

Глубокое обучение

Задачи с изображениями

Дьяконов А.Г.

**Московский государственный университет
имени М.В. Ломоносова (Москва, Россия)**

Задачи с изображениями

Классификация – что изображено

Локализация – где изображено

Детектирование – что и где

Сегментация – матрица меток сегментов

Преобразование изображений

- **удаление шума**
 - **стилизация**

Восстановление объектов (ex: 3D-модели)

Классификация изображений – почему нетривиальная задача

освещение меняется



ракурсы / деформации / позы



[<http://cs231n.stanford.edu/2017/syllabus.html>]

Классификация изображений – почему нетривиальная задача

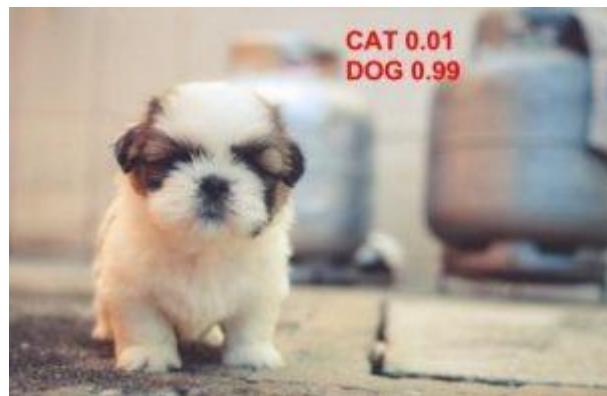
Окклюзия



Сливание с фоном



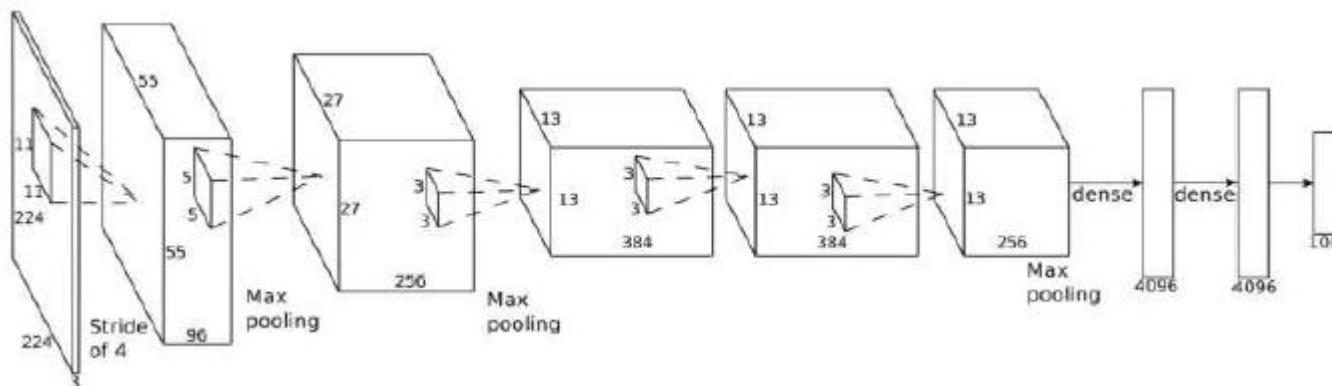
Классификация изображений



<http://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/>

Формально уже почти решённая задача!

**Можно использовать любую архитектуру /
преднастроенную НС**



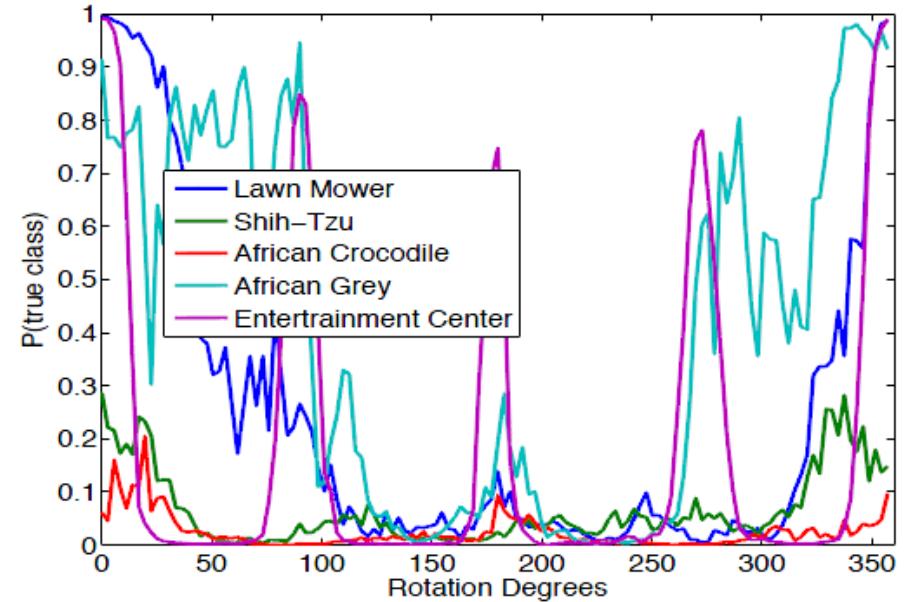
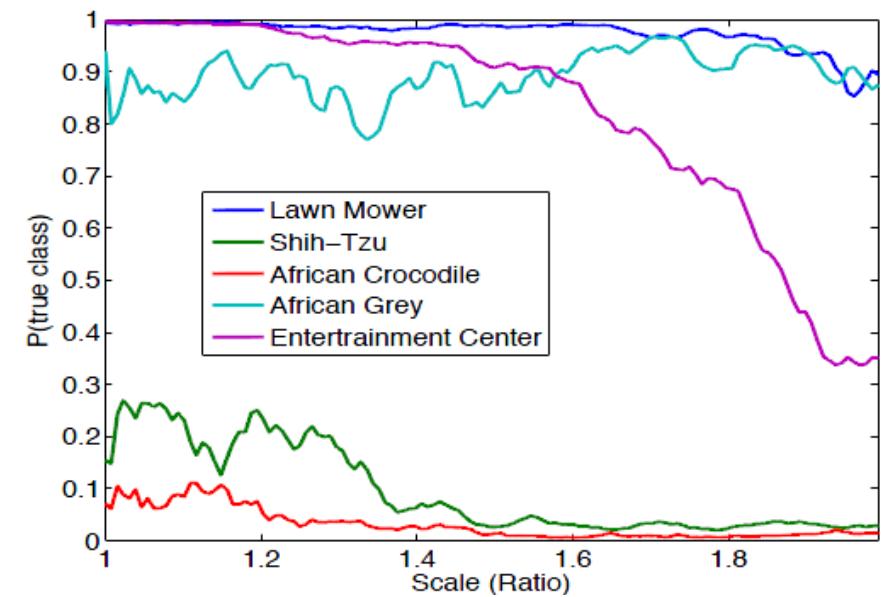
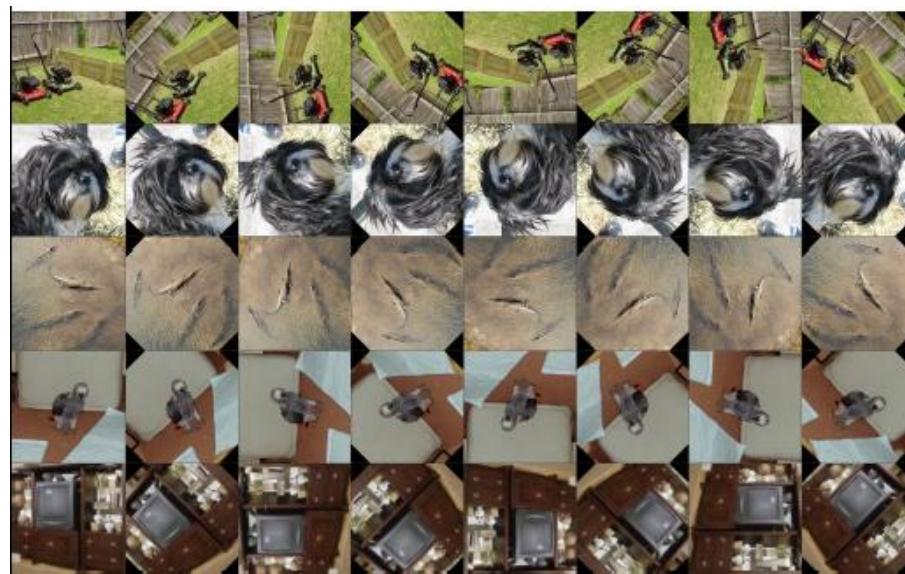
Вход – изображение

Выходы – классы

Ошибка – LogLoss

Сначала свёрточные слои, потом – полносвязные

Проблемы классификации: если менять масштаб и вращать



Другие задачи типа «что и где изображено»

