



Школа Анализа
Данных Яндекса



Факультет
компьютерных наук
НИУ ВШЭ



Лаборатория компьютерной графики
и мультимедиа ВМК МГУ

Курс «Компьютерное зрение» Лекция №6

«Свёрточные нейросети, часть 2 – визуализация,
архитектуры, классификация изображений и поиск
похожих»

Антон Конушин
1 апреля 2019 года

Нейросети – победители LSVR 2012 года



Car

Vegetable, veggie, veg
Edible seeds or roots or stems or leaves or bulbs or tubers or nonsweet fruits of any of numerous herbaceous plant

Number of images: the number of images in the subtree

ImagNet 2011 Winter Release (17)
animal, animalia having, beast, being, sport, athletics (165)
fabric, cloth, material, textile (2)
instrumentality, instrumentation, appliance (50)
structure, construction (1238)
fruit (308)
flower (461)
fungus (392)
tree (932)
vegetable, veggie, veg (175)
annual, biennial, perennial, fennel, finocchio (14)
cucumber, cucumber (14)
squash (14)
cucurbitaceous vegetable (18)
peppermint, rhubarb (0)
root vegetable (21)
solanaceous vegetable (23)
greens, green, leafy vegetable, poached (0)
legume (37)
raw vegetable, rabbit food (0)
artichoke, globe artichoke (0)
artichoke heart (0)
asparagus (0)
plantain (0)
truffle, earthnut (0)
pumpkin (0)
mushroom (0)

Treemap Visualization Images of the Synset Downloads

ImagNet 2011 Winter Release - Vegetable, veggie, veg

A treemap visualization of the ImagNet 2011 Winter Release dataset. The main area is divided into several large rectangular regions, each representing a category of objects. The categories include Legume, Greens, Roots, Fruits, and various vegetables like Carrot, Tomato, Onion, Celery, and Potato. Each region contains a grid of smaller images, with the size of each image indicating the number of images in that specific category. The overall layout is a hierarchical tree structure where the root node is "Vegetable, veggie, veg".

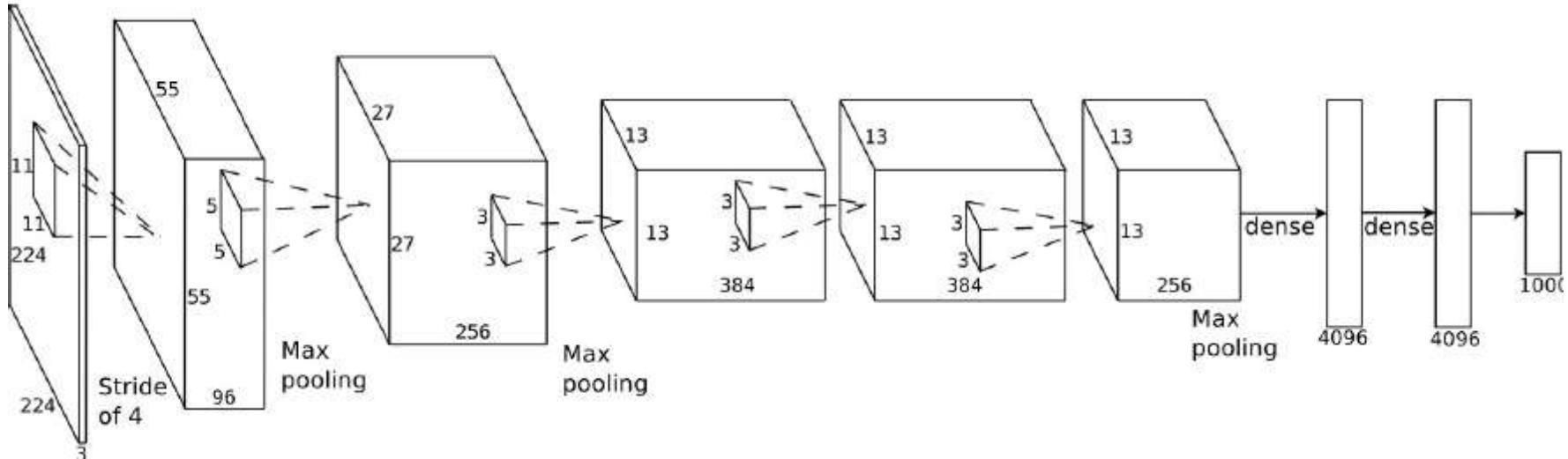
<http://www.image-net.org>

Winner

SuperVision

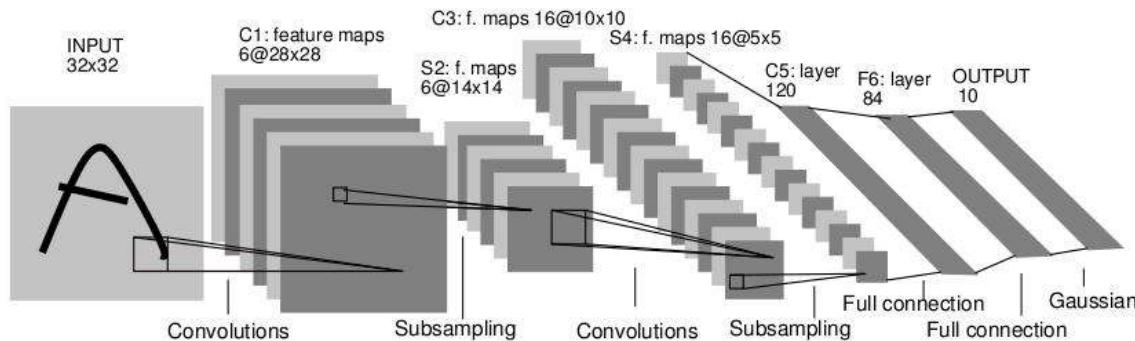
Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton
University of Toronto

SuperVision



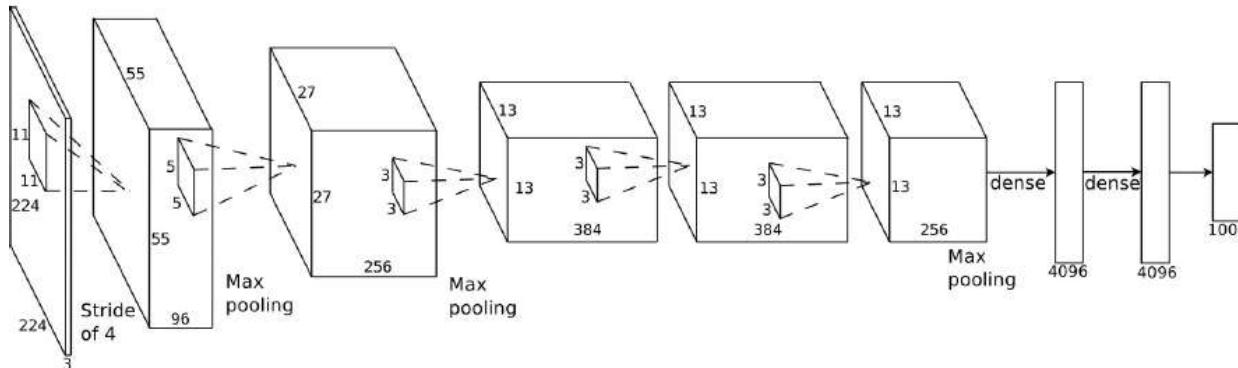
- 650,000 neurons
- 60,000,000 parameters
- 630,000,000 connections
- 1 машина, 2 GPU по 2Gb, 5GB Ram, 27Gb HDD, 1 неделя на обучение

CNN раньше и сейчас



1998 год

- 2 свёрточных слоя (6 и 6 фильтров)
- 2 полносвязанных (120 и 84 нейрона)

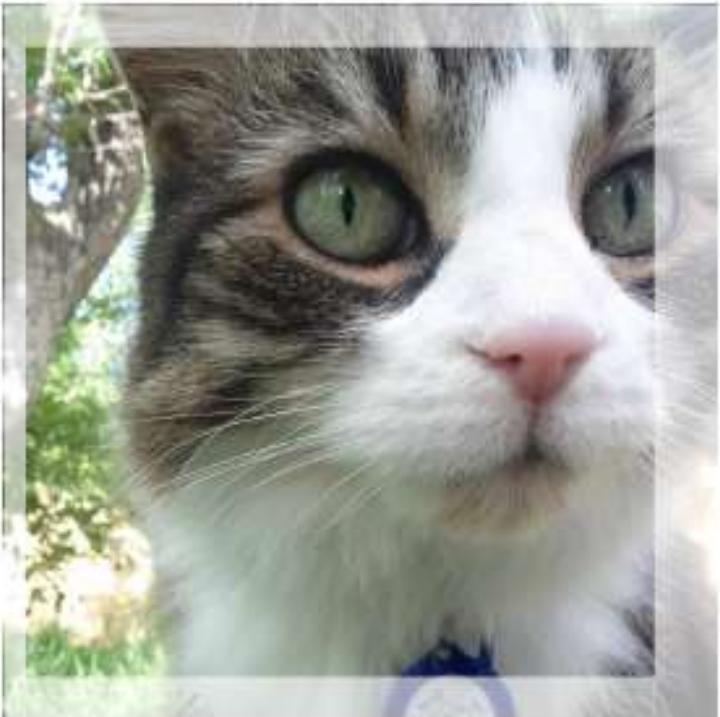


2012 год

- 5 свёрточных слоёв (96, 256, 384, 384, 256 фильтров)
- 2 полносвязанных (4096 и 4096 нейрона)

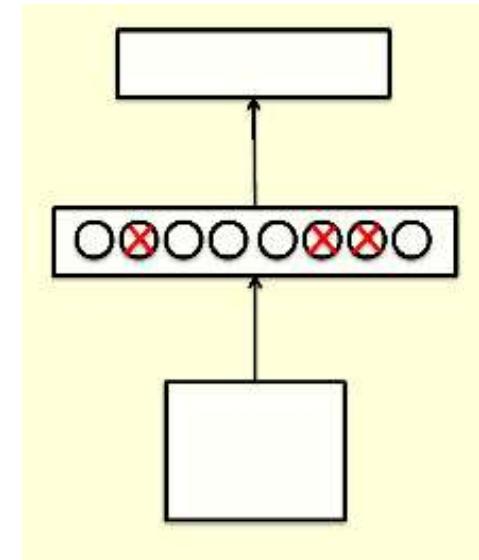
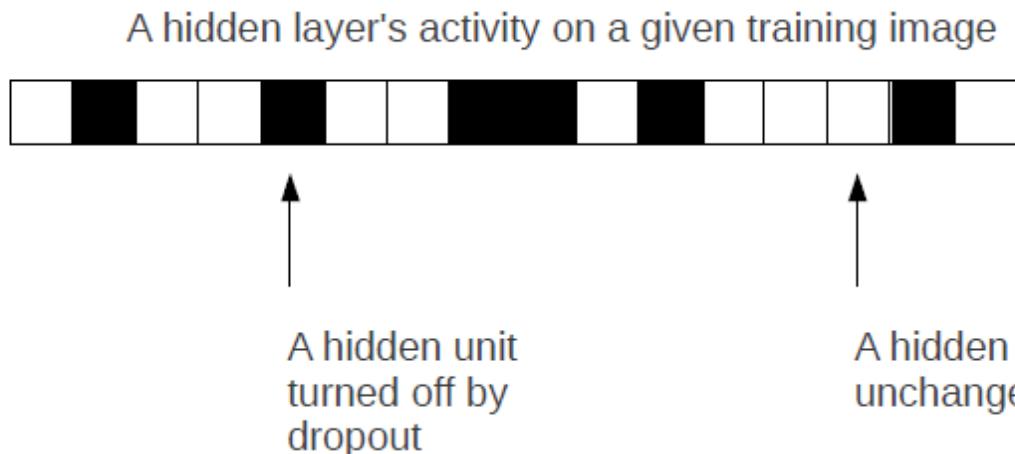
- Больше слоёв, фильтров, нейронов
- За счёт большого объёма данных, вычислительной мощности и некоторых улучшений (ReLU и т.д.) смогли обучить такую большую сеть

Размножение данных (data augmentation)



- Борьба с переобучением и повышение обучающей способности
- Геометрические методы:
 - Случайные crop (пр. вырезаем 224x224 из 256x256)
 - Масштабирование
 - Сдвиги
 - Повороты
- Цветовые искажения
 - Обычно сдвиги в RGB
- Можем сделать гораздо умнее!!

Dropout

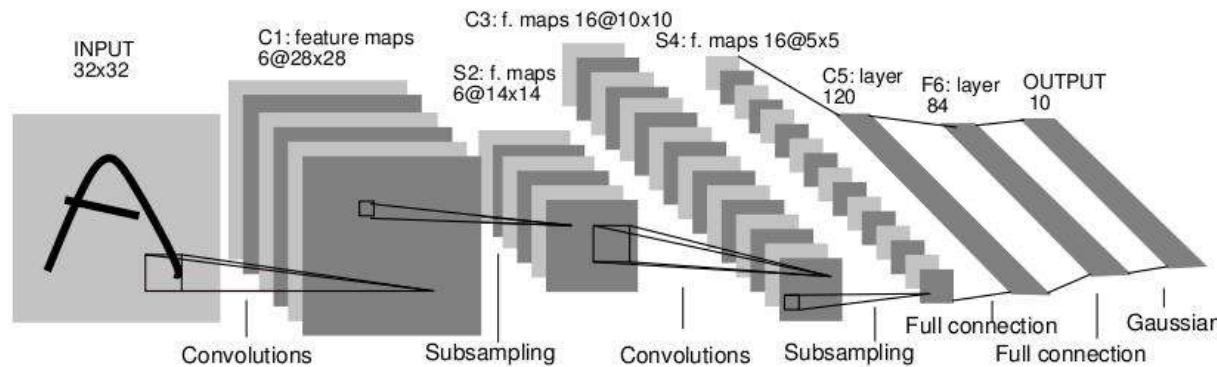


- Отключаем половину нейронов в каждом слое
- Получаем случайную выборку из множества сетей
- Во время тестирования используем «среднюю» сеть с уполовиненными весами

Примеры работы

			
mite mite black widow cockroach tick starfish	container ship container ship lifeboat amphibian fireboat drilling platform	motor scooter go-kart moped bumper car golfcart	leopard leopard jaguar cheetah snow leopard Egyptian cat
			
grille convertible grille pickup beach wagon fire engine	mushroom agaric mushroom jelly fungus gill fungus dead-man's-fingers	cherry dalmatian grape elderberry ffordshire bullterrier currant	Madagascar cat squirrel monkey spider monkey titi indri howler monkey

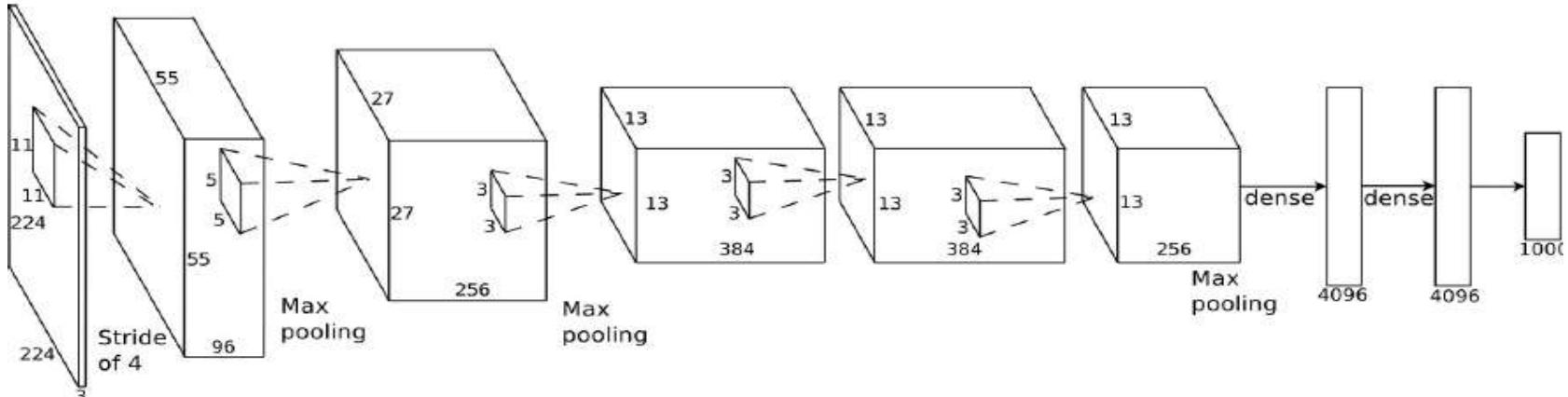
Важный вывод



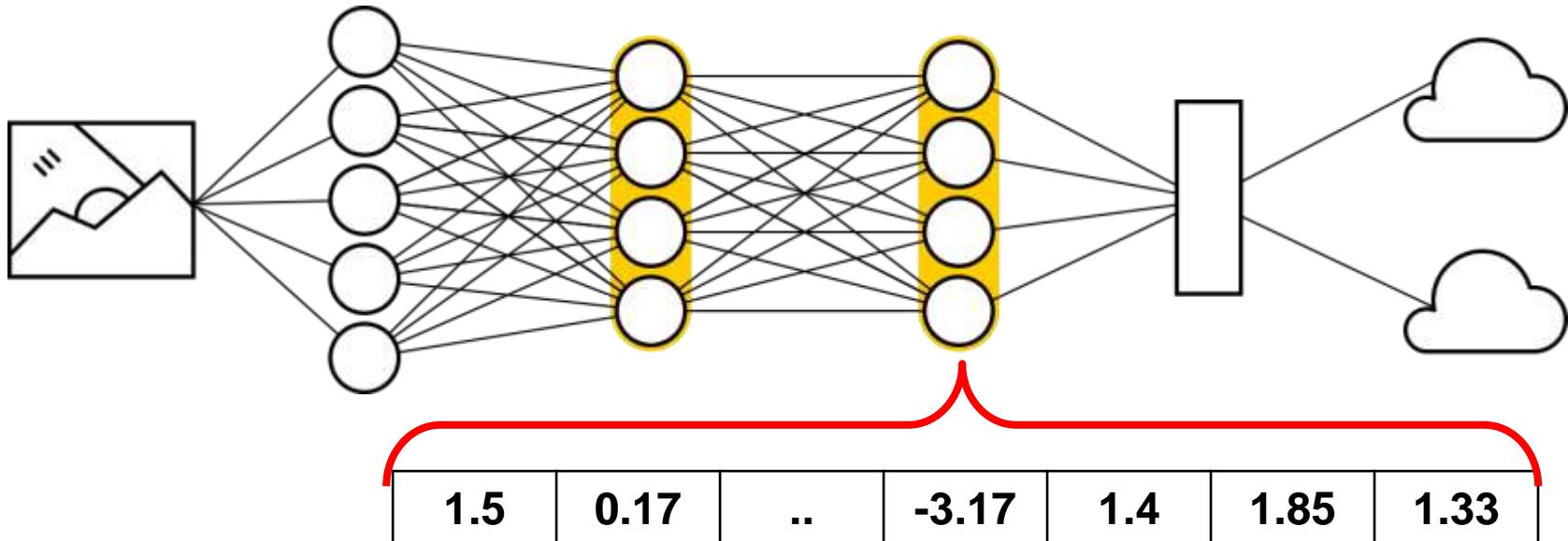
- С помощью свёрточной нейросети из 2х свёрточных слоёв можно реализовать большинство эвристических методов вычисления признаков изображения (гистограммы цветов, HOG, мешки визуальных слов)
- Последующие слои реализуют какие-то признаки «более высокого уровня»
- При обучении свёрточной сети эти признаки *обучаются* под решение поставленной задачи, а не задаются пользователем
- Визуализация работы нейросети позволяет немного понять, что именно выучивается на глубоких свёрточных слоях

Что происходит внутри сети?

Мы умеем обучать нейросеть для классификации изображений

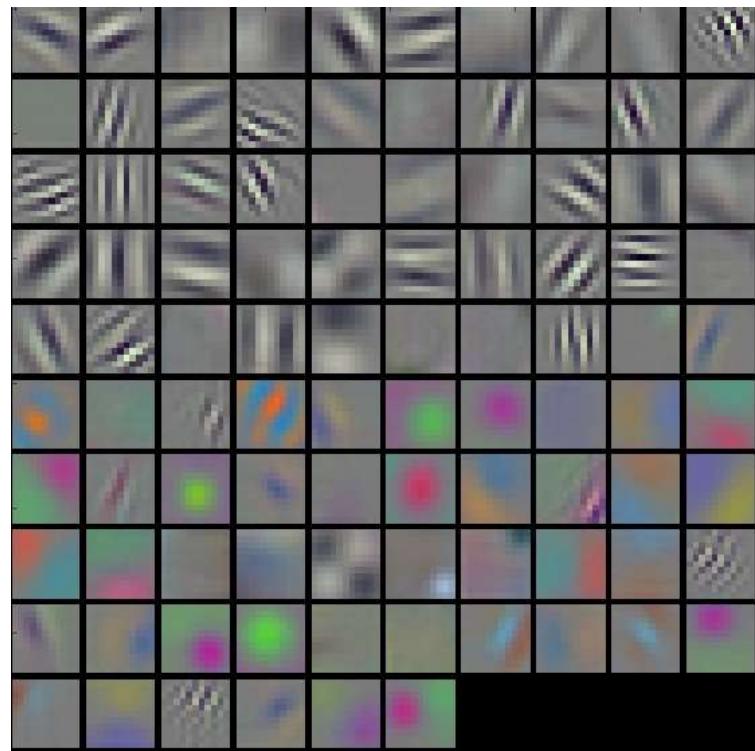


Что происходит внутри неё? Что будем визуализировать?

