

Основы глубинного обучения

Лекция 6
Архитектуры свёрточных сетей

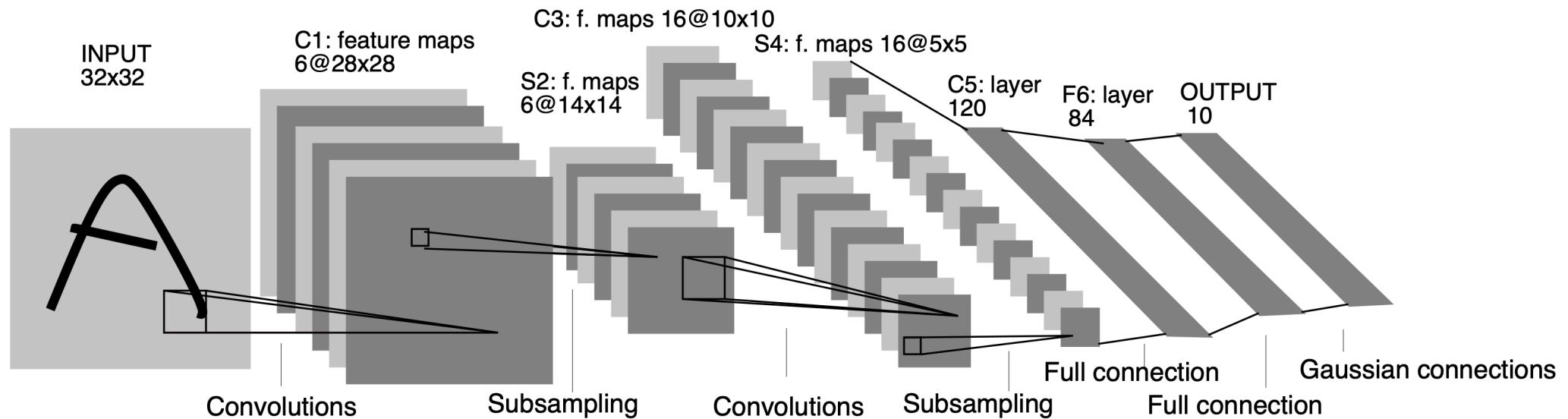
Евгений Соколов

esokolov@hse.ru

НИУ ВШЭ, 2023

Архитектуры свёрточных сетей

LeNet (1998)



LeNet (1998)

- Для данных MNIST
- Идея end-to-end обучения
- Использовали аугментацию
- Около 60.000 параметров
- Доля ошибок на тесте 0.8%

ImageNet



- ImageNet Large Scale Visual Recognition Challenge (ILSVRC)
- Около 1.000.000 изображений
- 1000 классов
- Обычно качество измерялось на основе лучшей гипотезы модели

AlexNet (2012)

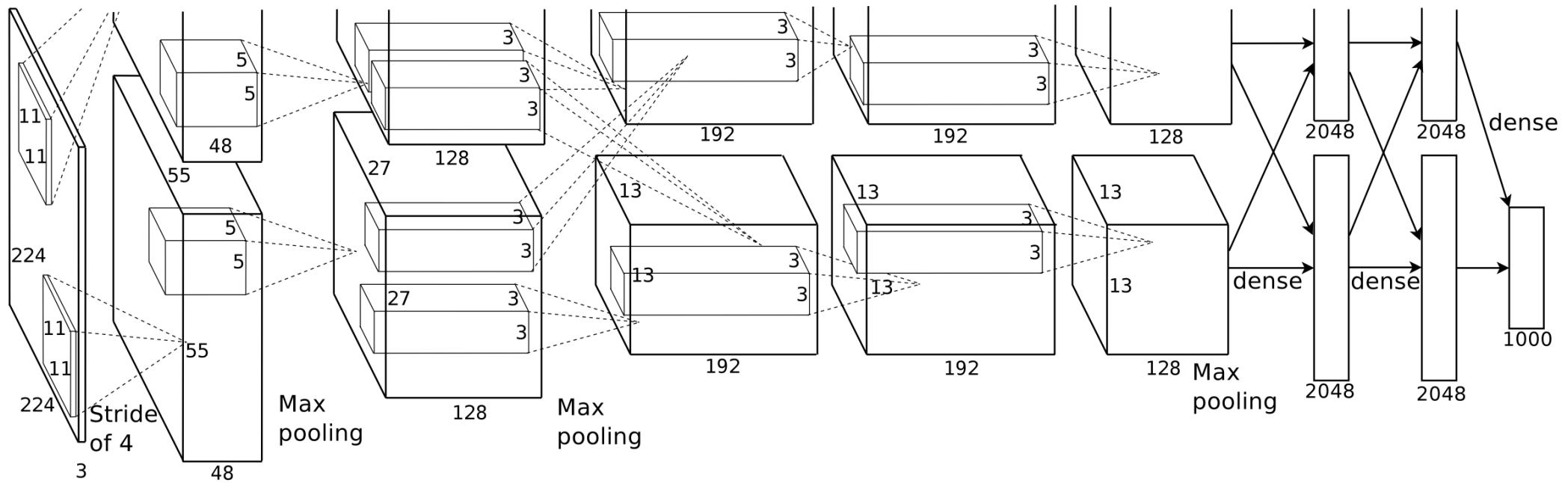
ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
`kriz@cs.utoronto.ca`

Ilya Sutskever
University of Toronto
`ilya@cs.utoronto.ca`

Geoffrey E. Hinton
University of Toronto
`hinton@cs.utoronto.ca`

AlexNet (2012)



AlexNet (2012)

- Используют ReLU, аугментацию, dropout
- Градиентный спуск с инерцией (momentum)
- Обучение на двух GPU (5-6 суток)
- Около 60 миллионов параметров
- Ошибка около 17%

VGG (2014)

VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

Karen Simonyan* & **Andrew Zisserman⁺**

Visual Geometry Group, Department of Engineering Science, University of Oxford
`{karen,az}@robots.ox.ac.uk`

VGG (2014)

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)

Table 2: Number of parameters (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

VGG (2014)

- Только маленькие свёртки
 - Меньше параметров
 - Больше нелинейностей (т.к. больше свёрточных слоёв)
- Градиентный спуск с инерцией
- Dropout для двух первых полносвязных слоёв
- Хитрая инициализация (сначала обучается вариант А со случайными начальными весами, потом им инициализируются более глубокие сети)

VGG (2014)

Table 3: ConvNet performance at a single test scale.

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
A	256	256	29.6	10.4
A-LRN	256	256	29.7	10.5
B	256	256	28.7	9.9
C	256	256	28.1	9.4
	384	384	28.1	9.3
	[256;512]	384	27.3	8.8
D	256	256	27.0	8.8
	384	384	26.8	8.7
	[256;512]	384	25.6	8.1
E	256	256	27.3	9.0
	384	384	26.9	8.7
	[256;512]	384	25.5	8.0

GoogLeNet (2014)

Going Deeper with Convolutions

Christian Szegedy¹, Wei Liu², Yangqing Jia¹, Pierre Sermanet¹, Scott Reed³,
Dragomir Anguelov¹, Dumitru Erhan¹, Vincent Vanhoucke¹, Andrew Rabinovich⁴

¹Google Inc. ²University of North Carolina, Chapel Hill

³University of Michigan, Ann Arbor ⁴Magic Leap Inc.

¹{szegedy,jiayq,sermanet,dragomir,dumitru,vanhoucke}@google.com

²wliu@cs.unc.edu, ³reedscott@umich.edu, ⁴arabinovich@magineleap.com

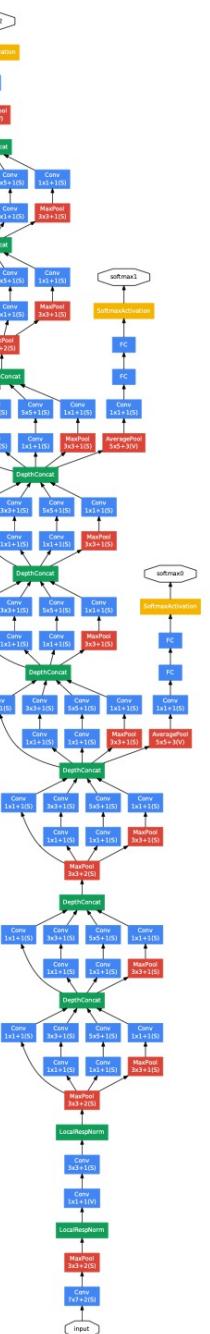
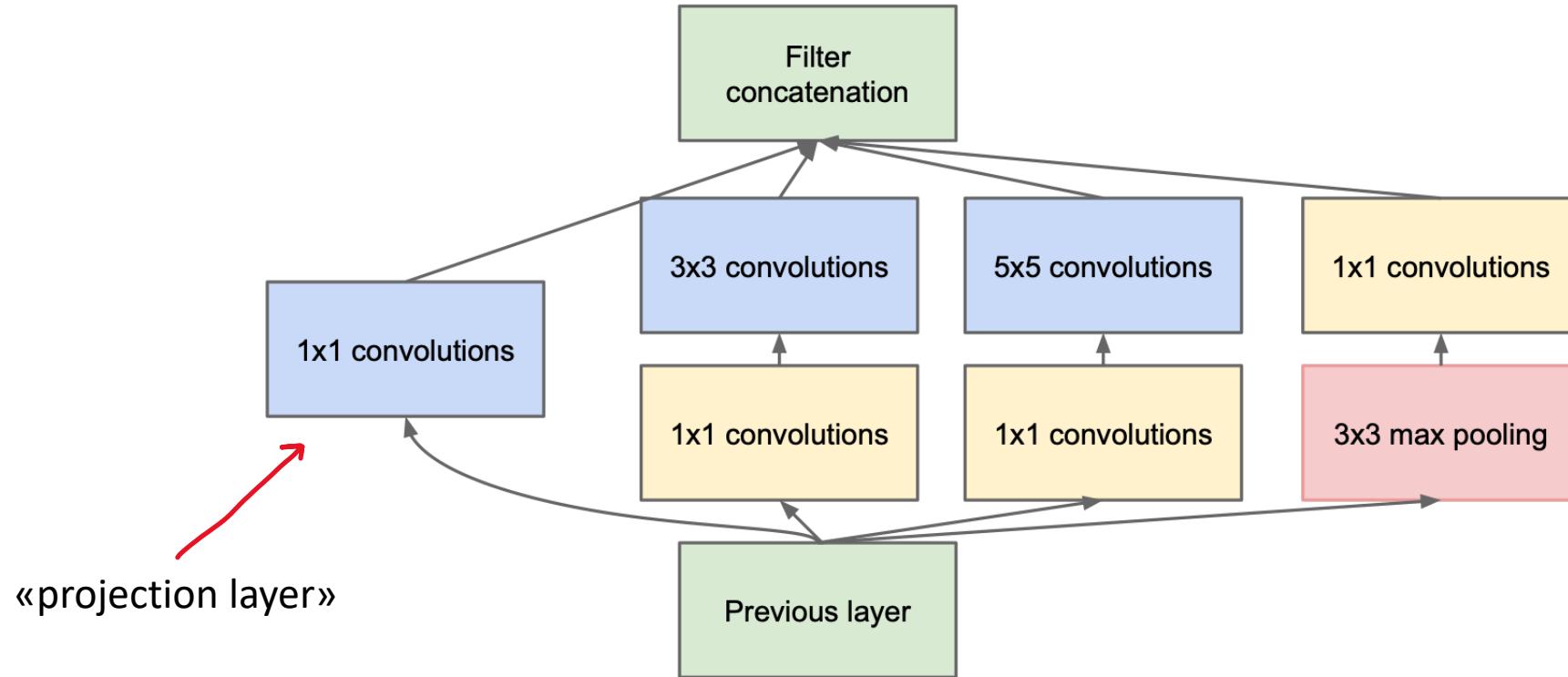


Figure 3: GoogLeNet network with all the bells and whistles.

GoogLeNet (2014)



(b) Inception module with dimensionality reduction

свёртки делаются с паддингом!

