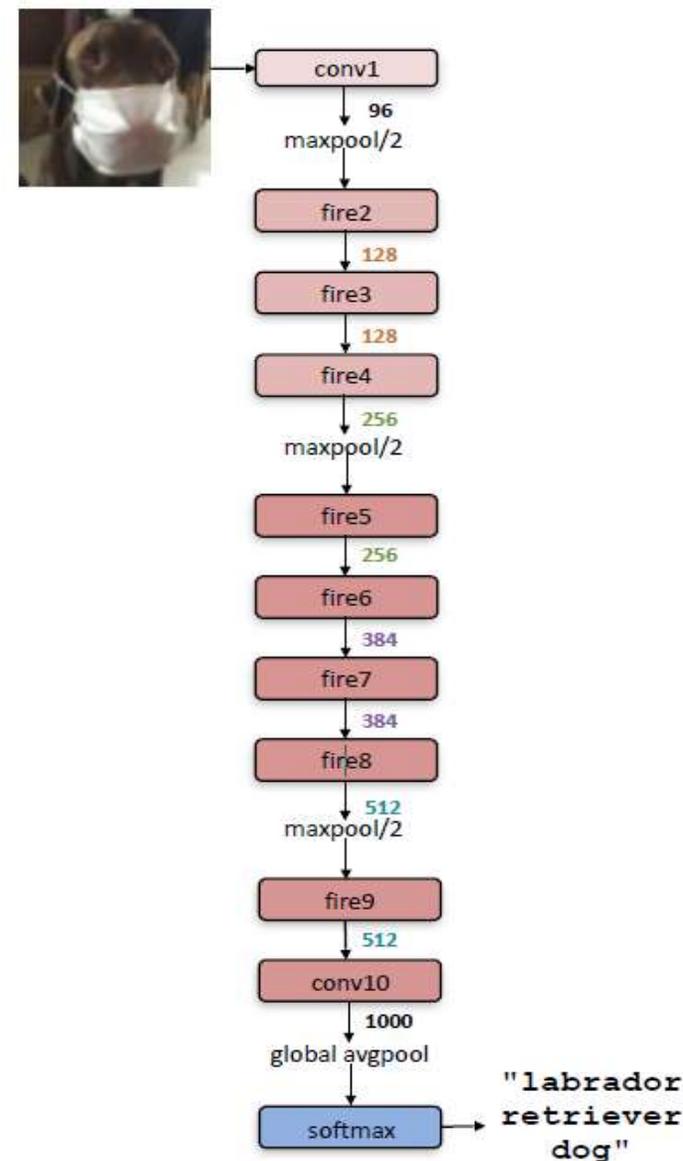


Быстрые свёрточные модели

SqueezeNet



- Активно использовать 1x1 свёртки для уменьшения числа параметров
- «Сжимать» мы будем для того, чтобы на вход 3x3 фильтрам подавать меньше данных



Факторизация свёртки

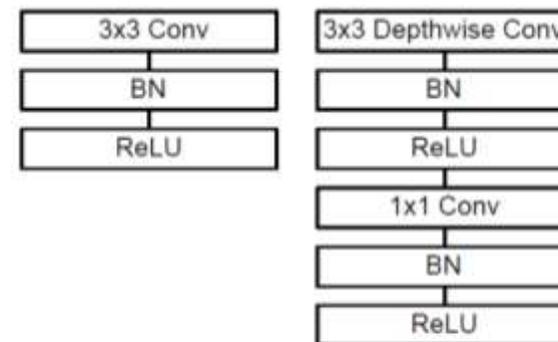
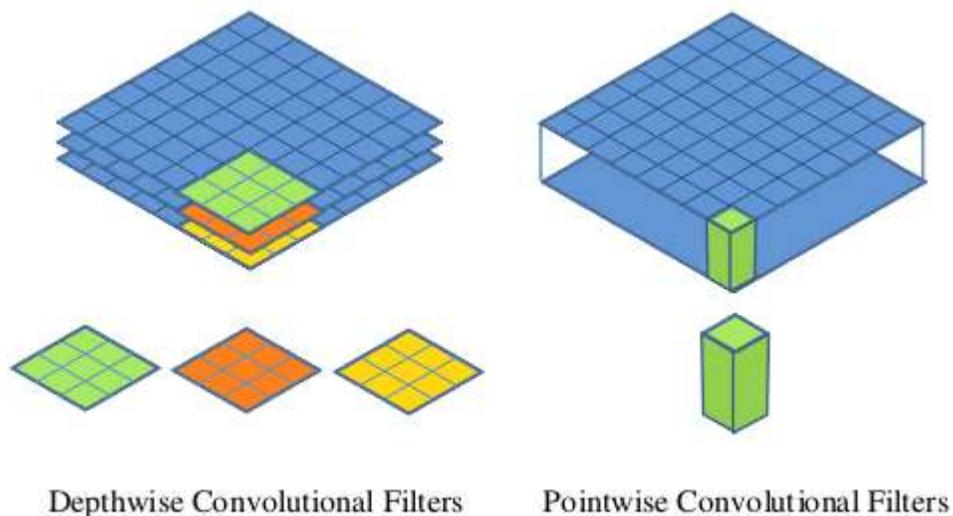


Figure 3. Left: Standard convolutional layer with batchnorm and ReLU. Right: Depthwise Separable convolutions with Depthwise and Pointwise layers followed by batchnorm and ReLU.