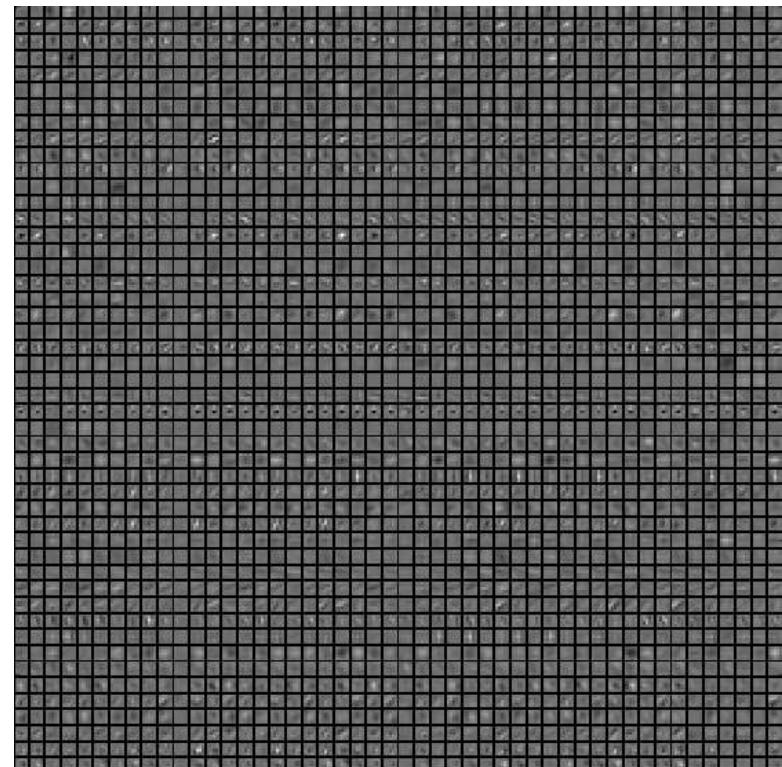


Визуализация работы нейросети

Визуализация фильтров

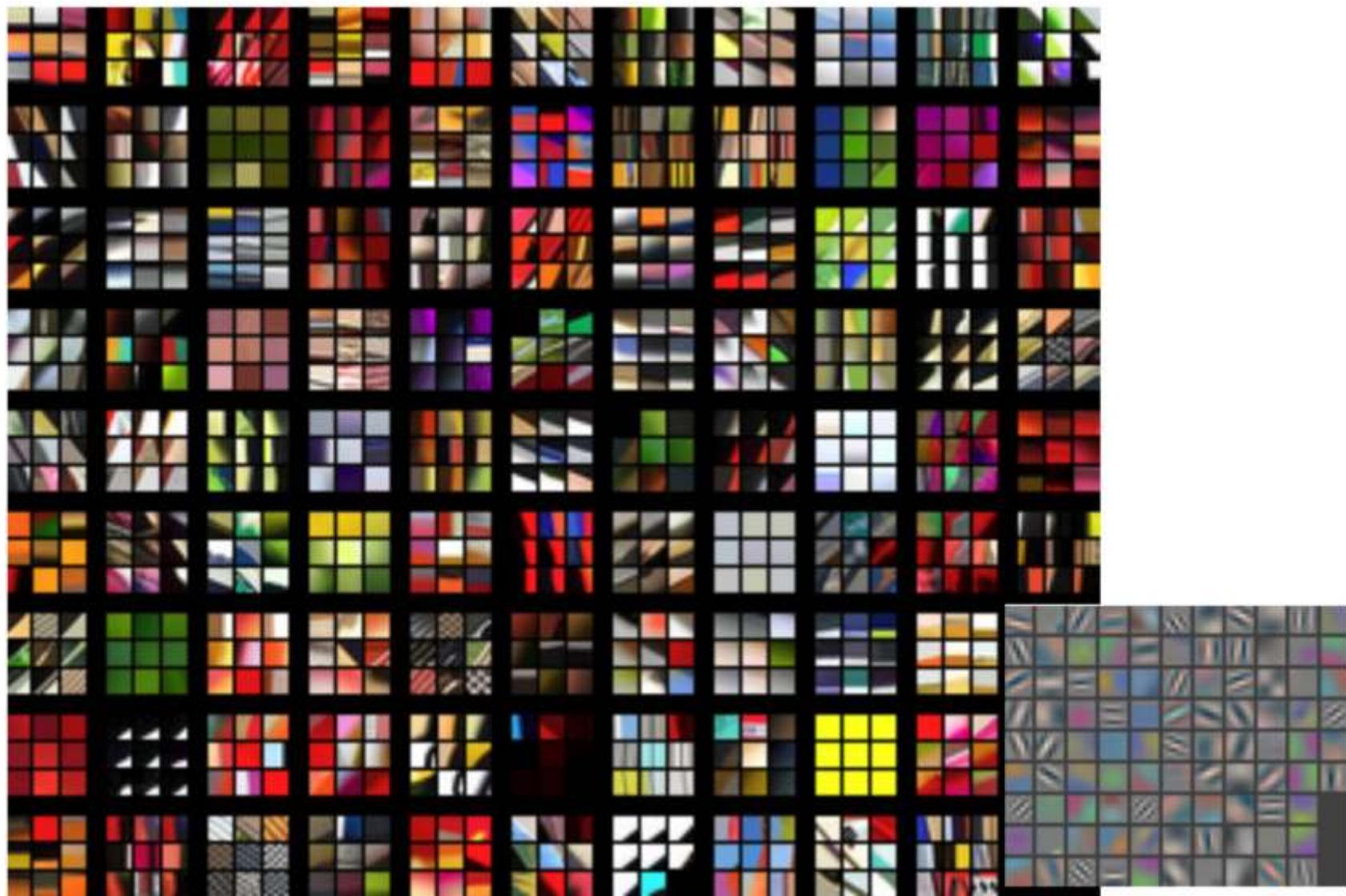


Слой conv1

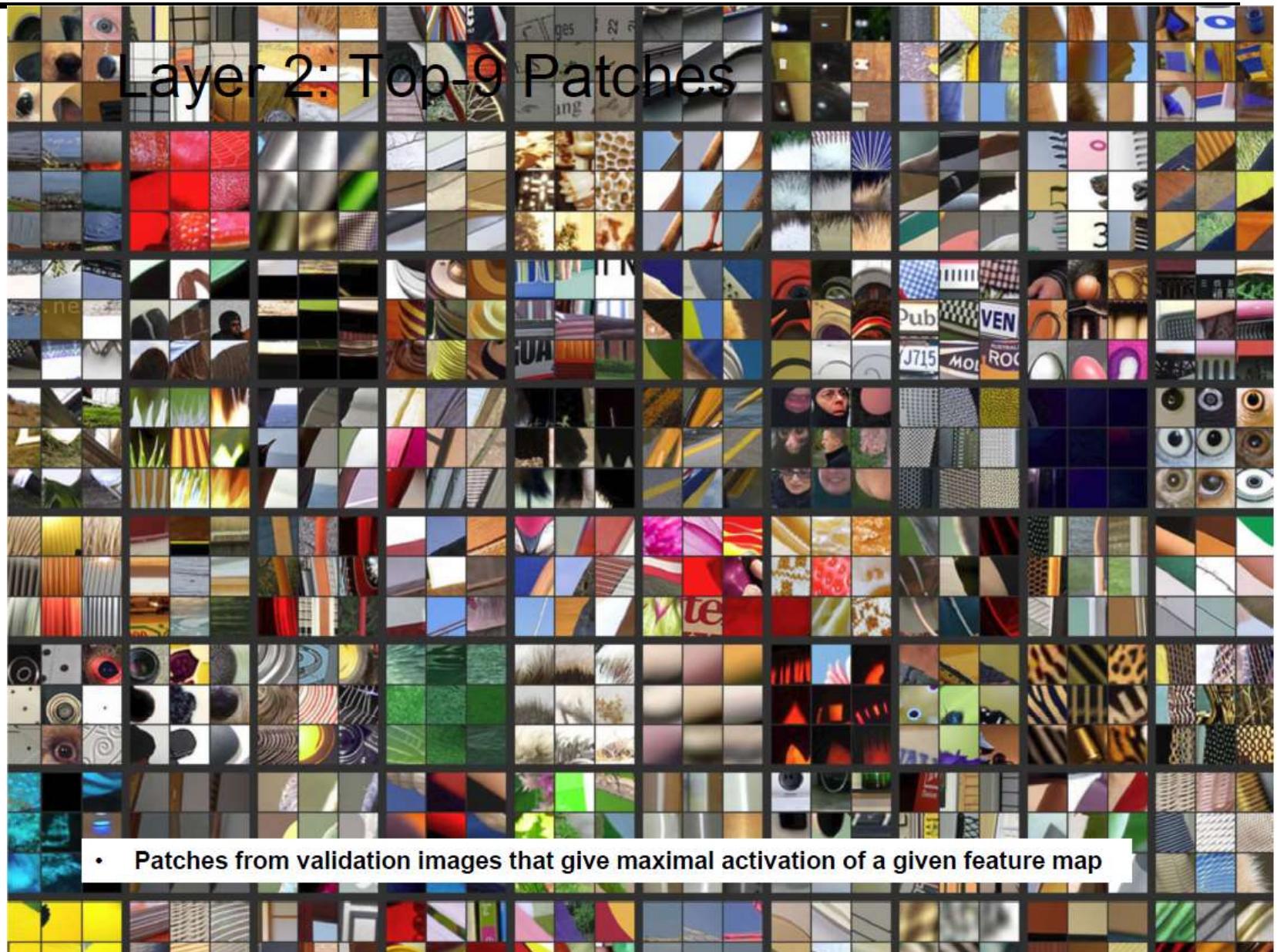


Слой conv2

Слой 1: Топ-9 фрагментов



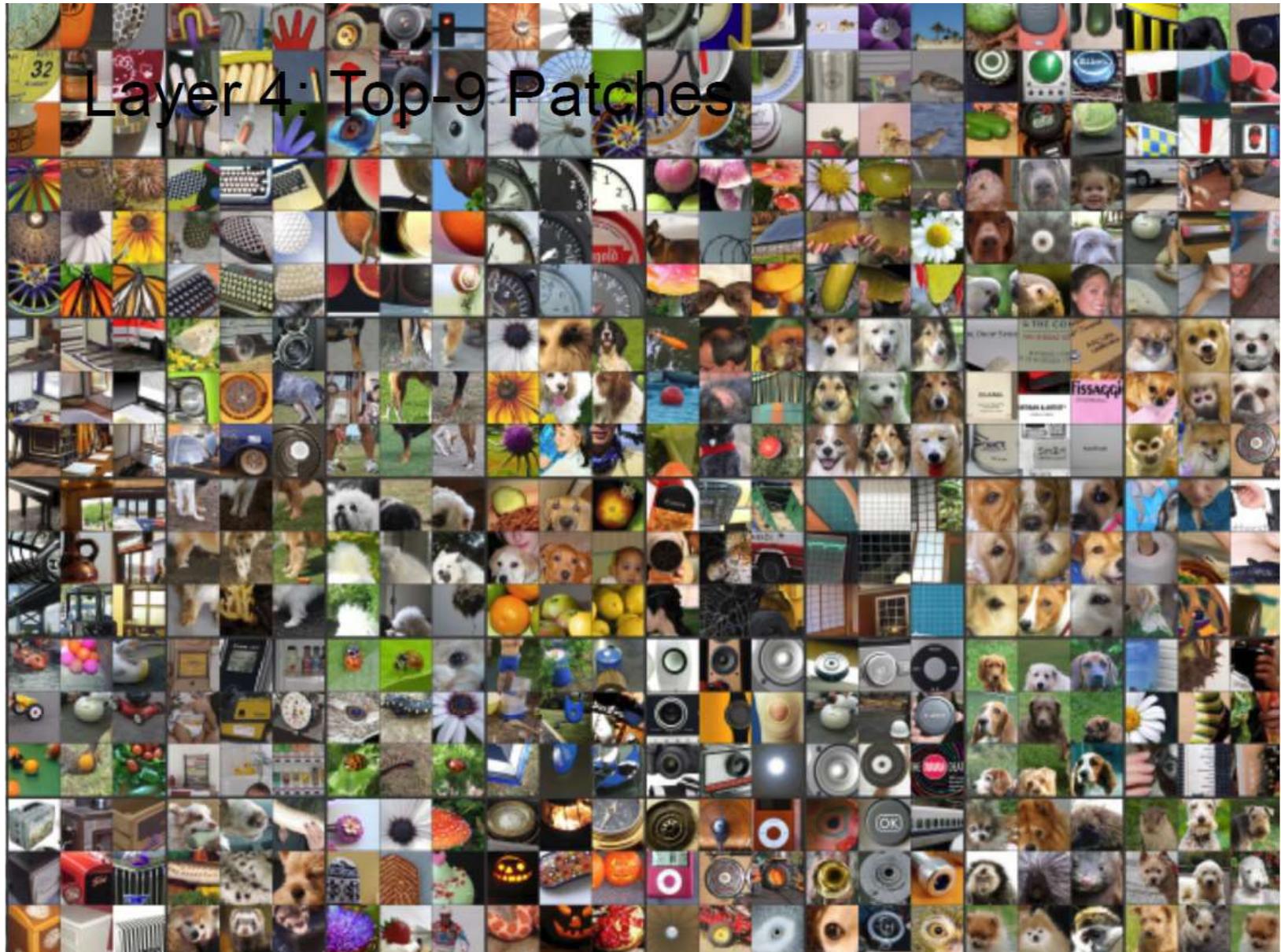
Слой 2: Топ-9 фрагментов



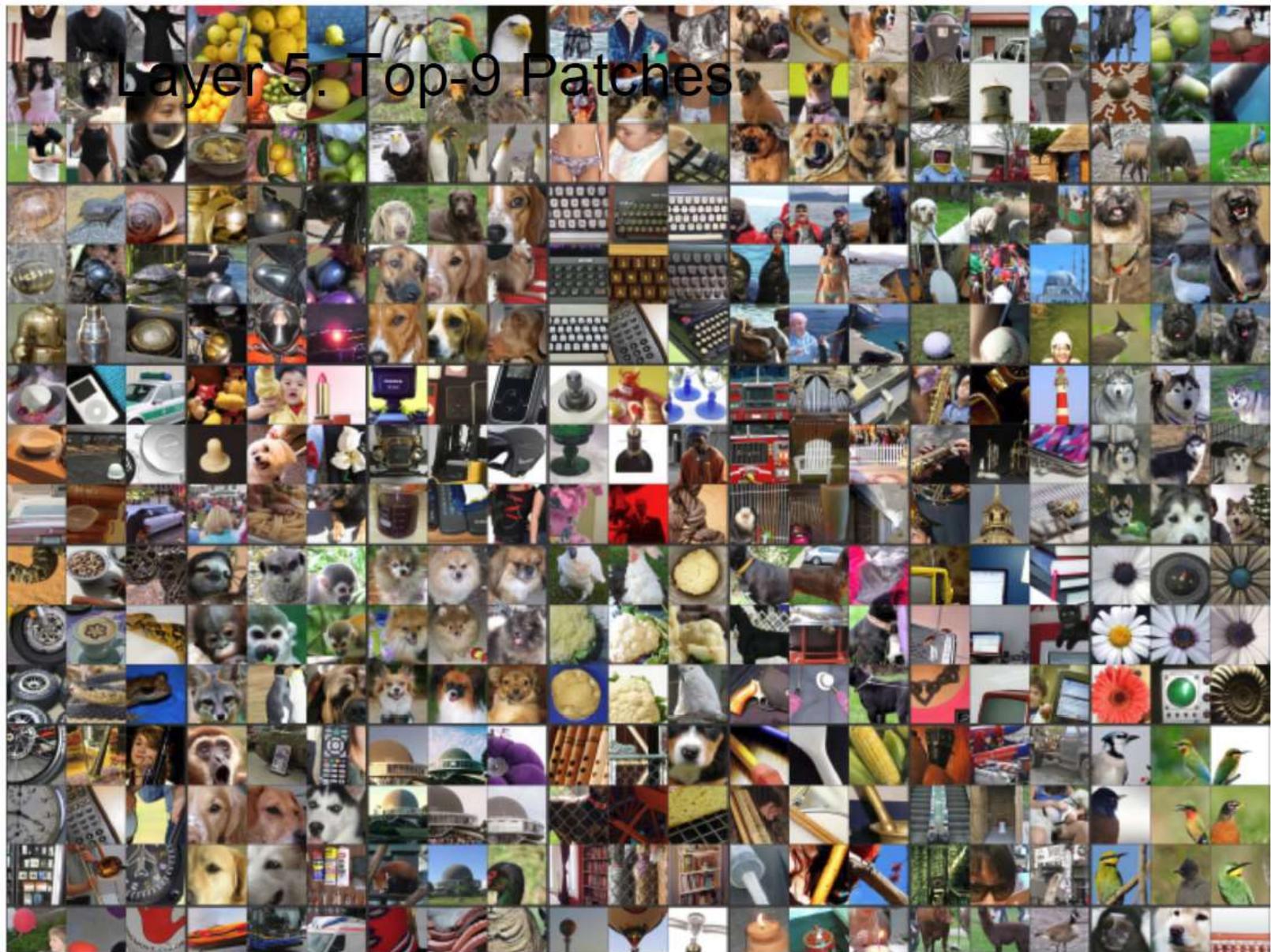
Слой 3: Топ-9 фрагментов



Слой 4: Топ-9 фрагментов



Слой 5: Топ-9 фрагментов



Layer 5: Топ-9 Patches

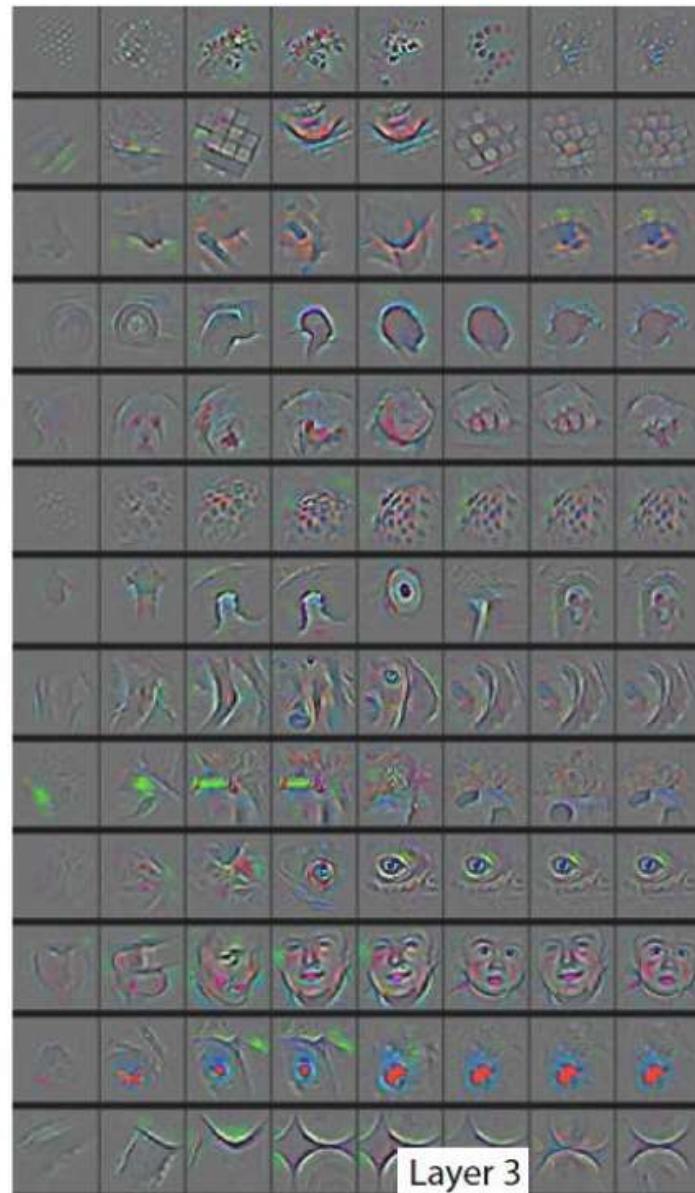
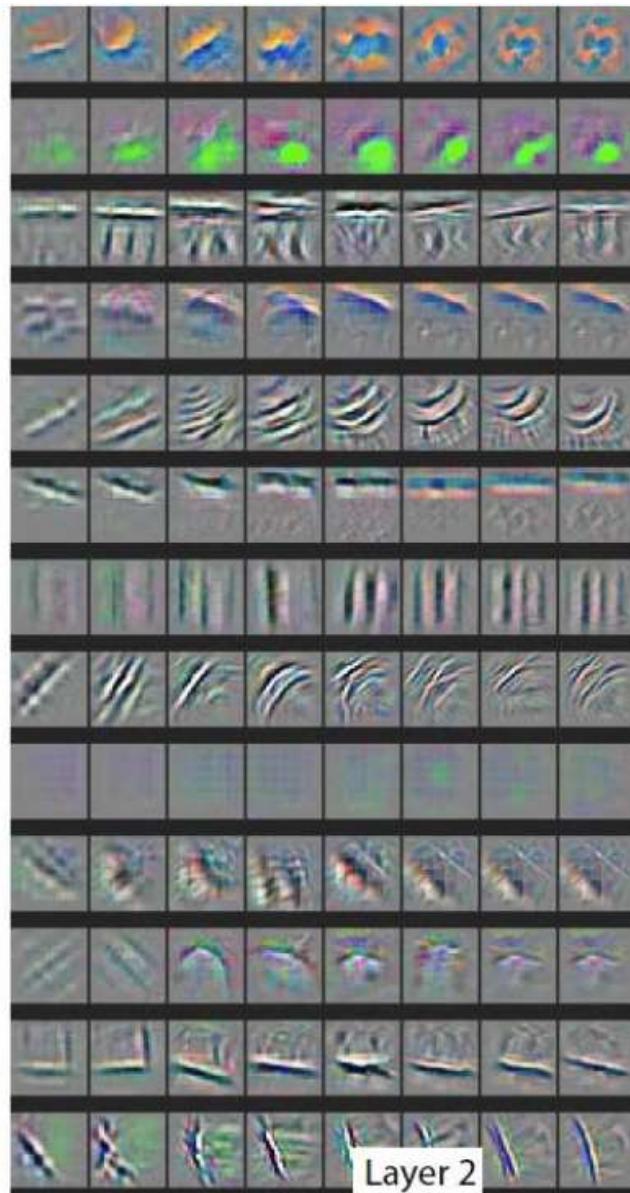
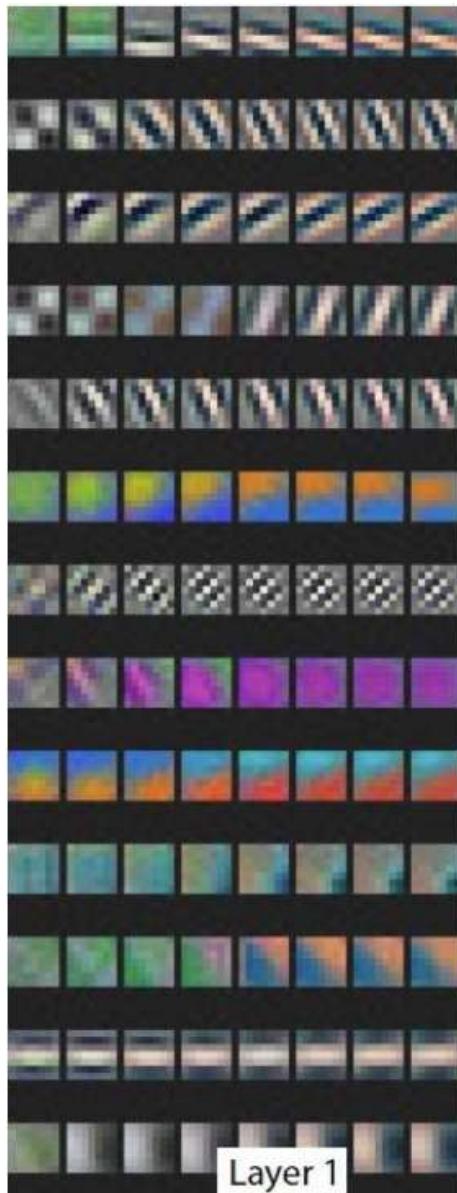
Визуализация работы нейросети

