

Основы глубинного обучения

Лекция 5
Архитектуры свёрточных сетей

Евгений Соколов

esokolov@hse.ru

НИУ ВШЭ, 2020

Инициализация весов

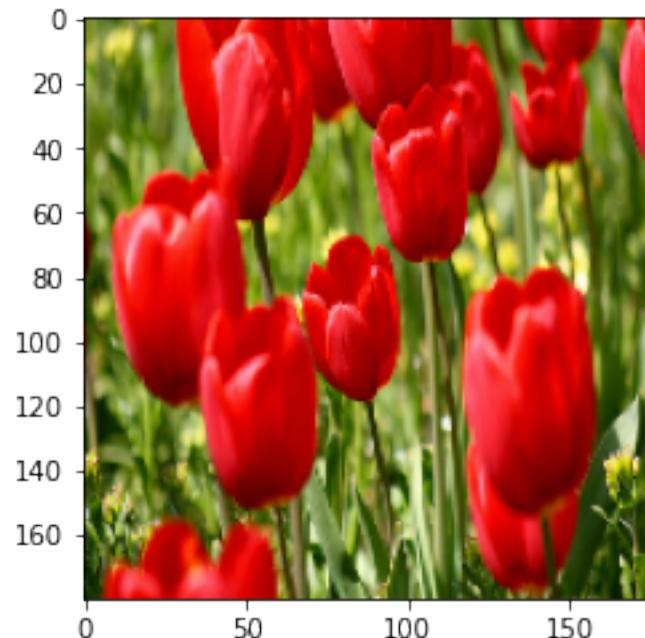
- Не должно быть симметрий (плохо инициализировать всё одним числом)
- Хороший вариант:

$$w_j \sim \frac{2}{\sqrt{n}} \mathcal{N}(0, 1)$$

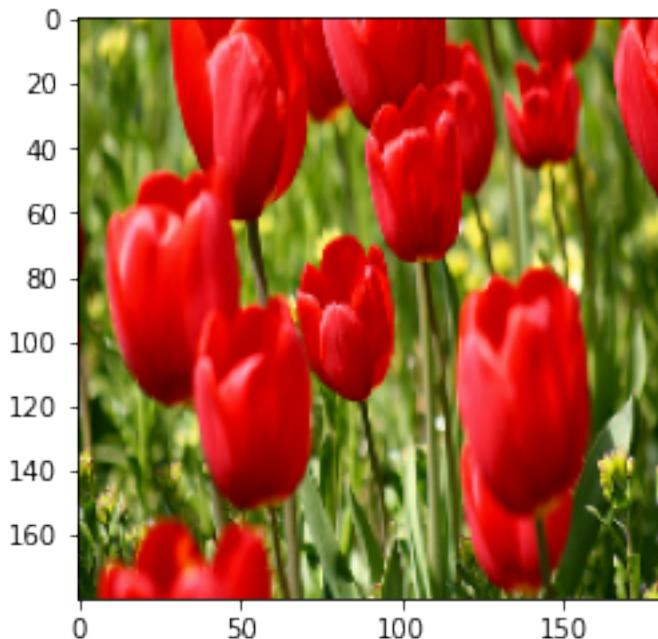
n — число входов

- Пытаемся сделать так, чтобы масштаб всех выходов был примерно одинаковым

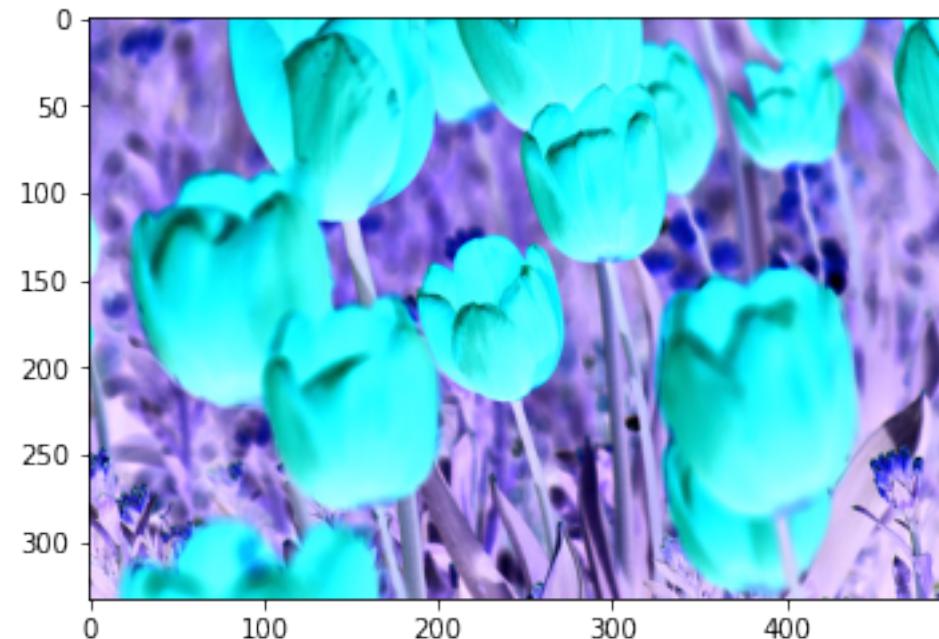
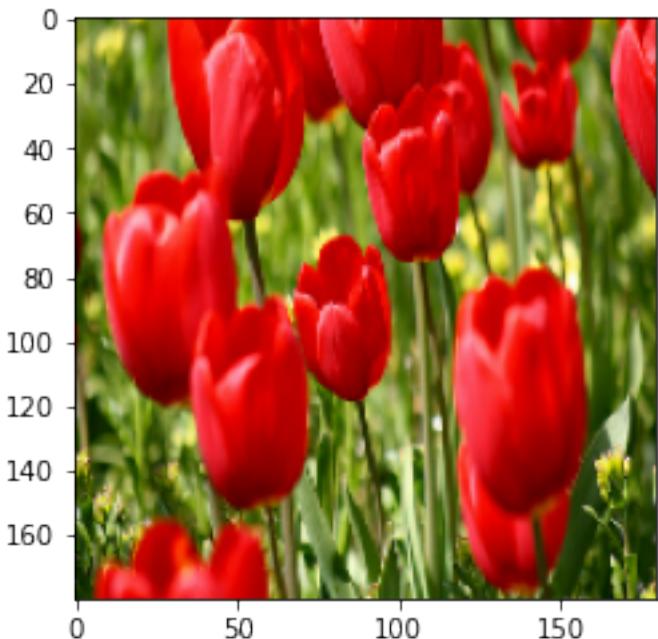
Аугментация