<http://arxiv.org/abs/1409.4842>

GoogLeNet (2014)

- Снижается число каналов перед «тяжёлыми» свёртками
- Несколько выходных слоёв для улучшения обучаемости
- Обучается градиентным спуском с инерцией
- Ошибка 6.67% на ImageNet

ResNet (2015)

Deep Residual Learning for Image Recognition

Kaiming He

Xiangyu Zhang

Shaoqing Ren

Jian Sun

Microsoft Research

{kahe, v-xiangz, v-shren, jiansun}@microsoft.com

ResNet (2015)

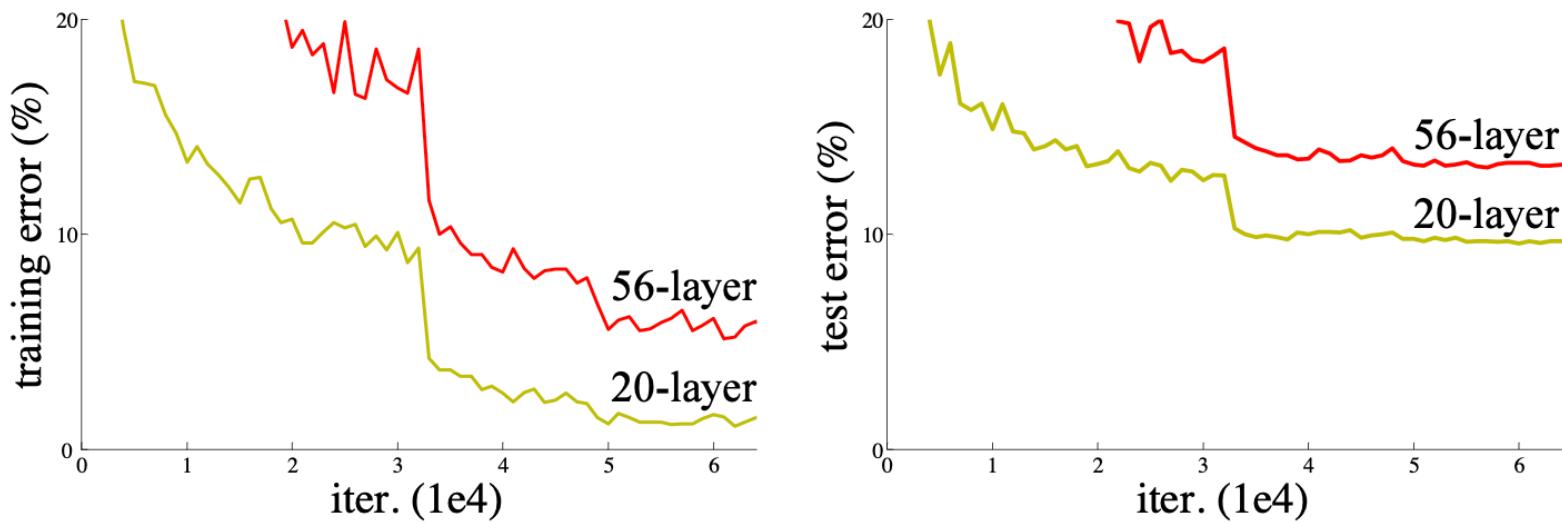
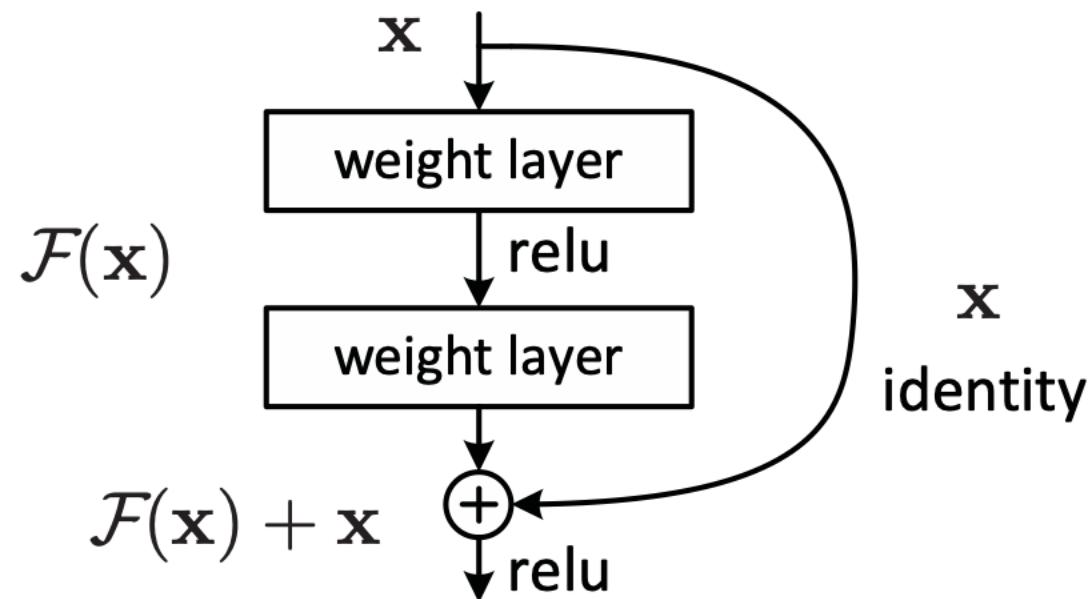


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer “plain” networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

ResNet (2015)

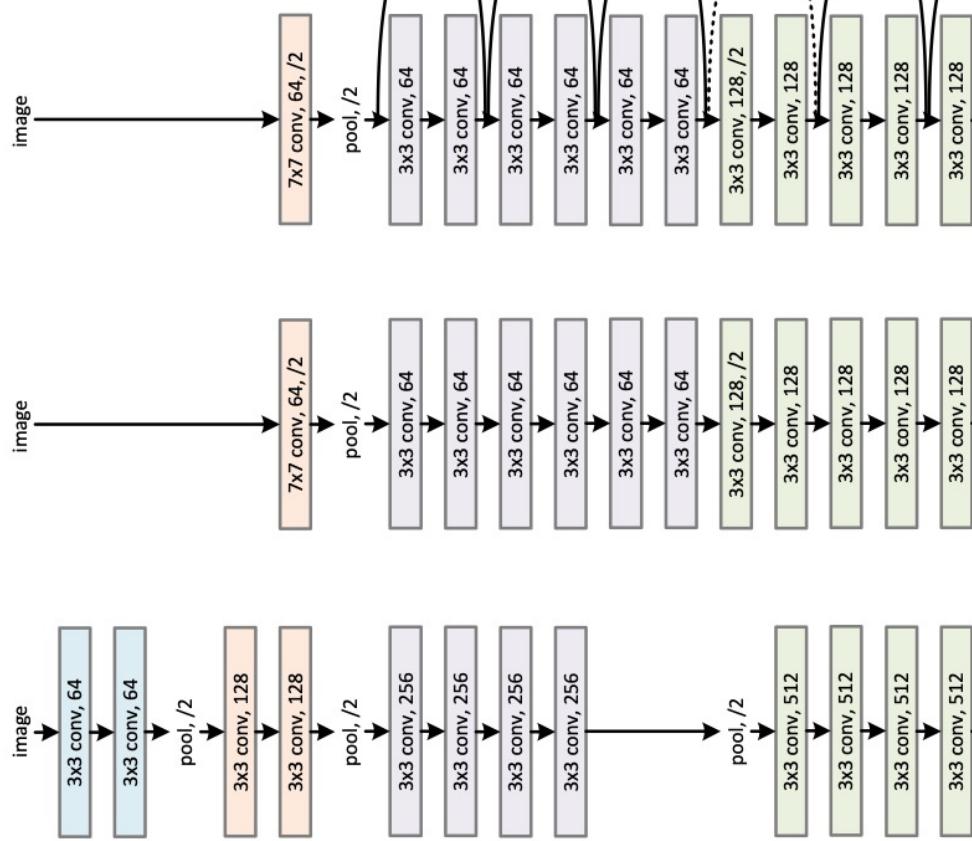
- Добавление слоёв в свёрточную сеть ухудшает качество даже на обучении
- Хотя возможностей для переобучения больше, сеть почему-то не может ими воспользоваться

ResNet (2015)

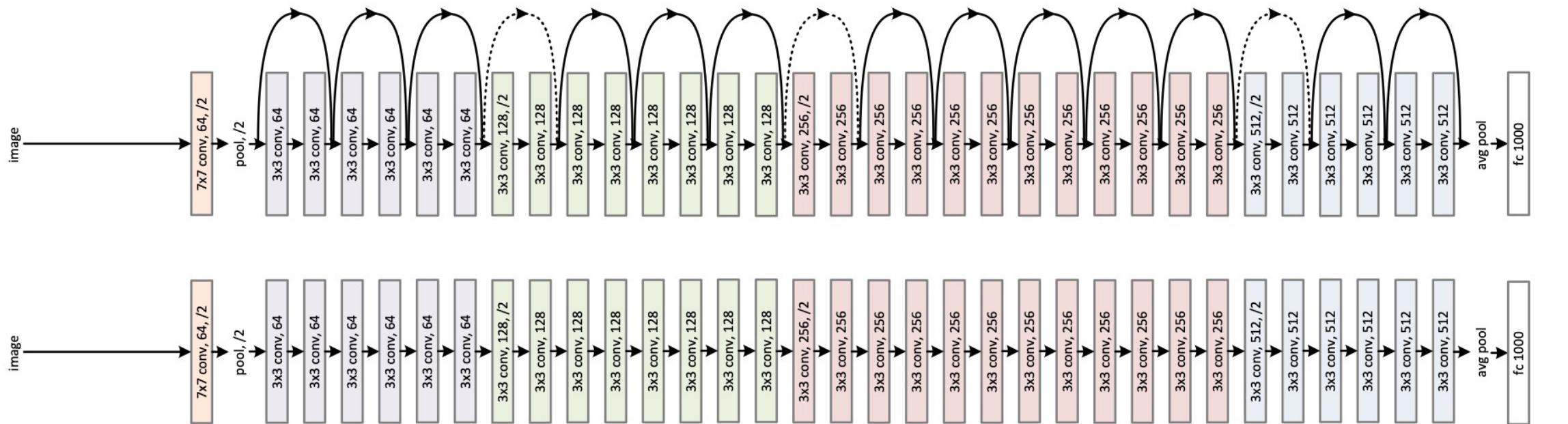


ResNet (2015)