Изображения, на которых достигается максимальный отклик фильтра

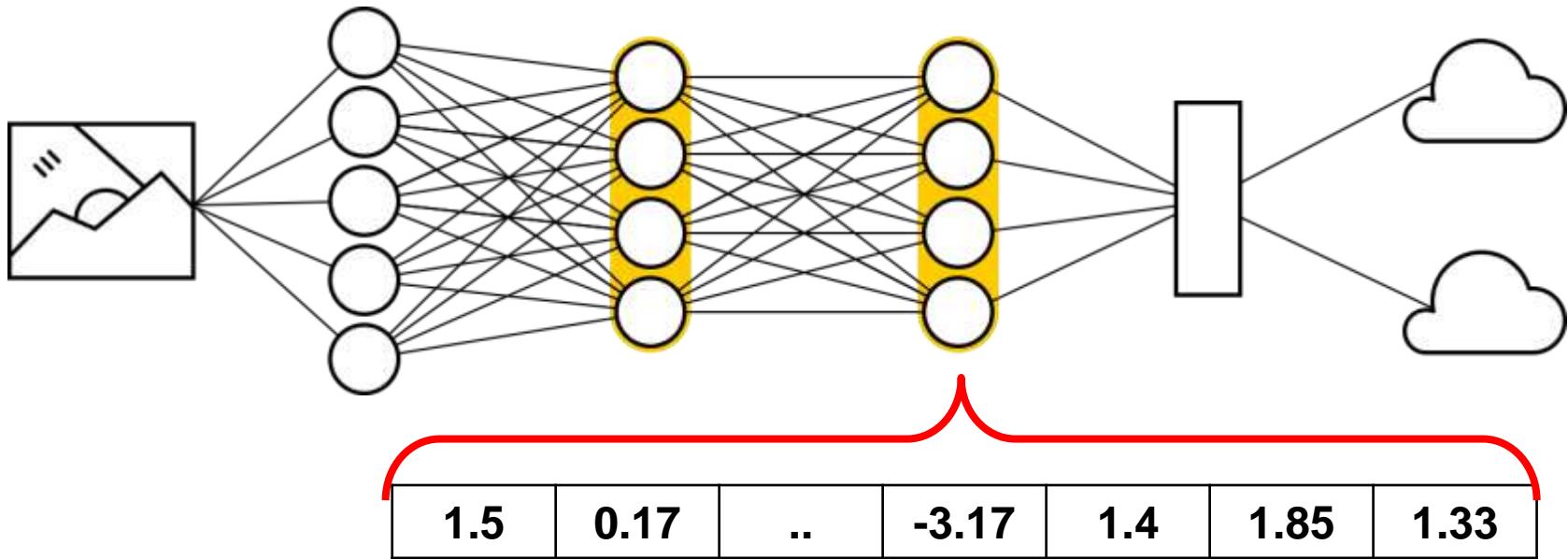


Фильтры слоя pool5

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
- Попробуем проанализировать его

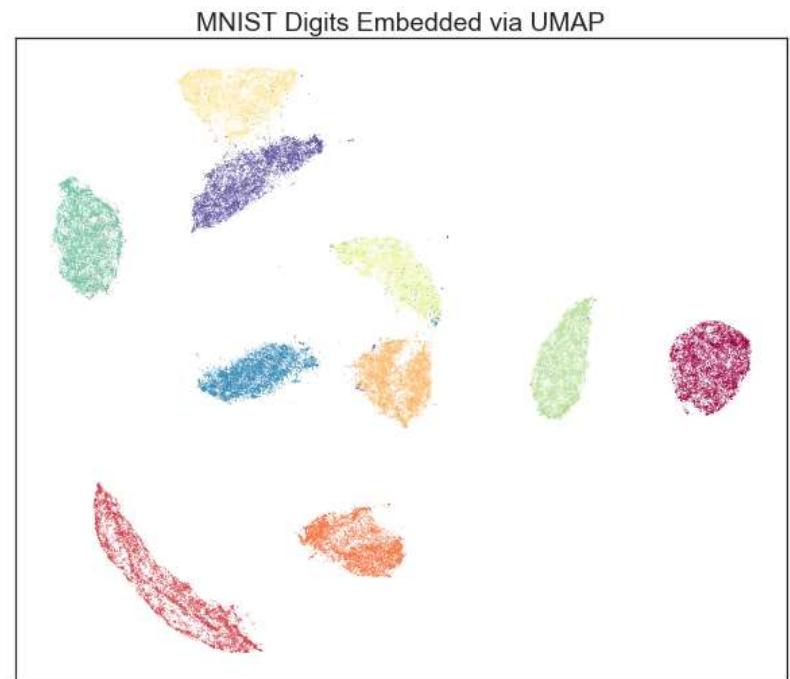
t-SNE

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

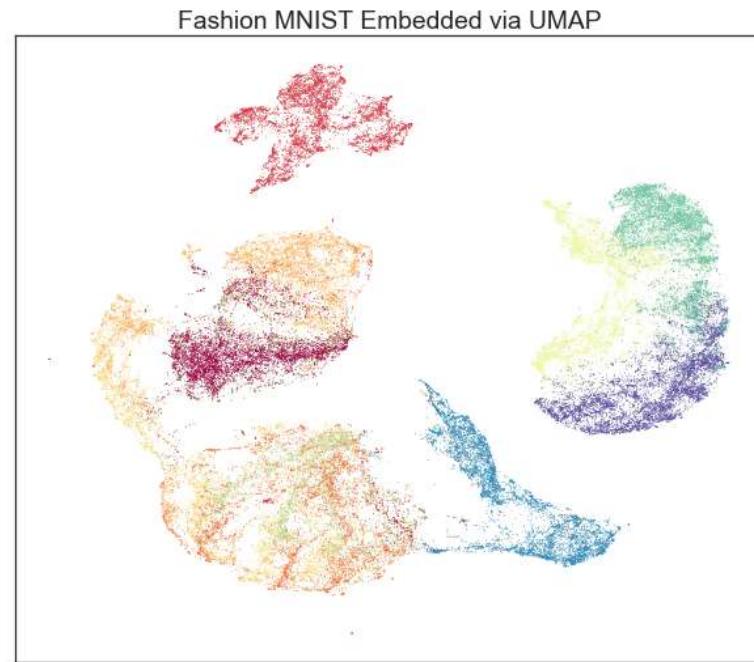




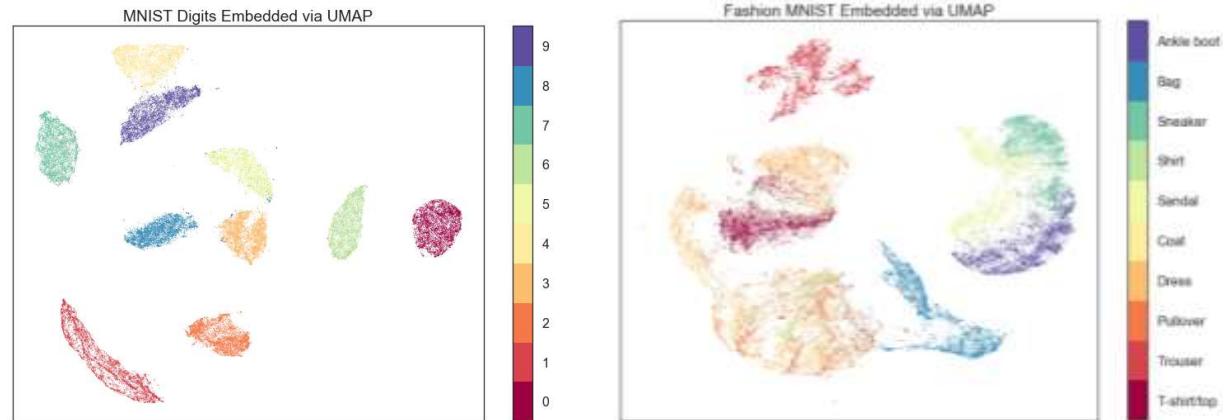
УМАР визуализация



УМАР визуализация

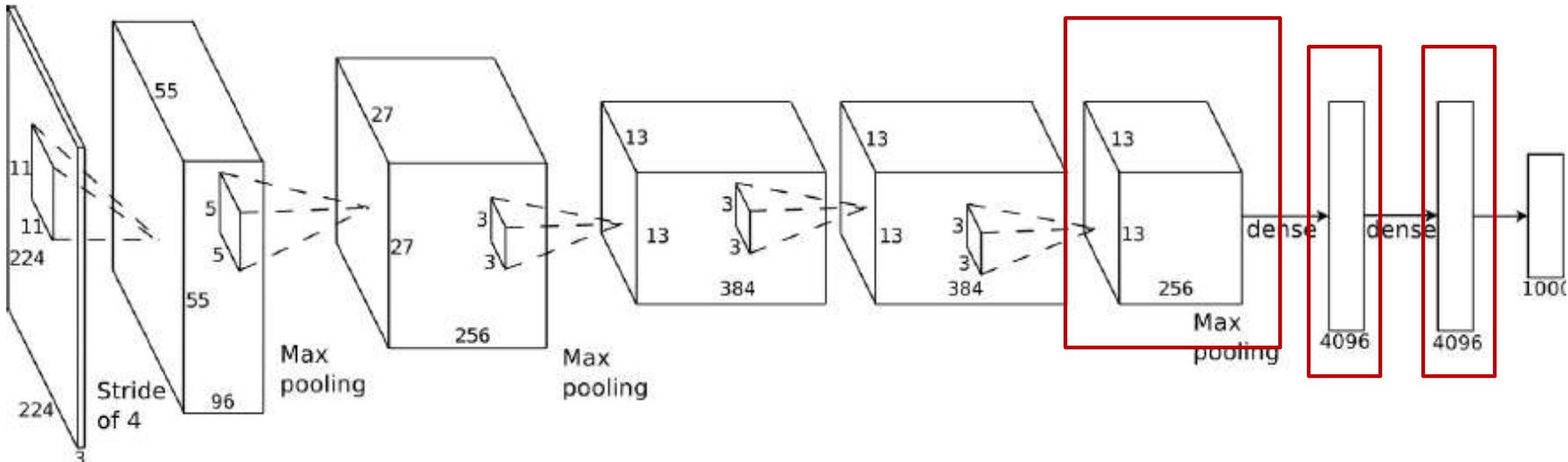


Выводы



- 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



- 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

Donahue et. al. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition, 2013

Распознавание на других базах

	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	

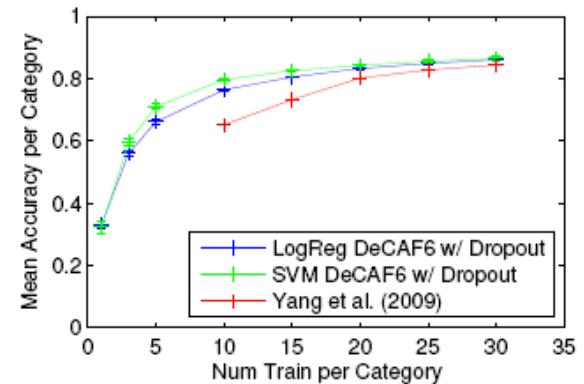


Figure 4. Left: average accuracy per class on Caltech-101 with 30 training samples per class across three hidden layers of the network and two classifiers. Our result from the training protocol/classifier combination with the best validation accuracy – SVM with Layer 6 (+ dropout) features – is shown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

Обучили нейросеть для извлечения признаков на ImageNet и применили для Caltech 101

Donahue et. al. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition, 2013

Классификация близких объектов



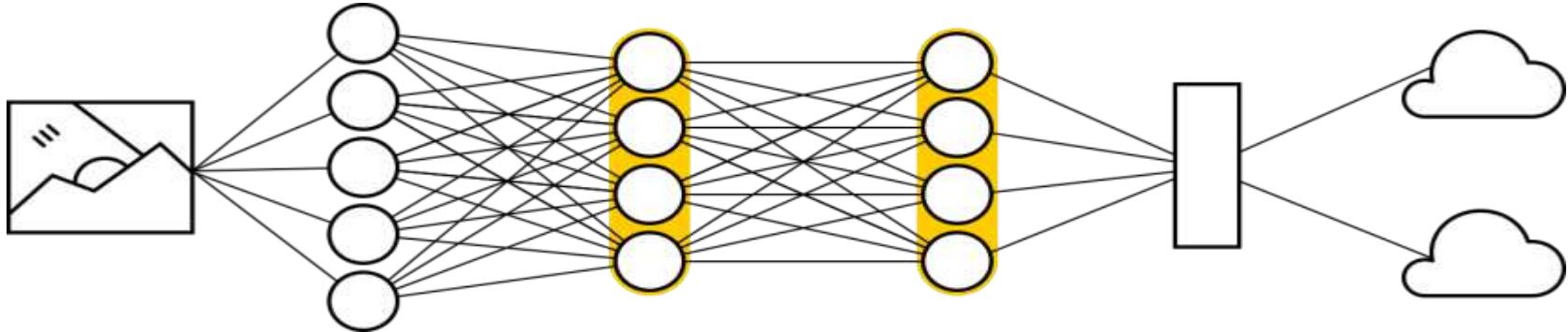
Method	Part info	mean Accuracy
Sift+Color+SVM[45]	✗	17.3
Pose pooling kernel[49]	✓	28.2
RF[47]	✓	19.2
DPD[50]	✓	51.0
Poof[5]	✓	56.8
CNN-SVM	✗	53.3
CNNaug-SVM	✗	61.8
DPD+CNN(DeCaf)+LogReg[10]	✓	65.0

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.

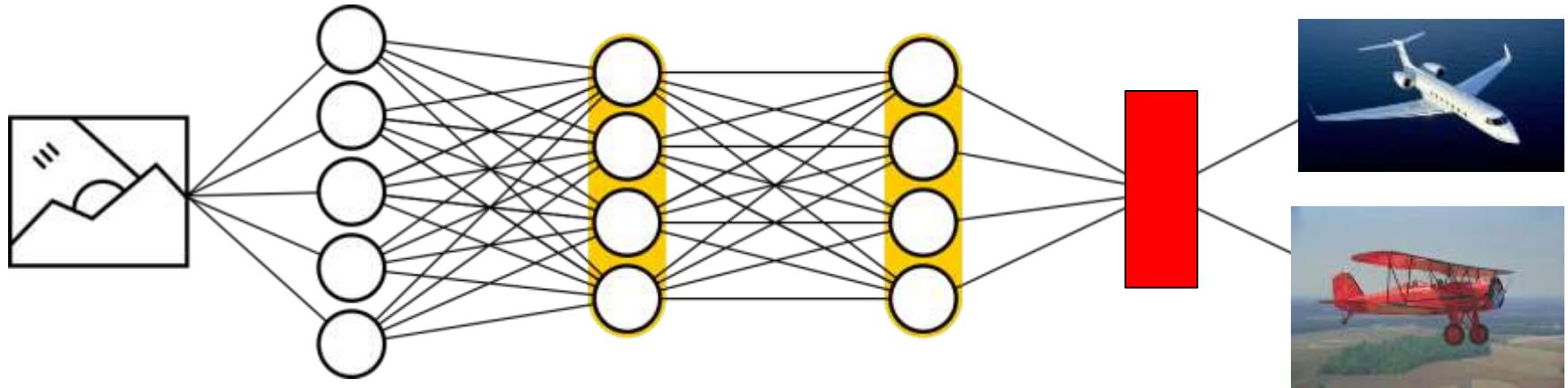
- Fine-graded classification – например, определение видов птиц
- Возьмём нейросеть, обученную для классификации ImageNet
- Применим её для получения вектор-признаков изображений
Обучаем классификатор поверх этих признаков
- Profit!

Дообучение (fine-tuning)

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

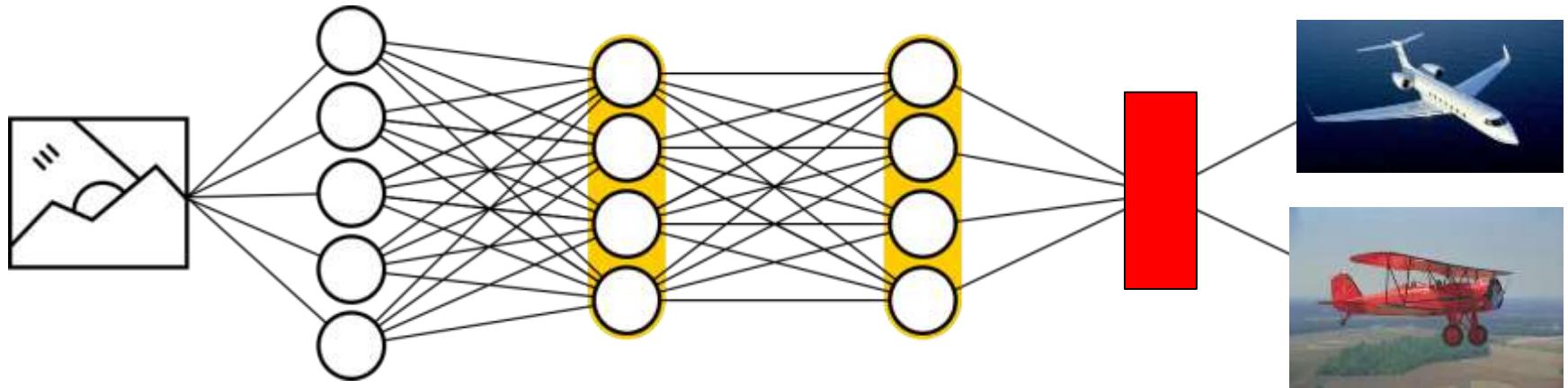


Заменим в ней последний слой (классификатор)



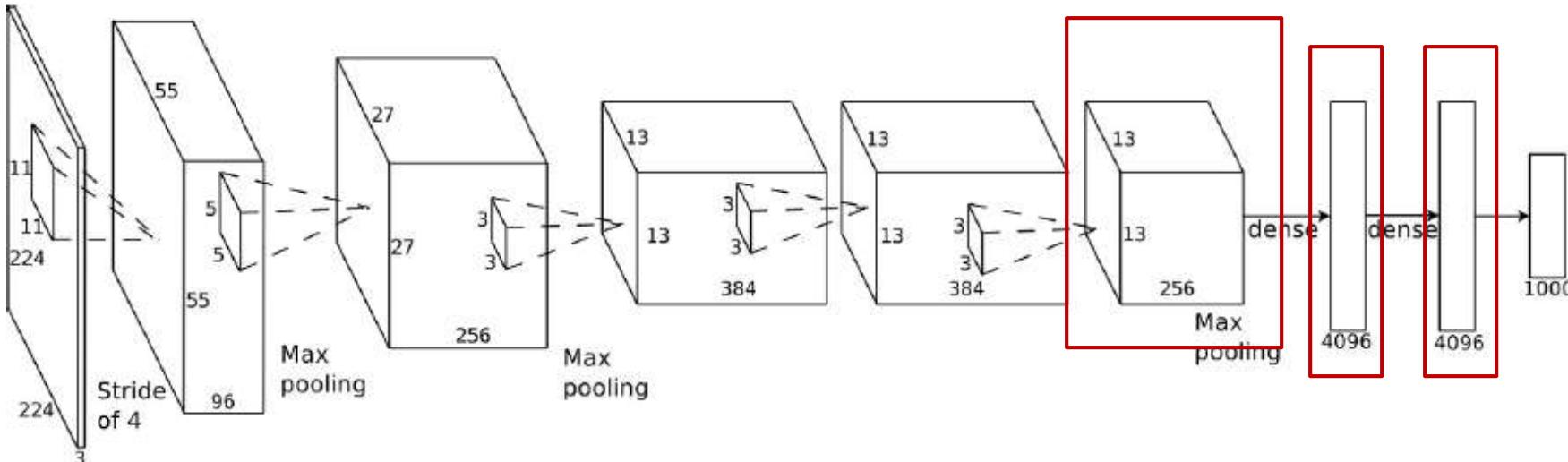
Обучим («дообучим») эту сеть на новой коллекции для решения новой задачи