Аугментация



Аугментация



https://www.tensorflow.org/tutorials/images/data_augmentation



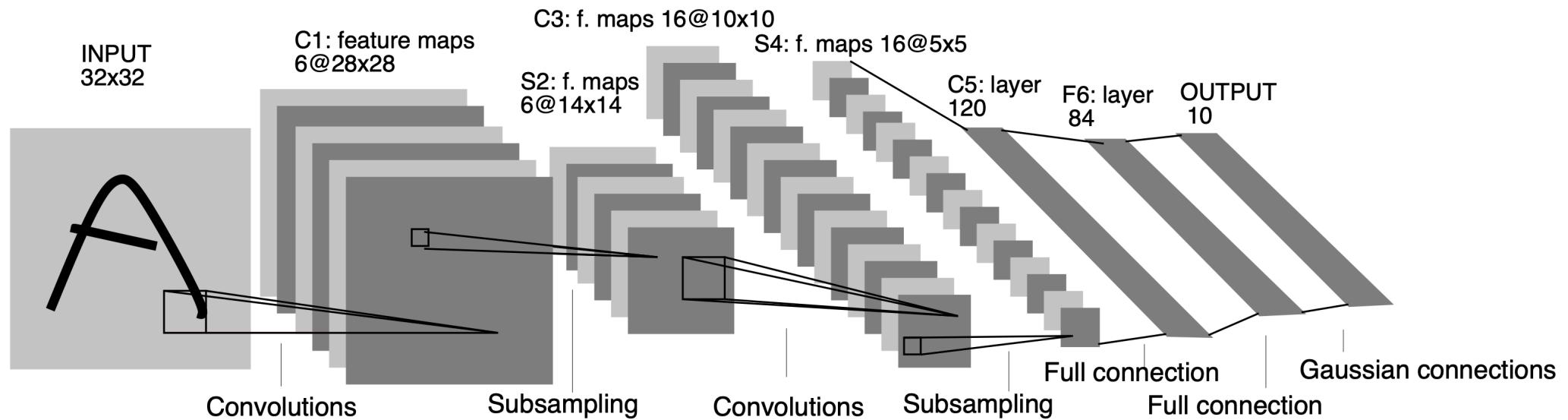
Аугментация

- Много разных вариантов
- «Бесплатное» расширение обучающей выборки
- В некотором смысле регуляризация модели

- Обычно аугментации случайно применяют к картинкам из текущего батча
- На этапе применения можно сделать несколько аугментаций картинки, применить сеть к каждой, усреднить предсказания

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)

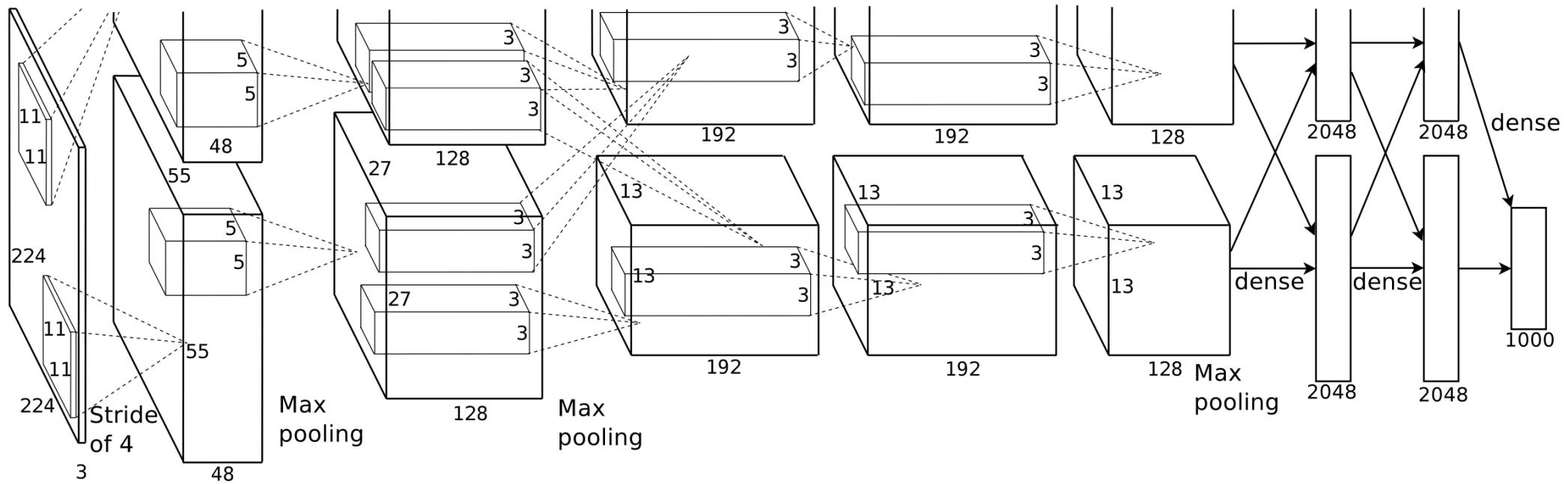
ImageNet Classification with Deep Convolutional Neural Networks

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
- Градиентный спуск с инерцией
- Dropout для двух первых полно связных слоёв
- Хитрая инициализация (сначала обучается вариант A 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

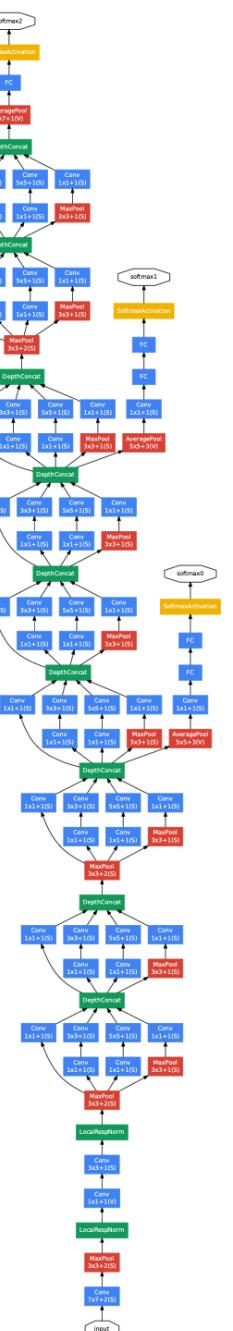
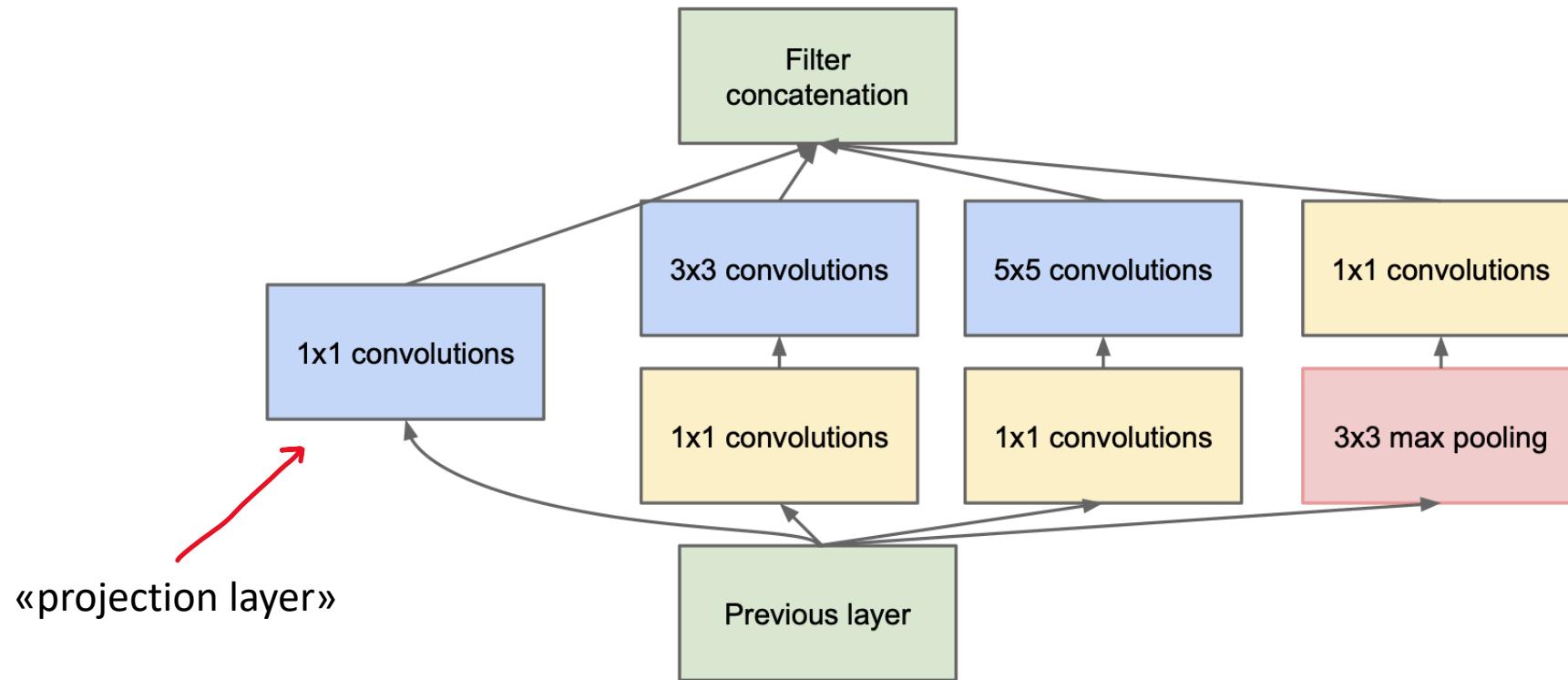