Классификация + локализация



«cat»

Детектирование объектов
(Object Detection)



DOG, DOG, CAT

<http://cs231n.stanford.edu/2017/syllabus.html>

Семантическая сегментация
(Semantic Segmentation)



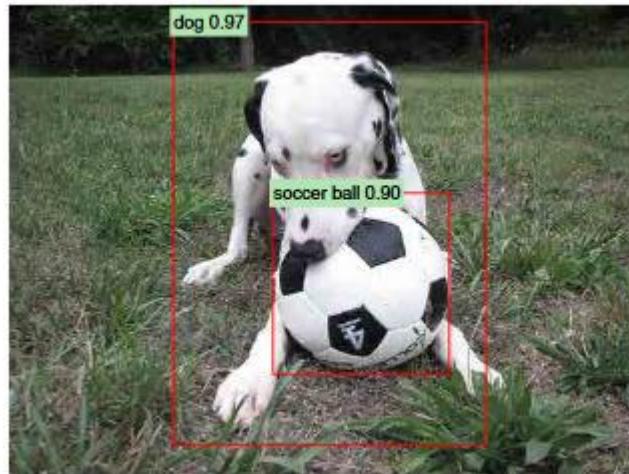
GRASS, CAT, TREE, SKY

Сегментация объектов
(Instance Segmentation)



DOG, DOG, CAT

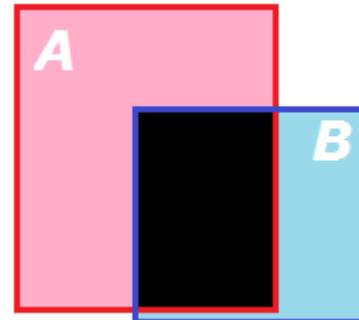
Детектирование объектов = Локализация + Классификация



**Локализация (localization)
объекта – где
Классификация – что**

<http://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/>

Правильное детектирование, если



$$\text{IoU} = \frac{|A \cap B|}{|A \cup B|} \geq \alpha$$

Intersection Over Union

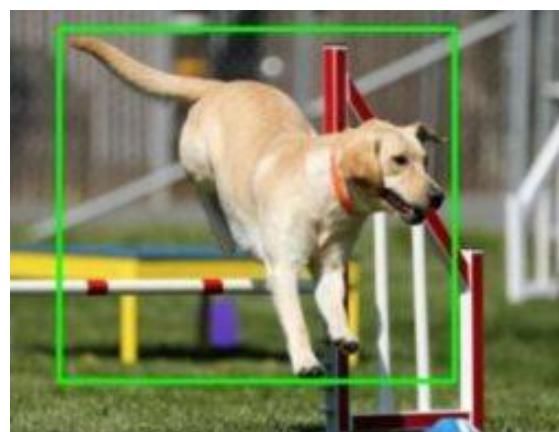
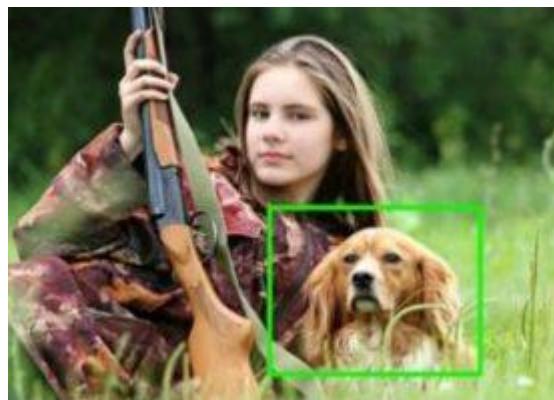
Детектирование объектов

**Формально – перебрать разные локализации
(чаще всего используют прямоугольники)
и для каждого – классификация**

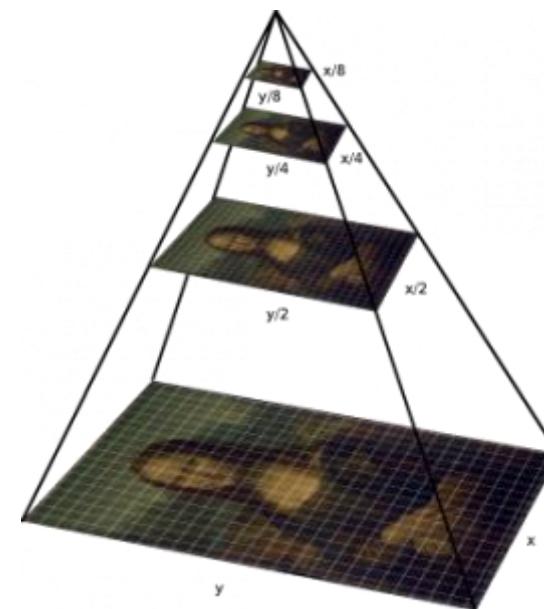


Детектирование объектов

**Реально –
проблема размера**



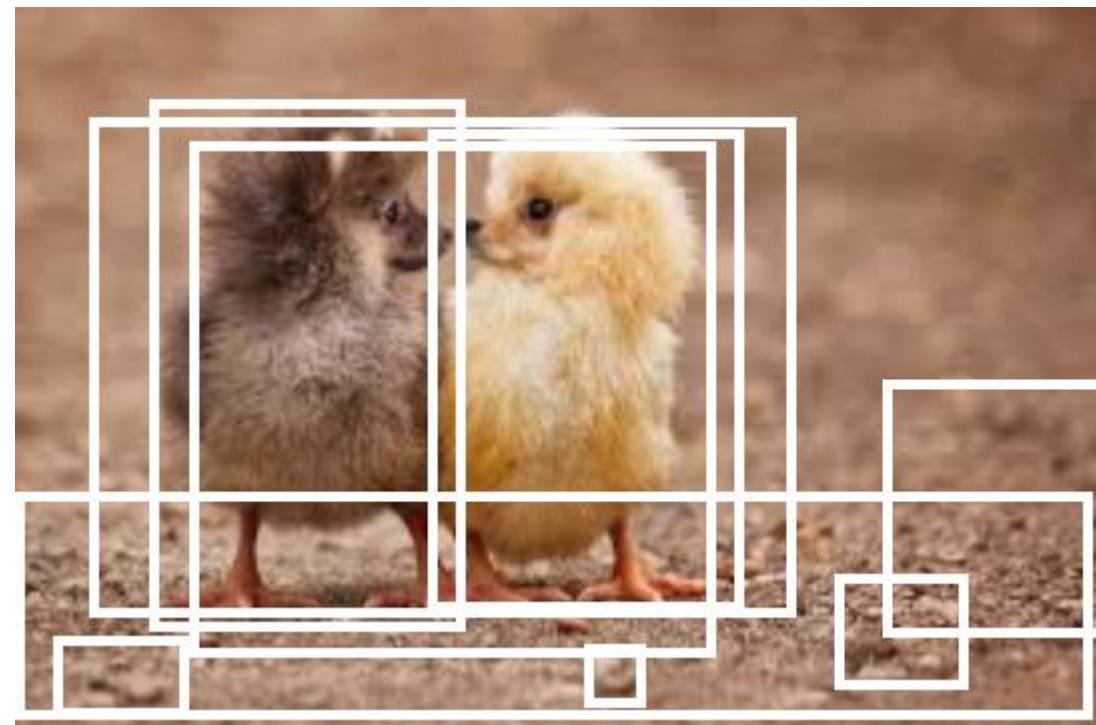
Решение –



**Использовать изображения разных
масштабов + фиксированный размер
области**

Генерация регионов (Region Proposals)