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

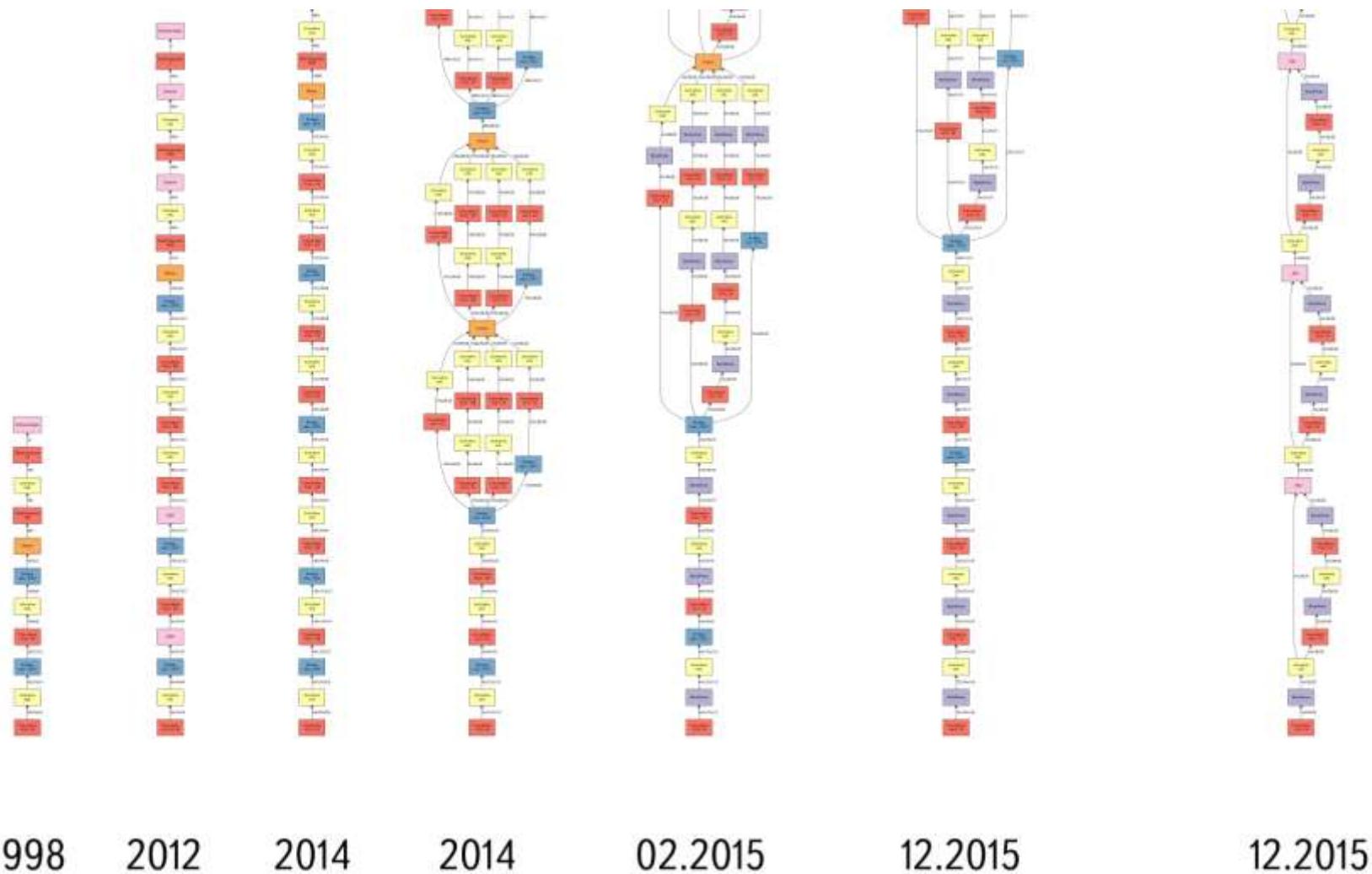
Идея «базовой» архитектуры



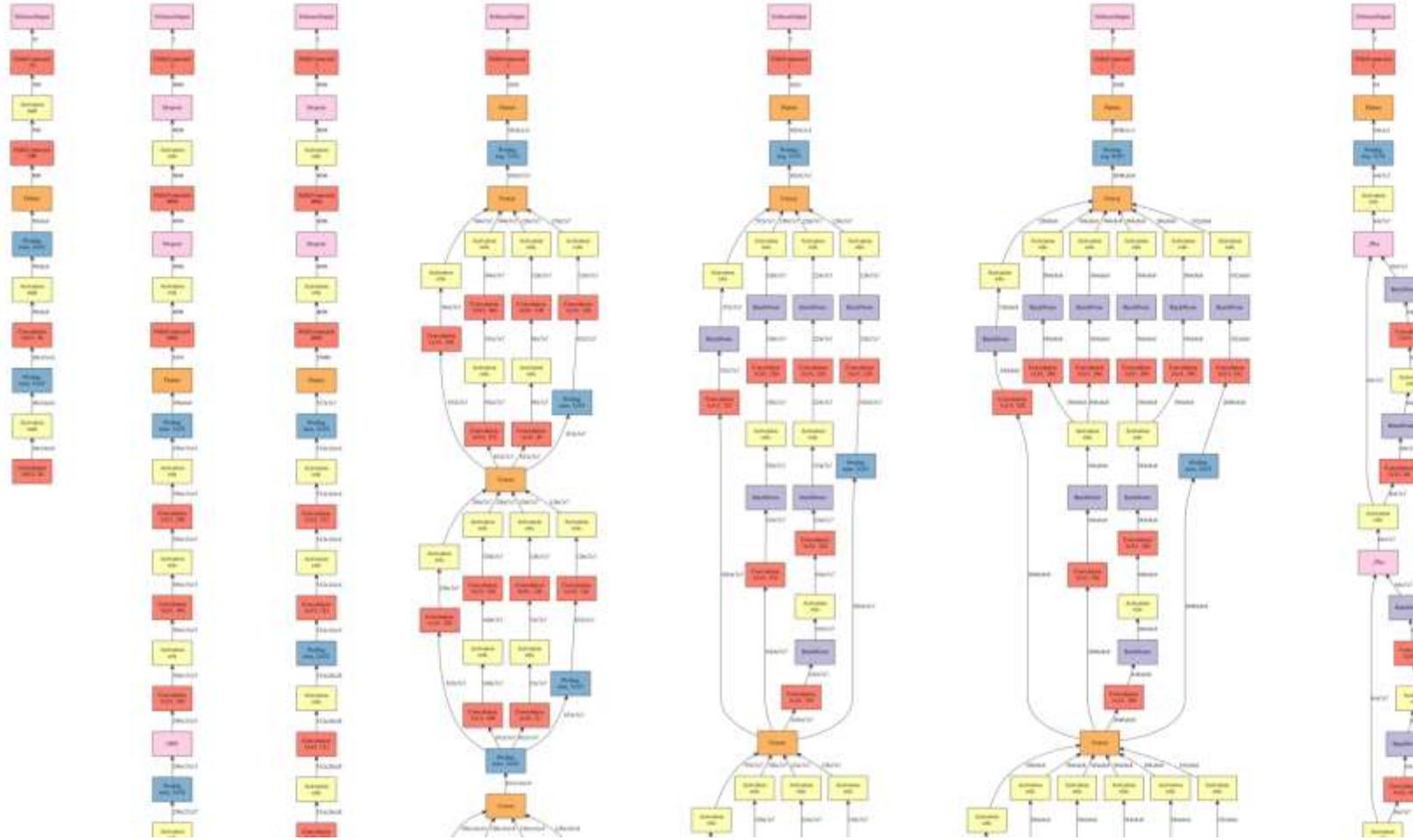
- Можем обучать нейросети-классификаторы на больших коллекциях («базовые сети»), использовать их как основу для решения новых задач (дообучением)
- Создали коллекции моделей (zoo – зоопарки)
- Пока в основном обучают на ImageNet



Развитие базовых архитектур

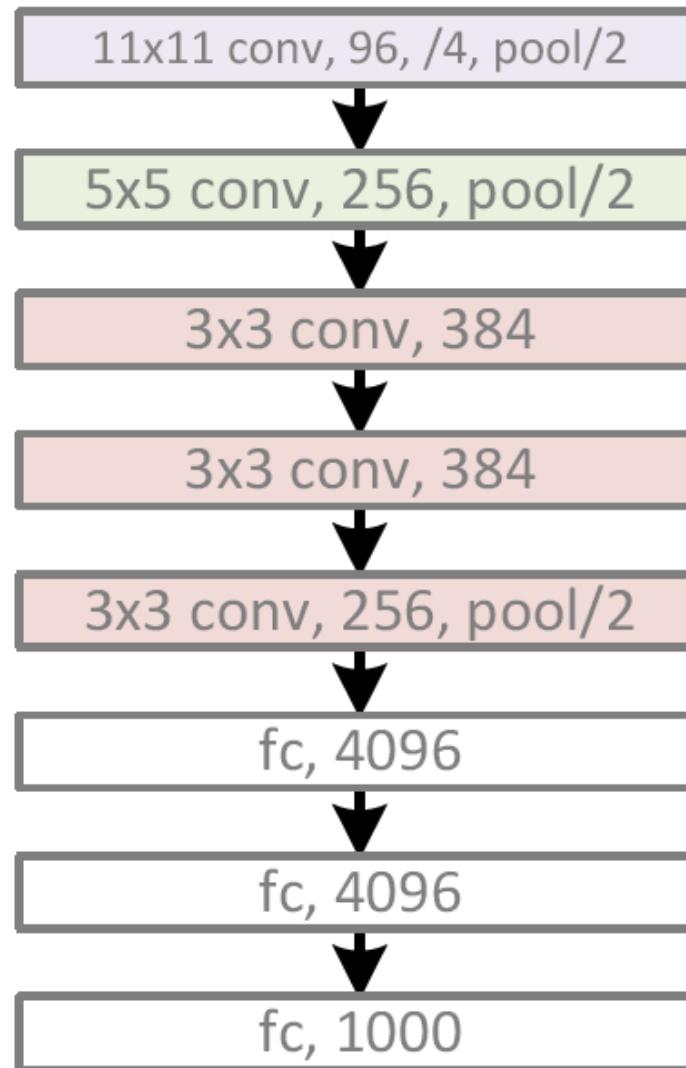


Развитие базовых архитектур



AlexNet

AlexNet, 8 layers
(ILSVRC 2012)



Применение AlexNet

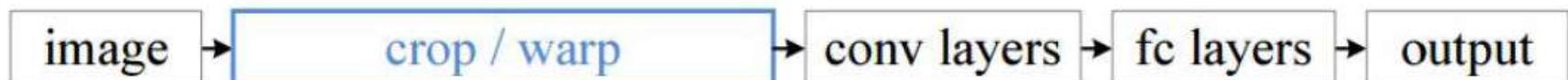
- На вход нейросеть принимает изображение фиксированного размера 224x224 пикселя. Можем ли мы подать другое?
- Как быть с изображениями других размеров?
- 2 варианта действий:



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

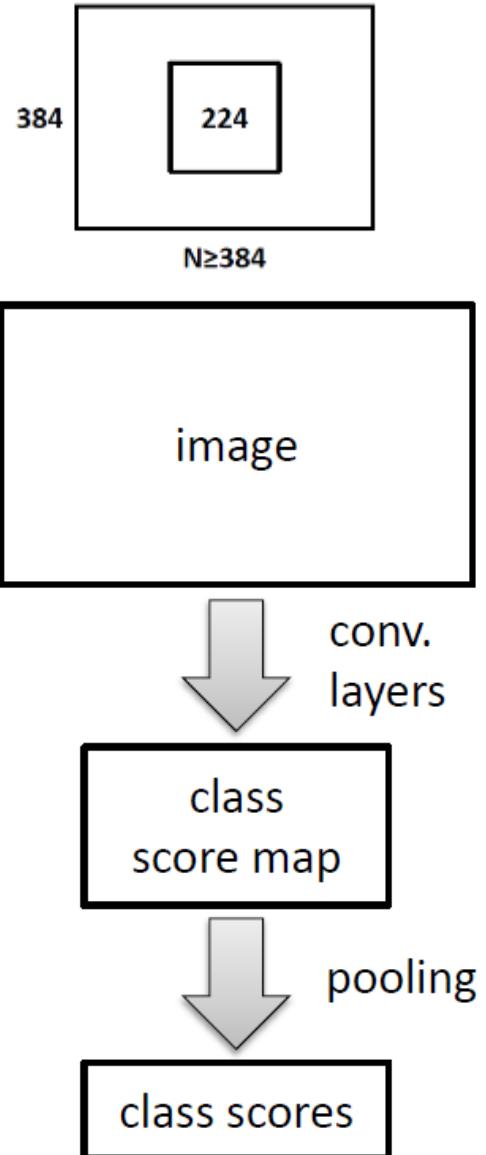
Приводить к фиксированному
разрешению (image warp)

- 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)

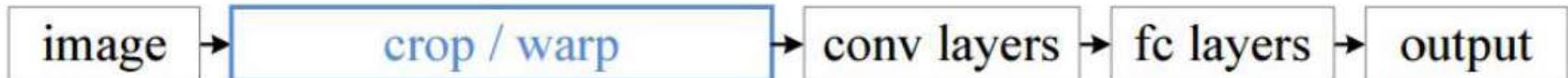
Идея: преобразовать выходы свёрточного слоя (признаки) произвольного разрешения к вектор-признаку фиксированной длины



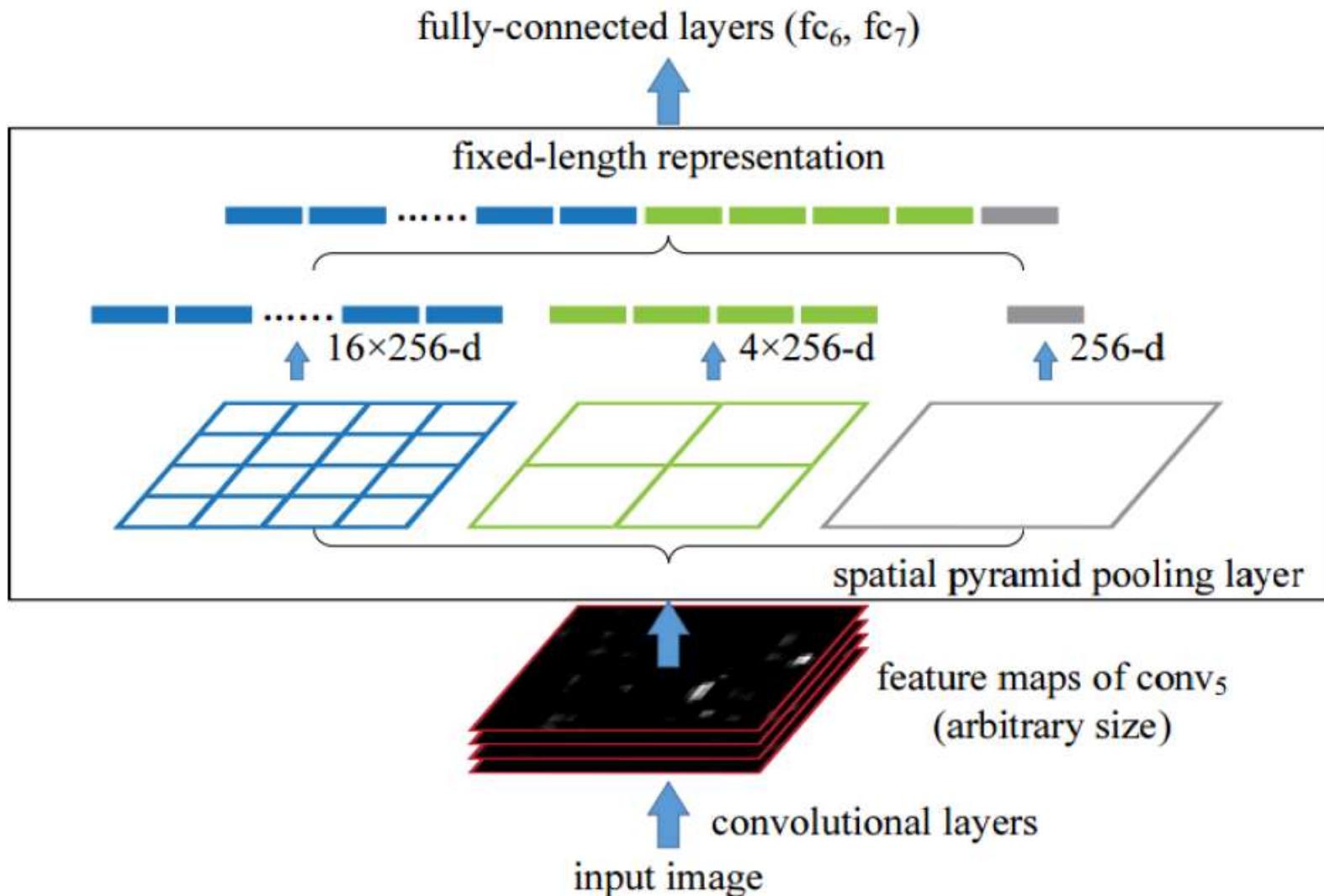
crop



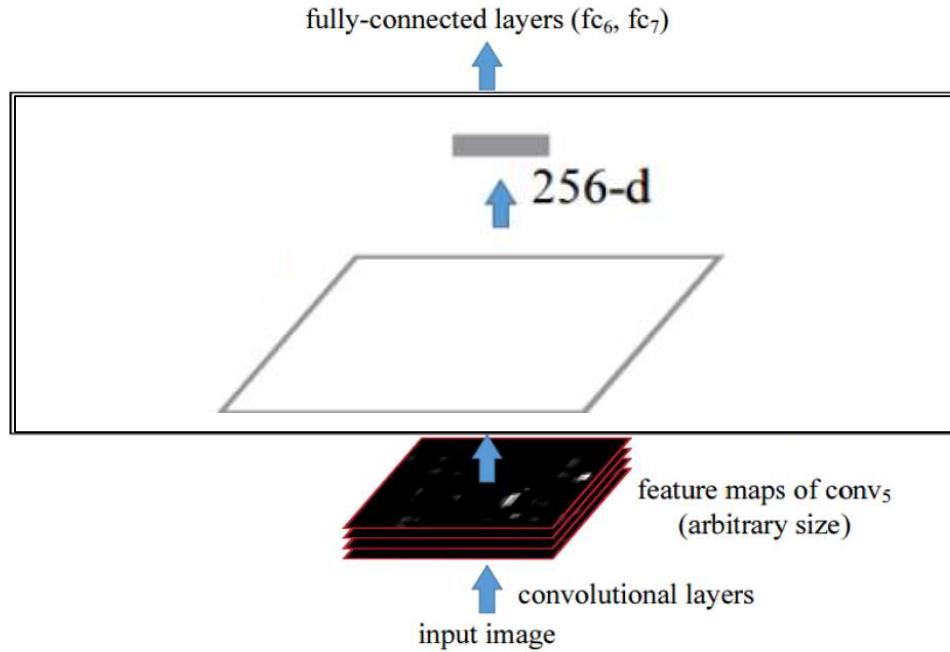
warp



Spatial Pyramid Pooling (SPP)



Average Pooling



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

Очень глубокие (VGG)

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

image

conv-64

conv-64

maxpool

conv-128

conv-128

maxpool

conv-256

conv-256

maxpool

conv-512

conv-512

maxpool

conv-512

conv-512

maxpool

FC-4096

FC-4096

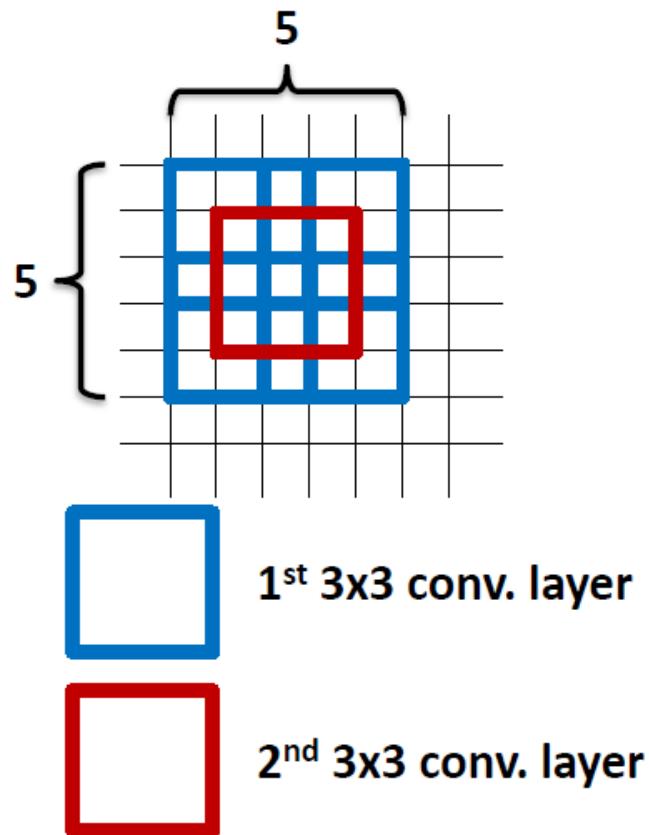
FC-1000

softmax

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

Свёртки 3x3

- Стэк свёрток позволяет обеспечить большее receptive field (reception field)
 - 5×5 для 2-x свёрток
 - 7×7 для 3-x свёрток
- Большая нелинейность за счёт ReLU активаций
- Меньше параметров
 - $18 \times (2 \text{ } 3 \times 3) \text{ vs } 25 \times (5 \times 5)$
 - $27 \times (3 \text{ } 3 \times 3) \text{ vs } 49 \times (7 \times 7)$



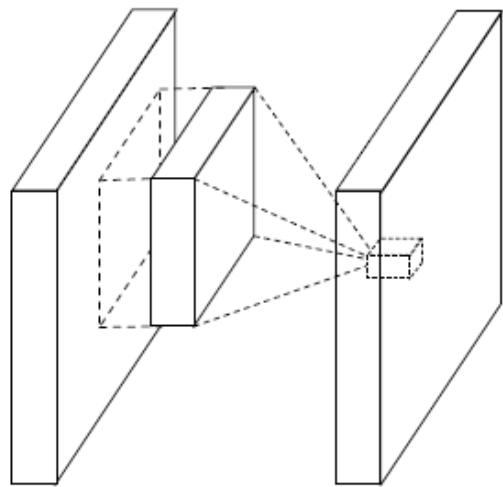
Исследование вариантов

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
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					