- Факторизуем свёртку в «послойную» (depthwise) и поточечную (pointwise)
- Сложность обычного свёрточного слоя:

$D_k \cdot D_k \cdot M \cdot N \cdot D_f \cdot D_f$, где D_k - разрешение входа, M - число каналов входа, N - число каналов выхода и D_f - разрешение выхода

- Сложность комбинации depthwise convolution + pointwise convolution:::

$$|D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F$$

MobileNet



Table 6. MobileNet Width Multiplier

Width Multiplier	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
0.75 MobileNet-224	68.4%	325	2.6
0.5 MobileNet-224	63.7%	149	1.3
0.25 MobileNet-224	50.6%	41	0.5

Table 7. MobileNet Resolution

Resolution	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
1.0 MobileNet-192	69.1%	418	4.2
1.0 MobileNet-160	67.2%	290	4.2
1.0 MobileNet-128	64.4%	186	4.2

Table 8. MobileNet Comparison to Popular Models

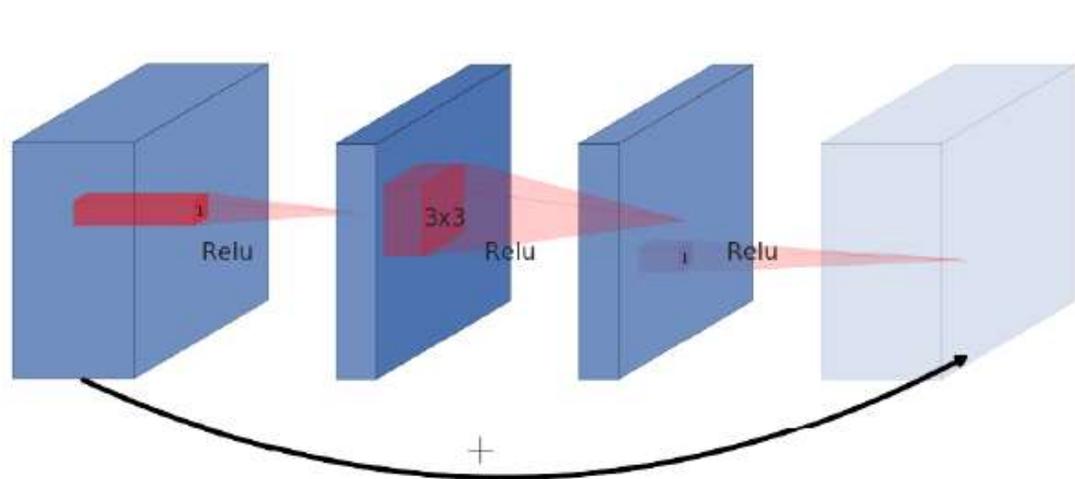
Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138

Table 4. Depthwise Separable vs Full Convolution MobileNet

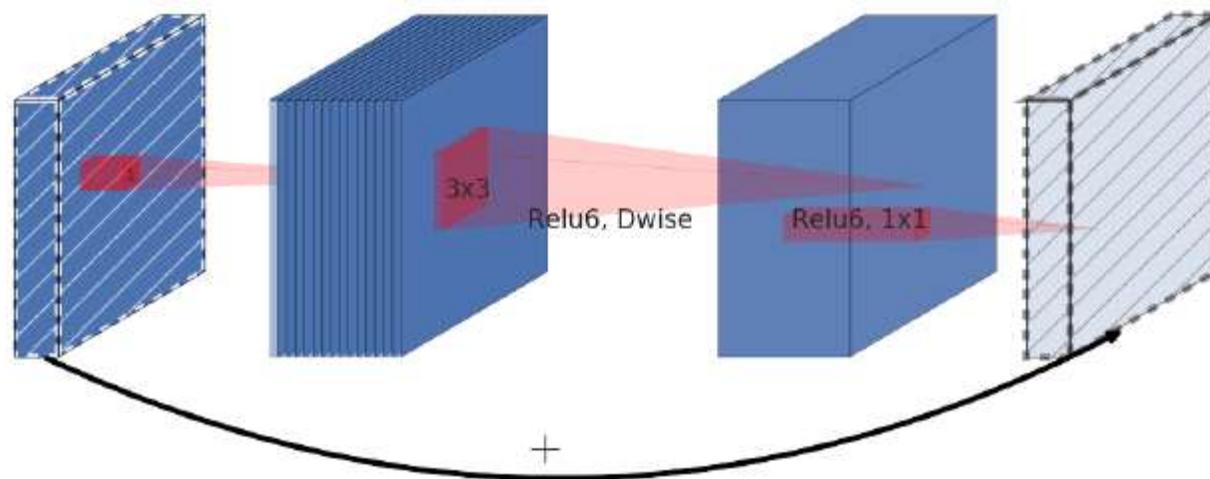
Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
Conv MobileNet	71.7%	4866	29.3
MobileNet	70.6%	569	4.2