Figure 3: GoogLeNet network with all the bells and whistles.

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)

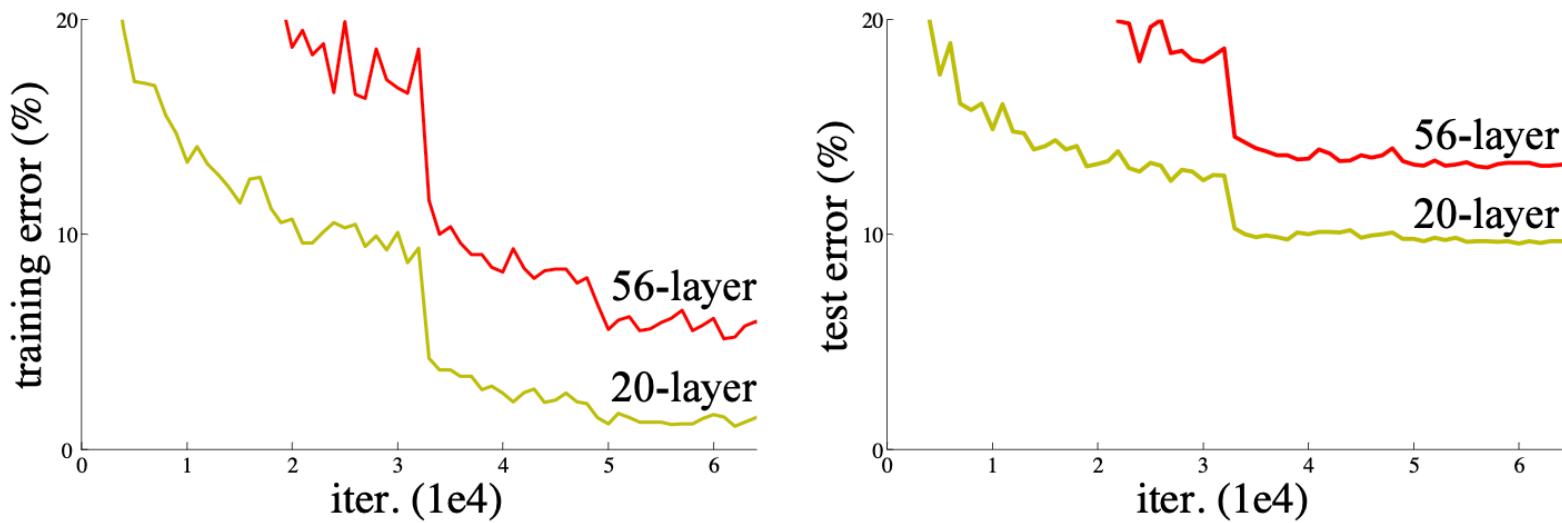
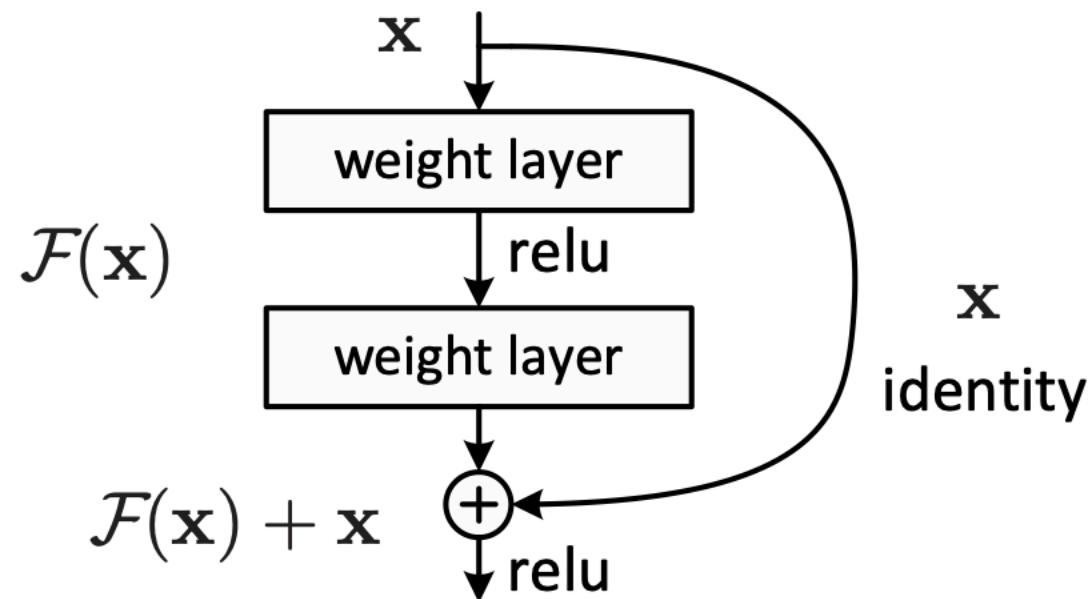


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer “plain” networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

ResNet (2015)