**Много разных методов генерации областей,
где потенциально есть объекты**



Работают очень быстро

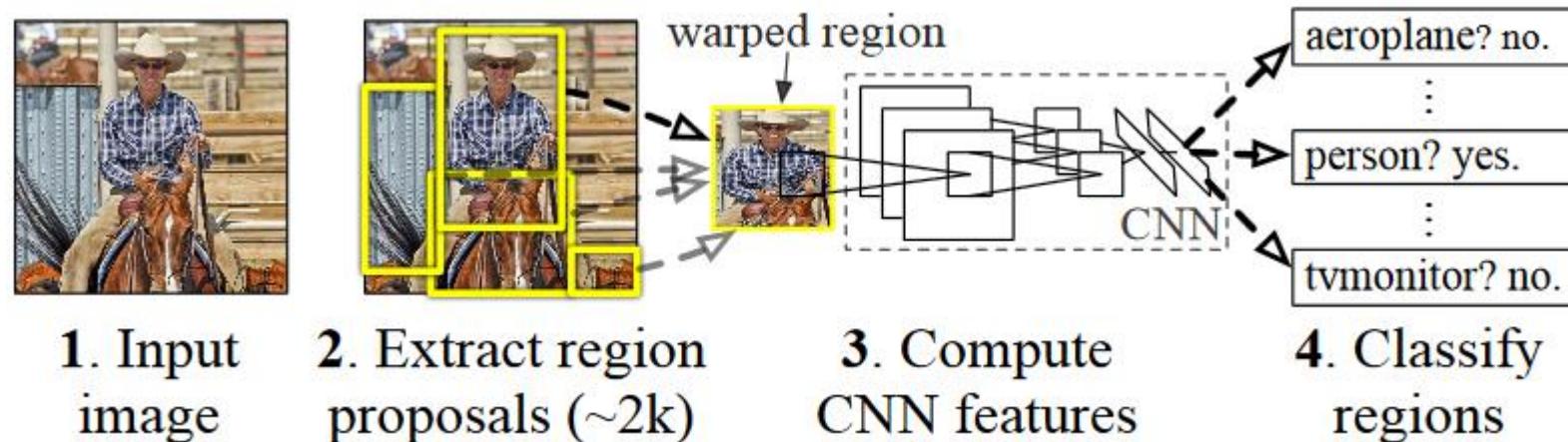
Детектирование объектов

«R-CNN – Object detection and semantic segmentation»

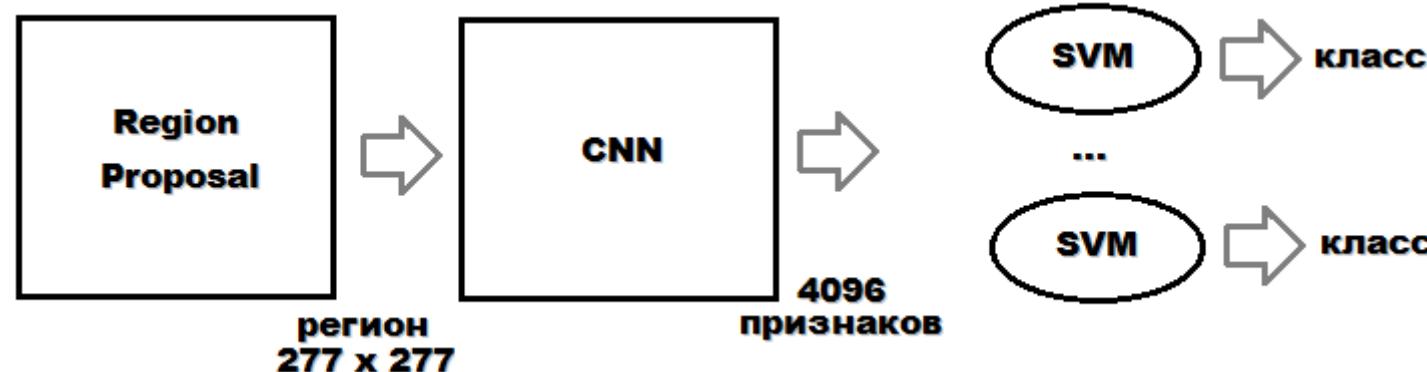
[Girshick et al., 2013 <https://arxiv.org/pdf/1311.2524.pdf>]

«Fast R-CNN» – [Girshick 2015 <https://arxiv.org/pdf/1504.08083.pdf>]

«Faster R-CNN» – [Ren et al., 2015 <https://arxiv.org/abs/1506.01497>]



R-CNN (=Regions with CNN features)



- **Selective Search для генерации регионов (гипотез, где объекты)**

Есть много методов просто выбран этот + resize ~2000 раз

- **Классификация регионов (CNN + SVM)**

Krizhevsky Caffe + линейный SVM

Предварительно натренированная CNN (ILSVRC2012 classification) + дотренировать на данных

- **Оптимизация регионов с помощью регрессии**

Bounding-box regression

R-CNN Дотренировка



**Если 30% (важный коэффициент) пересечения с истинным
объектом «машина», то это класс машина**

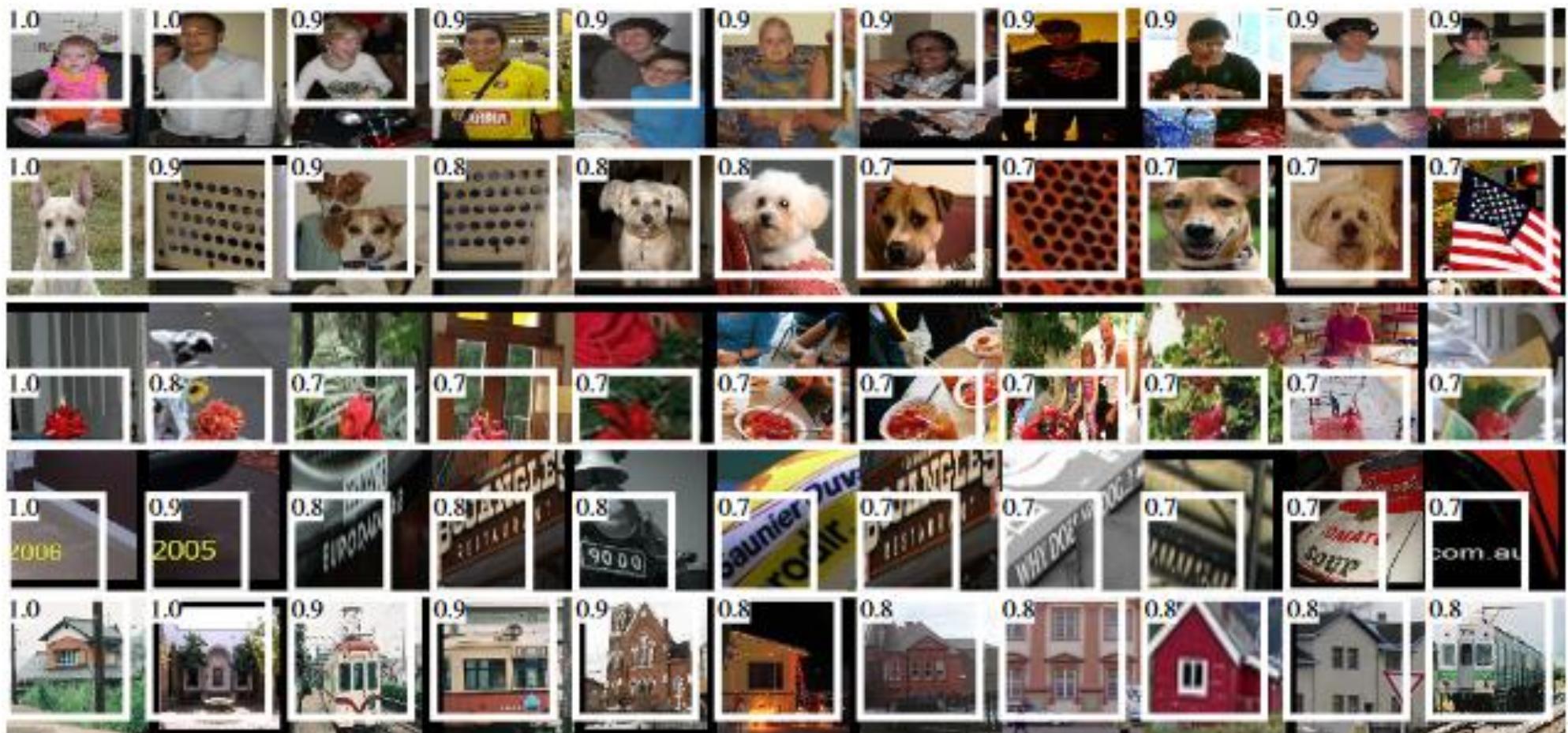
**Классов = число объектов детектирования + 1 (фон)
– теперь столько нейронов в последнем слое**