- VGG-подобная архитектура с факторизованными свёртками

MobileNetV2



Обычный residual block

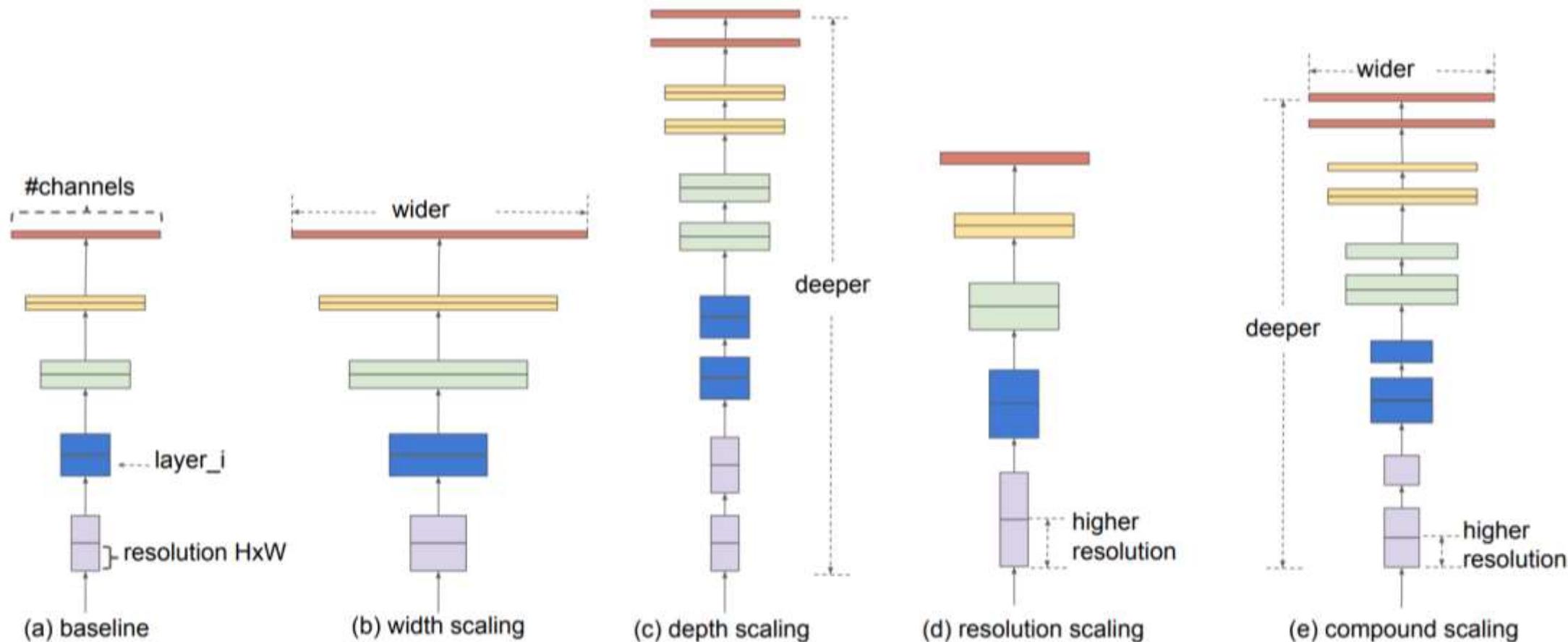


Инвертированный residual block

В чём разница?

<https://arxiv.org/abs/1801.04381>

EfficientNet



Выделяем коэффициент масштабирования для пропорционального увеличения сложности всех элементов