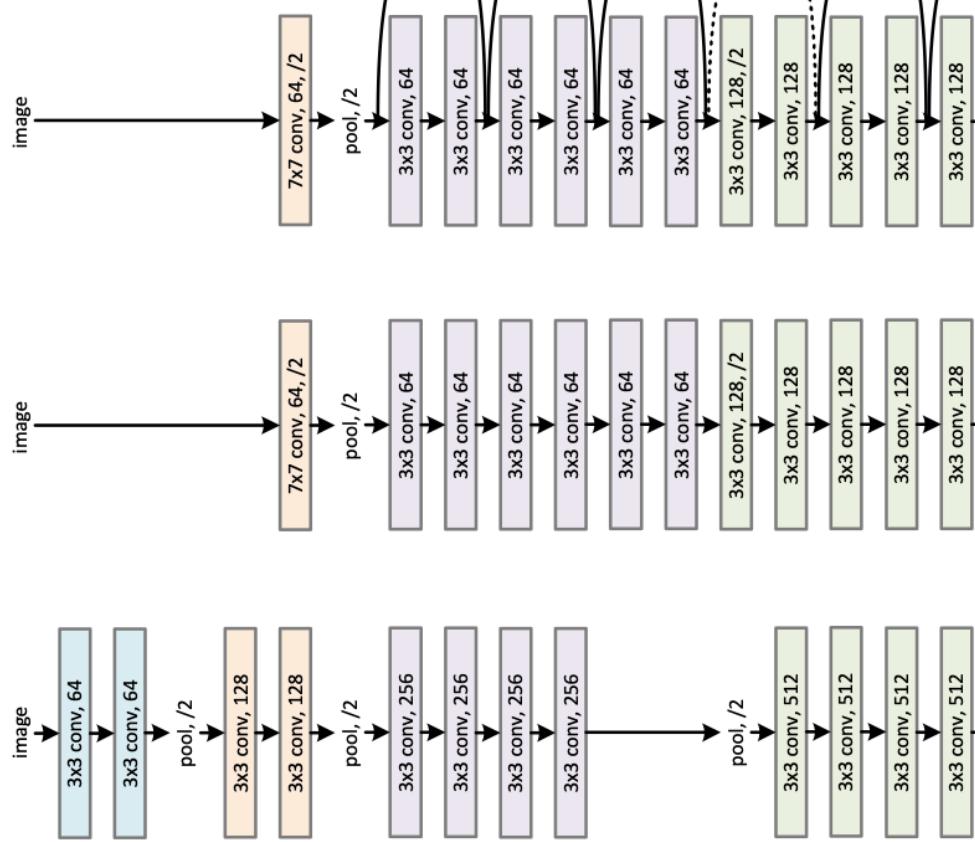
- Добавление слоёв в свёрточную сеть ухудшает качество даже на обучении
- Хотя возможностей для переобучения больше, сеть почему-то не может ими воспользоваться

ResNet (2015)

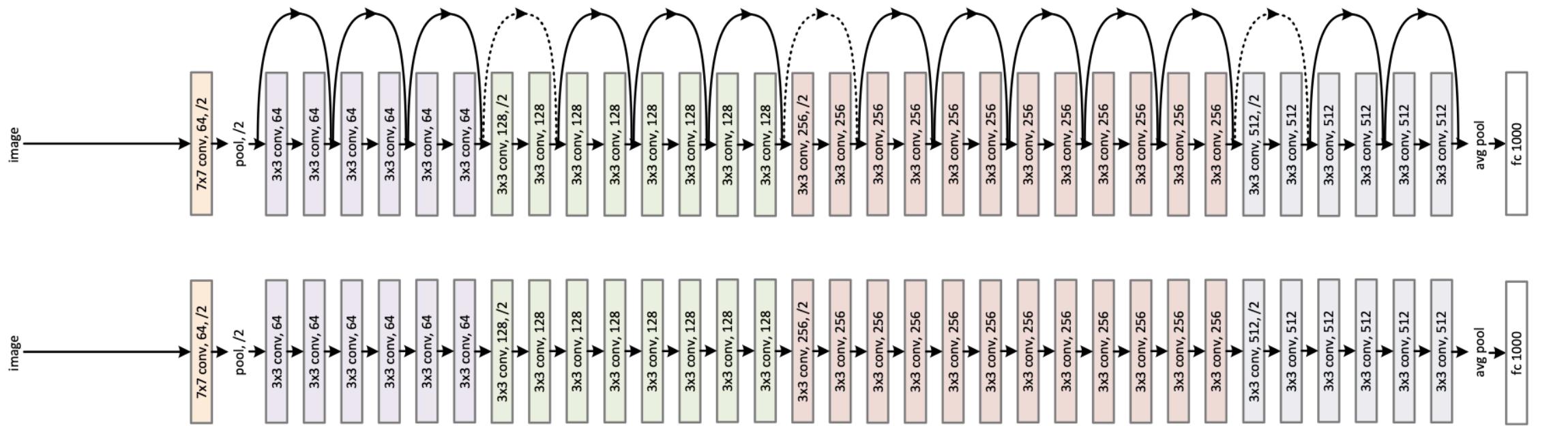


ResNet (2015)