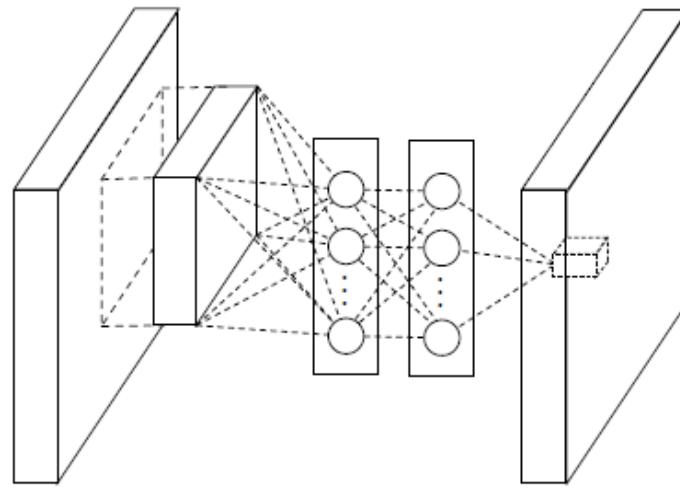
D - VGG-16

E - VGG-19

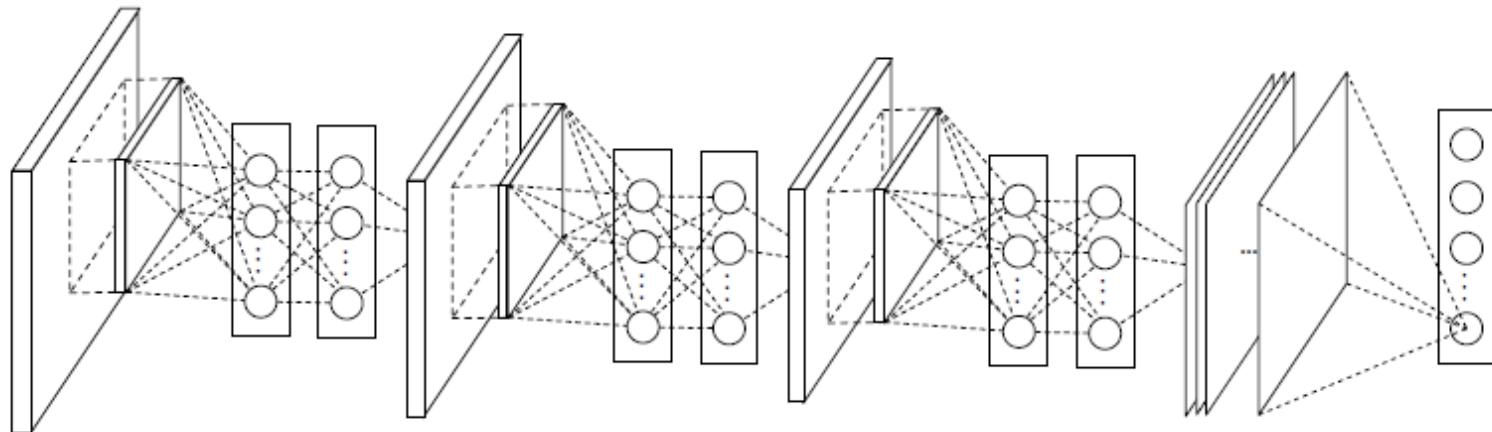
Network in Network



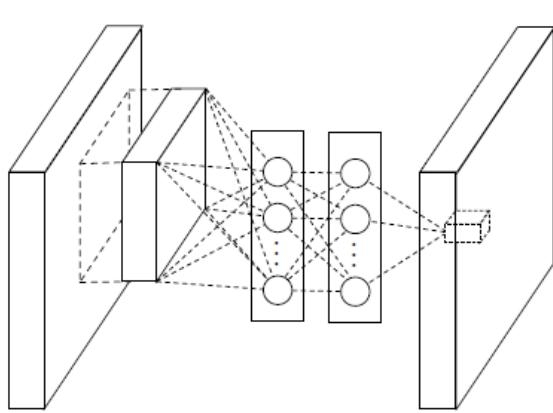
(a) Linear convolution layer



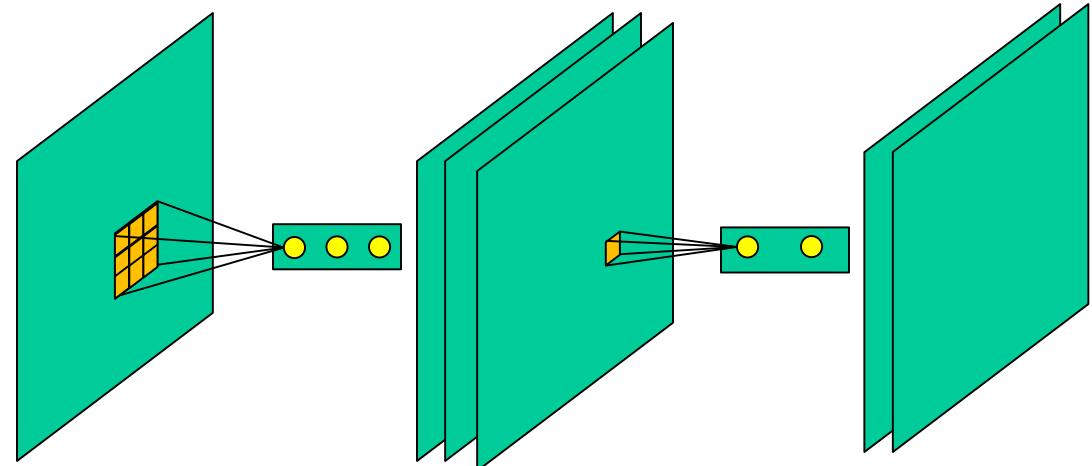
(b) Mlpconv layer



Свёртка 1x1

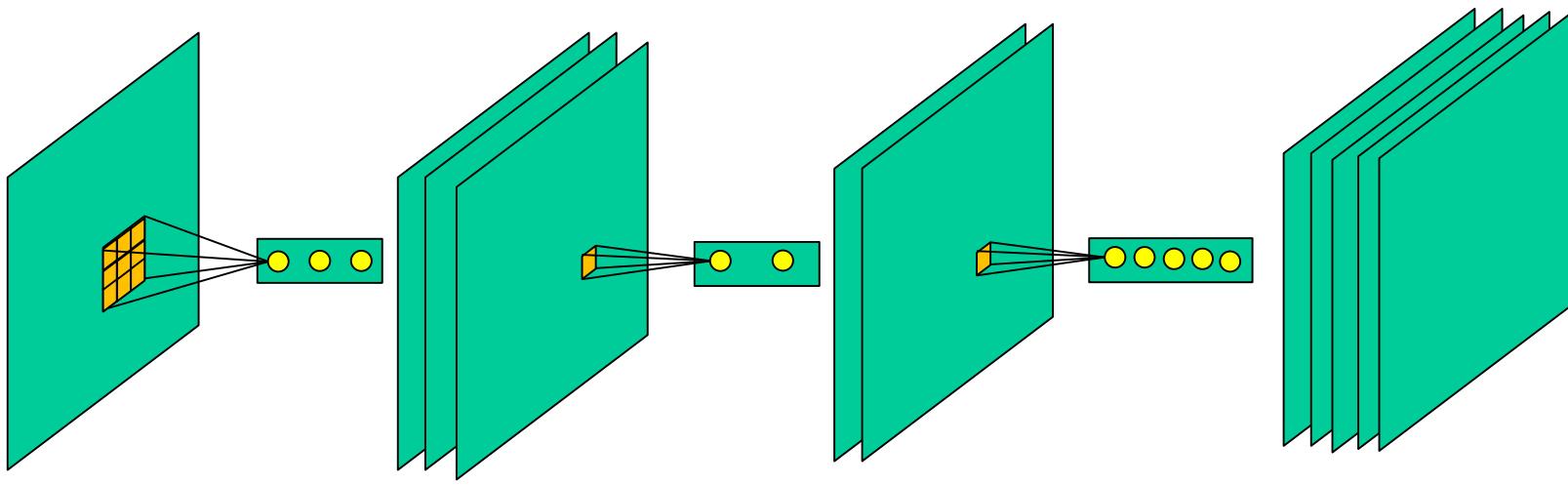


(b) Mlpconv layer



Мы можем реализовать второй и далее слои «вложенного» перспептрона как 1x1 свёртку с предыдущим слоем

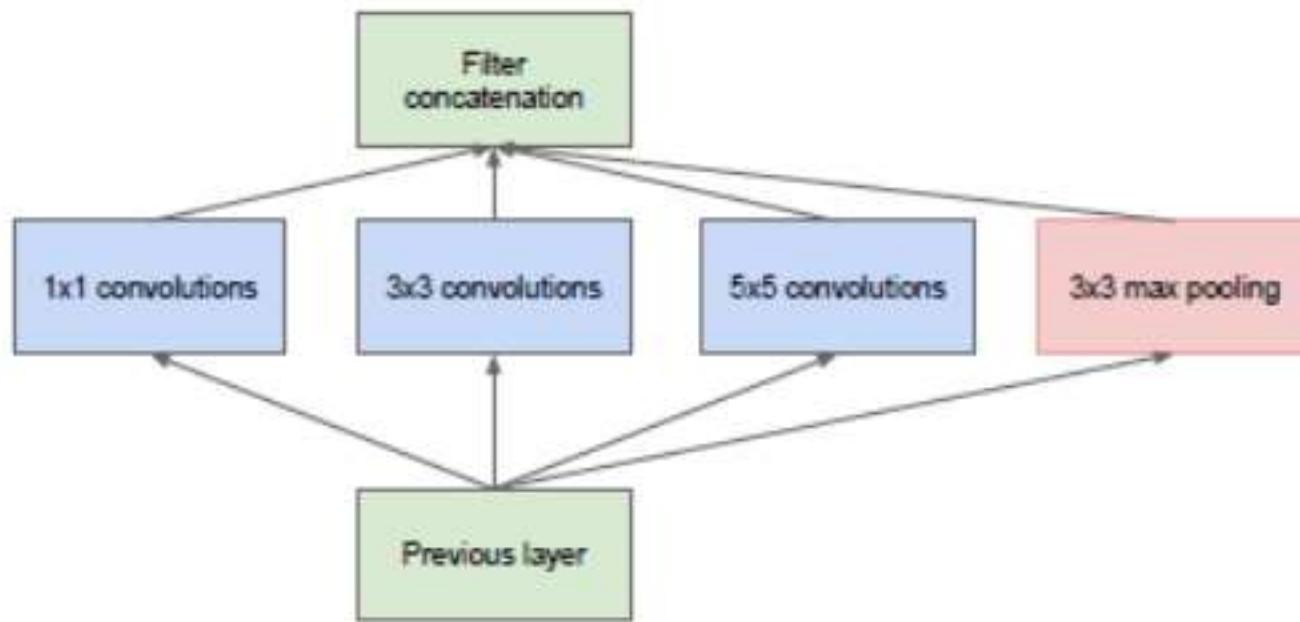
Свёртка 1x1



- 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

Архитектура Inception

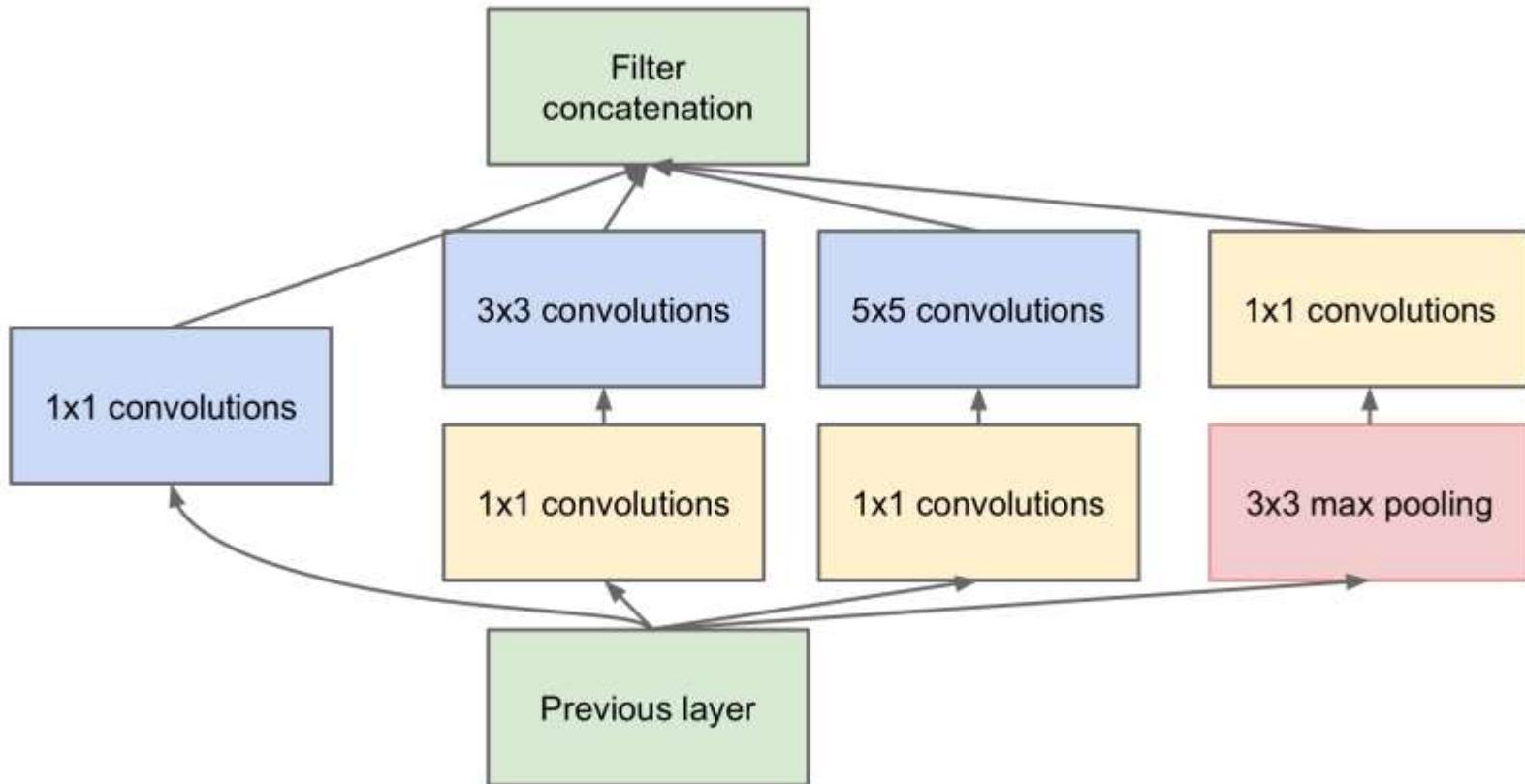
Построим нейросеть из модулей более сложной структуры, чем просто набор свёрточных слоёв VGG



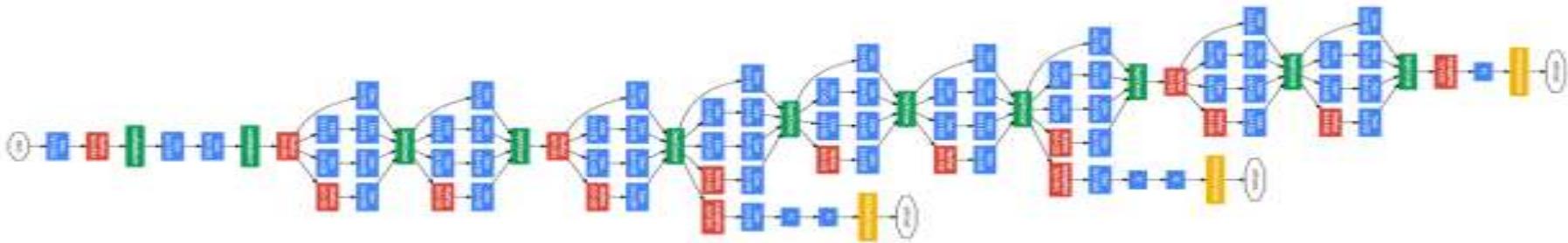
В чём смысл применять одновременно свёртки разных размеров?

Модуль Inception

Добавим 1×1 свёртки для повышения производительности

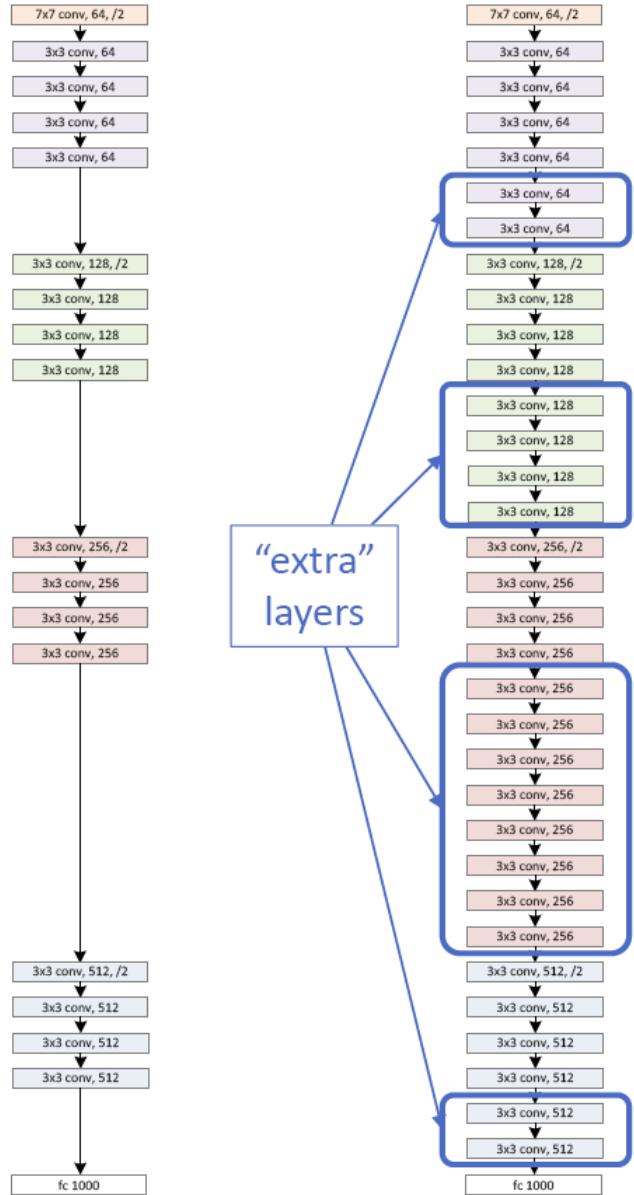


Архитектура Inception

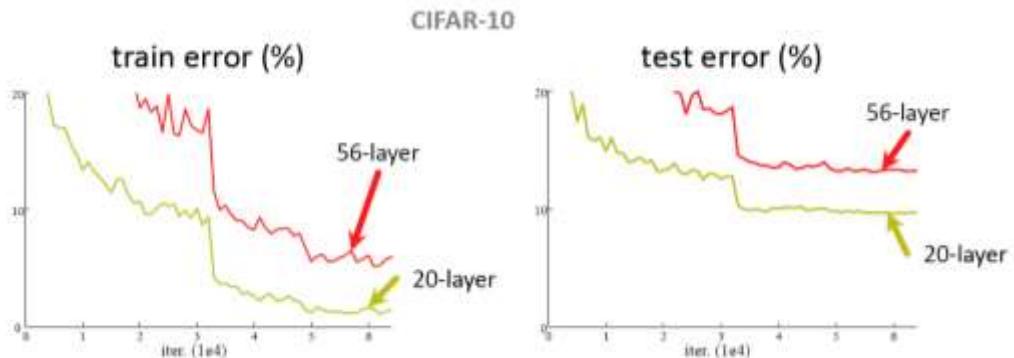


- Глубокая сеть
- Inception-модули
- Несколько уровней supervision

Проблема глубины



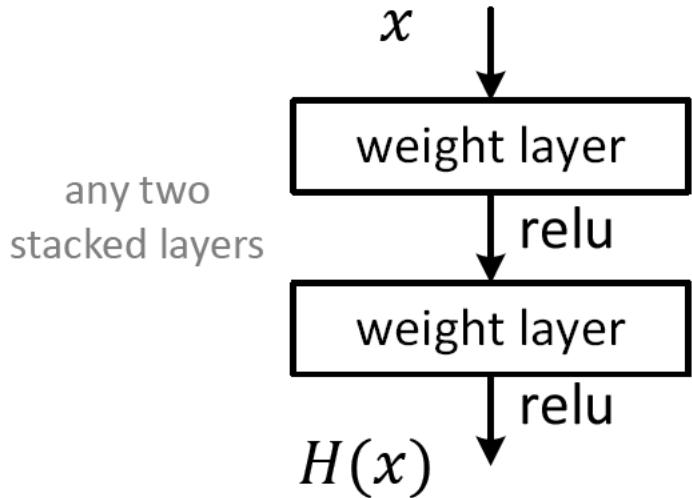
- Дальнейшее увеличение числа слоёв приводило к падению качества итоговой модели



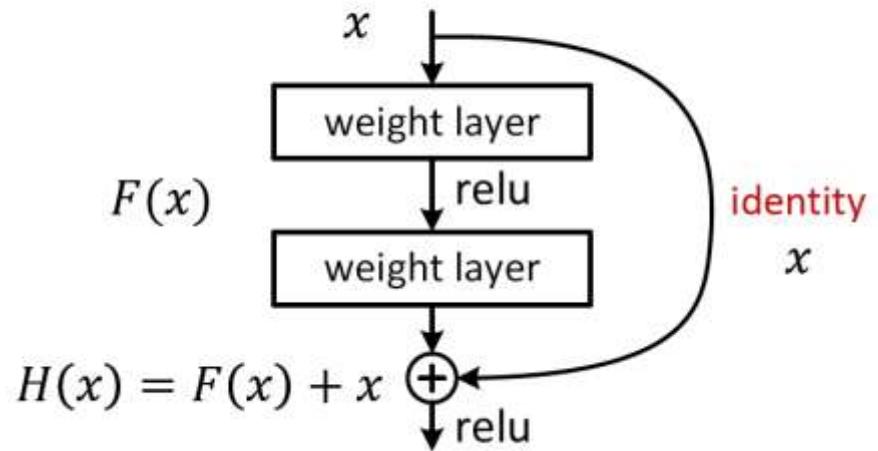
- При этом можно добавить «единичные» слои к сети, оставив ту же самую функцию и качество
- Значит, проблема в обучении сети

Residual net (или skip-connections)