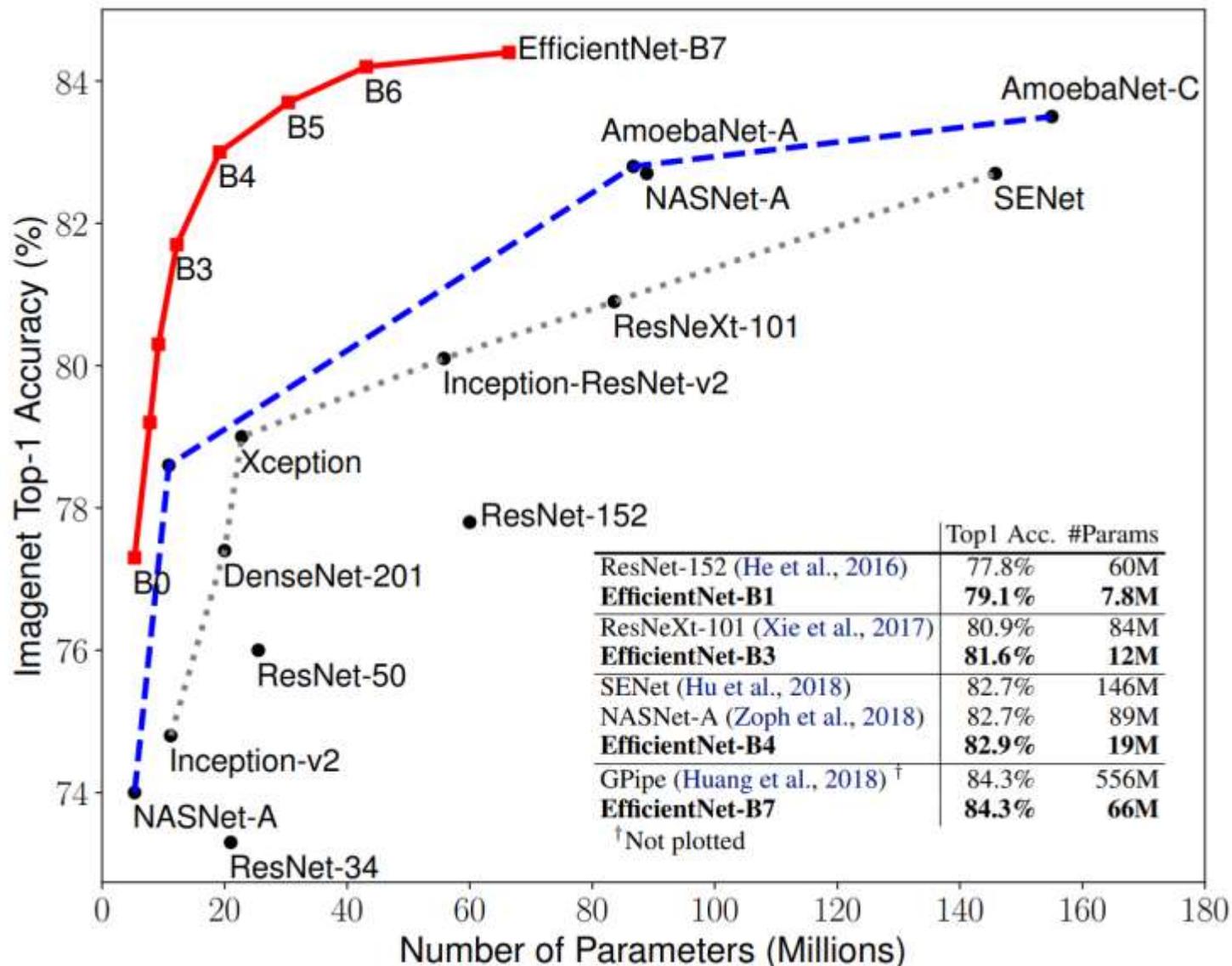
EfficientNet



depth: $d = \alpha^\phi$
width: $w = \beta^\phi$
resolution: $r = \gamma^\phi$
s.t. $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$
 $\alpha \geq 1, \beta \geq 1, \gamma \geq 1$

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels \hat{C}_i	#Layers \hat{L}_i
1	Conv3x3	224×224	32	1
2	MBCConv1, k3x3	112×112	16	1
3	MBCConv6, k3x3	112×112	24	2
4	MBCConv6, k5x5	56×56	40	2
5	MBCConv6, k3x3	28×28	80	3
6	MBCConv6, k5x5	28×28	112	3
7	MBCConv6, k5x5	14×14	192	4
8	MBCConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