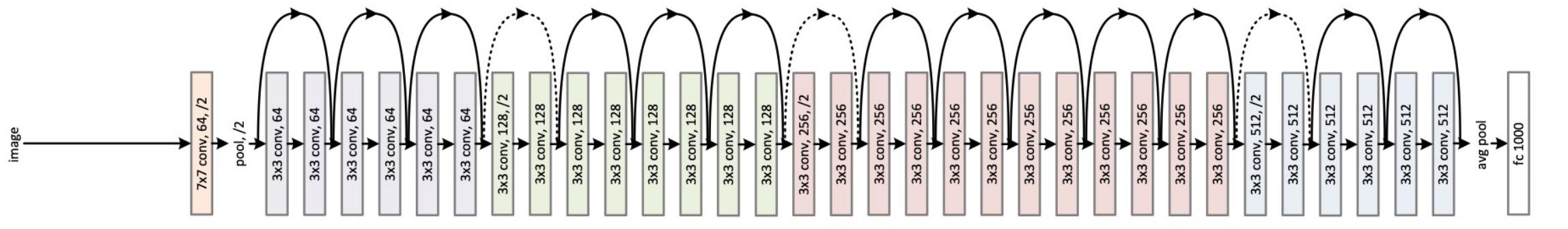
VGG-19



34-layer plain



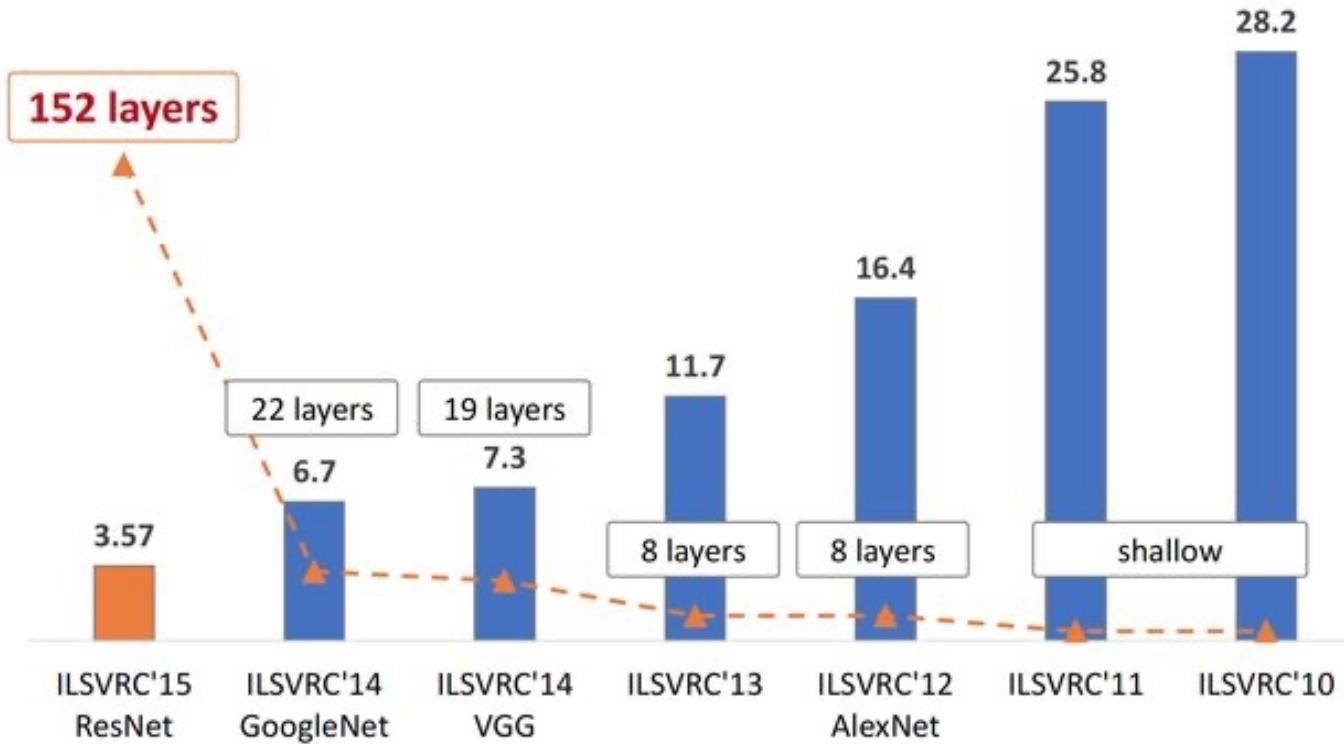
34-layer residual



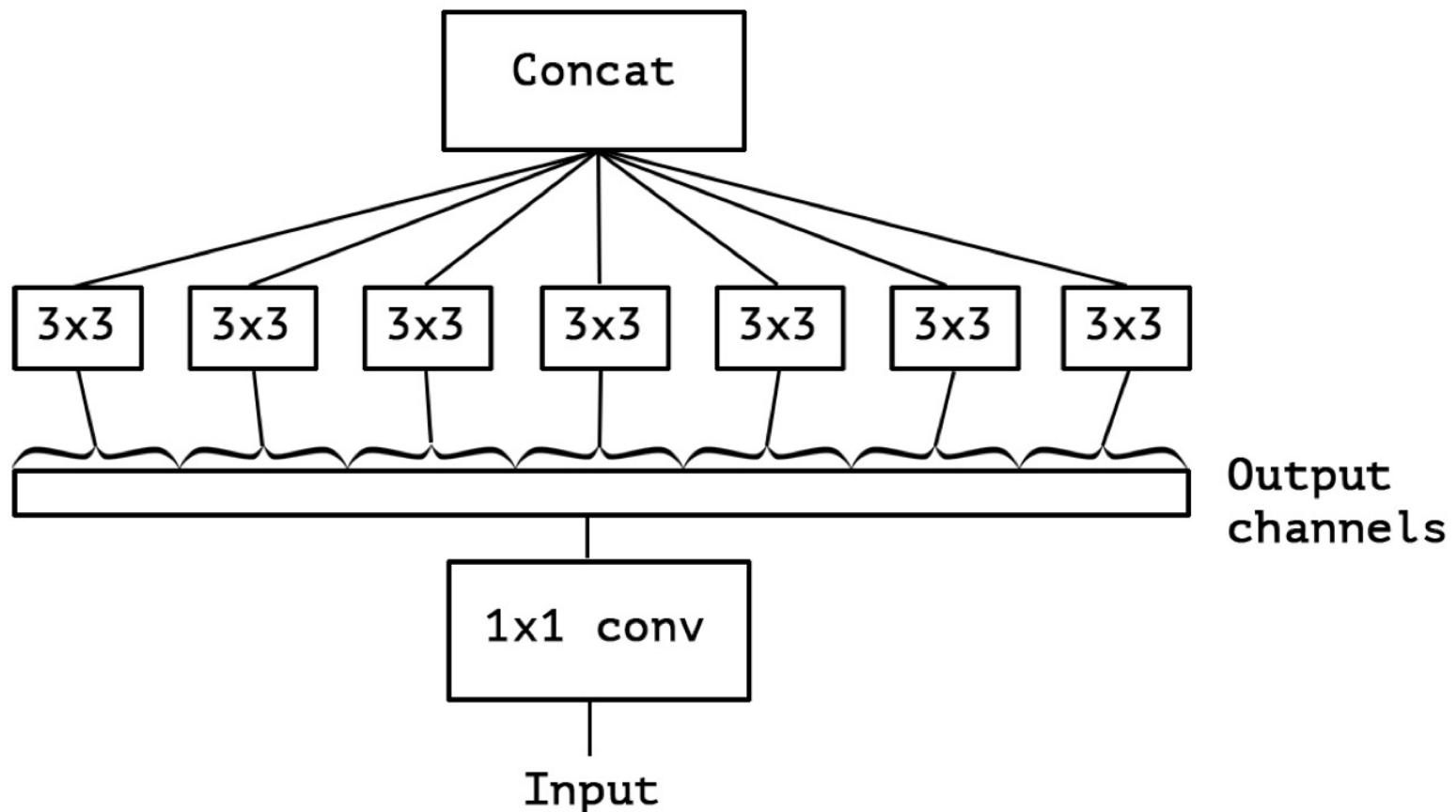
ResNet (2015)

- Даёт низкую ошибку на обучении даже с 1000 слоёв (но там плохо на тестовой выборке)
- Обучается градиентным спуском с инерцией со случайной инициализацией
- Ошибка 4.49% на ImageNet

Эволюция архитектур



Xception



Xception

- Разделяется роль свёрток: либо по каналам, либо по пространству
- Более эффективное использование параметров

Что ещё?

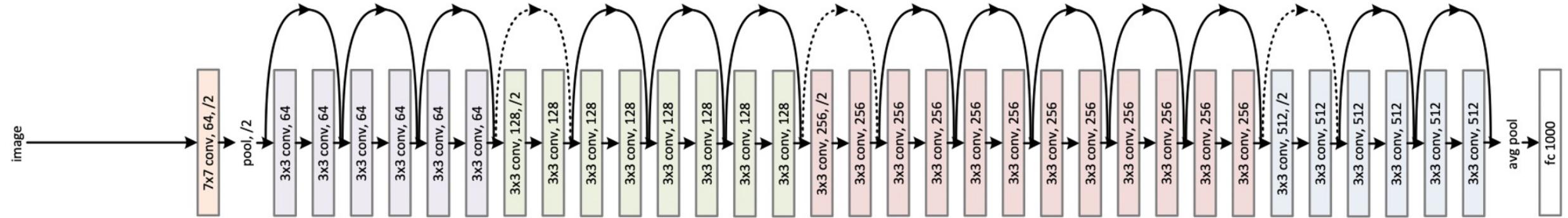
- Highway networks
- Inception-ResNet
- Squeeze and Excitation Network
- MobileNet
- EfficientNet
- ...

Transfer learning

Перенос знаний

- ImageNet:
 - Много данных (которые сложно собрать!)
 - Годы улучшений
- Не хотелось бы повторять это для новых задач

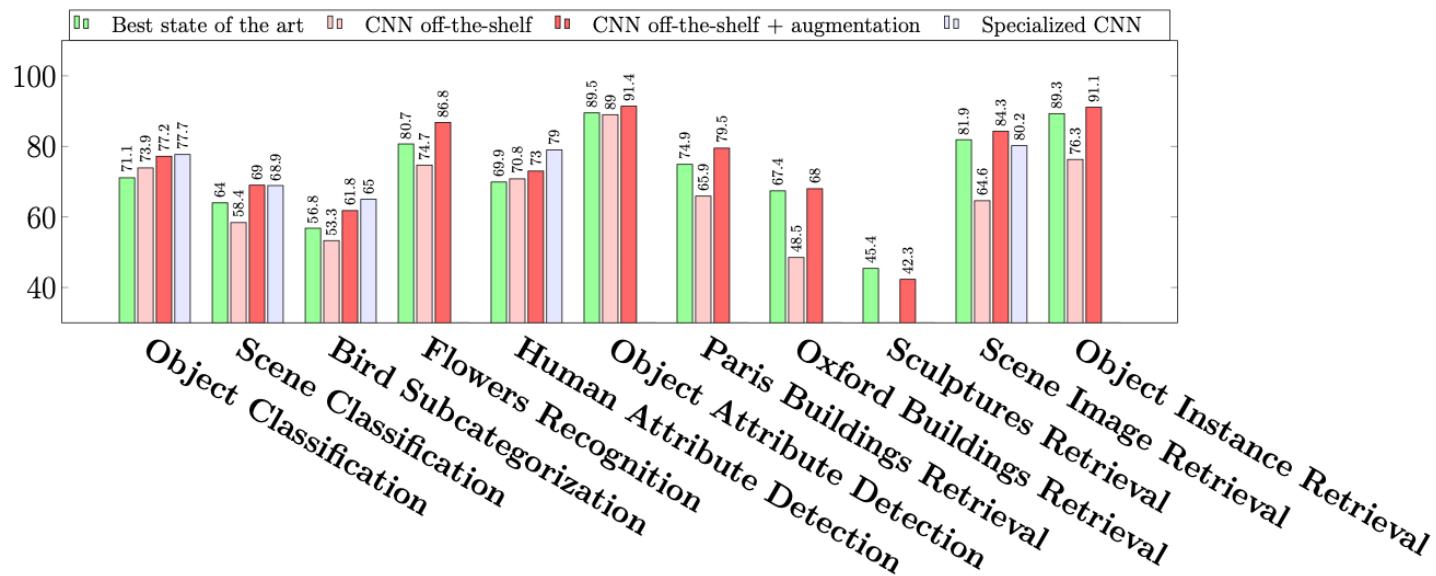
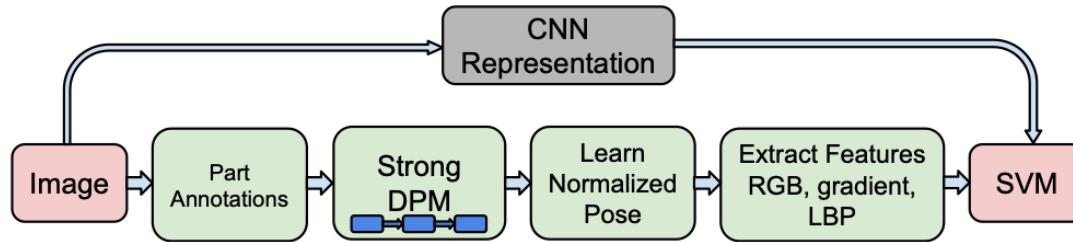
Дообучение