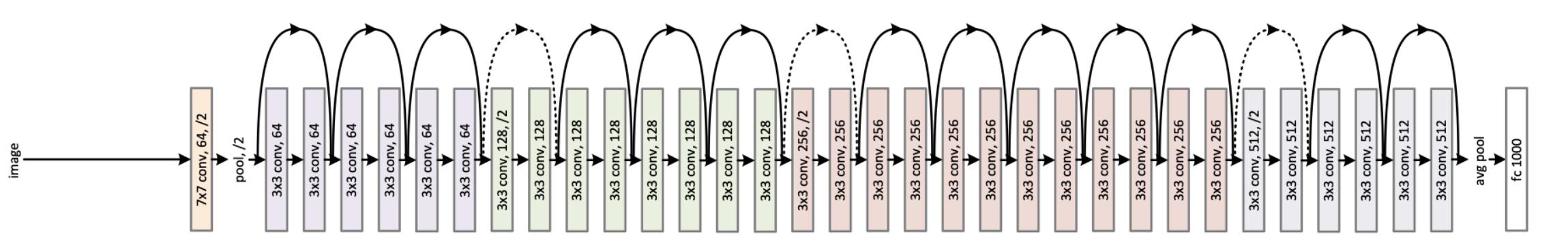
VGG-19



34-layer plain



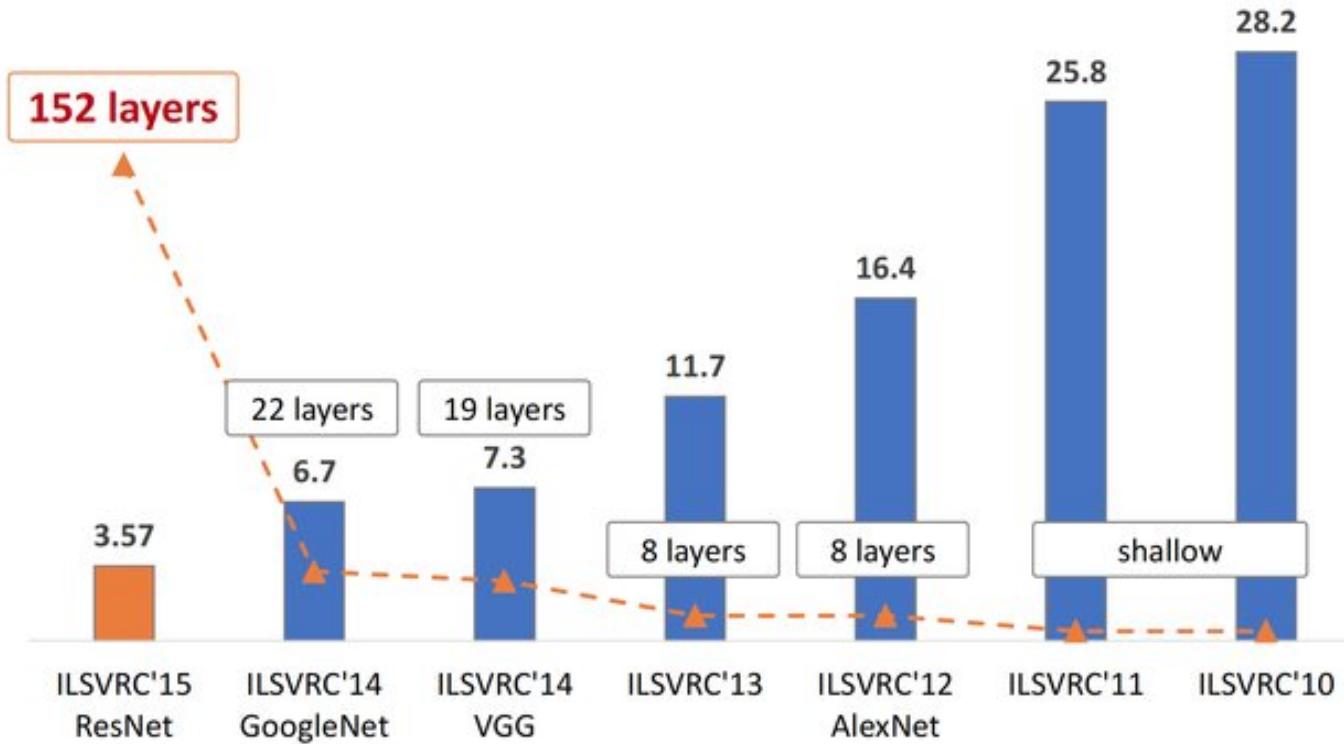
34-layer residual



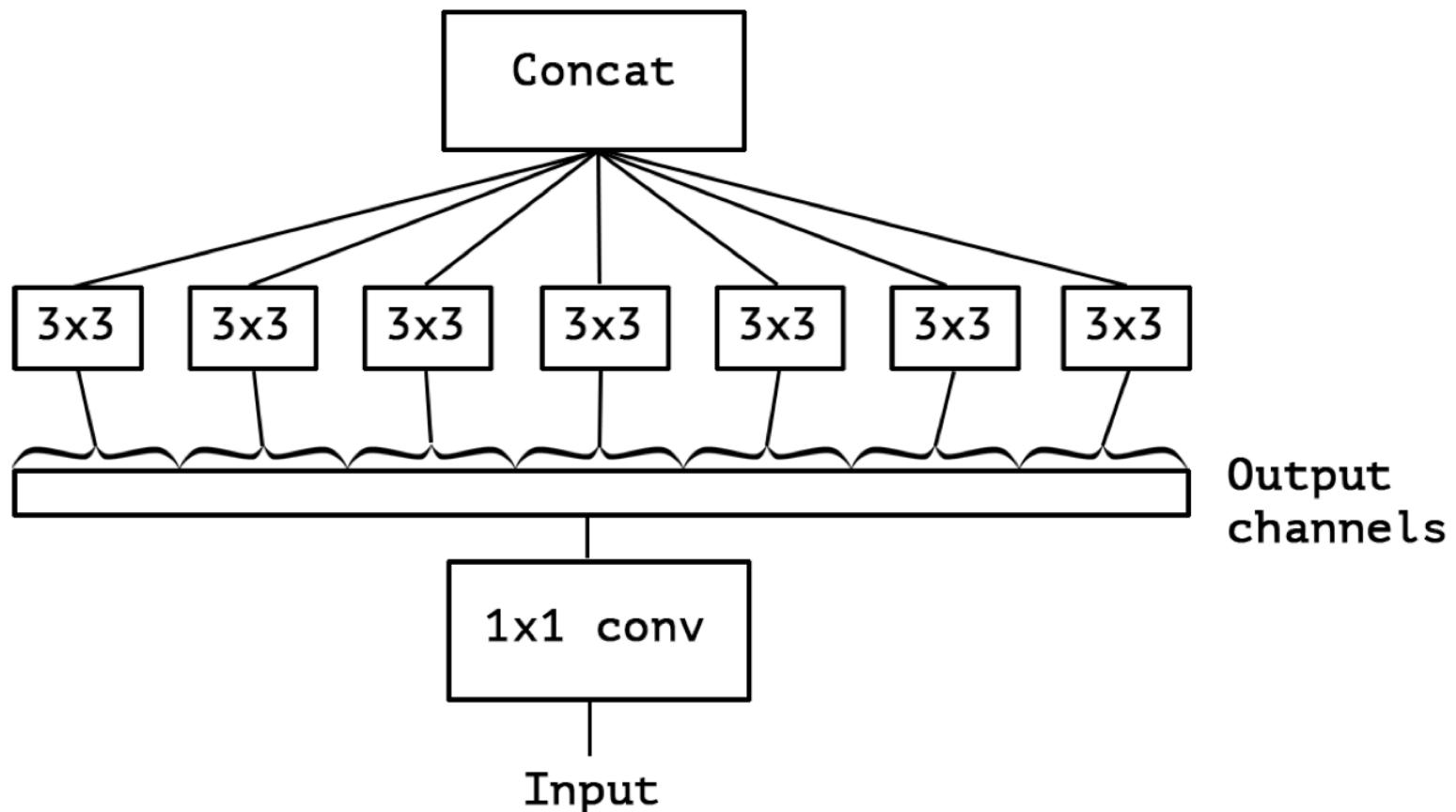
ResNet (2015)

- 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?

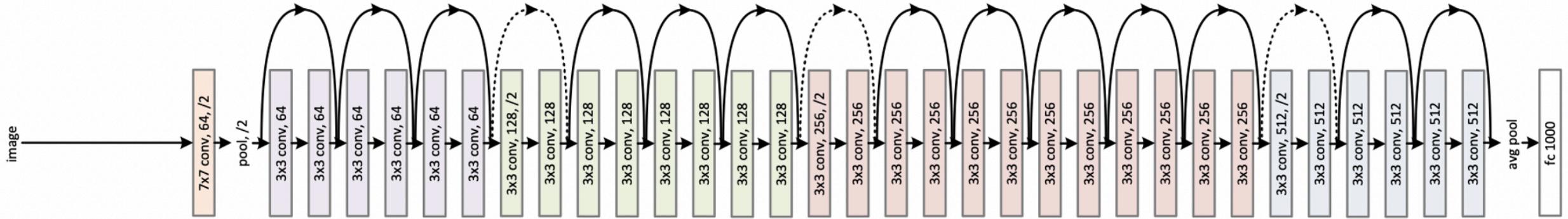
- Highway networks
- Inception-ResNet
- Squeeze and Excitation Network
- MobileNet
- ...