- Plain net

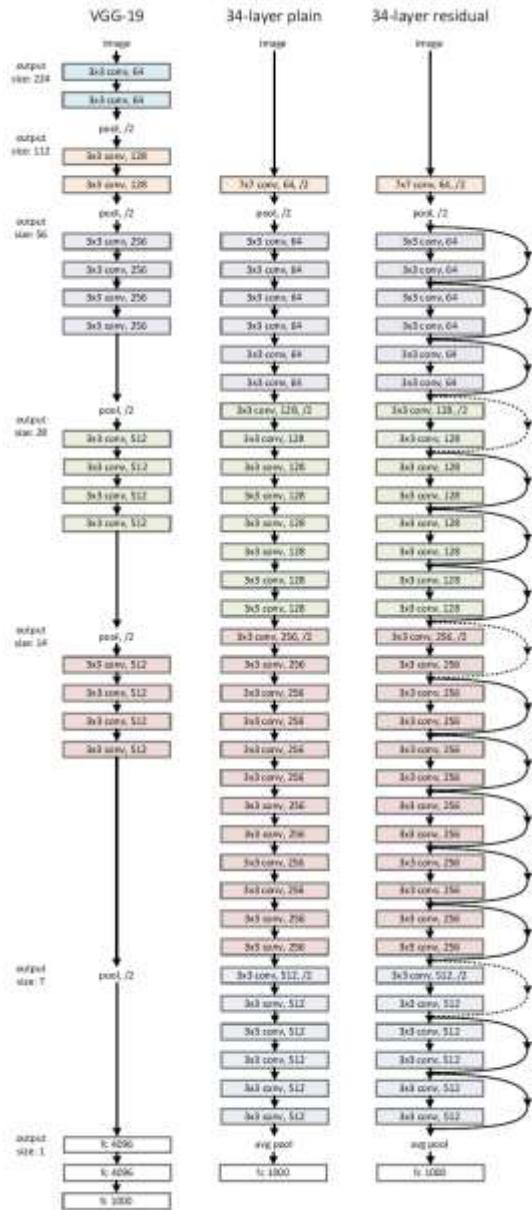


- Residual net



- Будем учить не преобразование, а добавку (пертурбацию) к тождественному преобразованию
 - Если единичное преобразование оптимально, тогда мы его сохраняем
 - Небольшие флюктуации оказывается обучать проще

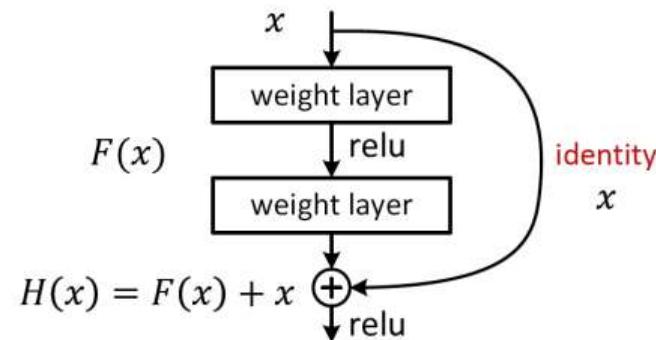
ResNet



Базовая модель

- Свёртки 3x3
- Subsampling через свёртку с шагом 2

• Residual net

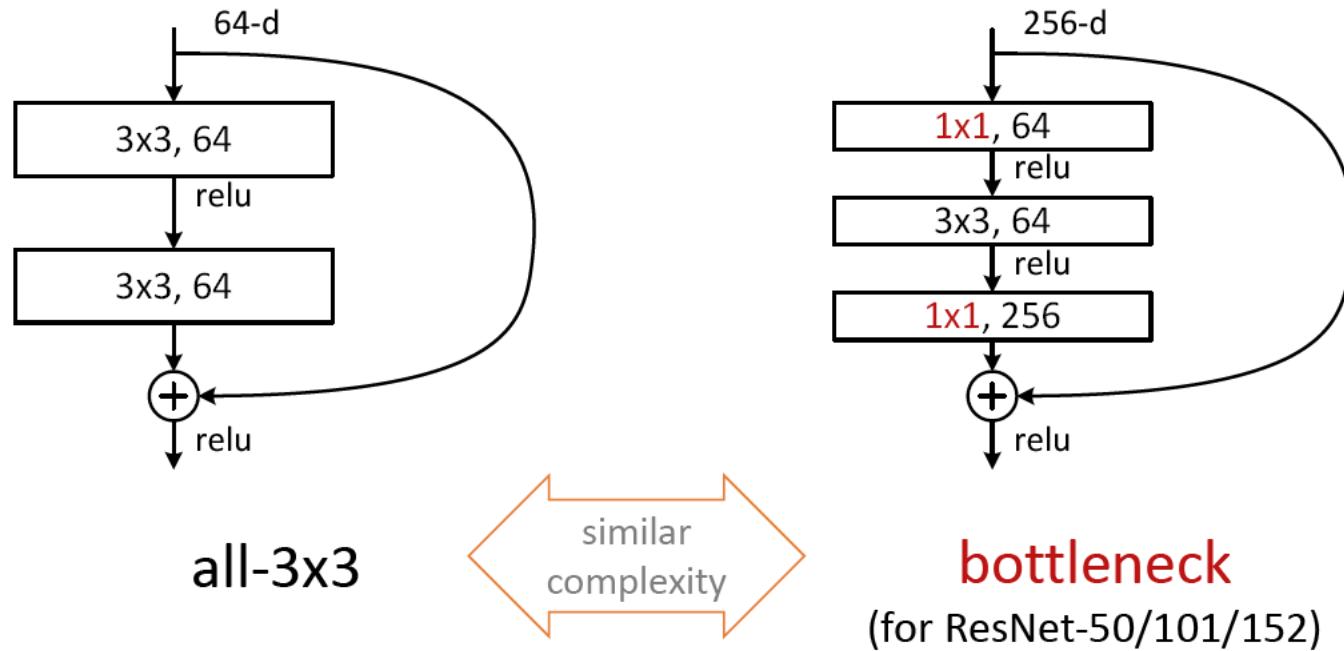


При изменении размеров тензоров пробуют варианты:

- Добавление нулями
- Линейная проекция
- Шаг 2

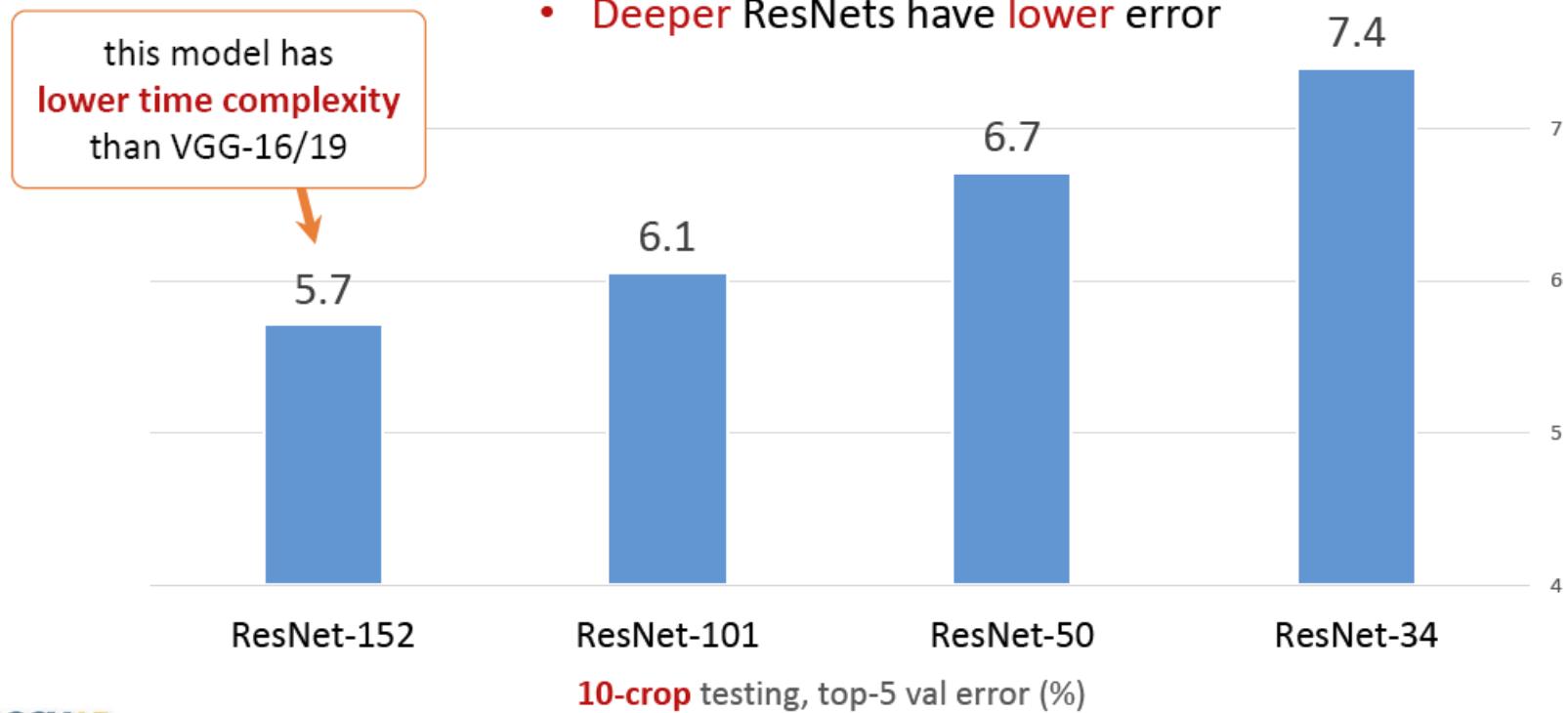
Блок для очень глубоких сетей

- A practical design of going deeper



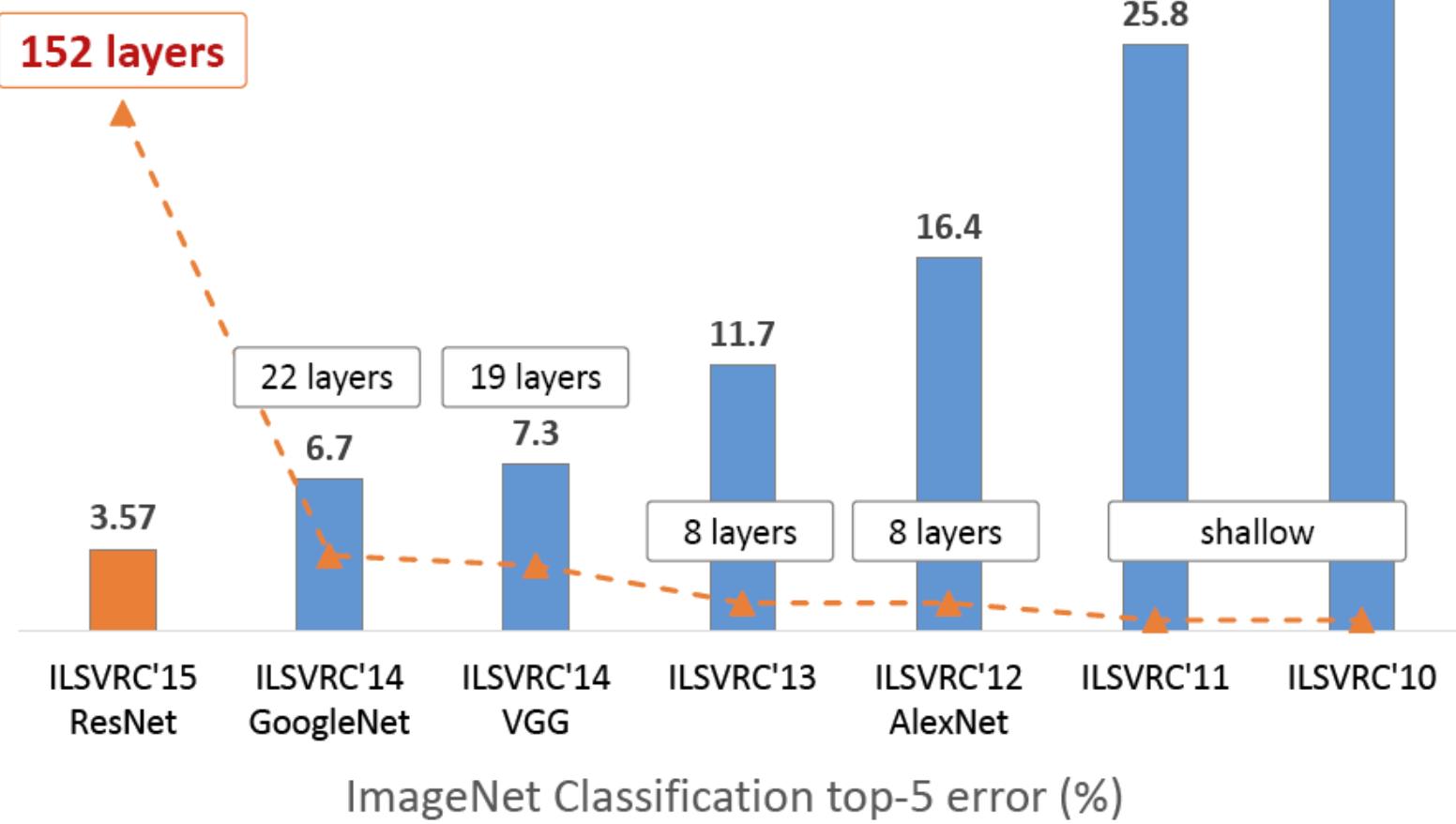
- Понижение размерности (256->64)
- Свёртка 3x3 на тензоре меньшей глубины
- Повышение размерности

Результаты на ImageNet

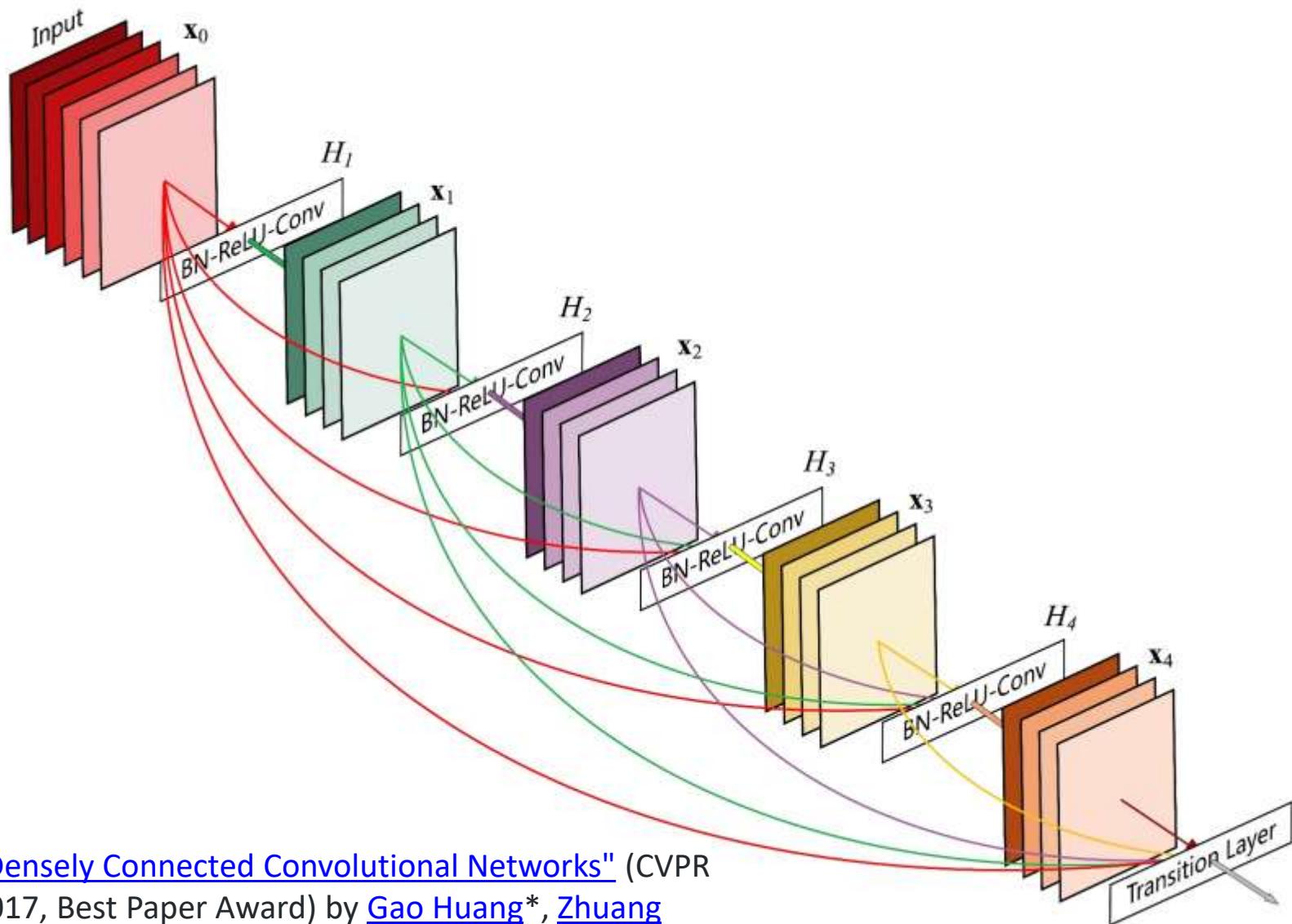


Результаты на ImageNet

ImageNet experiments

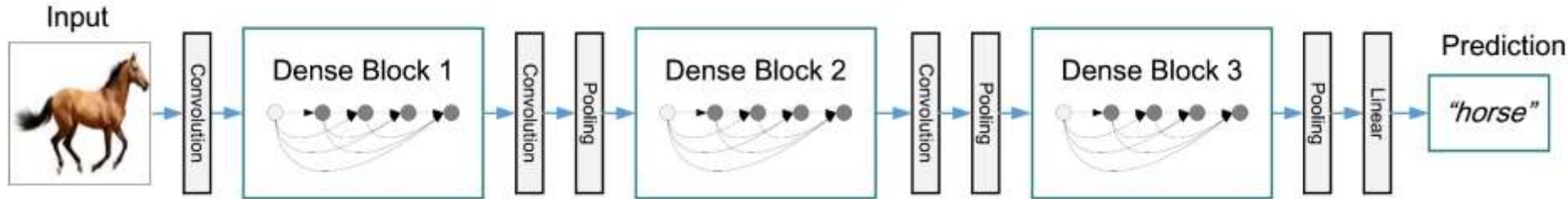


DenseNet



["Densely Connected Convolutional Networks"](#) (CVPR
2017, Best Paper Award) by [Gao Huang*](#), [Zhuang](#)
[Liu*](#), [Laurens van der Maaten](#) and [Kilian Weinberger](#)

DenseNet

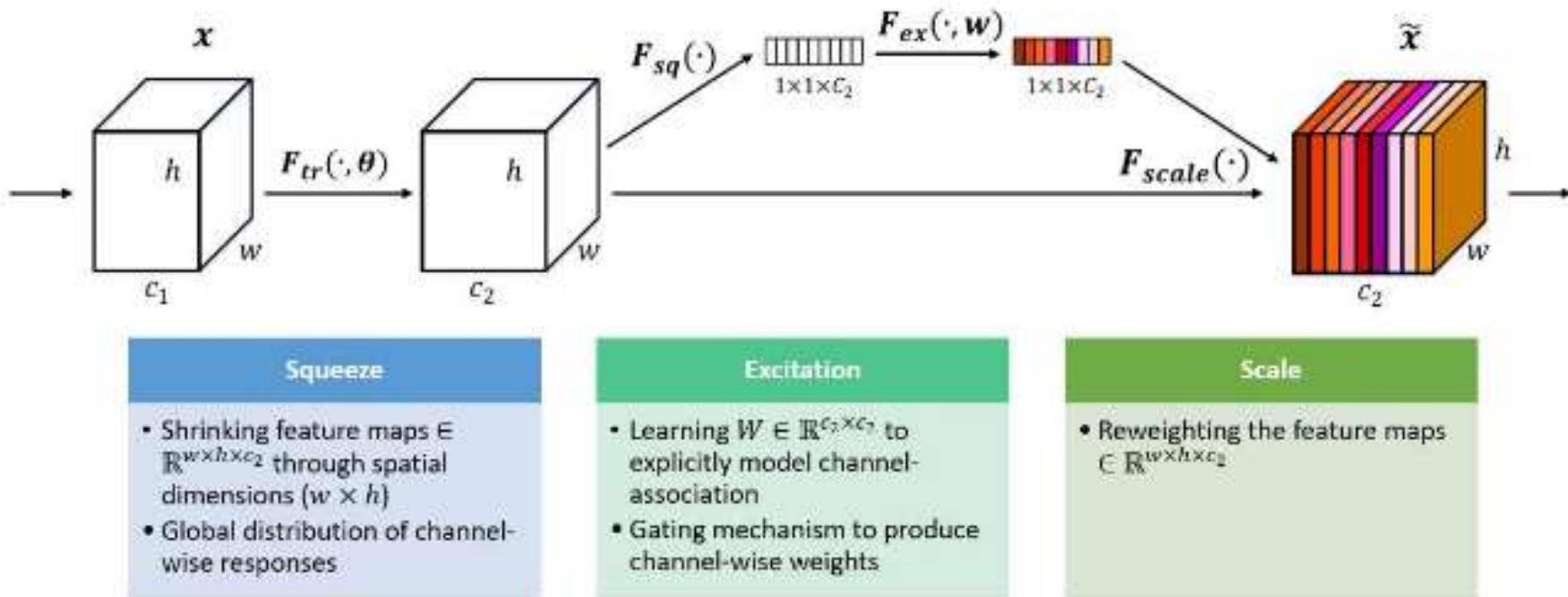


Network	Top-1 error	Torch Model
DenseNet-121 (k=32)	25.0	Download (64.5MB)
DenseNet-169 (k=32)	23.6	Download (114.4MB)
DenseNet-201 (k=32)	22.5	Download (161.8MB)
DenseNet-161 (k=48)	22.2	Download (230.8MB)