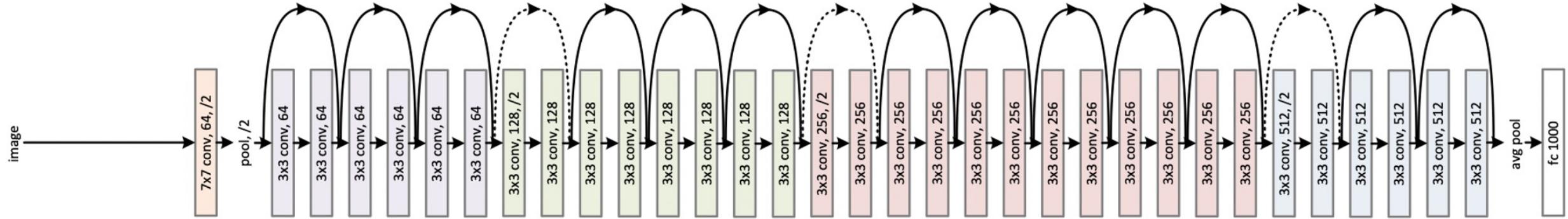
Если данных совсем мало:

- Берём модель из другой задачи
- Заменяем последний слой на слой с нужным числом выходов
- Обучаем только его
- По сути, это обучение линейной модели

Представления с последних слоёв



Дообучение



Если данных не очень мало:

- Берём модель из другой задачи
 - Заменяем последний слой на слой с нужным числом выходов
 - Обучаем его и несколько слоёв до него
 - Чем ближе к началу слой, тем ниже стоит делать градиентный шаг

Дообучение

- Как правило, на первых слоях фильтры похожие для всех задач
- Чем сильнее новая задача отличается от исходной, тем больше слоёв нужно переучивать
- В любом случае выходы последних слоёв модели, обученной на ImageNet, лучше случайных признаков

Интерпретация моделей

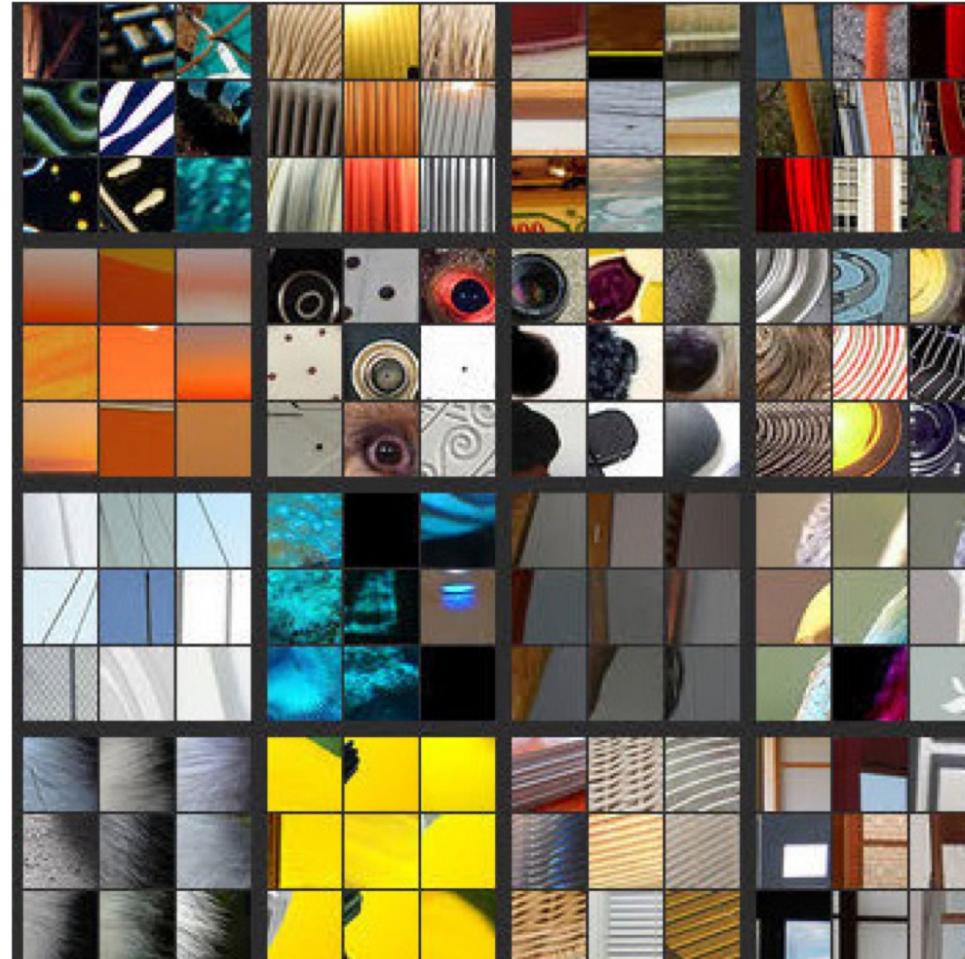
Что находят свёрточные сети?

- Можно для каждого фильтра найти кусочки картинок, дающие самый сильный отклик
- Сделаем это для AlexNet

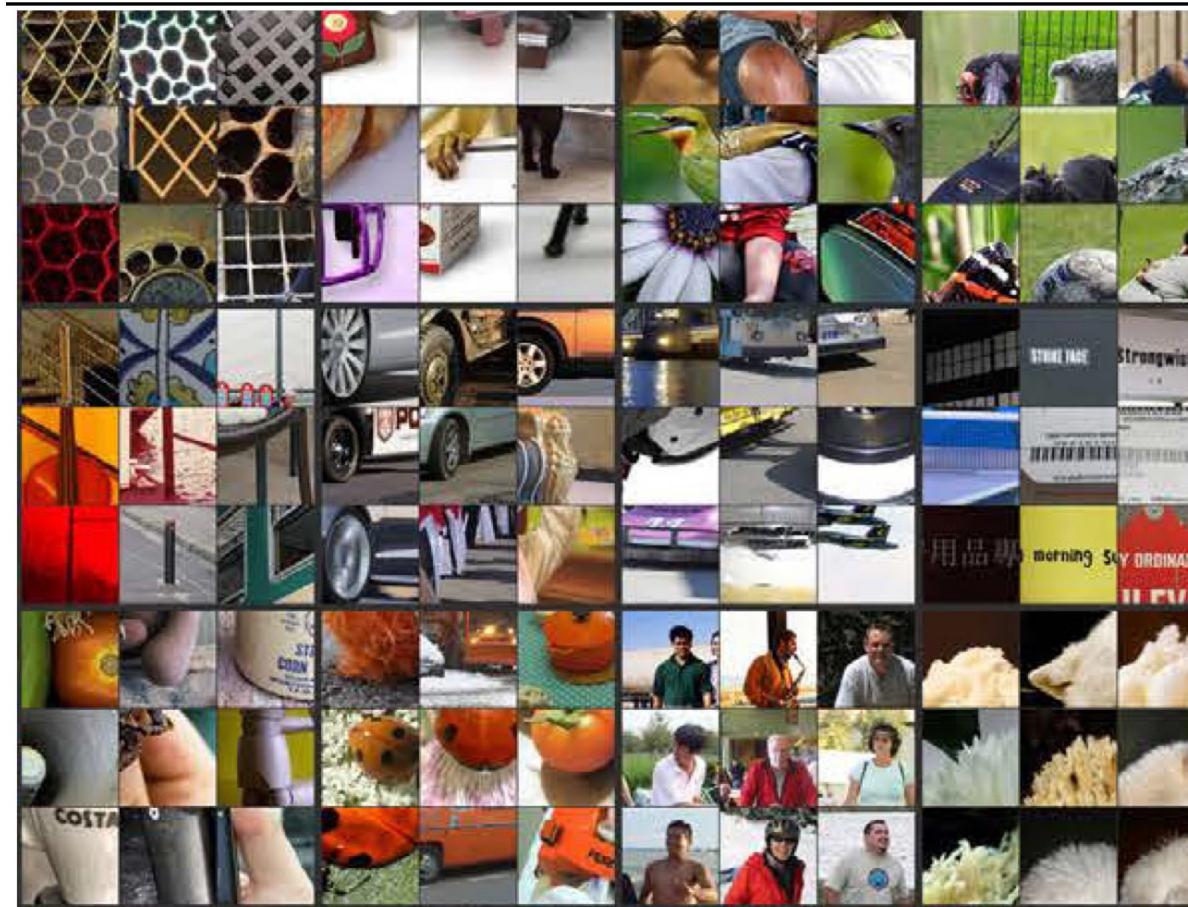
Слой 1



Слой 2



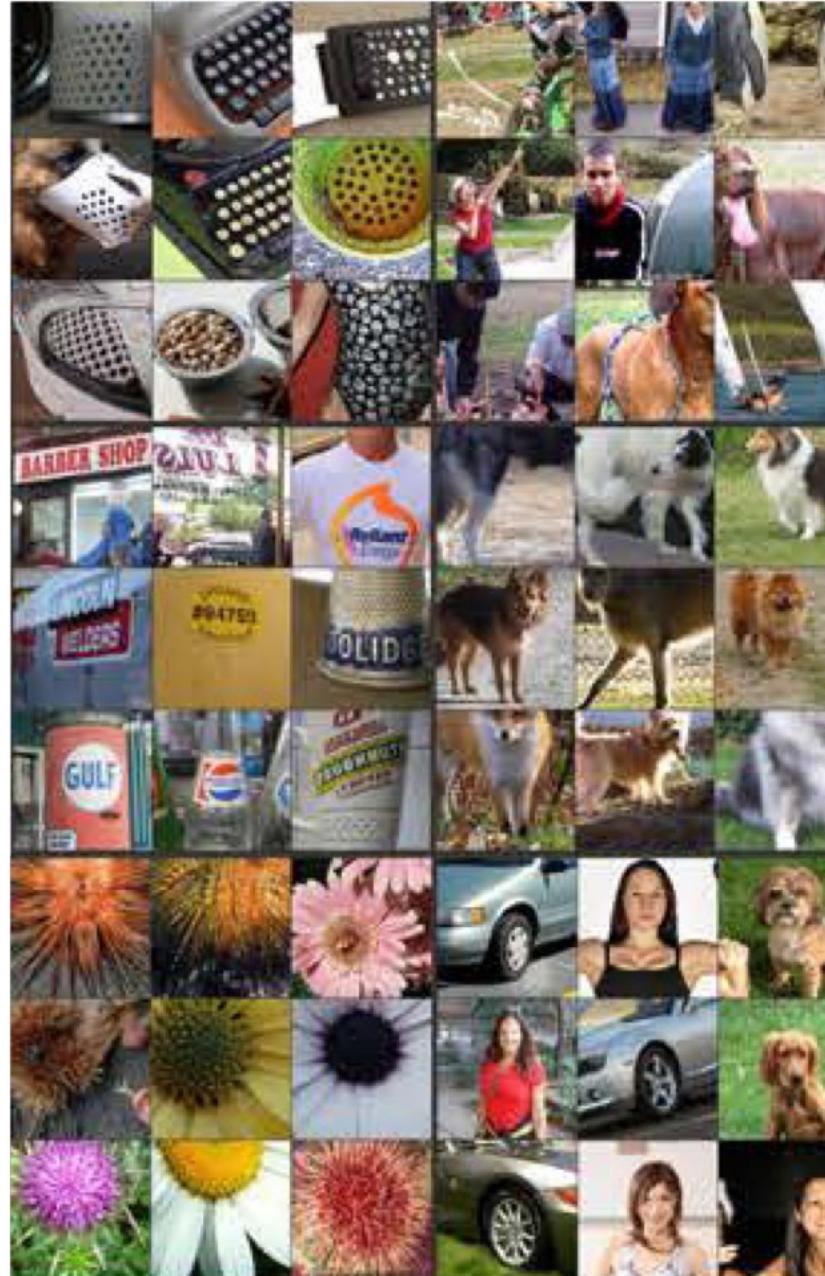
Слой 3



Слой 4



Слой 5

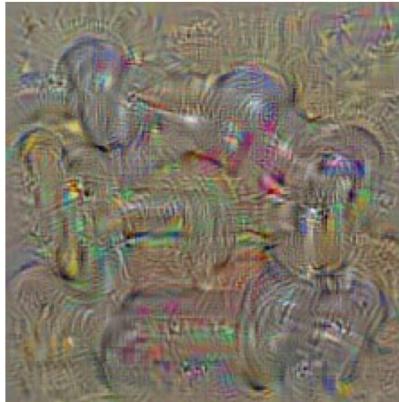


Максимизация вероятности класса

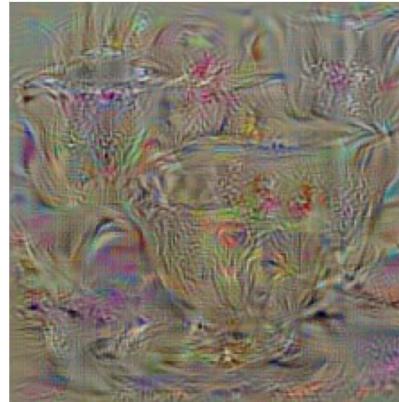
$$a_y(x) - \lambda \|x\|_2^2 \rightarrow \max_x$$