Transfer learning

Перенос знаний

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

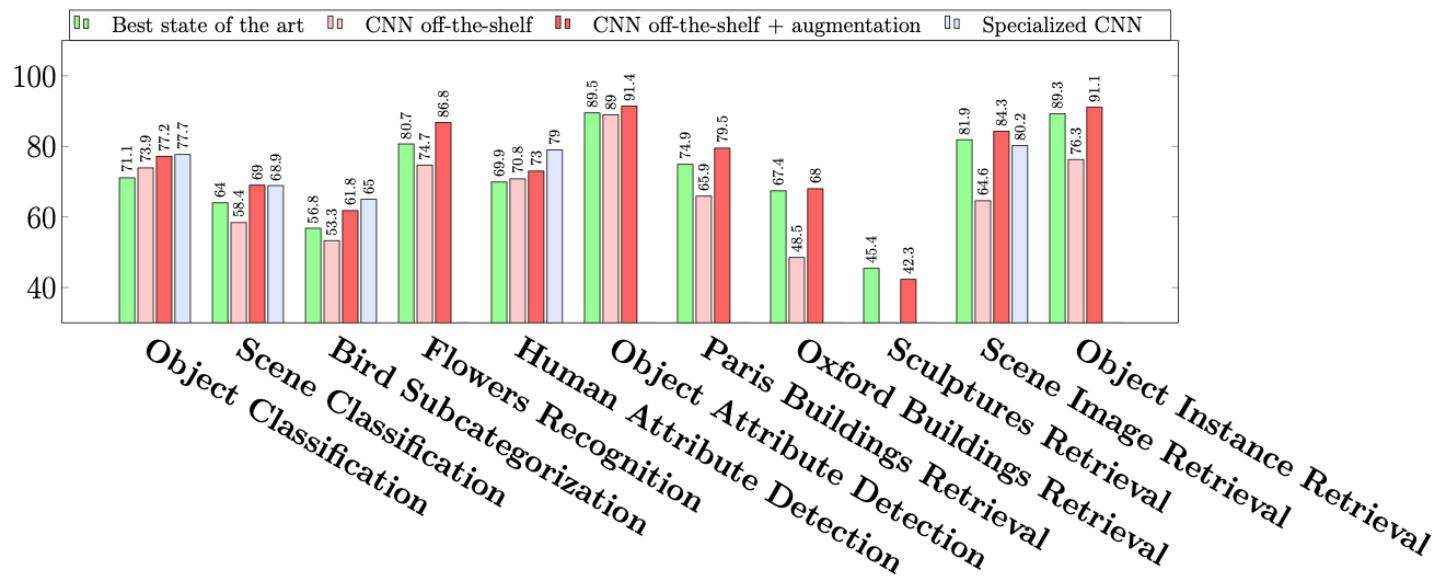
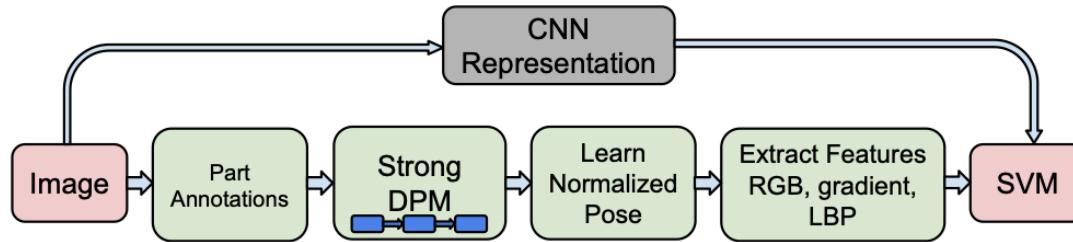
Дообучение



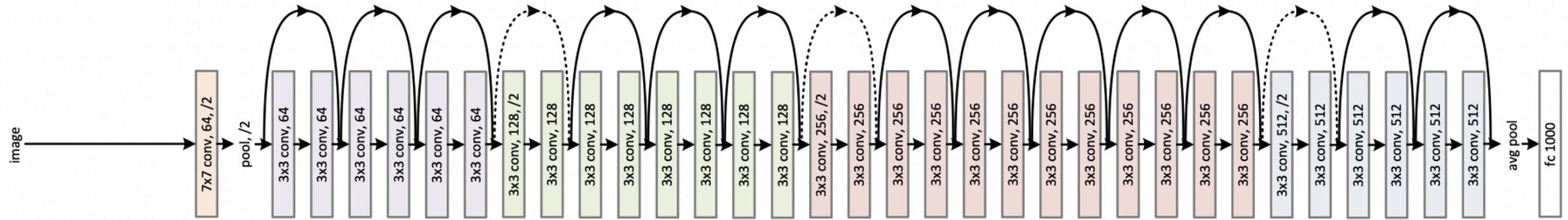
Если данных совсем мало:

- 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

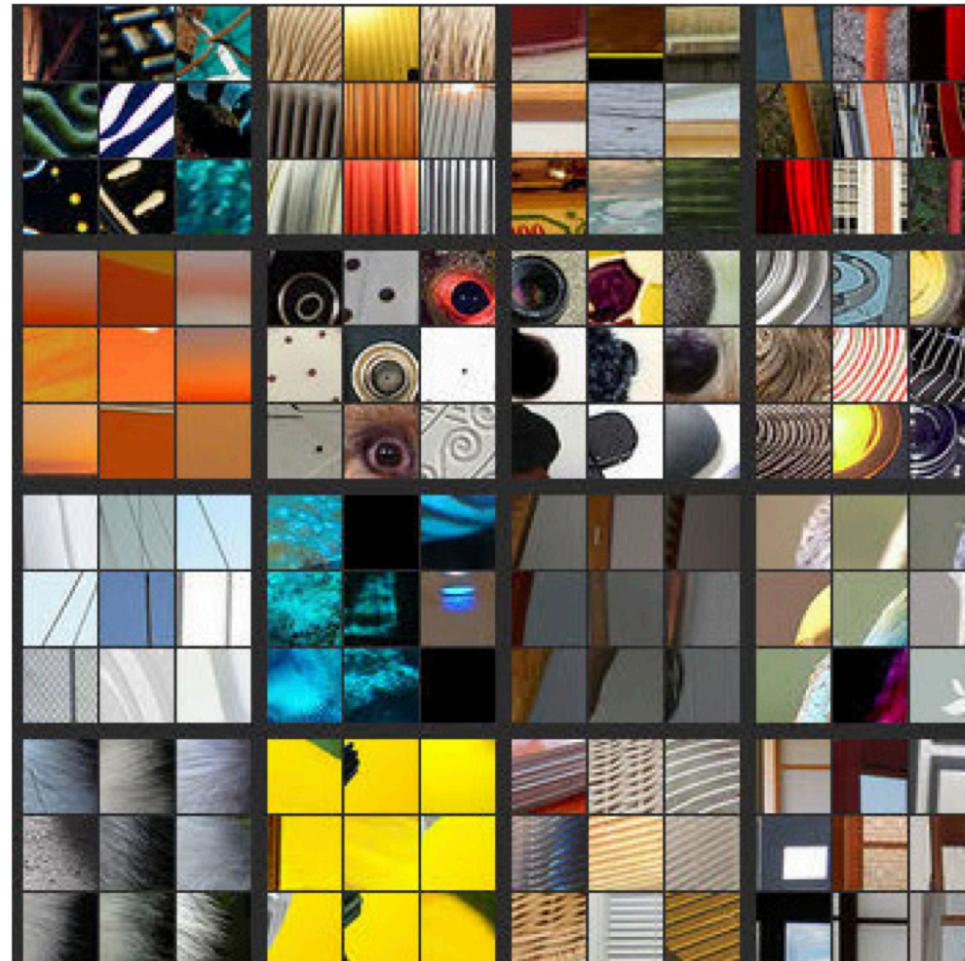
Что находят свёрточные сети?