Семейство EfficientNetB0-B7



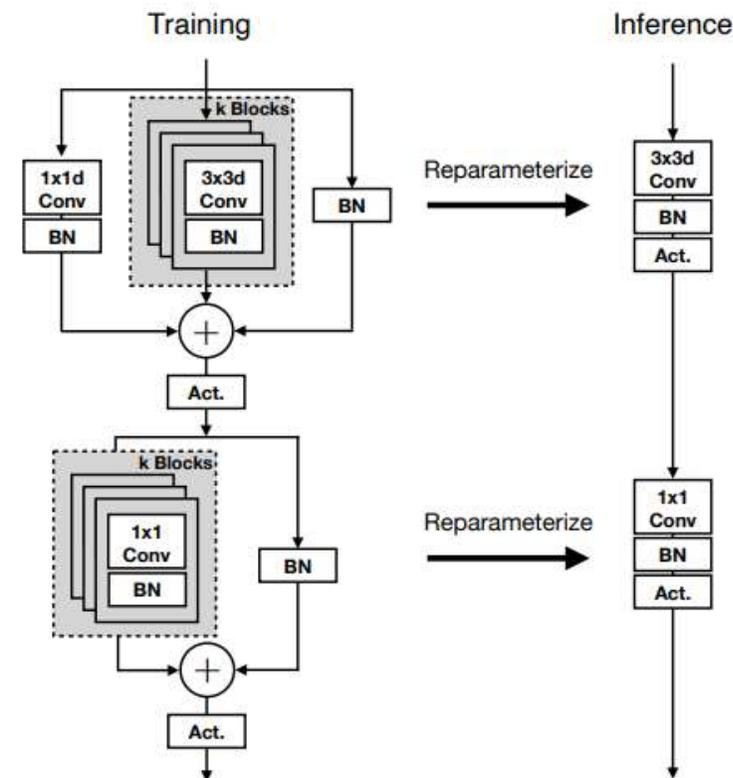
MobileOne



- Тщательное измерение влияния разных факторов на время инференса именно на мобильных устройствах
- Ветки, сложные функции активации, размер изображения резко снижают производительность
- Берём MobileNetV1, учим с ветками + репараметризация
- Учим хорошо, со всеми трюками + дистилляцией

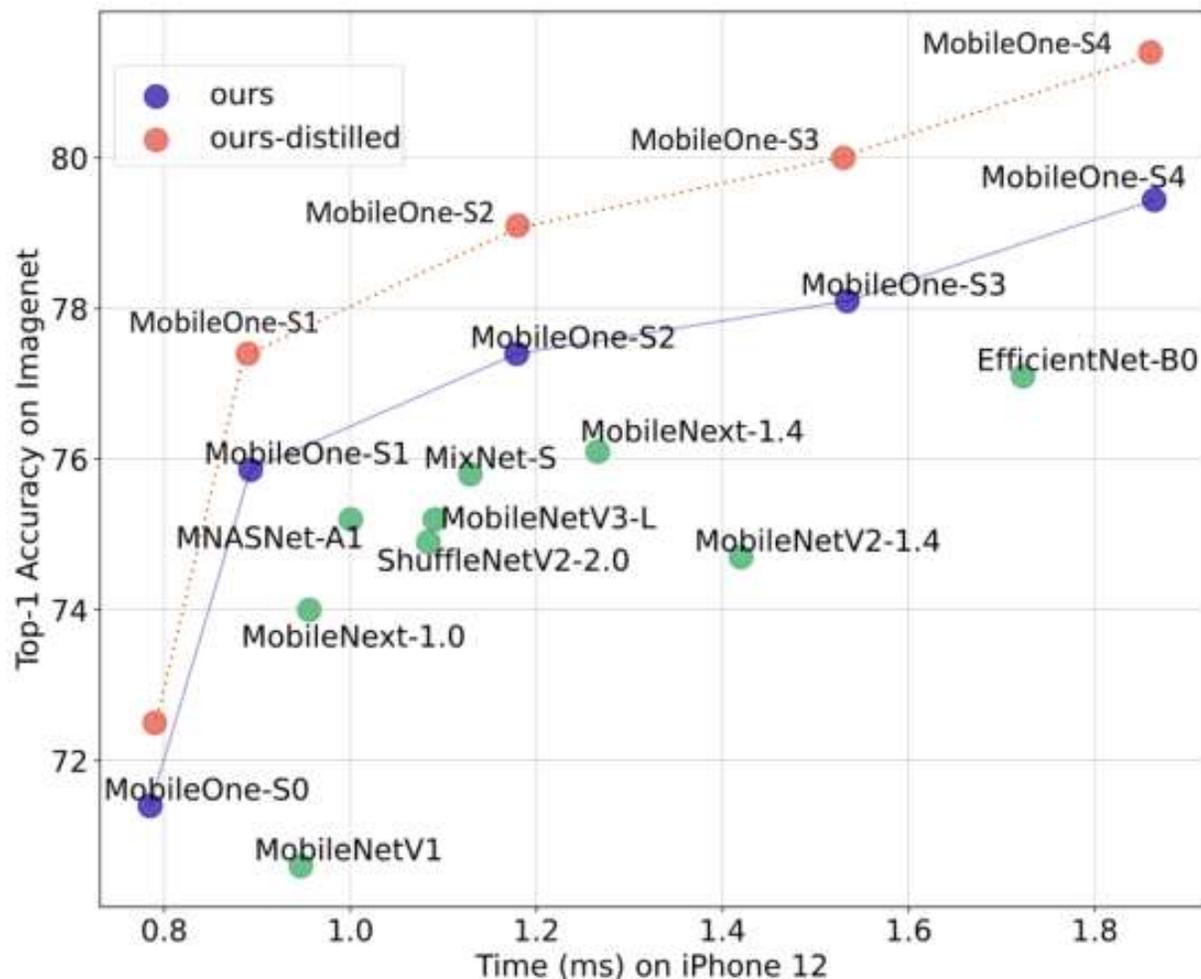
Activation Function	Latency (ms)
ReLU [1]	1.53
GELU [27]	1.63
SE-ReLU [32]	2.10
SiLU [15]	2.54
Dynamic Shift-Max [36]	57.04
DynamicReLU-A [6]	273.49
DynamicReLU-B [6]	242.14

Table 2. Comparison of latency on mobile device of different activation functions in a 30-layer convolutional neural network.