- Инициализируем картинку случайным шумом
- Ищем оптимальную для данного класса картинку градиентным спуском

Максимизация вероятности класса



dumbbell



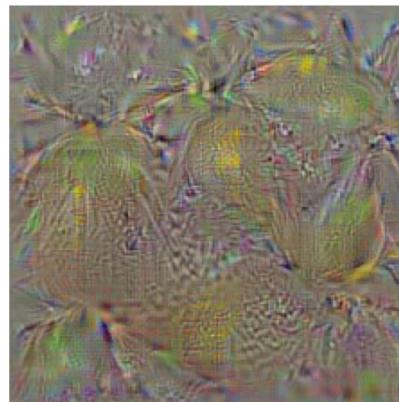
cup



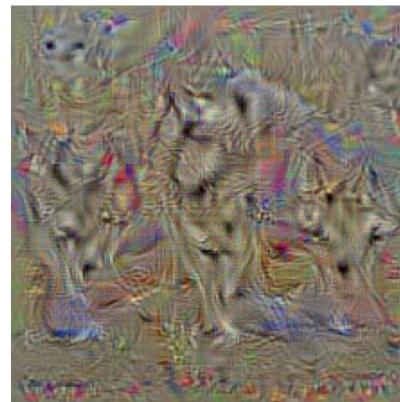
dalmatian



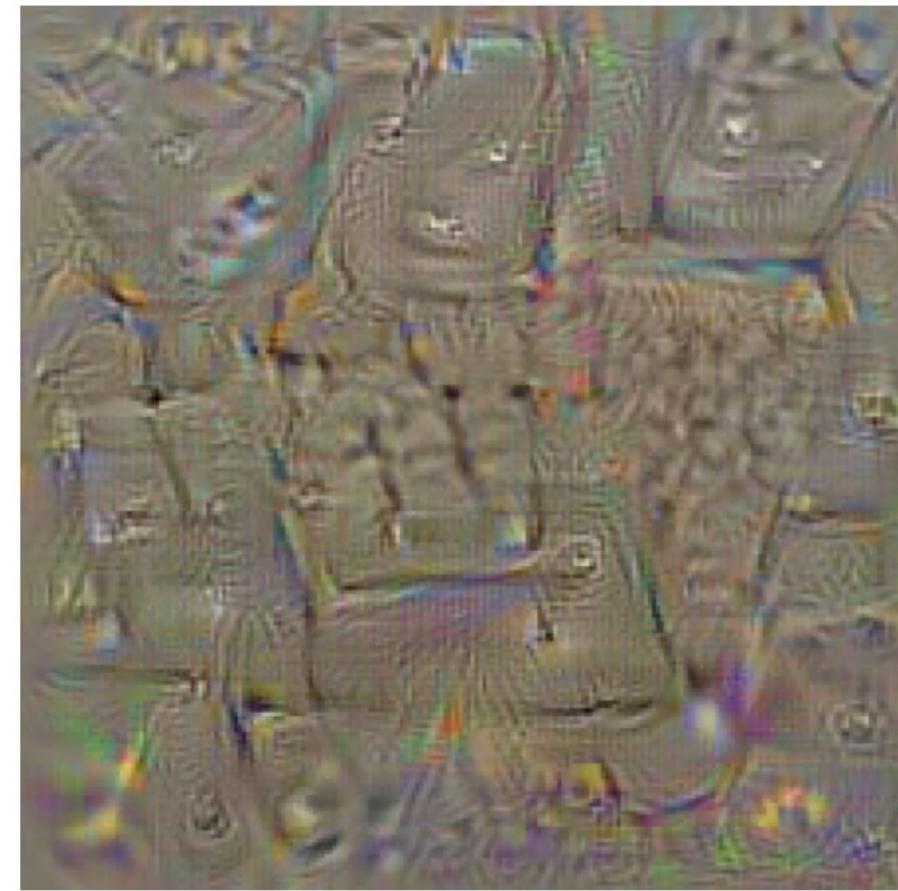
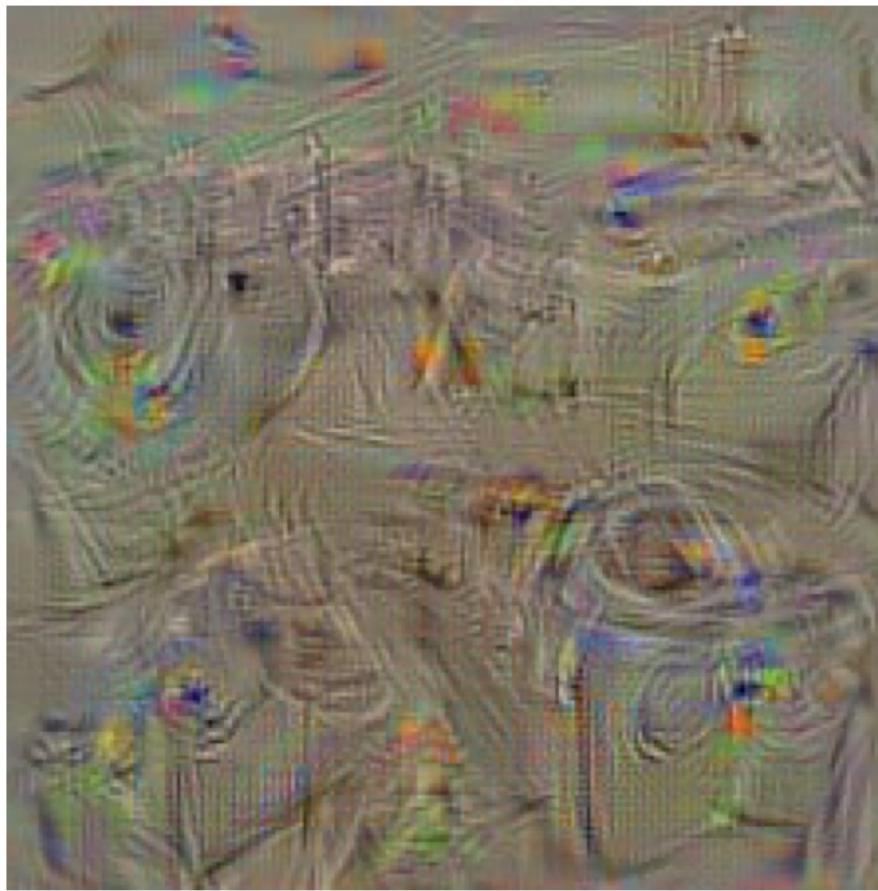
bell pepper