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

Слой 1



Слой 2



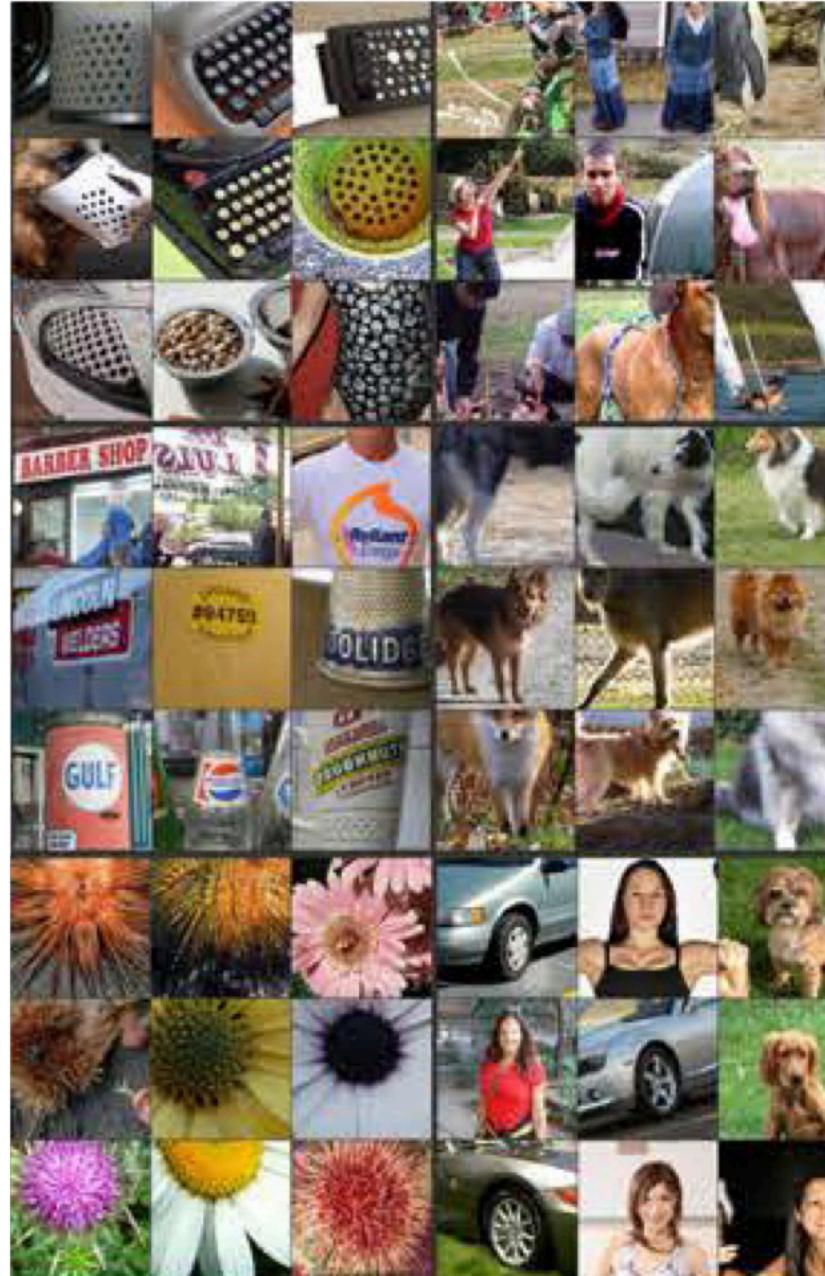
Слой 3



Слой 4



Слой 5

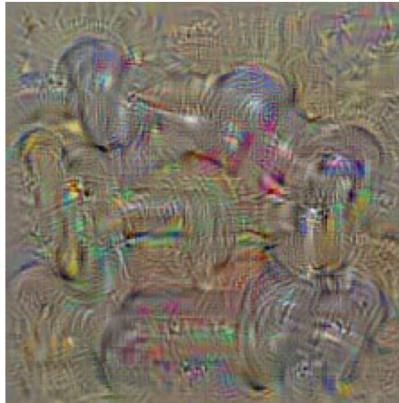


Максимизация вероятности класса

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

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

Максимизация вероятности класса



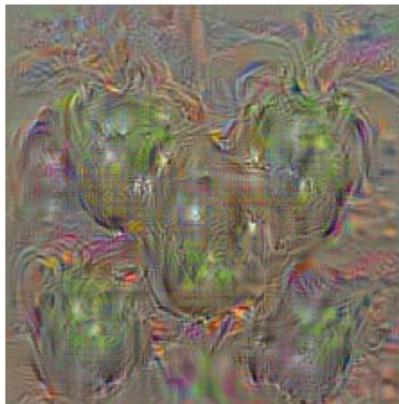
dumbbell



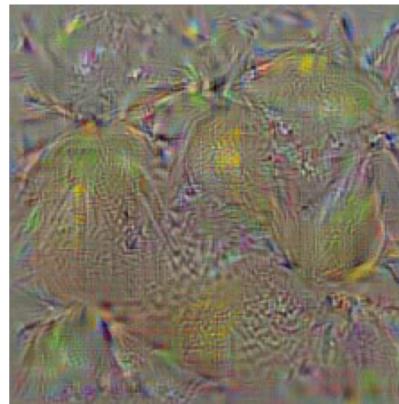
cup



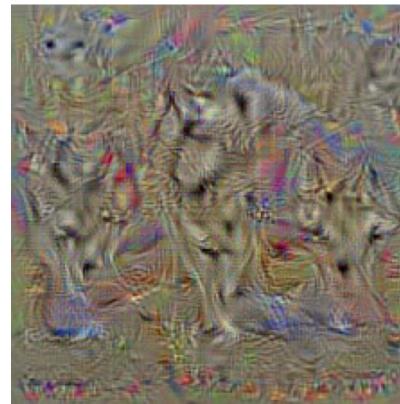
dalmatian



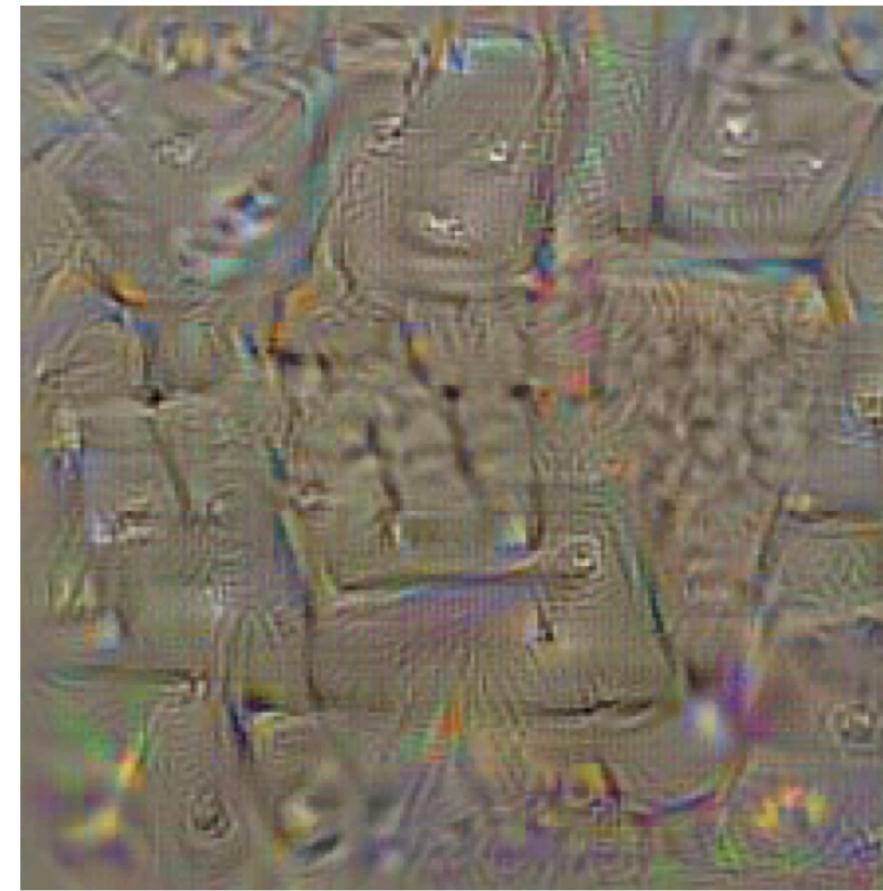
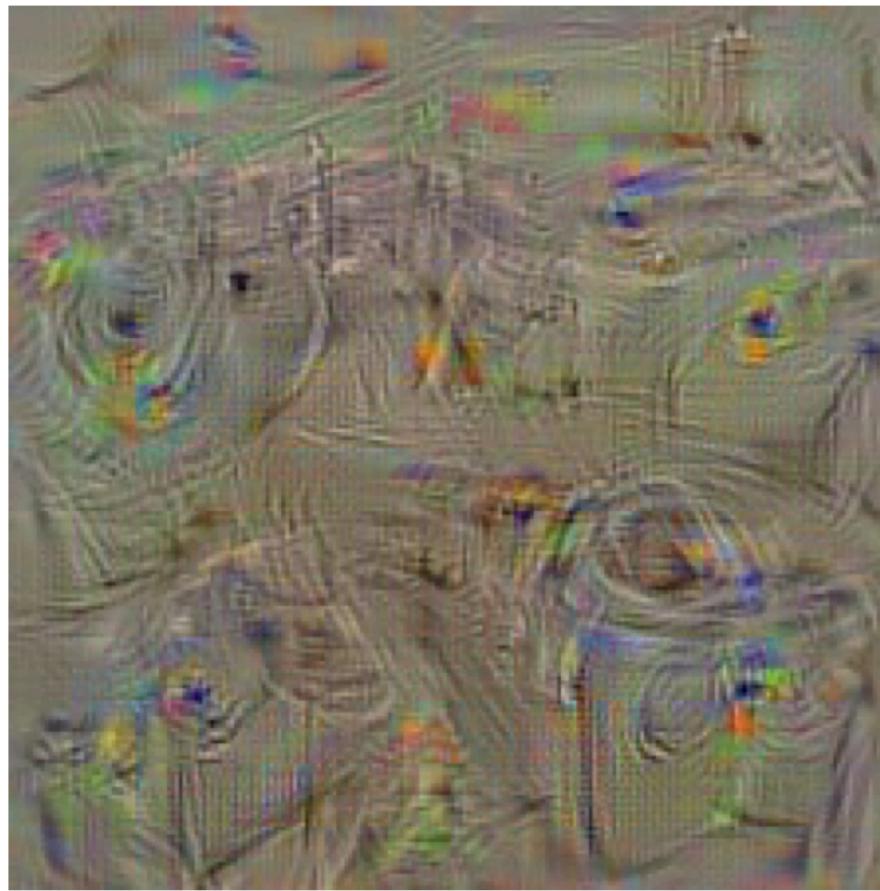