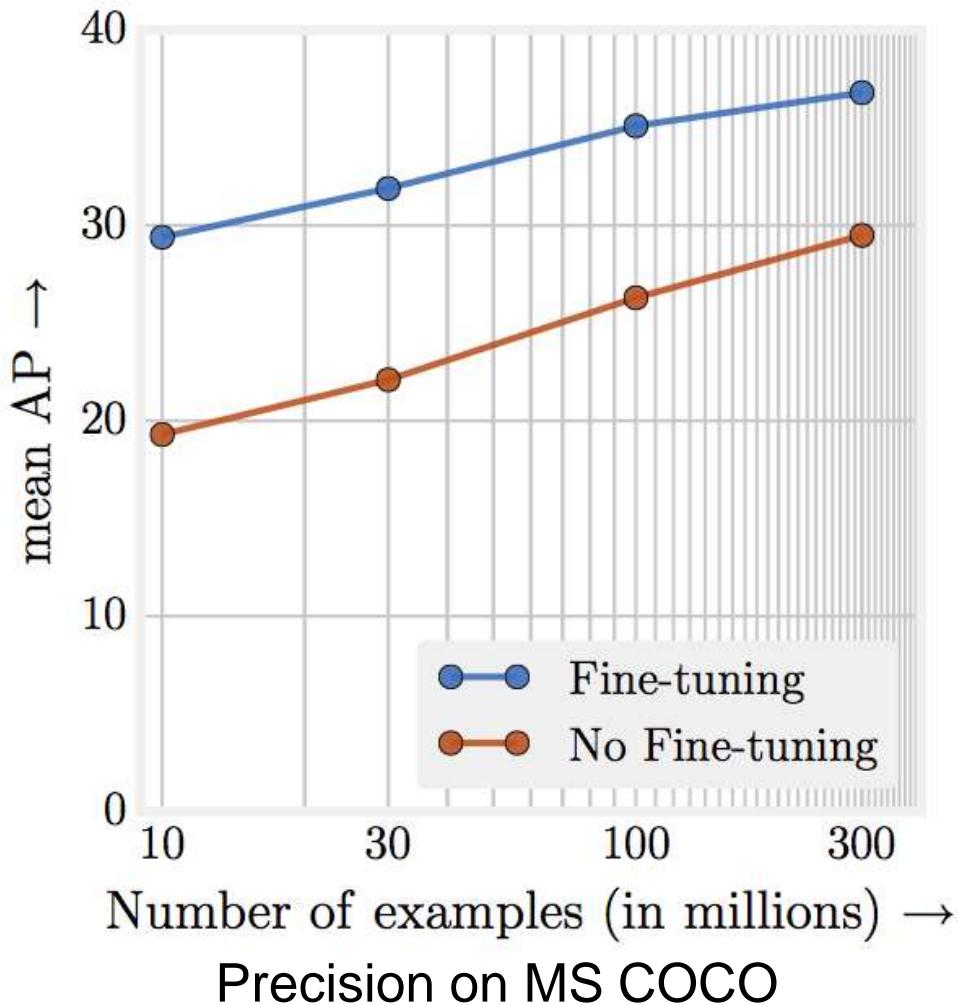
Model	Parameters	CIFAR-10	CIFAR-10+	CIFAR-100	CIFAR-100+
DenseNet (L=40, k=12)	1.0M	7.00	5.24	27.55	24.42
DenseNet (L=100, k=12)	7.0M	5.77	4.10	23.79	20.20
DenseNet (L=100, k=24)	27.2M	5.83	3.74	23.42	19.25
DenseNet-BC (L=100, k=12)	0.8M	5.92	4.51	24.15	22.27
DenseNet-BC (L=250, k=24)	15.3M	5.19	3.62	19.64	17.60
DenseNet-BC (L=190, k=40)	25.6M	-	3.46	-	17.18

<https://github.com/liuzhuang13/DenseNet>

Squeeze and excitation networks

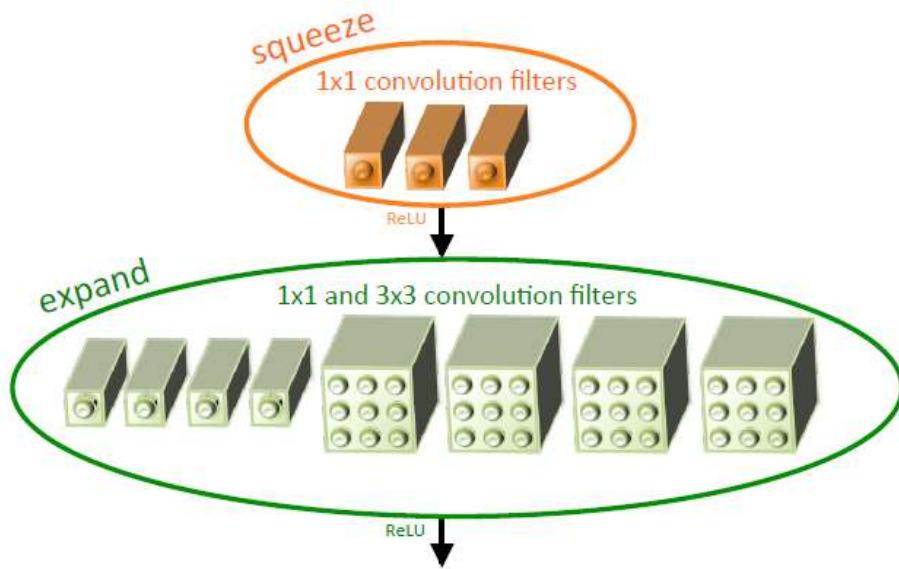


Предобучение (JFT-300M by Google)

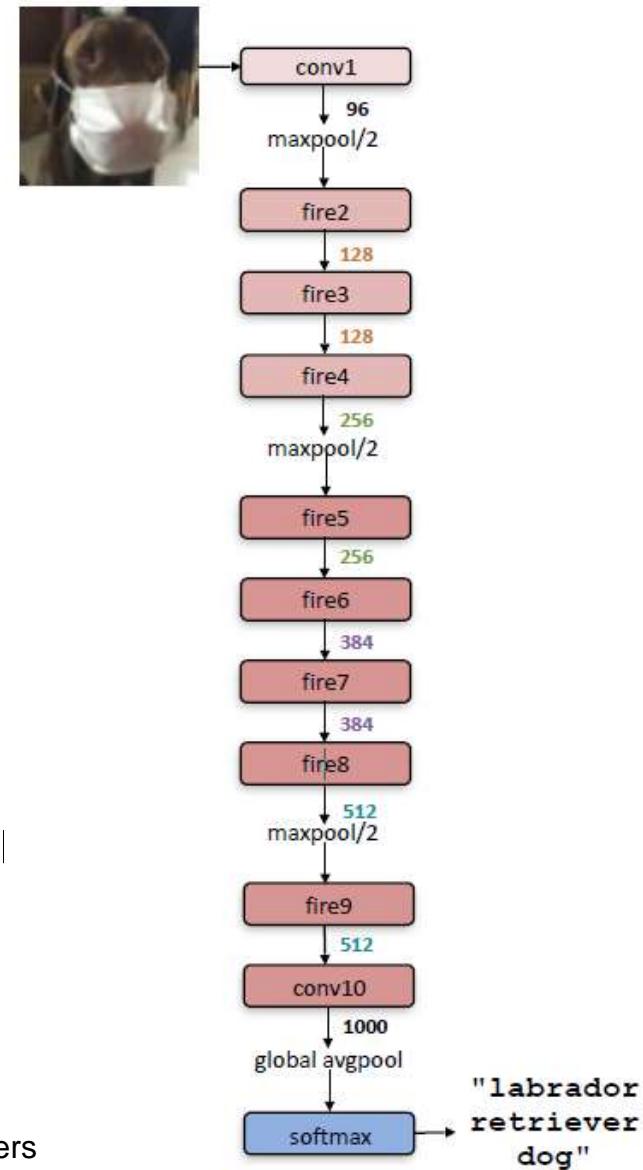


- 300M annotated images, 1B labels of 18291 categories
- 375M labels selected to maximize label precision, still ~20% errors
- ResNet-101 training on 50 x K80 for month
- Uses as pre-training for other tasks (object detection, semantic segmentation, pose estimation)

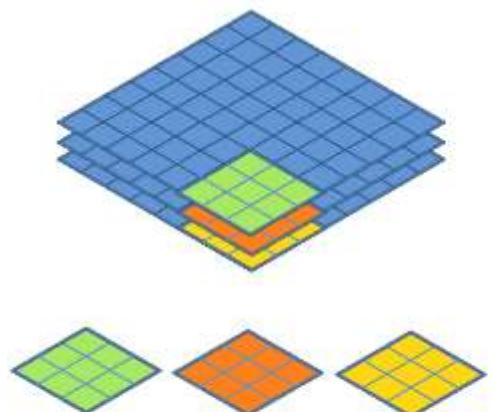
SqueezeNet



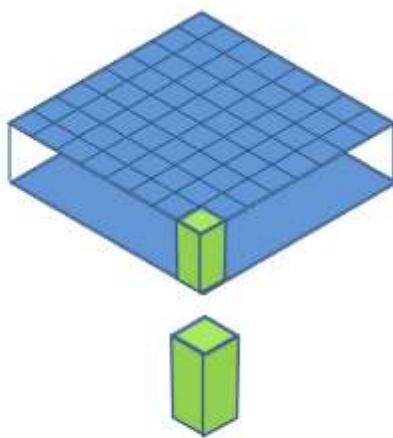
- Активно использовать 1x1 свёртки для уменьшения числа параметров
- «Сжимать» мы будем для того, чтобы на вход 3x3 фильтрам подавать меньше данных



Факторизация свёрток



Depthwise Convolutional Filters



Pointwise Convolutional Filters

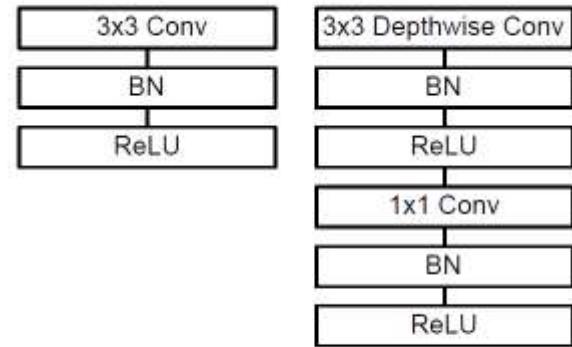


Figure 3. Left: Standard convolutional layer with batchnorm and ReLU. Right: Depthwise Separable convolutions with Depthwise and Pointwise layers followed by batchnorm and ReLU.

- Сложность обычного свёрточного слоя:

$$D_K * D_K * M * N * D_F * D_F$$

- 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 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5× Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Table 2. Resource Per Layer Type

Type	Mult-Adds	Parameters
Conv 1×1	94.86%	74.59%
Conv DW 3×3	3.06%	1.06%
Conv 3×3	1.19%	0.02%
Fully Connected	0.18%	24.33%

Table 4. Depthwise Separable vs Full Convolution MobileNet

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

Оценка вариантов MobileNet

Table 6. MobileNet Width Multiplier

Width Multiplier	ImageNet	Million	Million
	Accuracy	Mult-Adds	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	Million	Million
	Accuracy	Mult-Adds	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

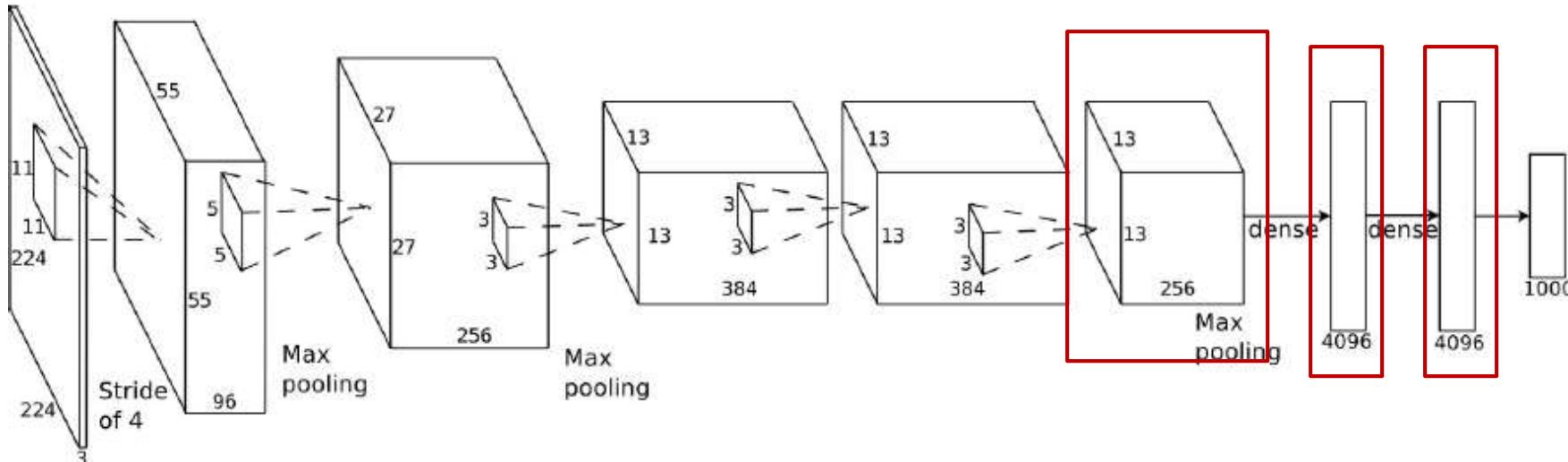
Table 8. MobileNet Comparison to Popular Models

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

Резюме базовых архитектур

- Основные принципы построения архитектур
 - Строить глубокие модели из похожих блоков
 - Большие свёртки можно приближать последовательностью свёрток 3x3
 - Свёртки 1x1 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
- Предобучать лучше на как можно большей и разносторонней коллекции
- Придумывают «быстрые» архитектуры

Нейросетевые признаки для поиска похожих



- Выходы верхних слоёв можно использовать как вектор-признак изображения
- Попробуем искать похожие изображения по этим вектор-признакам
 - L1, L2 метрики

Посмотрим работу

Важны мелкие текстурные детали

Слой 5:

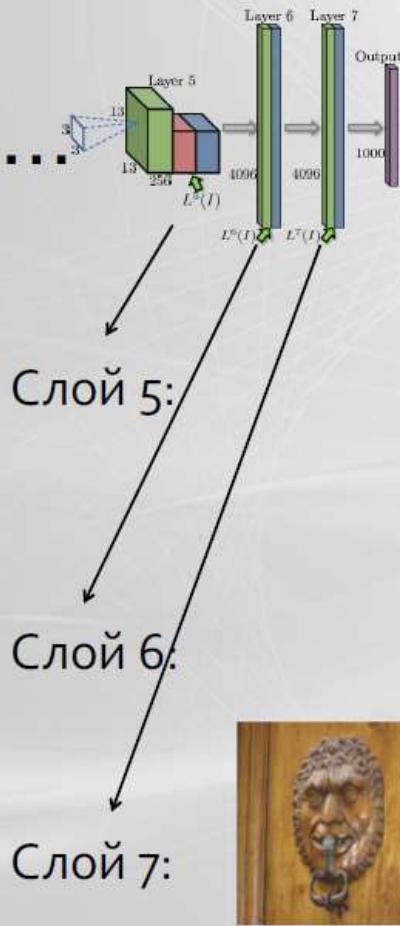
Слой 6:

Слой 7:

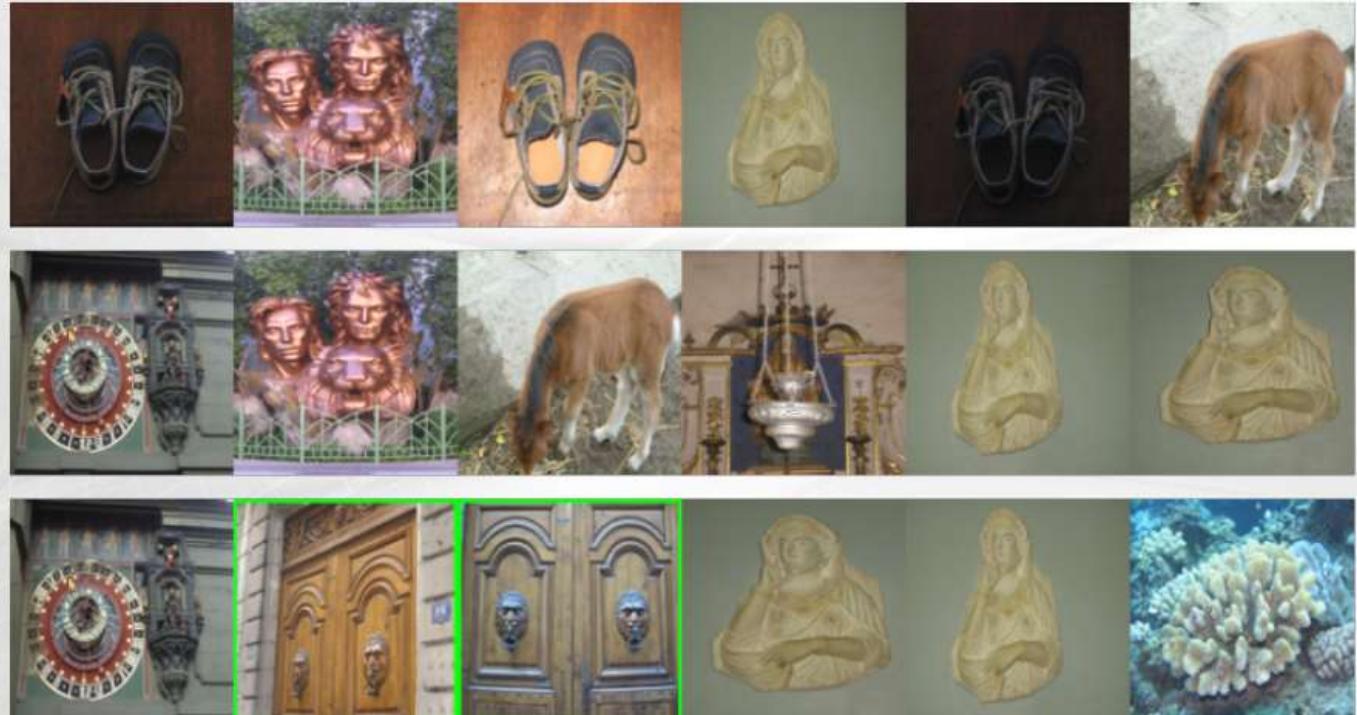
Запрос

Выдача

Посмотрим работу



Важна семантика



Запрос

Выдача

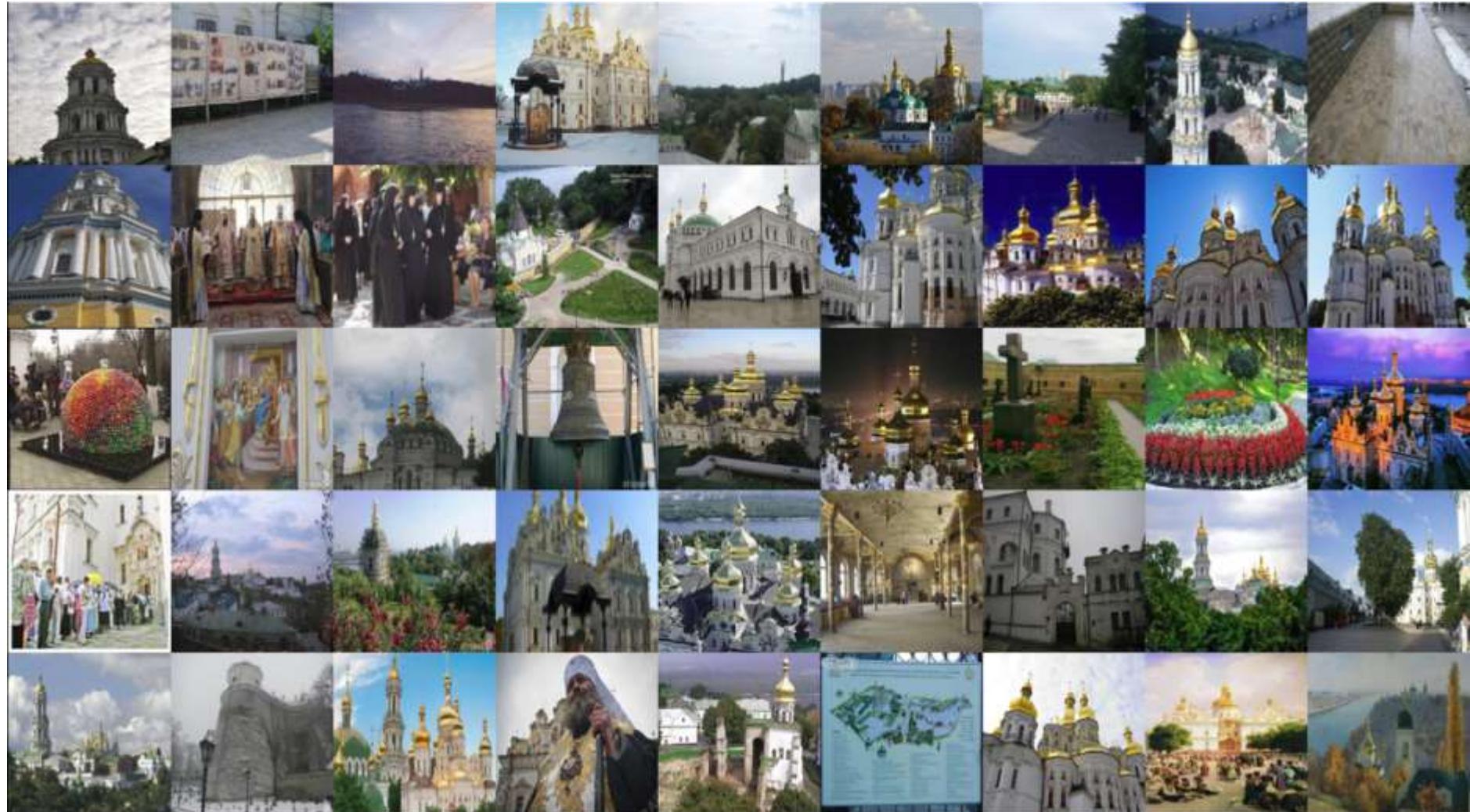
Адаптация к конкретной задаче

- В приложении для туризма (Яндекс.Променад) большинство запросов являются фотографиями достопримечательностей
- Нейросеть дообучалась на коллекции изображений достопримечательностей, чтобы формируемые дескрипторы «настроились» на семантику конкретной поисковой задачи
- Сбор коллекции проводился в полуавтоматическом режиме путем запросов к поисковой системе

Новая коллекция: Leeds Castle

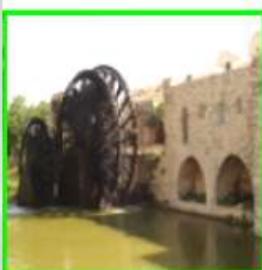
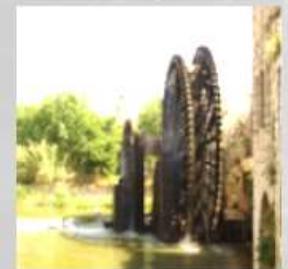