batch = 32 окна с объектом + 96 с фоном

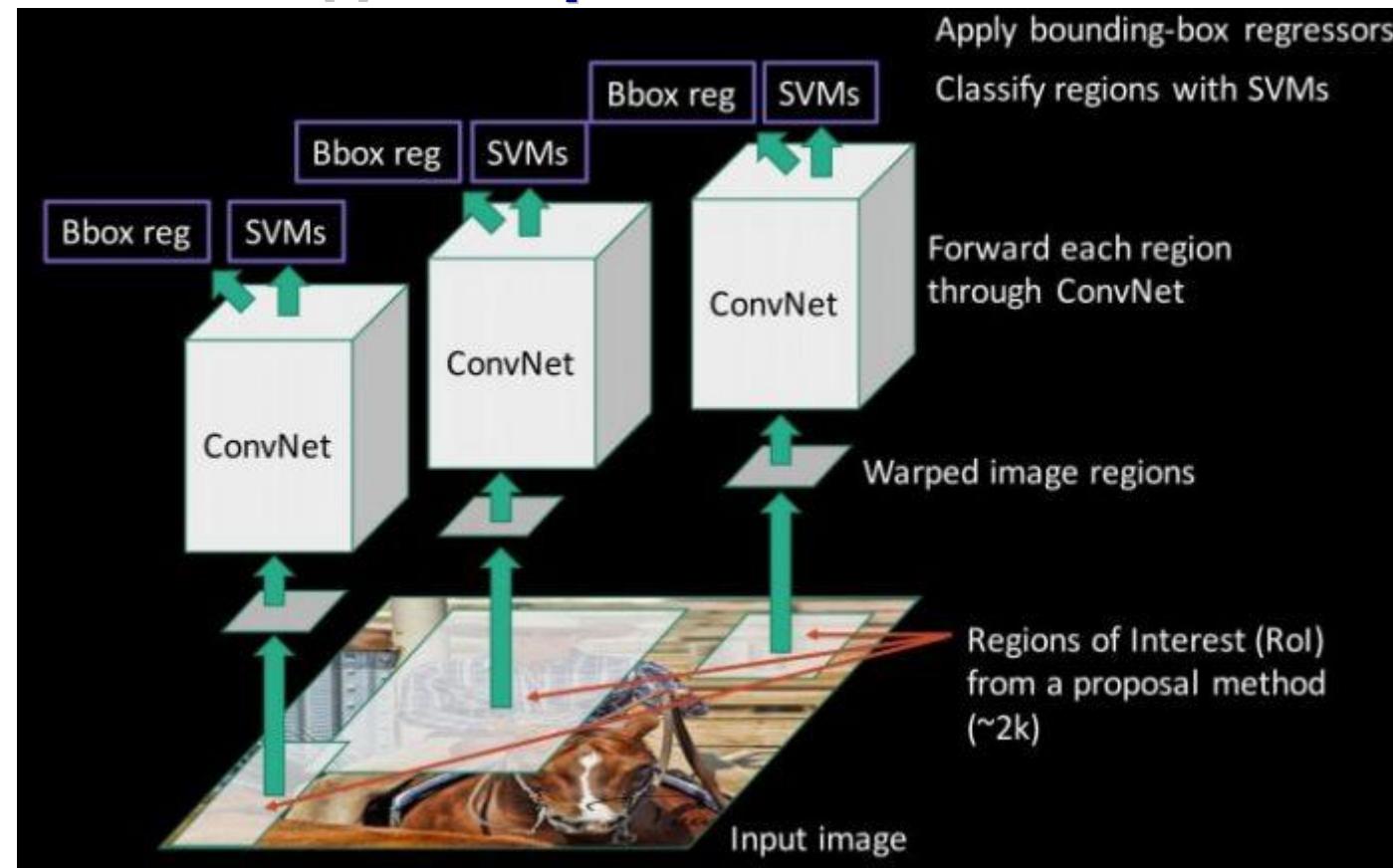
R-CNN Визуализация

Как понять, что сеть учится...

Для нейрона (в последних слоях) упорядочить все входы по его значению, посмотреть на Топ входов



Детектирование объектов



Недостатки R-CNN

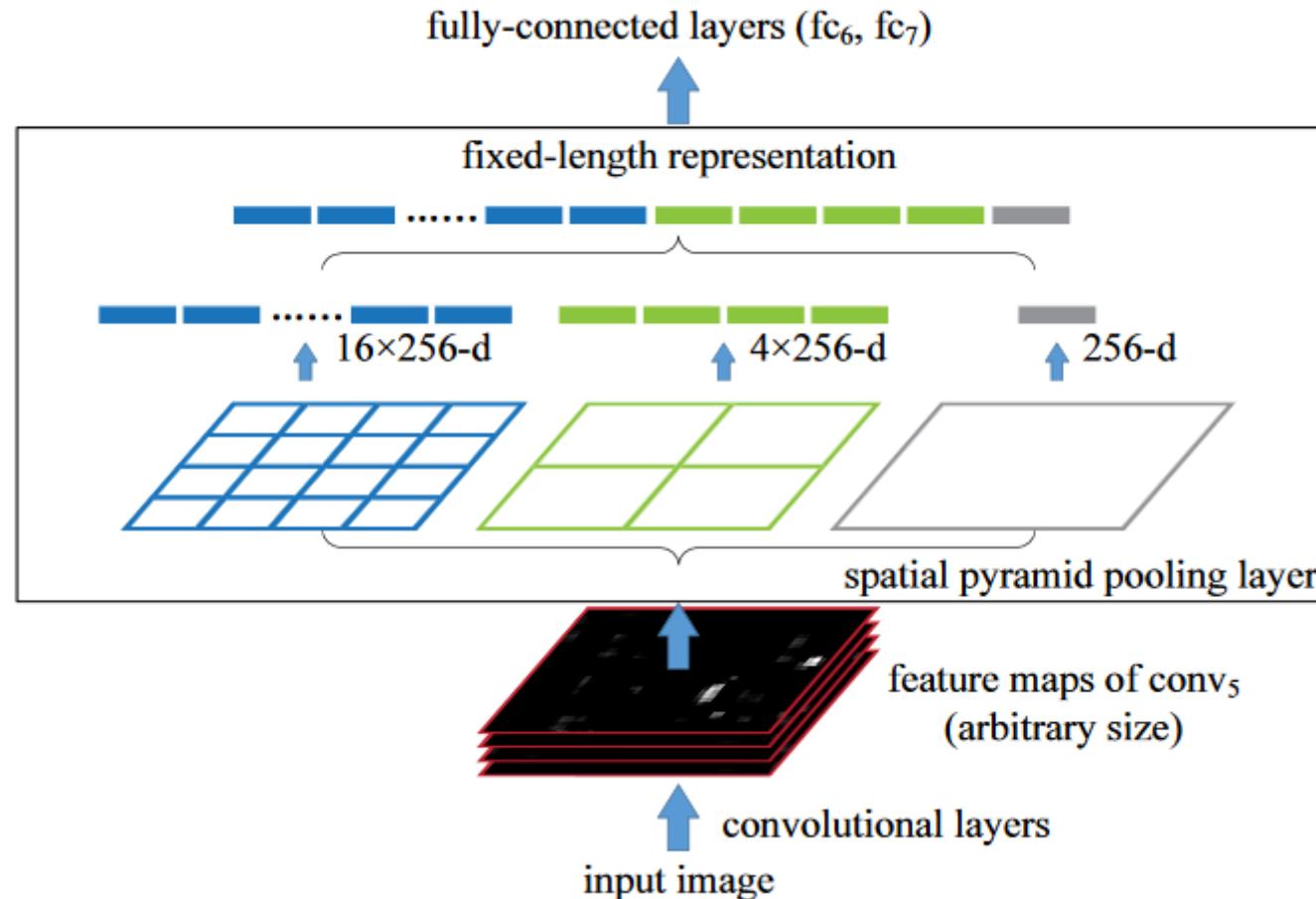
- **Дообучиваем CNN (log_loss), потом ещё SVM, потом bounding-box-регрессию...**
 - **Для каждого региона запускаем CNN**

Детектирование объектов Spatial Pyramid Pooling (SPP-net)

**Полносвязной НС нужен вход определённых размеров.
Чисто свёрточная НС может работать со входом любых размеров!
Идея – добавим к свёрточной сети слой, который может генерировать фиксированный размер**



Детектирование объектов The Spatial Pyramid Pooling Layer

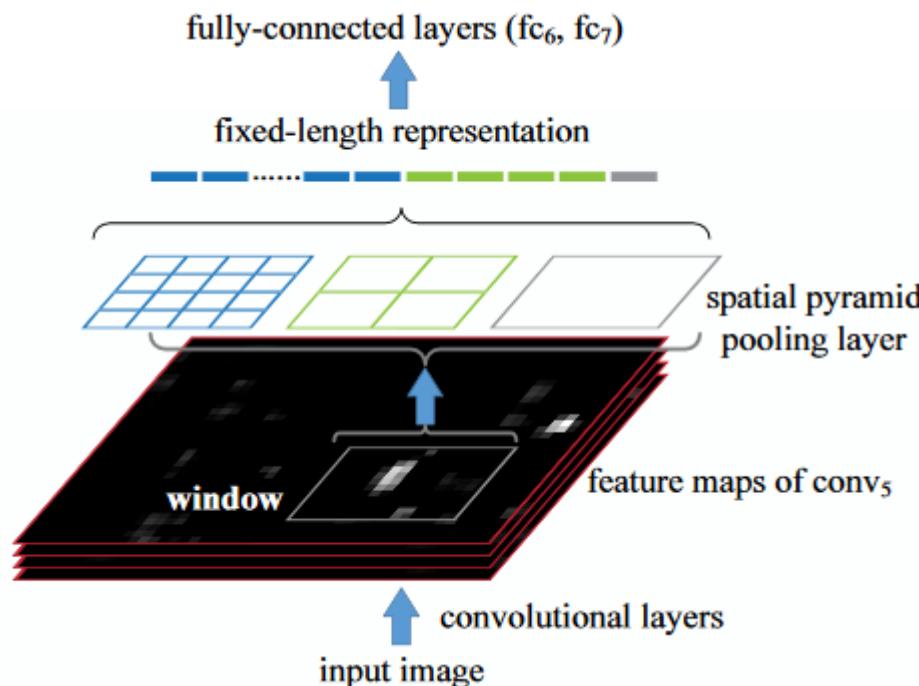


Идея из Bag-of-Words (BoW)

Spatial Pyramid Pooling in Deep Convolutional

Networks for Visual Recognition [He и др. 2015, <https://arxiv.org/pdf/1406.4729.pdf>]

Детектирование объектов



Считаем CNN-представление для изображений один раз.