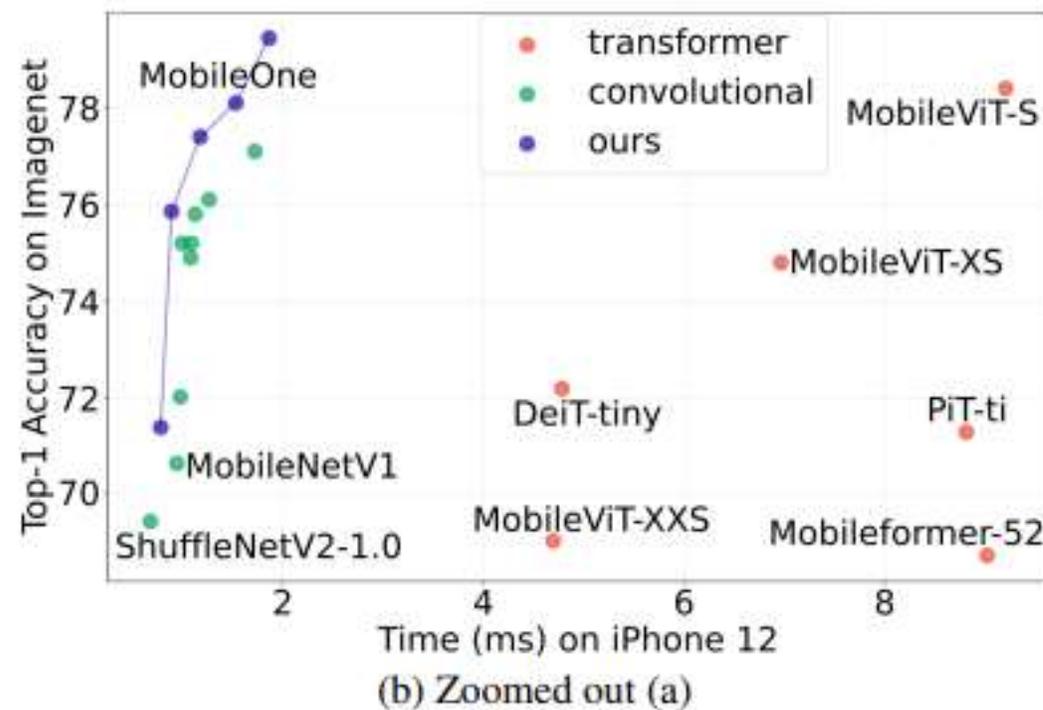


Базовый блок, похожий на MobileNetV1 + skip connection + репараметризация

MobileOne



(a) Top 1 accuracy vs Latency on iPhone 12.



Эффективные свёрточные сети пока обходят трансформенные модели на мобильных устройствах



Насколько хороша ImageNet?

Примеры работы



mite container ship motor scooter leopard

mite	container ship	motor scooter	leopard
black widow	lifeboat	go-kart	jaguar
cockroach	amphibian	moped	cheetah
tick	fireboat	bumper car	snow leopard
starfish	drilling platform	golfcart	Egyptian cat



grille mushroom cherry Madagascar cat

grille	mushroom	cherry	Madagascar cat
convertible	agaric	dalmatian	squirrel monkey
pickup	mushroom	grape	spider monkey
beach wagon	jelly fungus	elderberry	titi
fire engine	gill fungus	ffordshire bullterrier	indri
	dead-man's-fingers	currant	howler monkey

Проблемы ImageNet



Old label: pier
ReaL: dock; pier;
speedboat; sandbar;
seashore



Old label: quill
ReaL: feather boa



Old label: sunglass
ReaL: sunglass;
sunglasses



Old label: hammer
ReaL: screwdriver;
hammer; power drill;
carpenter's kit



Old label: water jug
ReaL: water bottle



Old label: sunglasses
ReaL: sunglass;
sunglasses



Old label: monitor
ReaL: mouse; desk;
desktop computer; lamp;
studio couch; monitor;
computer keyboard