lemon



husky



Максимизация вероятности класса



Hartebeest



Measuring Cup



Ant



Starfish



Anemone Fish



Banana

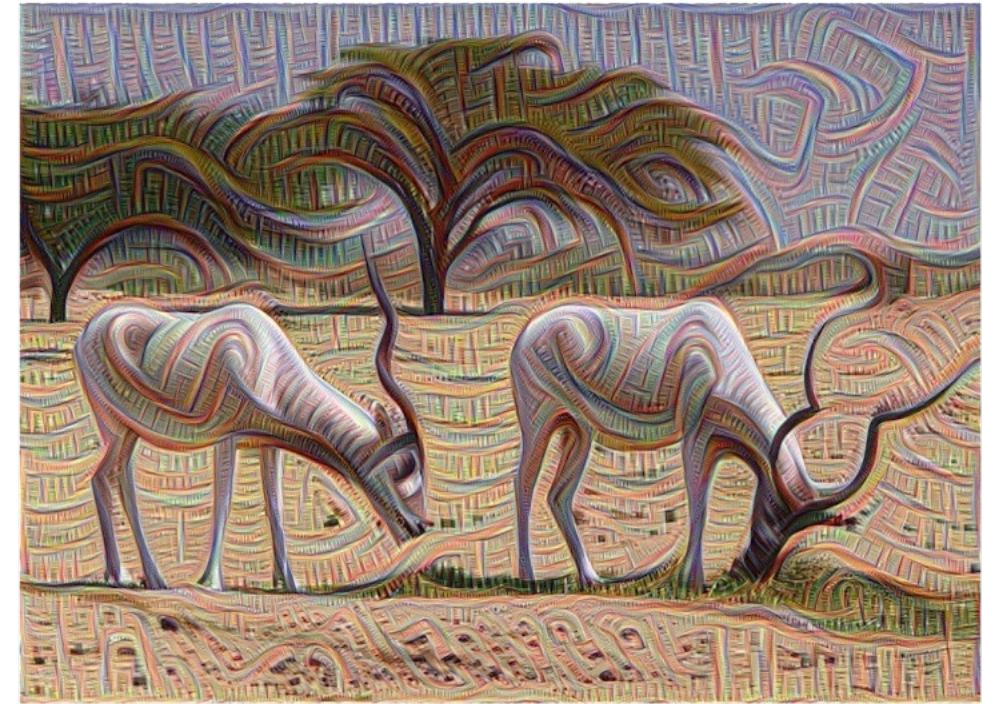


Parachute

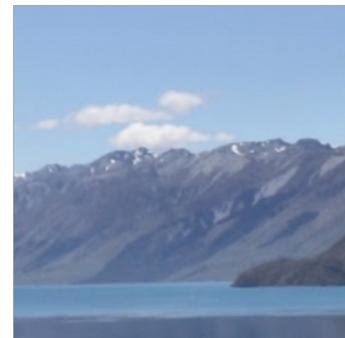


Screw

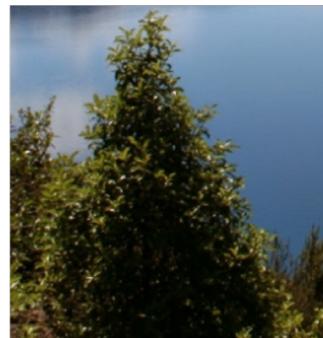
Максимизация вероятности класса



Максимизация вероятности класса



Horizon



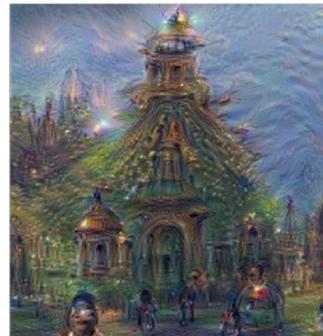
Trees



Leaves



Towers & Pagodas



Buildings



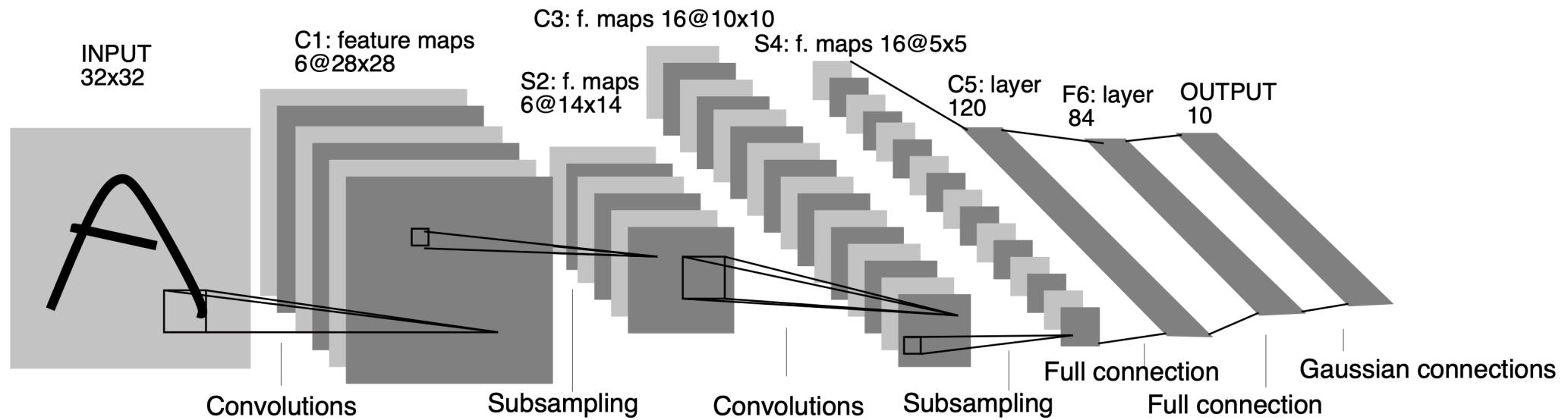
Birds & Insects

Максимизация вероятности класса



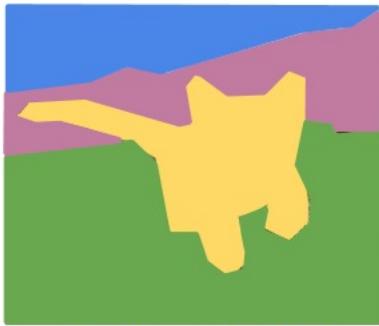
Компьютерное зрение

Классификация изображений



Обычно хочется другого

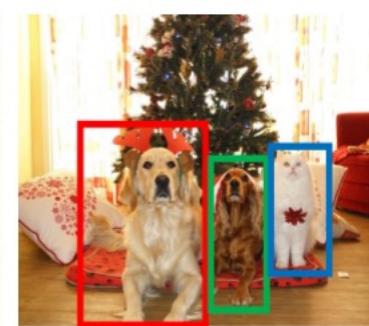
Semantic
Segmentation



Classification
+ Localization



Object
Detection



Instance
Segmentation

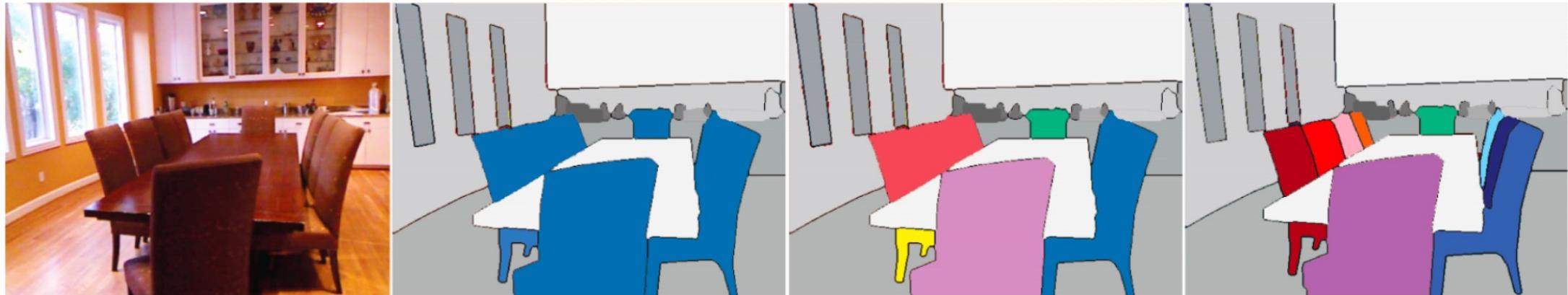


Обычно хочется другого

Но мы будем сводить все задачи к классификации!

Семантическая сегментация

Терминология



Semantic
segmentation

Instance
segmentation

Постановка задачи

- Данные: изображения и их корректные сегментации
- Пример: PASCAL VOC 2012



Постановка задачи

Table 2: Statistics of the segmentation image sets.

	train		val		trainval		test	
	img	obj	img	obj	img	obj	img	obj
Aeroplane	88	108	90	110	178	218	—	—
Bicycle	65	94	79	103	144	197	—	—
Bird	105	137	103	140	208	277	—	—
Boat	78	124	72	108	150	232	—	—
Bottle	87	195	96	162	183	357	—	—
Bus	78	121	74	116	152	237	—	—
Car	128	209	127	249	255	458	—	—
Cat	131	154	119	132	250	286	—	—
Chair	148	303	123	245	271	548	—	—
Cow	64	152	71	132	135	284	—	—
Diningtable	82	86	75	82	157	168	—	—
Dog	121	149	128	150	249	299	—	—
Horse	68	100	79	104	147	204	—	—
Motorbike	81	101	76	103	157	204	—	—
Person	442	868	445	865	887	1733	—	—
Pottedplant	82	151	85	171	167	322	—	—
Sheep	63	155	57	153	120	308	—	—
Sofa	93	103	90	106	183	209	—	—
Train	83	96	84	93	167	189	—	—
Tvmonitor	84	101	74	98	158	199	—	—
Total	1464	3507	1449	3422	2913	6929	—	—

Постановка задачи

- Каждый объект содержит сильно больше информации, чем при классификации!
- Можно хорошо обучаться на небольших выборках

Метрики качества

- Попиксельная доля верных ответов:

$$L(y, a) = \frac{1}{n} \sum_{i=1}^n [y_i = a_i]$$

- Мера Жаккара (считается отдельно для каждого класса):

$$J_k(y, a) = \frac{\sum_{i=1}^n [y_i = k][a_i = k]}{\sum_{i=1}^n \max([y_i = k], [a_i = k])}$$

(можно усреднить по всем классам)

ФУНКЦИЯ ПОТЕРЬ

- Для одного изображения:

$$L(y, a) = \sum_{i=1}^n \sum_{k=1}^K [y_i = k] \log a_{ik}$$

сумма по пикселям

сумма по классам

истинный класс в i -м пикселе

вероятность k -го класса в i -м пикселе
(из модели)

ФУНКЦИЯ ПОТЕРЬ

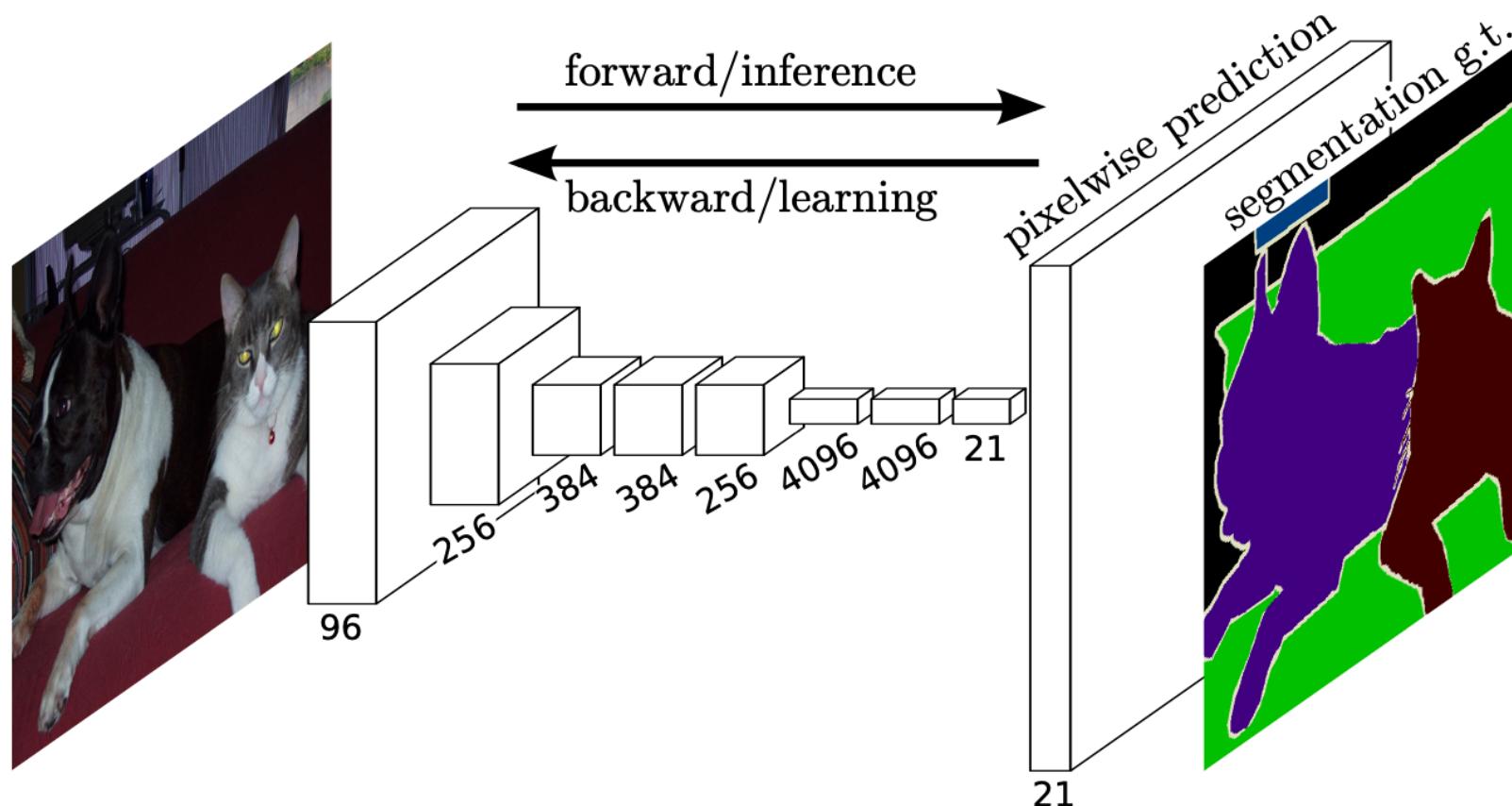
- Для одного изображения (categorical cross-entropy, CCE):

$$L(y, a) = \sum_{i=1}^n \sum_{k=1}^K [y_i = k] \log a_{ik}$$

- Если модель в i -м пикселе выдаёт какие-то числа b_{i1}, \dots, b_{iK} , то их можно превратить в вероятности через softmax:

$$a_{ik} = \frac{\exp(b_{ik})}{\sum_{m=1}^K \exp(b_{im})}$$

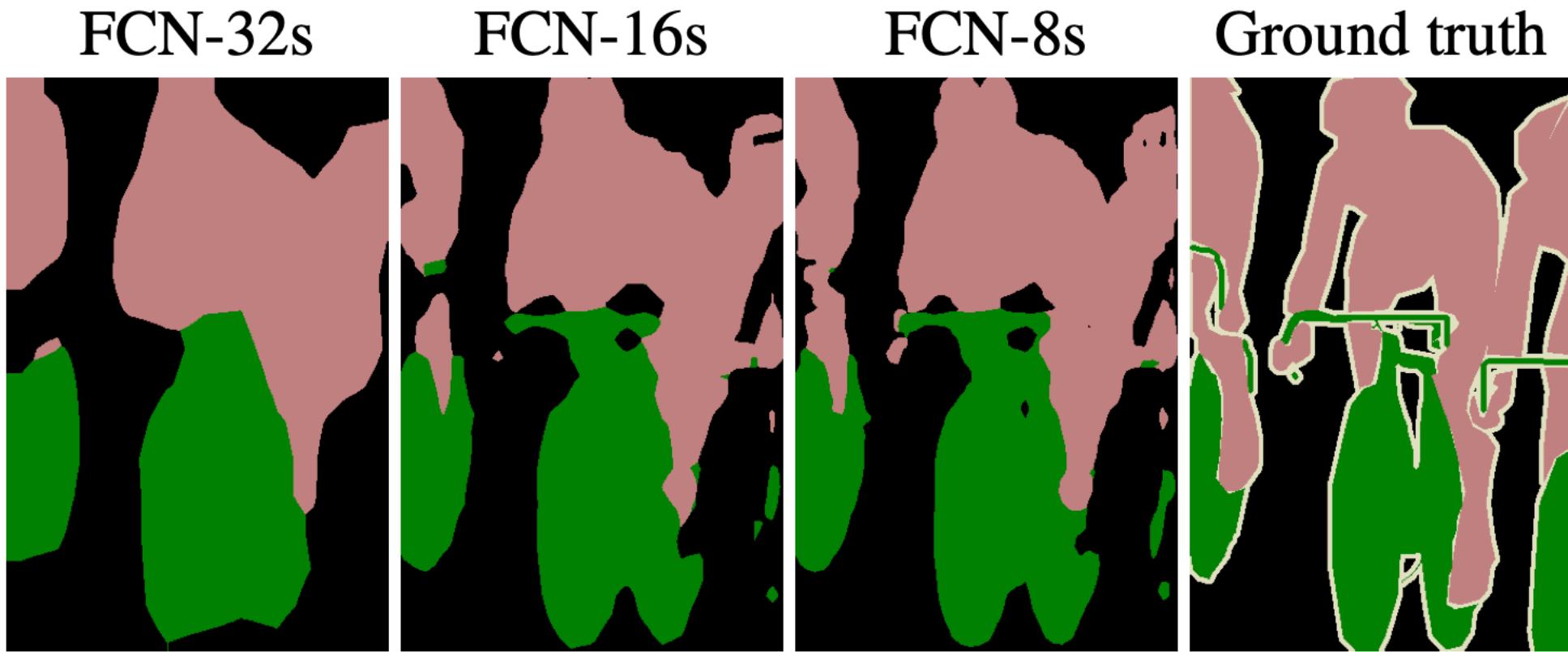
Fully Convolutional Network



Fully Convolutional Network

- Убираем полносвязные слои
- Остаются только свёртки
- Тензор с последнего слоя преобразуем свёртками 1×1 в тензор такого же размера, но с K каналами (по числу классов)
- Повышаем разрешение
 - Можно простым размножением пикселей
 - Можно хитрее

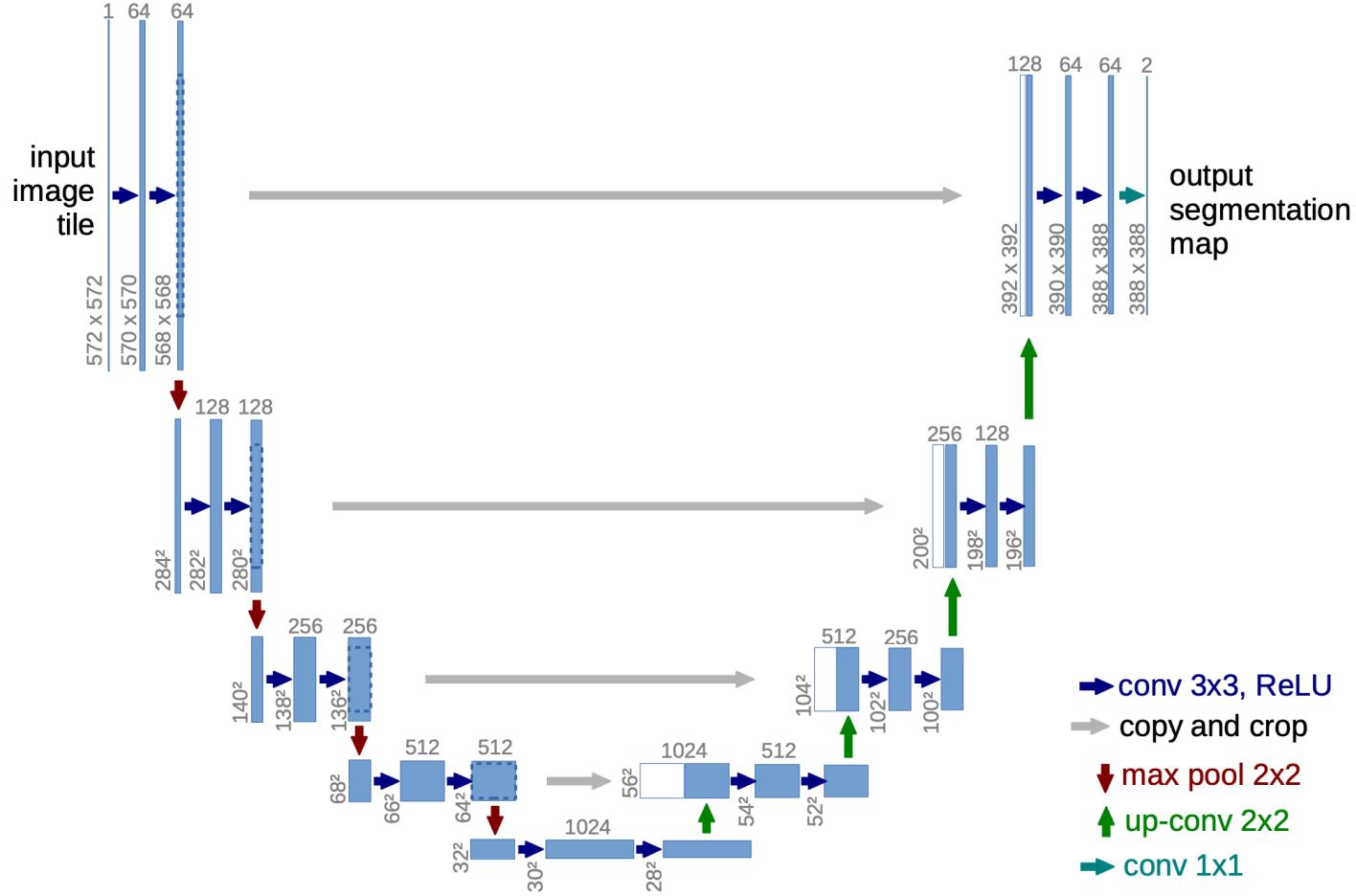
Fully Convolutional Network



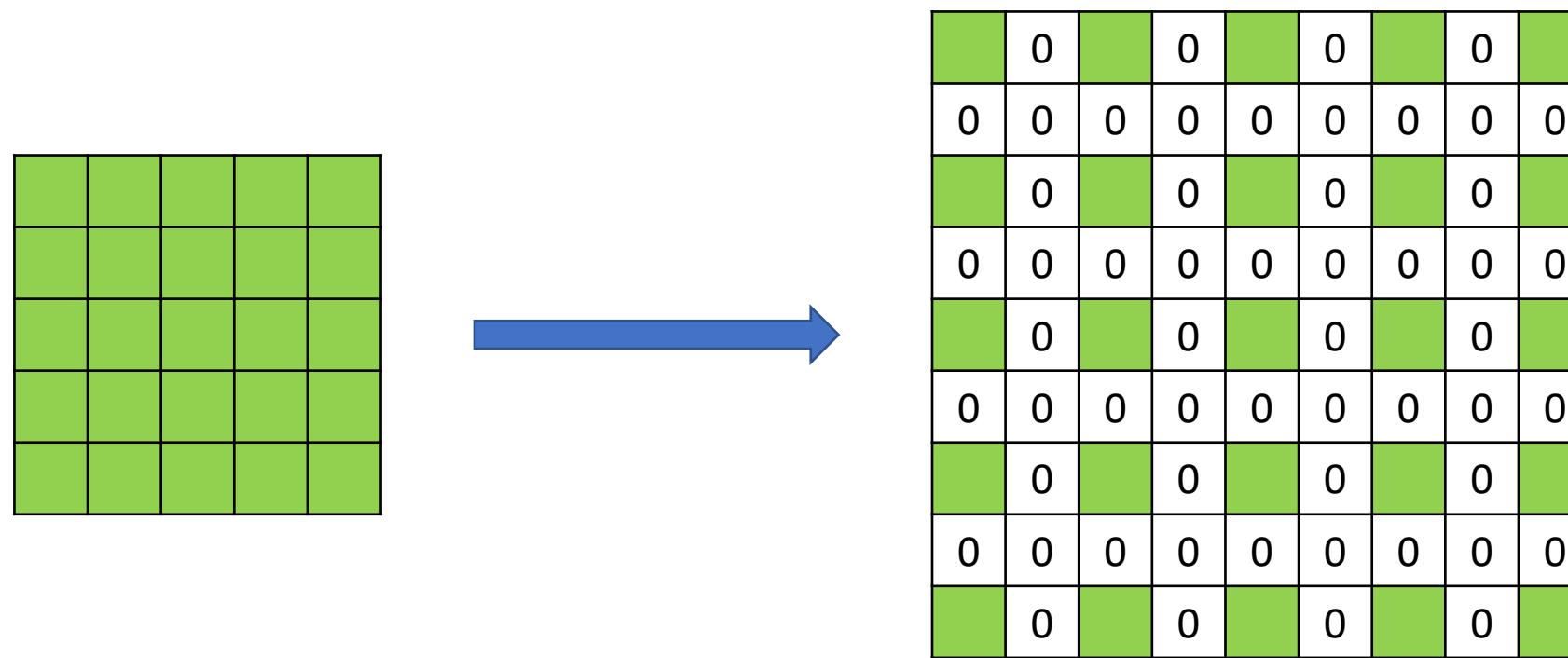
Чем это плохо?

- Тензор на последнем слое довольно маленький
- Классы предсказываются грубо, у объектов нечёткие границы
- Можно делать меньше пулингов
- Но тогда будут проблемы с размером поля восприятия

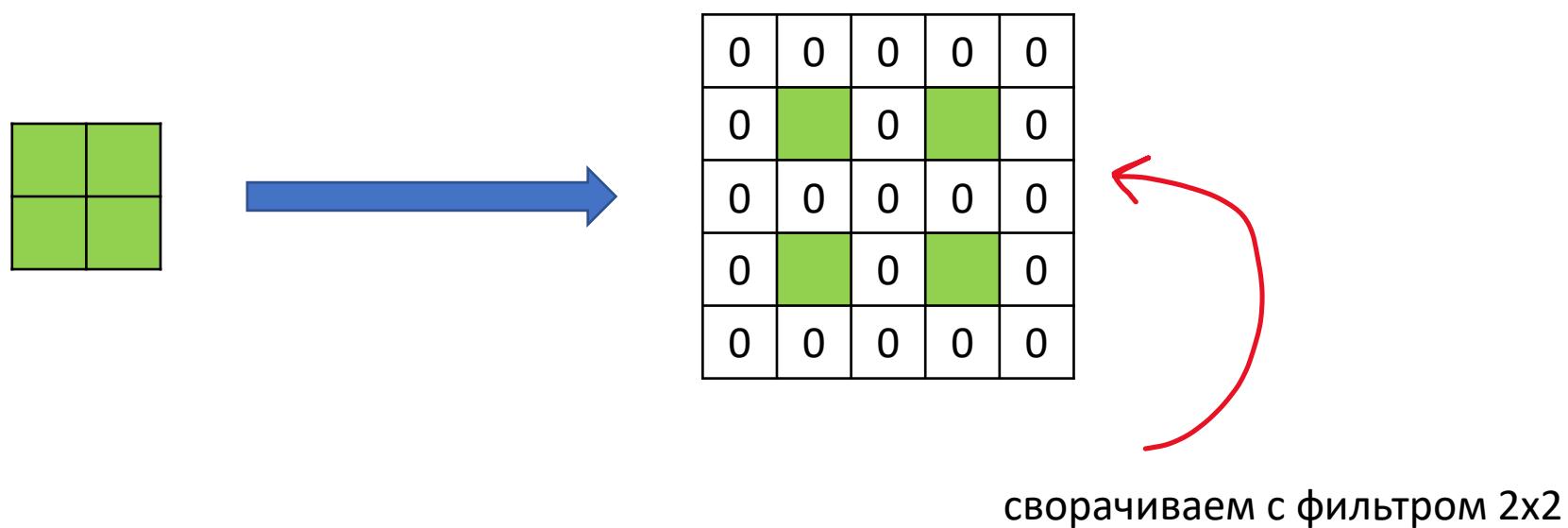
U-Net



Upsampling, вариант 1 (bed of nails)