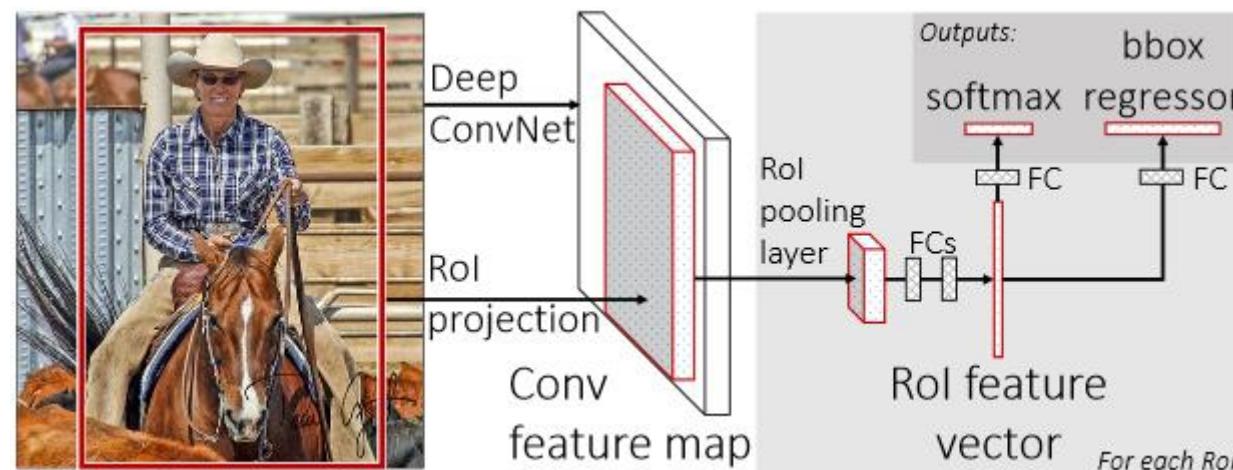
С помощью SPP-net для каждого региона получаем признаковое представление (фиксированной длины)

Быстрое решение проблемы того, что регионы разных размеров!
+ сначала-то мы считаем CNN-представление!
а SPP-net просто эффективный способ перевода его в признаки
Сеть не работает на каждом регионе! Сразу на изображении!
Выигрыш по скорости 10x – 100x!

Детектирование объектов

Fast R-CNN = R-CNN + SPP + регрессию встроили в НС

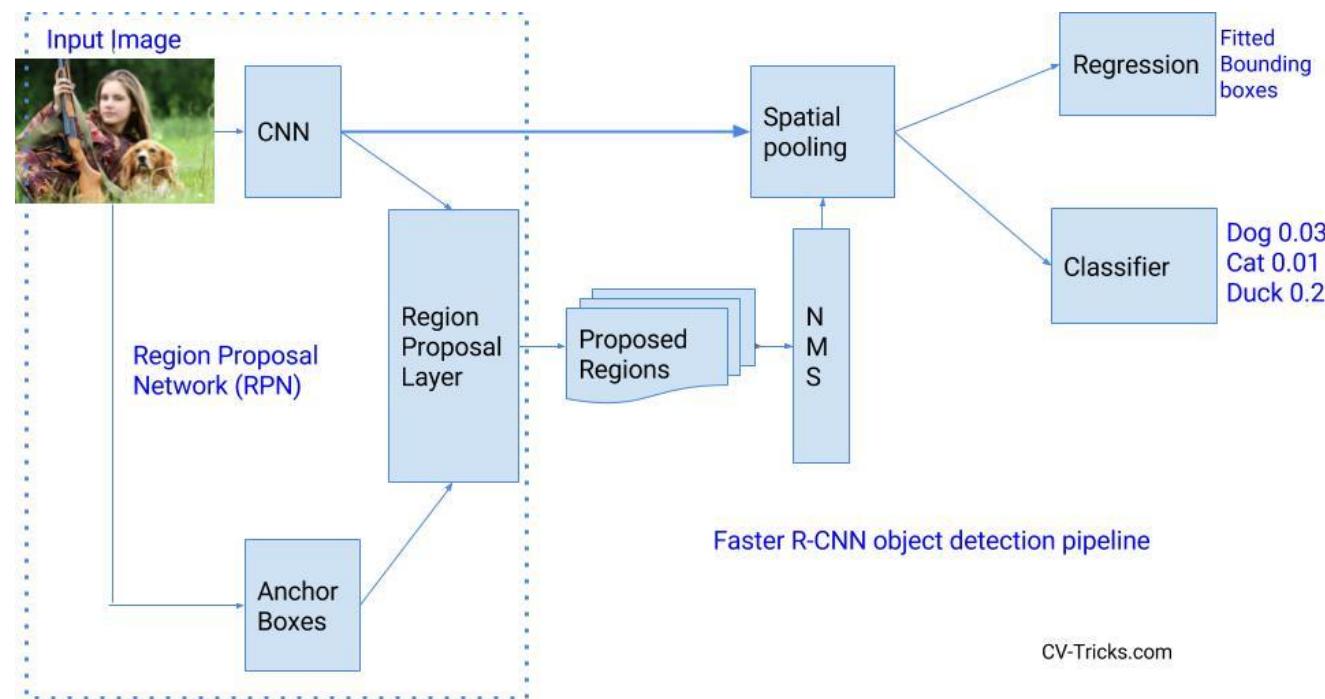
Обучение в одну стадию (раньше CNN → SVM → regression)!



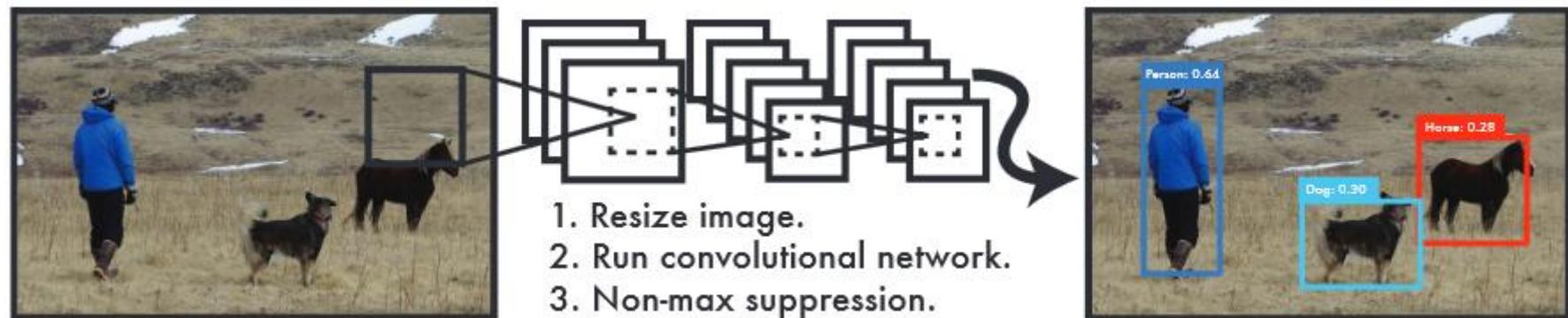
Вход: изображение + параметры регионов

Faster R-CNN = Selective Search → Region Proposal Network
«Где смотреть» решает НС

Детектирование объектов



Детектирование объектов YOLO (You only Look Once)