Old label: chain
ReaL: necklace



Old label: laptop
ReaL: notebook;
laptop; computer keyboard



Old label: zucchini
ReaL: broccoli;
zucchini; cucumber;
orange; lemon; banana



Old label: purse
ReaL: wallet



Old label: notebook
ReaL: notebook;
laptop; computer keyboard



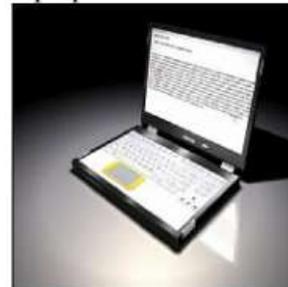
Old label: ant
ReaL: ant; ladybug



Old label: passenger car
ReaL: school bus



Old label: laptop
ReaL: notebook;
laptop



Переоценка последних методов

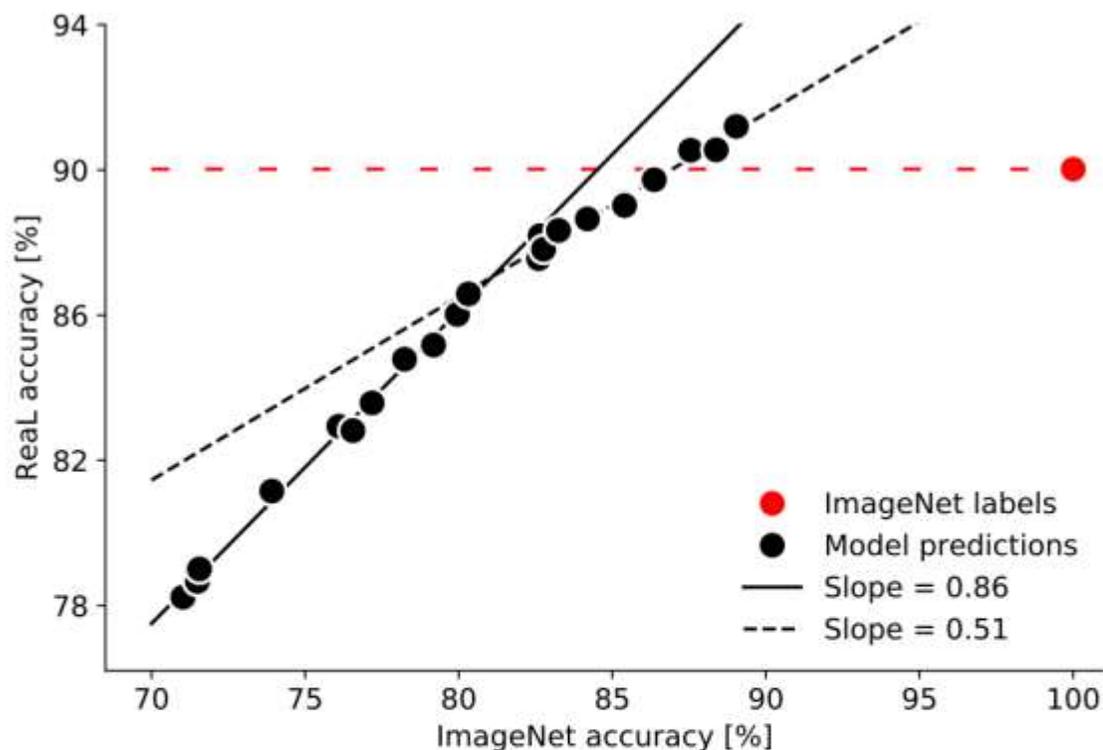


Figure 4: Comparing progress on RealL accuracy and the original ImageNet accuracy. We measured the association between both metrics by regressing ImageNet accuracy onto RealL accuracy for the first (solid line) and second half (dashed line) of the models in our pool.