bell pepper



lemon



husky



Максимизация вероятности класса



Hartebeest



Measuring Cup



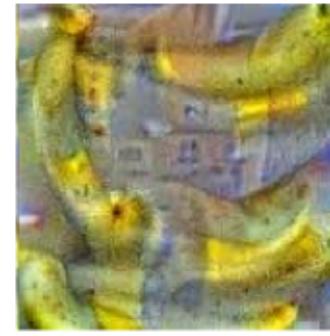
Ant



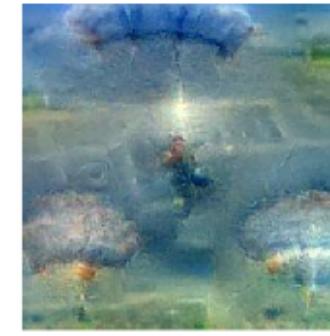
Starfish



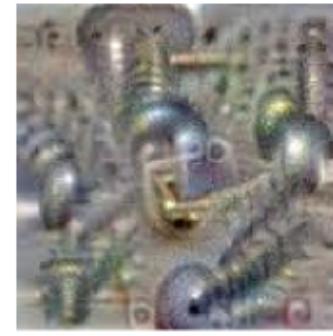
Anemone Fish



Banana

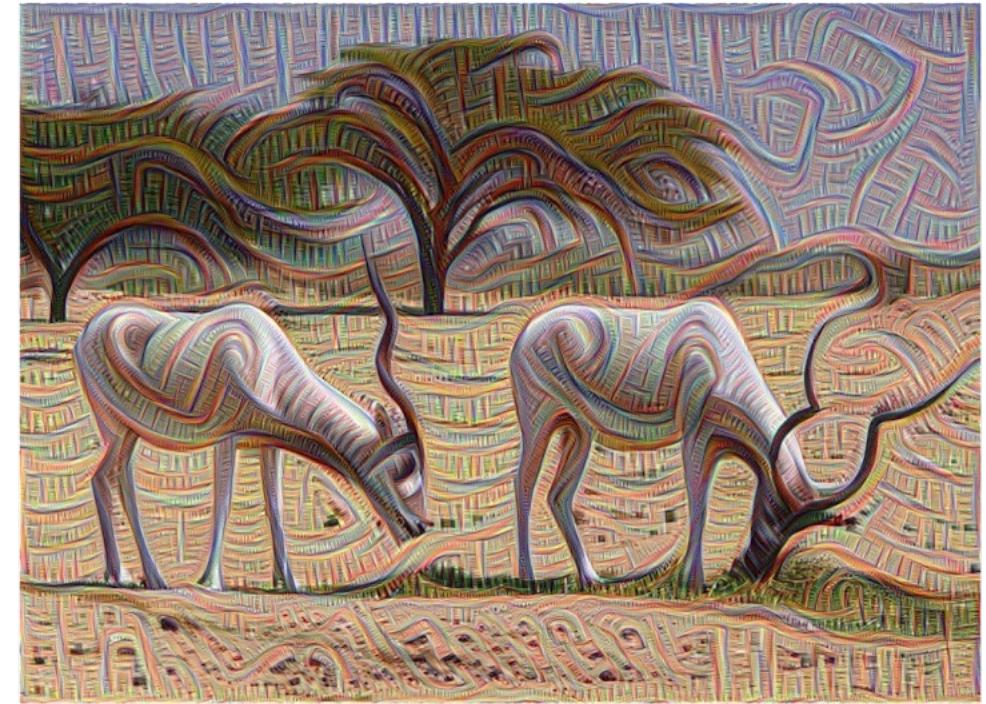


Parachute

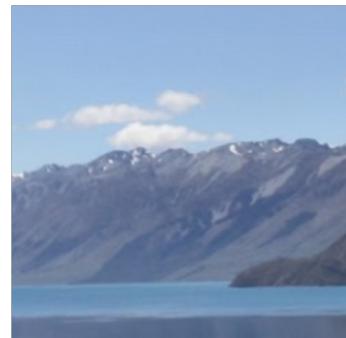


Screw

Максимизация вероятности класса



Максимизация вероятности класса



Horizon



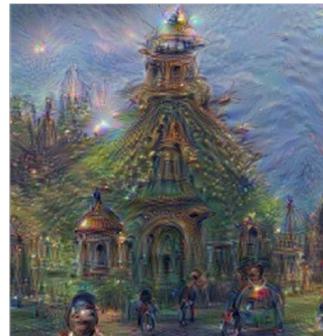
Trees



Leaves



Towers & Pagodas



Buildings



Birds & Insects

Максимизация вероятности класса