- Изменение масштаба → 448×448
 - CNN
 - Пороговое принятие решения

Сеть видит изображение целиком, а не регионами

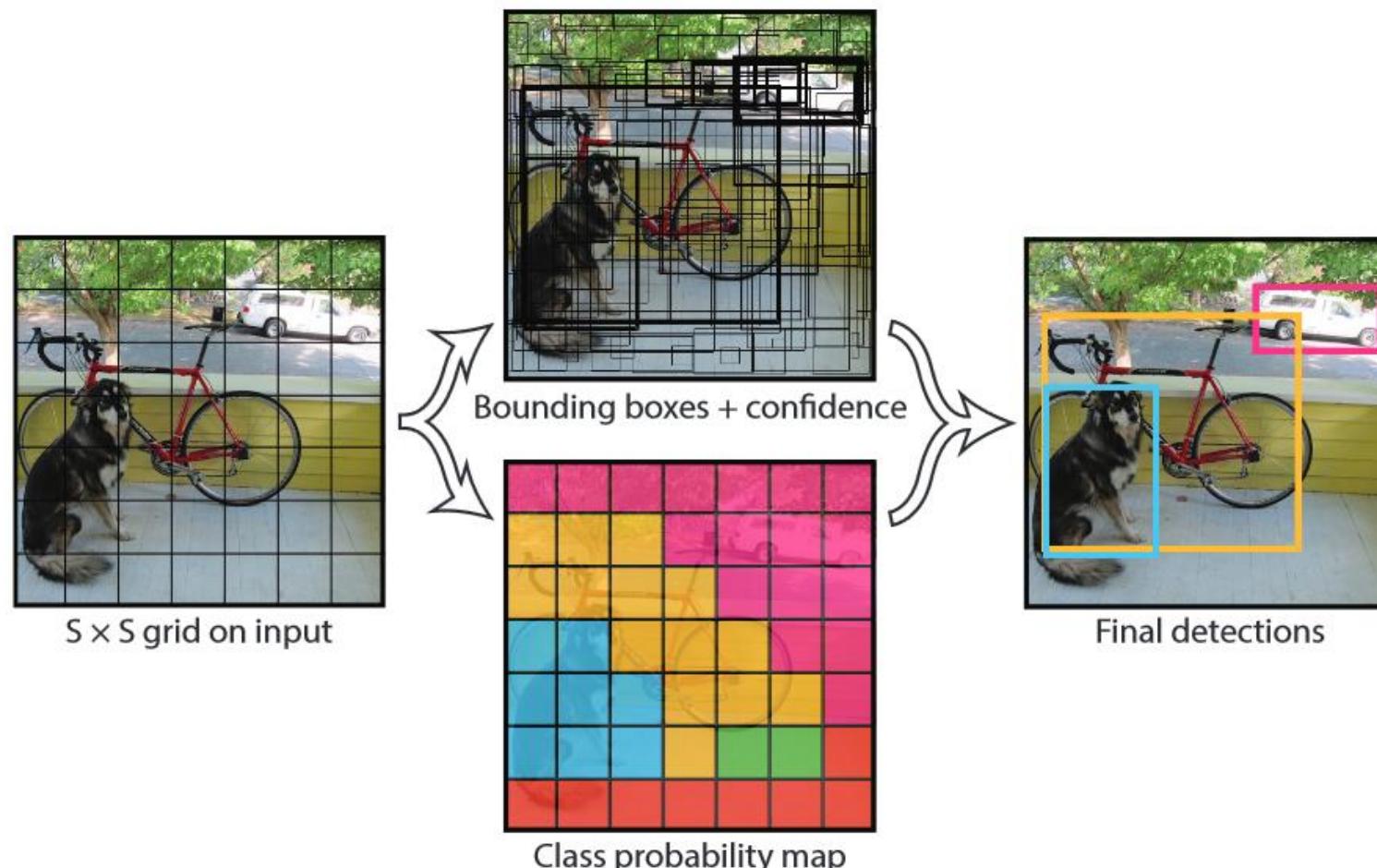
Очень быстрая, но точность хуже (особенно для мелких объектов)

https://pjreddie.com/media/files/papers/yolo_1.pdf

Детектирование объектов

Вход: Изображение

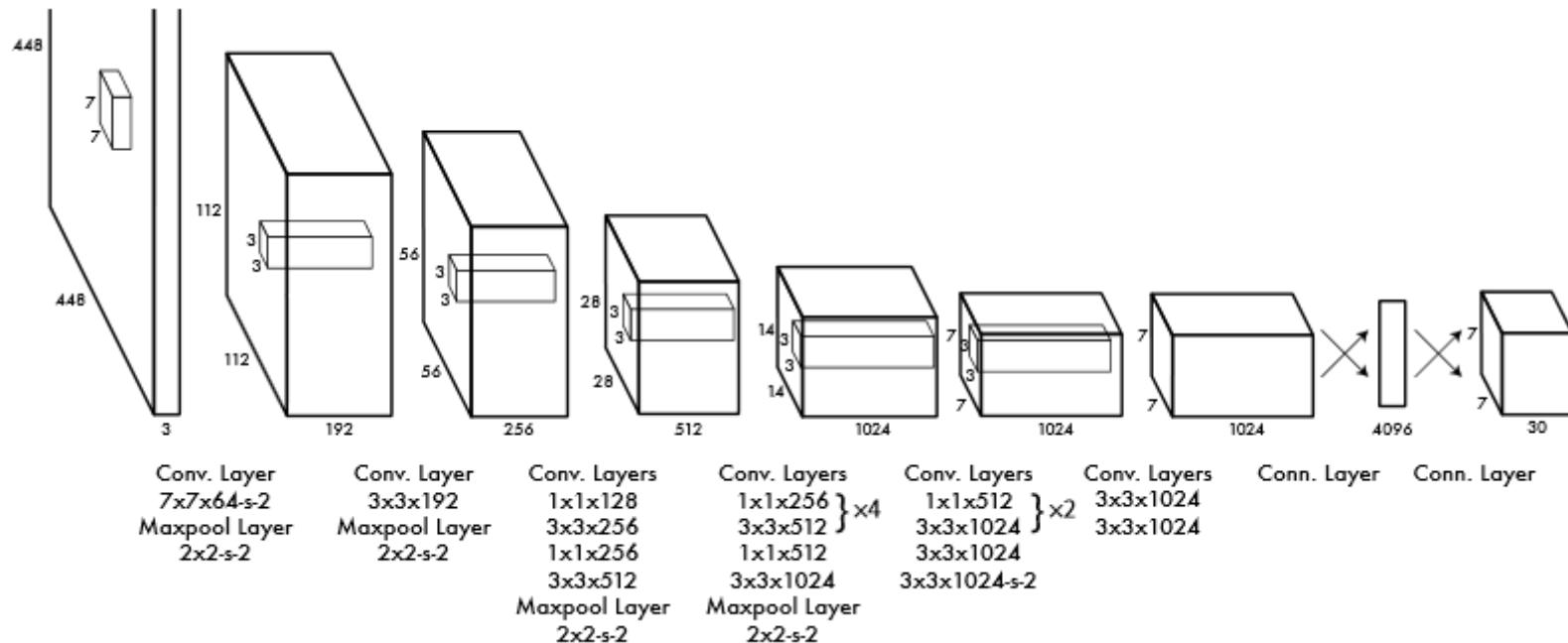
Выход: класс (вероятности) + координаты рамки