Возможность улучшения

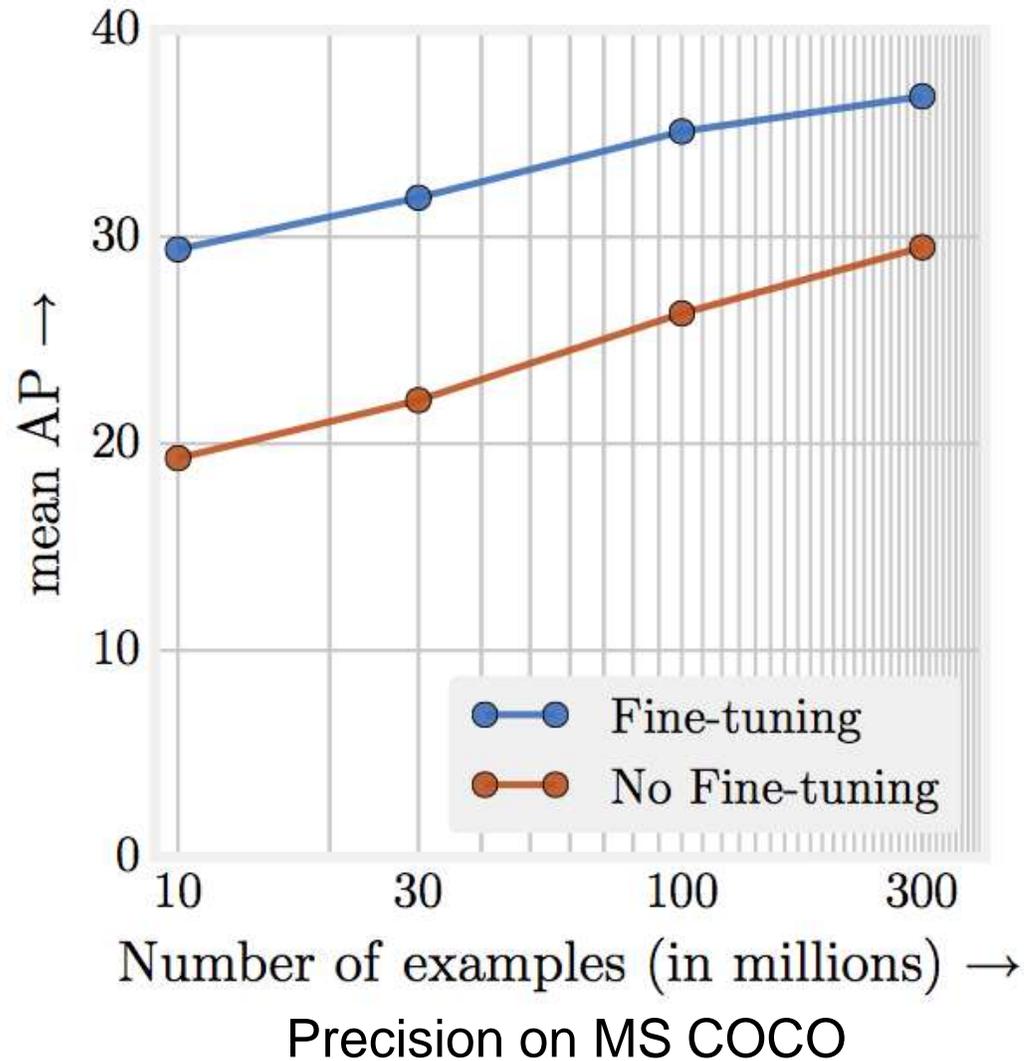


Table 2: Top-1 accuracy (in percentage) on ImageNet with our proposed sigmoid loss and clean label set. Median accuracy from three runs is reported for all the methods. Either sigmoid loss or clean label set leads to consistent improvements over baseline. Using both achieves the best performance. The improvement of our proposed method is more pronounced with longer training schedules.

Model		ImageNet accuracy			ReaL accuracy		
		90 epochs	270 epochs	900 epochs	90 epochs	270 epochs	900 epochs
ResNet-50	Baseline	76.0	76.9 (+0.9)	75.9 (-0.1)	82.5	82.9 (+0.4)	81.6 (-0.9)
	+ Sigmoid	76.3 (+0.3)	77.8 (+1.8)	76.9 (+0.9)	83.0 (+0.5)	83.9 (+1.4)	82.7 (+0.2)
	+ Clean	76.4 (+0.4)	77.8 (+1.8)	77.4 (+1.4)	82.8 (+0.3)	83.7 (+1.2)	83.3 (+0.8)
	+ Both	76.6 (+0.6)	78.2 (+2.2)	78.5 (+2.5)	83.1 (+0.6)	84.3 (+1.8)	84.1 (+1.6)
ResNet-152	Baseline	78.0	78.3 (+0.3)	77.1 (-0.9)	84.1	83.8 (-0.3)	82.3 (-1.8)
	+ Sigmoid	78.5 (+0.5)	78.7 (+0.7)	77.4 (-0.6)	84.6 (+0.5)	84.3 (+0.2)	82.7 (-1.4)
	+ Clean	78.6 (+0.6)	79.6 (+1.6)	79.0 (+1.0)	84.4 (+0.3)	85.0 (+0.9)	84.4 (+0.3)
	+ Both	78.7 (+0.7)	79.8 (+1.8)	79.3 (+1.3)	84.6 (+0.5)	85.2 (+1.1)	84.5 (+0.4)



Использование других данных



- JFT-300M by Google
- 300M изображений,, 1B меток 18291 категорий
- После фильтрации осталось 375M меток, но всё равно ~20% ошибок
- Обучение ResNet-101 на 50 x K80 в течение месяца
- Полученную модель использовали как предобученную для решения других задач

Chen Sun et. al. Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. <https://arxiv.org/abs/1707.02968>

Резюме базовых классификаторов



- Основные принципы построения архитектур
 - Строим глубокие модели из похожих блоков
 - Большие свёртки можно приближать последовательностью свёрток 3×3
 - Свёртки 1×1 позволяют управлять «толщиной» слоя и уменьшать вычислительную сложность
 - Можно обработать данные параллельно в нескольких ветвях и затем объединить
 - Residual-связи (или skip-connections) позволяют снизить проблему затухания градиента и обучать очень глубокие сети, но на инференсе их лучше убирать
 - Развиваем attention и факторизуем свёртки
- Обучить лучше на больших коллекциях, использовать хитрые (и странные) аугментации и аккуратно чистить тестовые данные