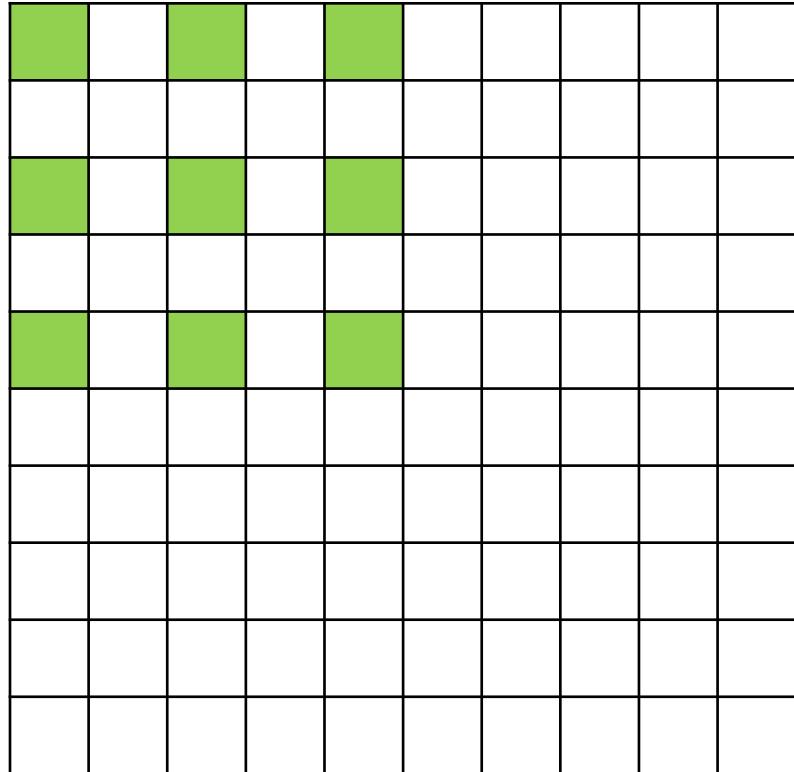
Upsampling, вариант 2 (up-convolution)



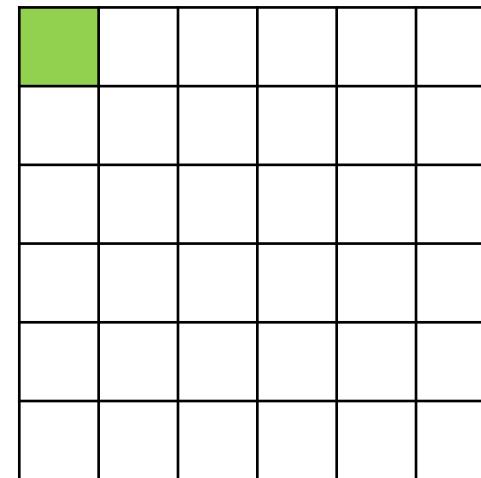
U-Net

- «Декодировщик» имеет очень большое поле восприятия
- Главное отличие от FCN — копирование слоёв между ветками сети

Dilated convolutions («раздутые» свёртки)

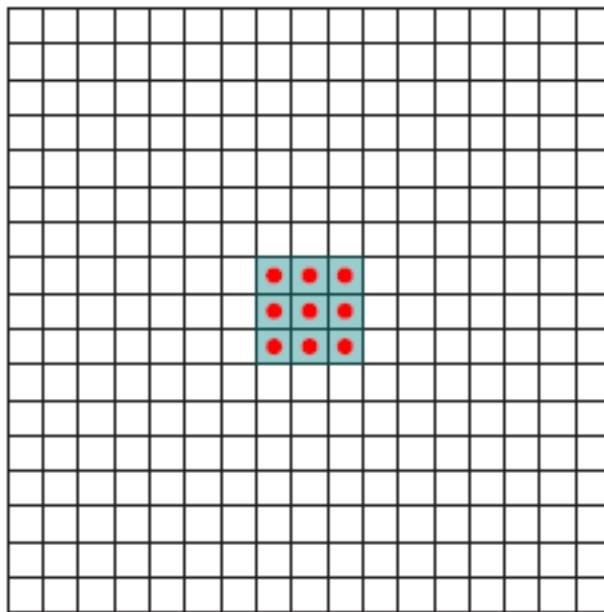


$$\begin{matrix} * & \begin{matrix} \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \end{matrix} & = \end{matrix}$$

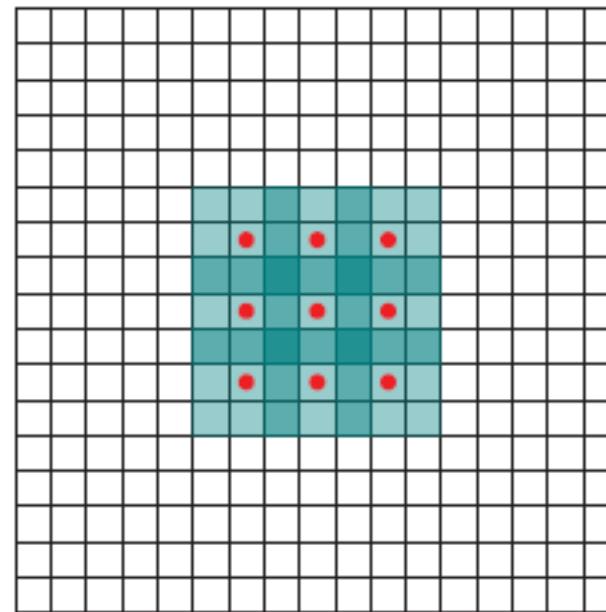


$$l = 2$$

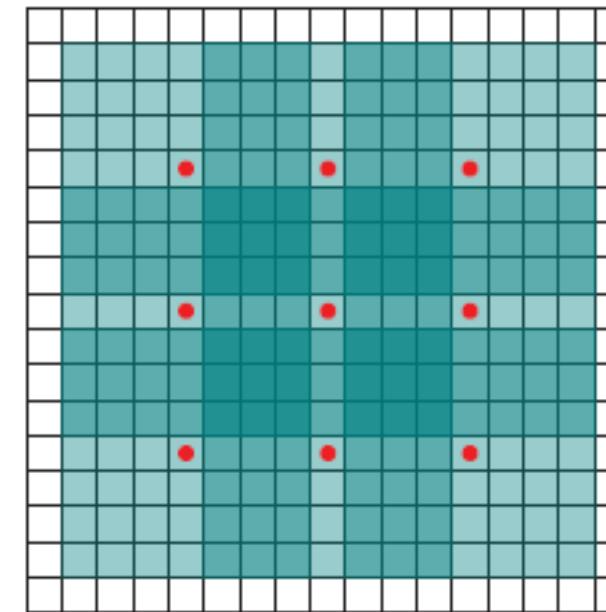
Dilated convolutions («раздутые» свёртки)



$$l = 1$$



$$l = 2$$



$$l = 4$$

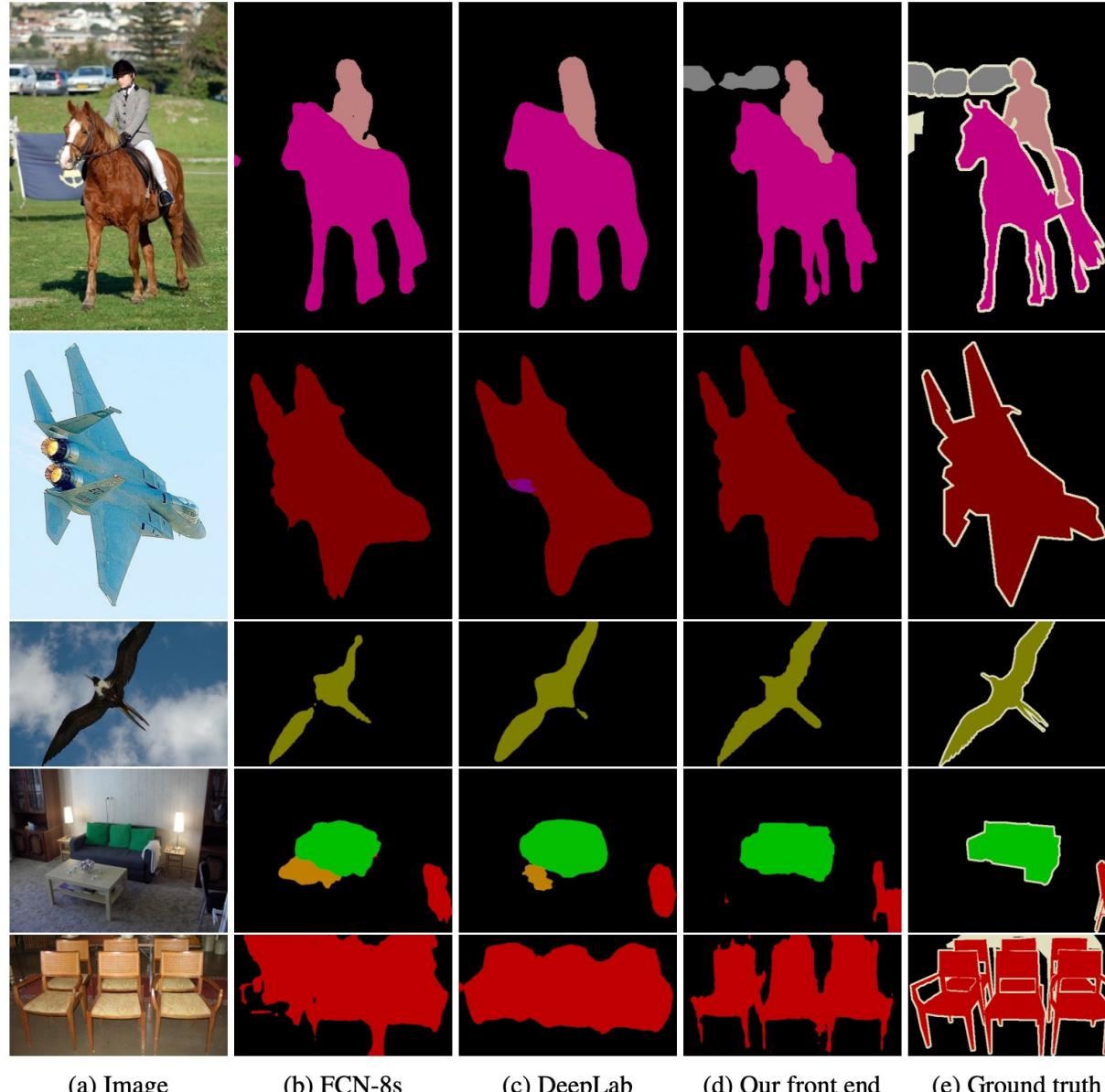
Dilated convolutions

Layer	1	2	3	4	5	6	7	8
Convolution	3×3	3×3	3×3	3×3	3×3	3×3	3×3	1×1
Dilation	1	1	2	4	8	16	1	1
Truncation	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Receptive field	3×3	5×5	9×9	17×17	33×33	65×65	67×67	67×67
Output channels								
Basic	C	C	C	C	C	C	C	C
Large	$2C$	$2C$	$4C$	$8C$	$16C$	$32C$	$32C$	C

Table 1: Context network architecture. The network processes C feature maps by aggregating contextual information at progressively increasing scales without losing resolution.

Dilated convolutions

- Можем сохранять размер тензора и при этом увеличивать поле восприятия



(a) Image

(b) FCN-8s

(c) DeepLab

(d) Our front end

(e) Ground truth