Классика: свёртки → полно связные слои

Детектирование объектов

24 слоя



**В тензоре $7 \times 7 \times 30$ закодированы все регионы оценки за классы
 7×7 – это сетка;)**

**$30 = 5 + 5$ (почему-то 2 региона для каждой ячейки) +
 $+ 20$ (# классов ~ Pascal VOC)**

$$5 = |(x, y, w, h, c)|$$

**x, y – координаты в центре соотв. ячейки
 c – уверенность, что регион правильный**

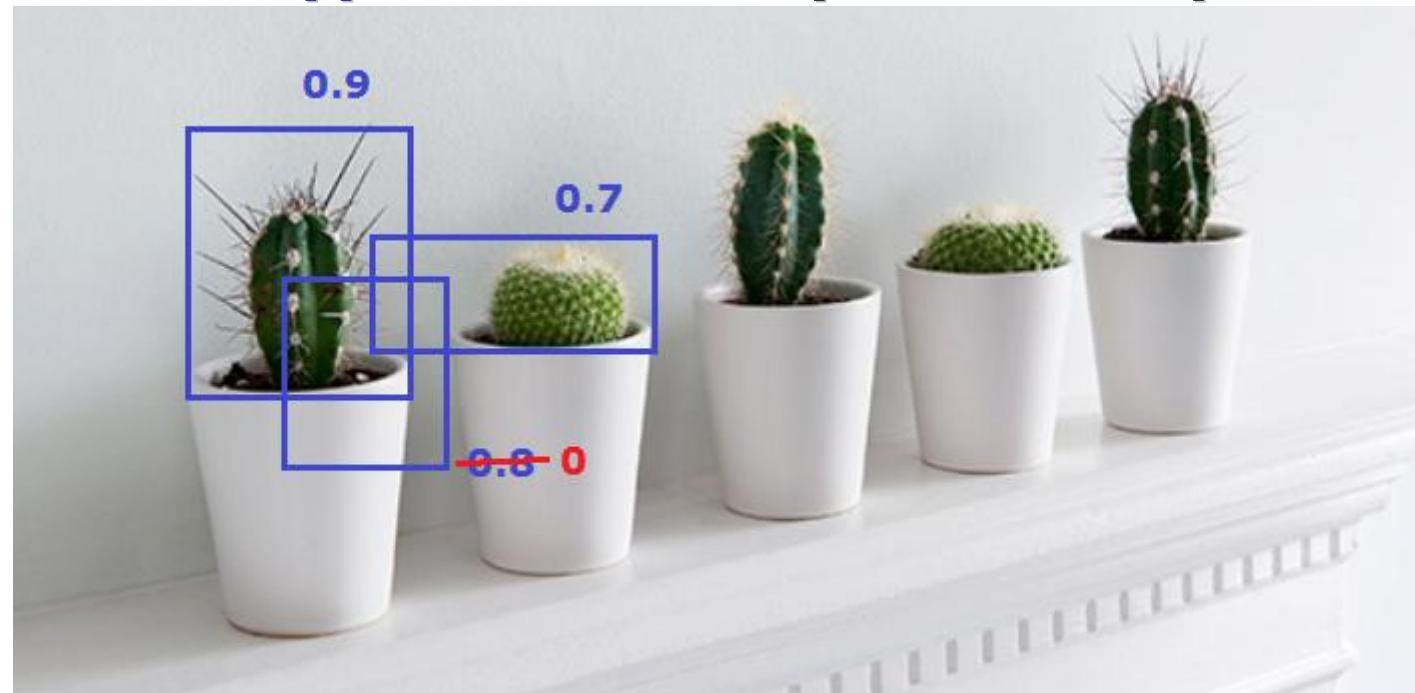
Детектирование объектов

- **Предположение: в каждой ячейке центры максимум 2х классов**
- **Первые 20 свёрточных слоёв предтренированы на ImageNet**
- **448 × 448 вместо 224 × 224,**
- **Leaky ReLU (везде)**
- **1 × 1 bottleneck filters**
- **dropout после 1го полносвязного слоя**
- **квадратичная ошибка (для координат, уверенности и оценок!)**
- **борьба с очень большими регионами и пустыми регионами (...)**
- **momentum 0.9, decay $5 \cdot 10^{-4}$**
- **аугментация (scaling, translation, HSV transformation)**

J. Redmon, S. Divvala, R. Girshick, A. Farhadi «You Only Look Once: Unified, Real-Time Object Detection» <https://arxiv.org/abs/1506.02640>

Детектирование объектов

Non-max Suppression – Как принимается решение?



Класс «Кактус»

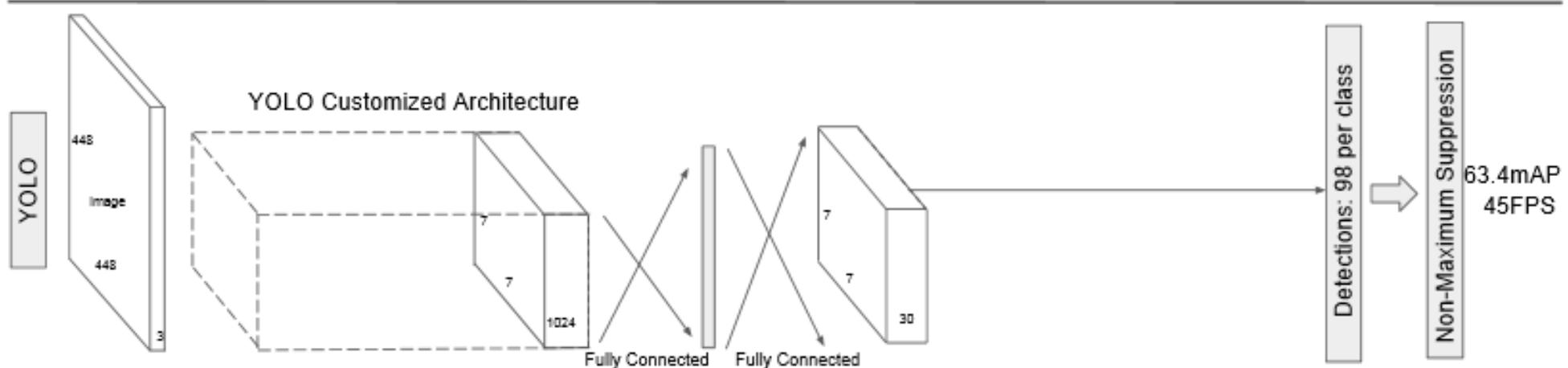
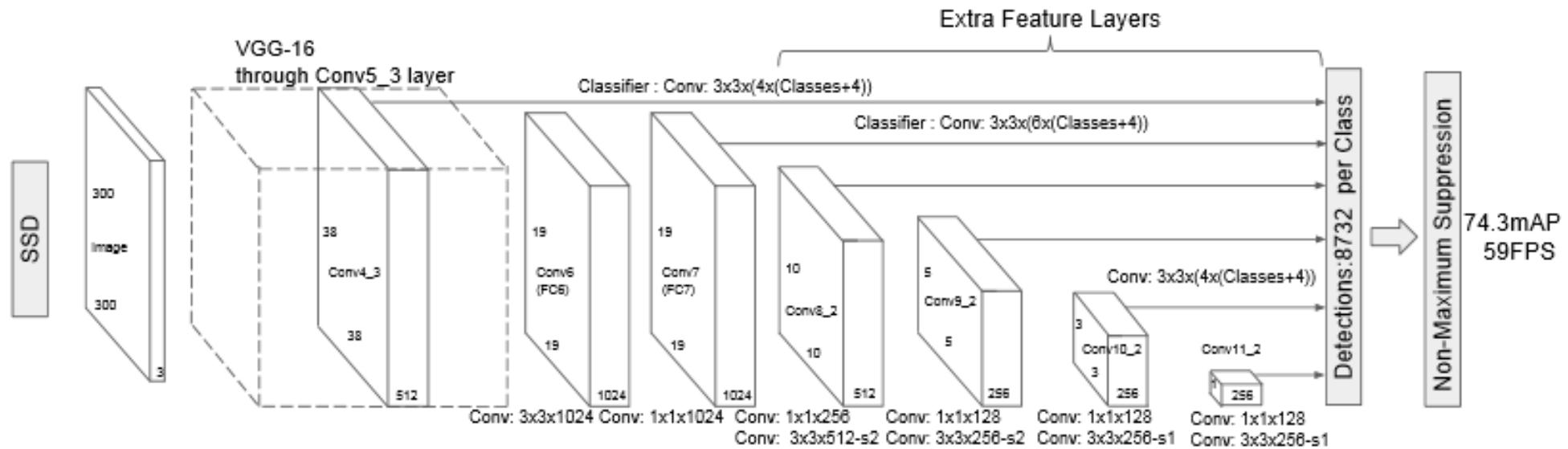
Упорядочим регионы: 0.9, 0.8, 0.7, ... (ниже порога сразу занулить)

Идём от MAX (i):

Идём от текущего (j):

регион j имеет большое пересечение с i \Rightarrow зануляем

Детектирование объектов Single Shot Detector (SSD)



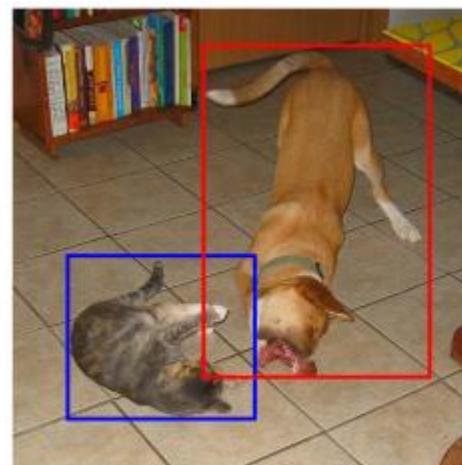
SSD: Single Shot MultiBox Detector

[W. Liu и др. <https://arxiv.org/pdf/1512.02325.pdf>]

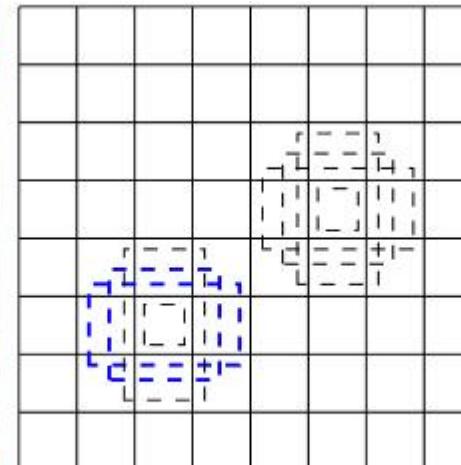
Детектирование объектов Single Shot Detector (SSD)