Новая коллекция: Kiev Pechersk Lavra



До и после адаптации

Запрос

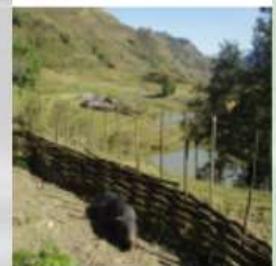


Выдача



До и после адаптации

Запрос

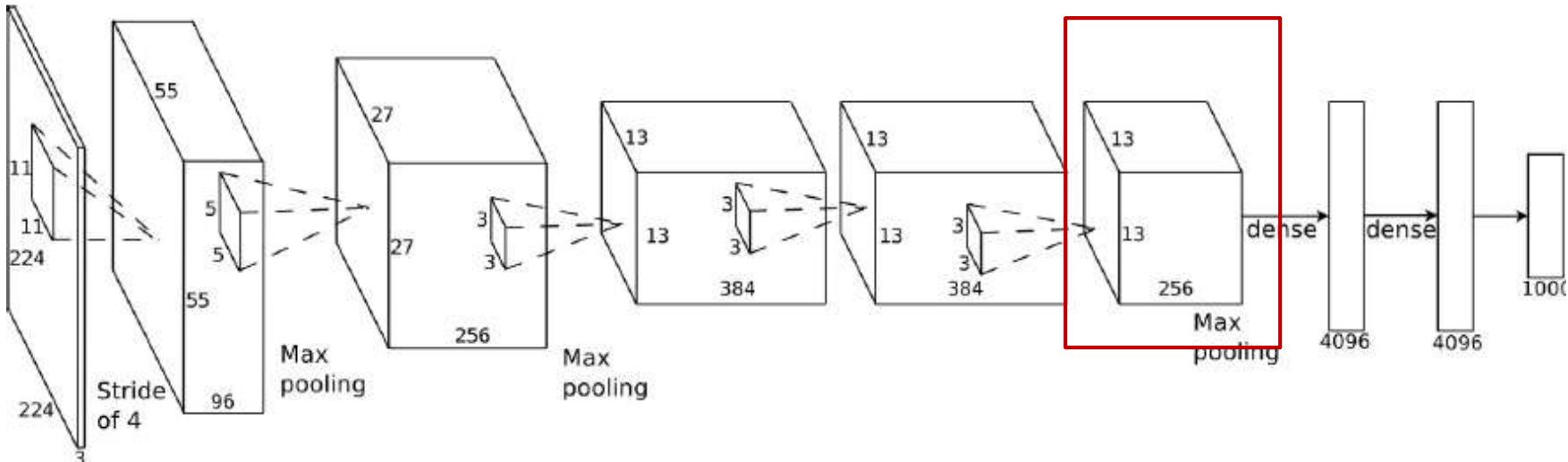


Выдача

Сжатие РСА до 128

Дескриптор	Oxford	Oxford105K	Holidays
Fisher+color	-	-	0.723
VLAD+adapt+innorm	0.448	0.374	0.625
Sparse-coded features	-	-	0.727
Triangulation embedding	0.433	0.353	0.617
Нейродескрипторы			
Слой 6	0.433	0.386	0.727
Нейродескрипторы с адаптацией			
Слой 6	0.557	0.523	0.769

Высокоуровневые признаки



- Дескрипторы могут быть вычислены только для квадратных изображений небольшого фиксированного размера
- При уменьшении изображений большого разрешения теряются маленькие детали
- Решение: использовать выходы сверточных слоев, они могут быть вычислены для любых изображений
- Делать average по выходу каждого свёрточного слоя

A Babenko, V Lempitsky [Aggregating local deep features for image retrieval](#)
CVPR 2015

Полносвязанные vs свёрточные

Результаты с полносвязным дескриптором:



Запрос

Яндекс Картинки Загруженная картинка Найти

A screenshot of a Yandex Images search results page. The search bar at the top shows the text 'Картинки' (Images) and 'Загруженная картинка' (Loaded image). To the right is a magnifying glass icon and the button 'Найти' (Find). The main area displays a 4x5 grid of images. The first row contains a black cat, a black kitten, a black dog, a black goat, and a black cat. The second row contains a black monkey, a black kitten, a black monkey, a black dog, and a black gorilla. The third row contains a black monkey, a black kitten, a black monkey, a black dog, and a black monkey. The fourth row contains a black monkey, a black dog, a black monkey, a black dog, and a black monkey. Some images have text overlaid: '说是出去打猎了' (Said to go hunting) and 'АЧЗХ?' (ACH?)

Полносвязанные vs свёрточные

Результаты со свёрточным дескриптором:



Запрос

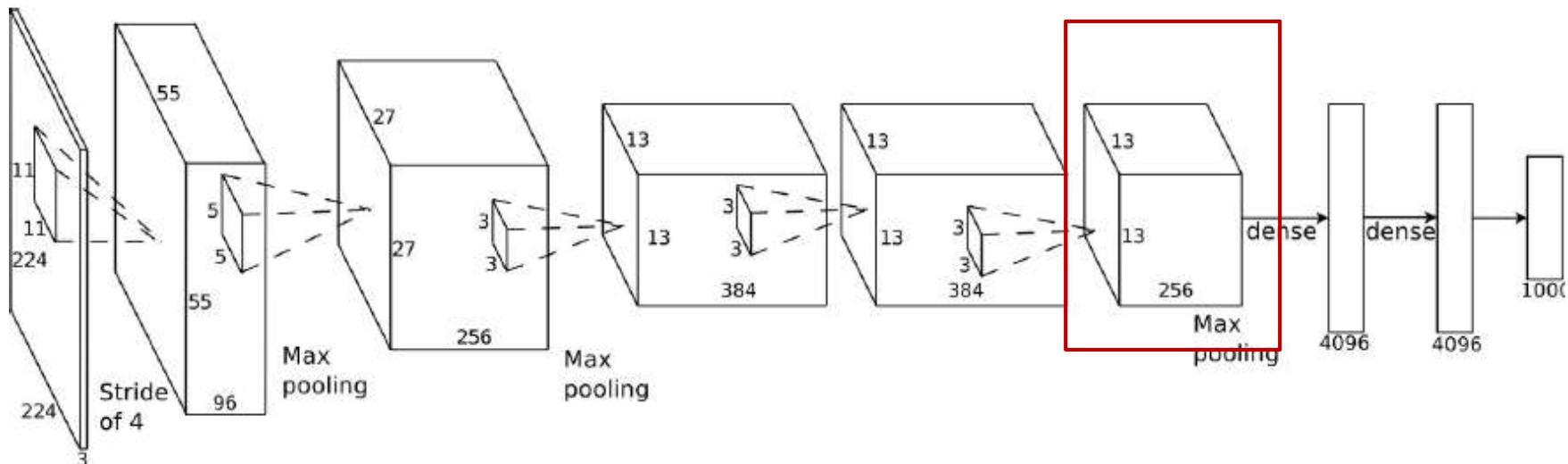
Яндекс Картинки Загруженная картинка Найти

A search results page from Yandex Images. At the top, there's a header with the Yandex logo, the word 'Картинки' (Images), a placeholder for 'Загруженная картинка' (Loaded image), a magnifying glass icon, and a 'Найти' (Find) button. Below the header is a grid of 24 smaller images arranged in four rows of six. All the images show black cats with varying features like eye color (green, yellow, blue) and patterns (solid black, tabby). In the bottom right corner of the grid, there's a button labeled 'Ещё похожие' (More similar) with a right-pointing arrow.

Сжатие РСА до 128

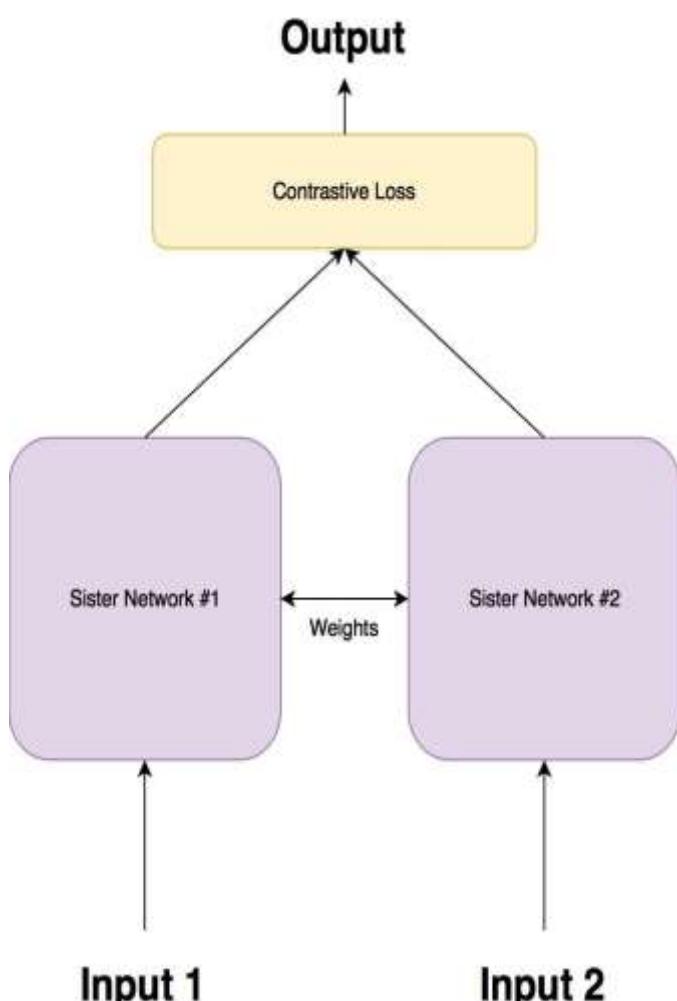
Дескриптор	Oxford	Oxford105K	Holidays
Fisher+color	-	-	0.723
VLAD+adapt+innorm	0.448	0.374	0.625
Sparse-coded features	-	-	0.727
Triangulation embedding	0.433	0.353	0.617
Нейродескрипторы			
Слой 6	0.433	0.386	0.727
Нейродескрипторы с адаптацией			
Слой 6	0.557	0.523	0.769
Сверточный слой	0.657	0.642	0.802

Что плохо в таком подходе?



- Мы строим отображение в вектор-признак косвенно, решая другую задачу – классификации
- Мы можем переформулировать задачу, строя специально хорошие признаки

Обучение представлений



Классический contrastive loss с использованием евклидовой метрики:

$$L(\theta, (X_1, X_2, Y)) = (1 - Y)L_G(\|X_1 - X_2\|_2) + YL_I(\|X_1 - X_2\|_2)$$

L_G - функция потерь для положительной пары, L_I - для отрицательной.

Как правило, $L_G(d) = d^2$, $L_I(d) = \max(0, m - d)_+^2$

[Chopra, et.al., 2005]

В качестве меры близости можно использовать и косинусную меру.

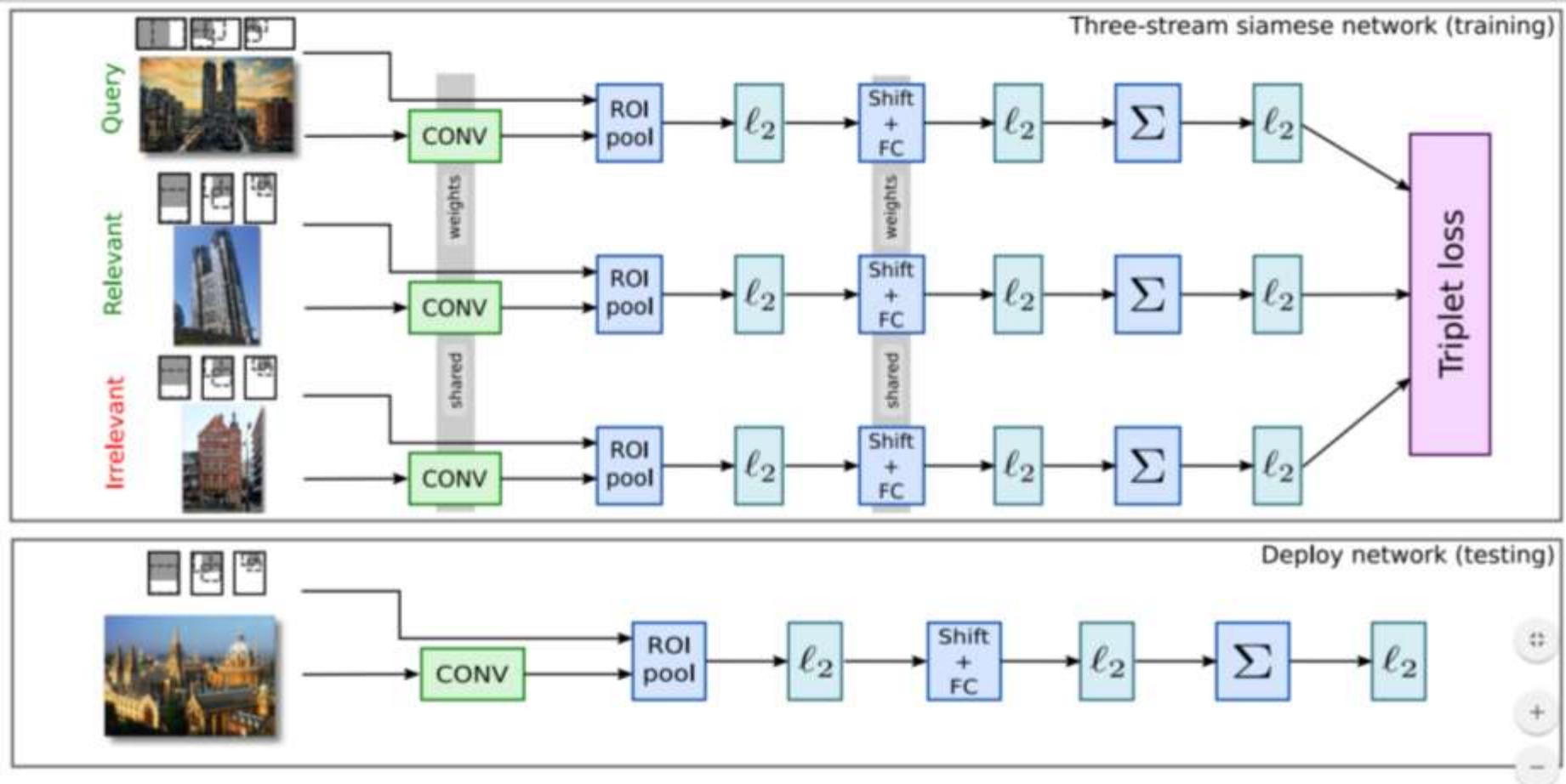
$$L(\theta, X_1, X_2, Y) = \frac{1}{2} (Y - \sigma(wd + b)),$$

где близость $d = \frac{X_1 X_2}{\|X_1\|_2 \|X_2\|_2}$

[Sun, et.al., 2014]

Radenović F., Tolias G., Chum O. CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples. ECCV 2016

Triplet loss



Триплет состоит из запроса q (query), релевантного изображения и нерелевантного изображения

Triplet-loss

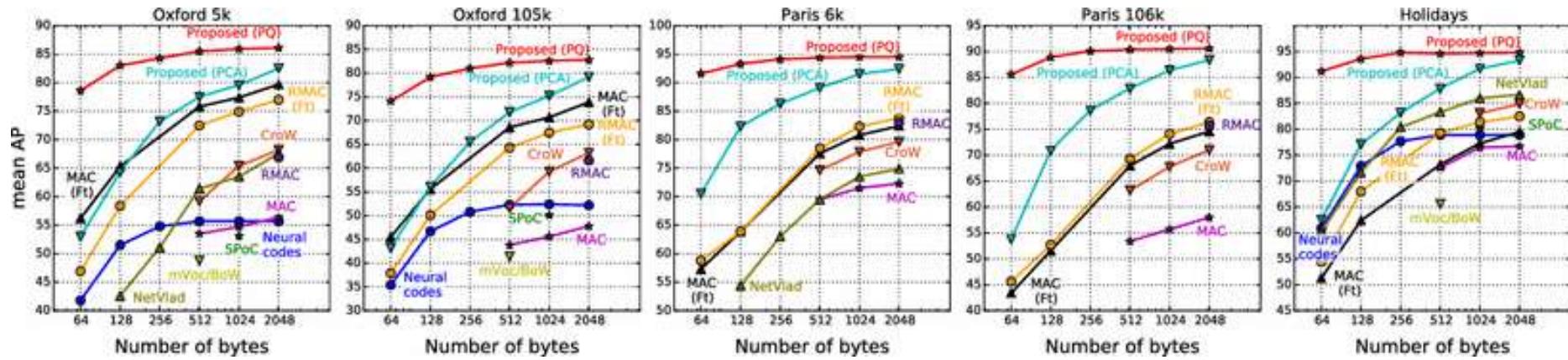
We use the following ranking loss. Let I_q be a query image with R-MAC descriptor q , I^+ be a relevant image with descriptor d^+ , and I^- be an irrelevant image with descriptor d^- . We define the ranking triplet loss as

$$L(I_q, I^+, I^-) = \frac{1}{2} \max(0, m + \|q - d^+\|^2 - \|q - d^-\|^2), \quad (1)$$

where m is a scalar that controls the margin. Given a triplet that produces a non-zero loss, the sub-gradients are given by:

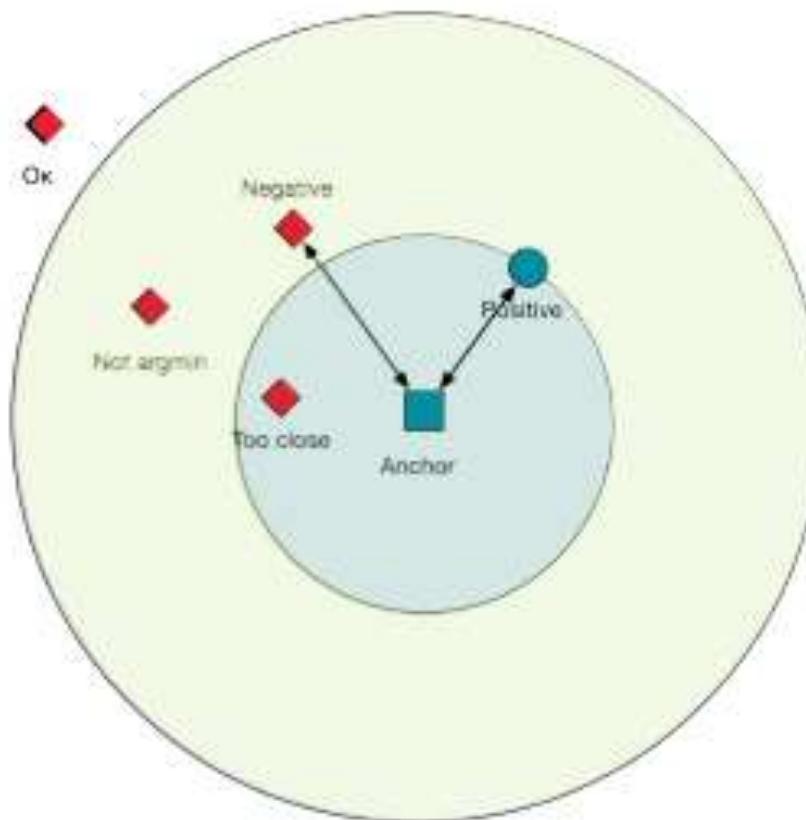
$$\frac{\partial L}{\partial q} = d^- - d^+, \quad \frac{\partial L}{\partial d^+} = d^+ - q, \quad \frac{\partial L}{\partial d^-} = q - d^-. \quad (2)$$

Сжатие



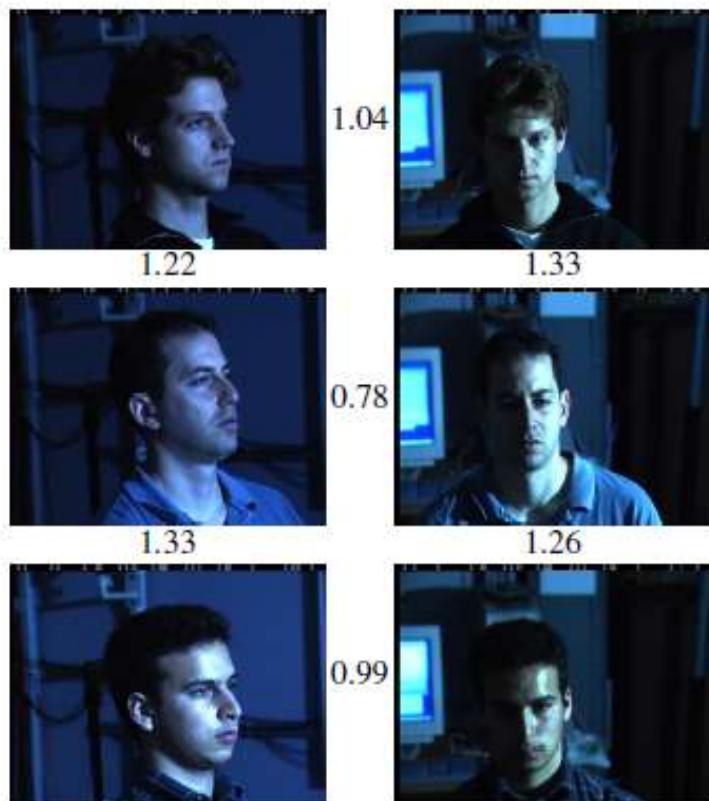
Сжатие обученных признаков с помощью РСА и PQ(product quantization)

Выбор триплетов



Пространство, полученное отображением $f(\cdot)$.

FaceNet – пример подобного подхода



- Обучение представлений
- Функция $y=f(I)$ отображает изображение I в 128-мерное пространство, в котором L2-метрика соответствует близости людей
- $f(x)$ – глубокая нейросеть
- Обучается triplet loss



Резюме по части поиска

- Для поиска похожих нужно дообучение на близкой по смыслу коллекции
- Стоит явно учить «представление», чтобы у похожих объектов были похожие вектор-признаки
 - Contrastive loss
 - Triplet loss и производные варианты