Вход 300×300 изображение

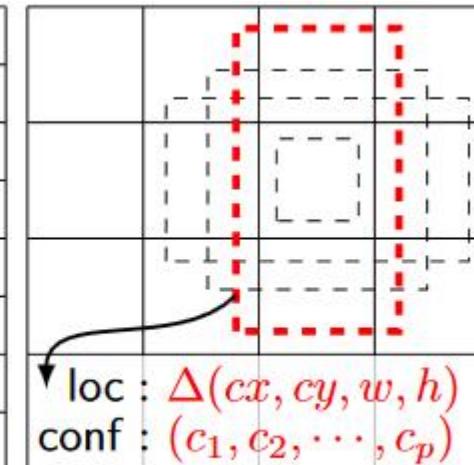
Нужны истинные рамки (на обучении)



(a) Image with GT boxes



(b) 8×8 feature map



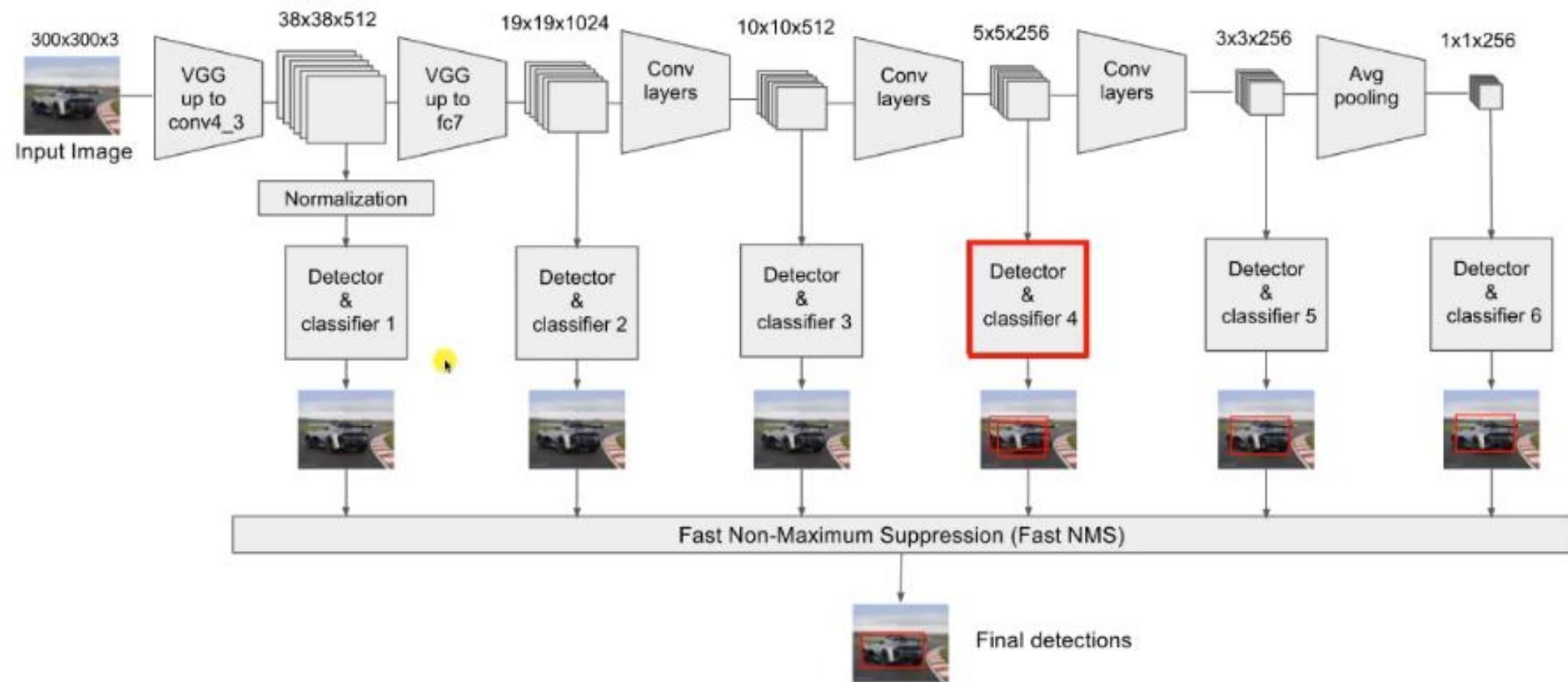
loc : $\Delta(cx, cy, w, h)$
conf : (c_1, c_2, \dots, c_p)

(c) 4×4 feature map

Задаём несколько (тут 4) «default boxes» – для каждого

- коррекция положения
- оценки принадлежности к классам

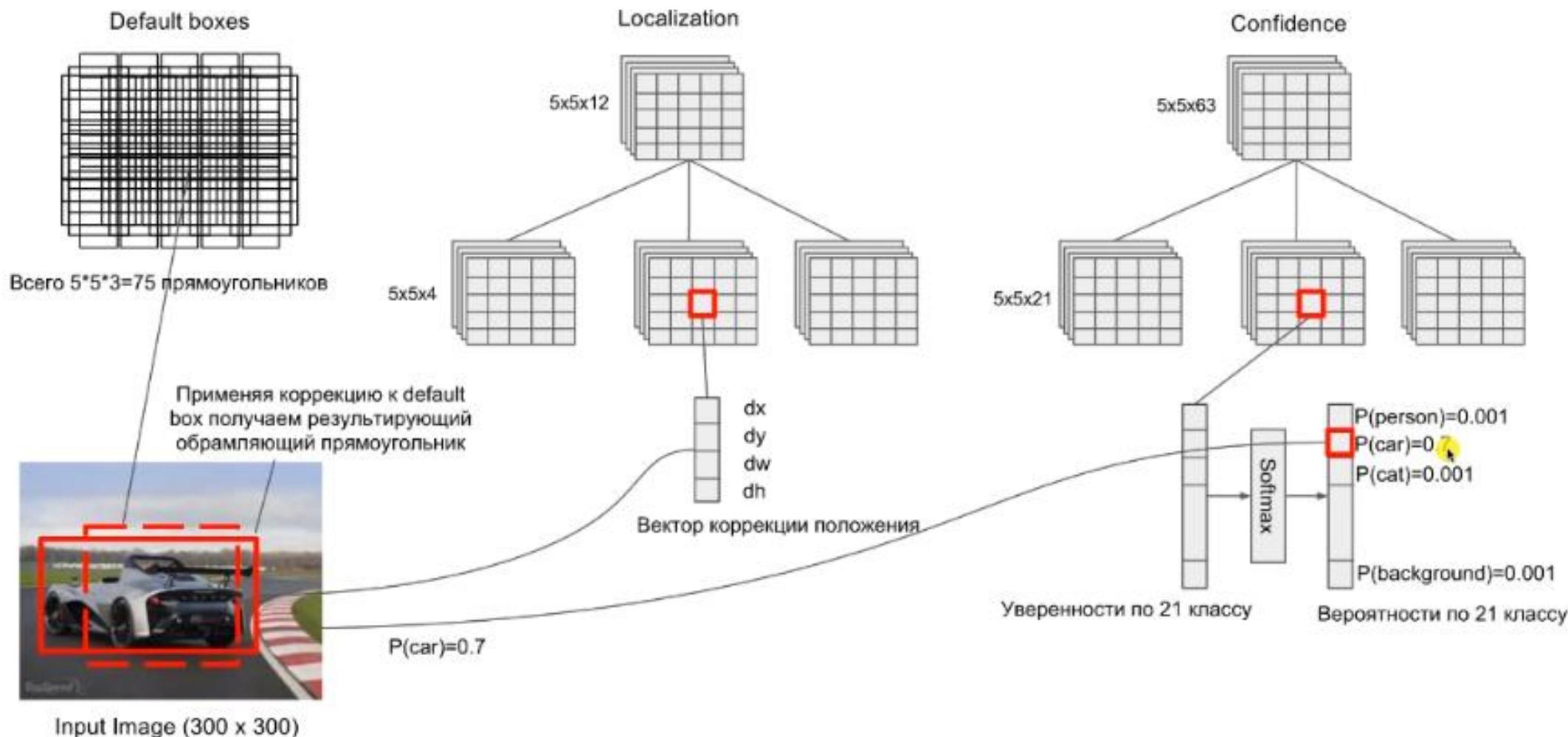
Детектирование объектов



Много регионов «default boxes» на разных масштабах

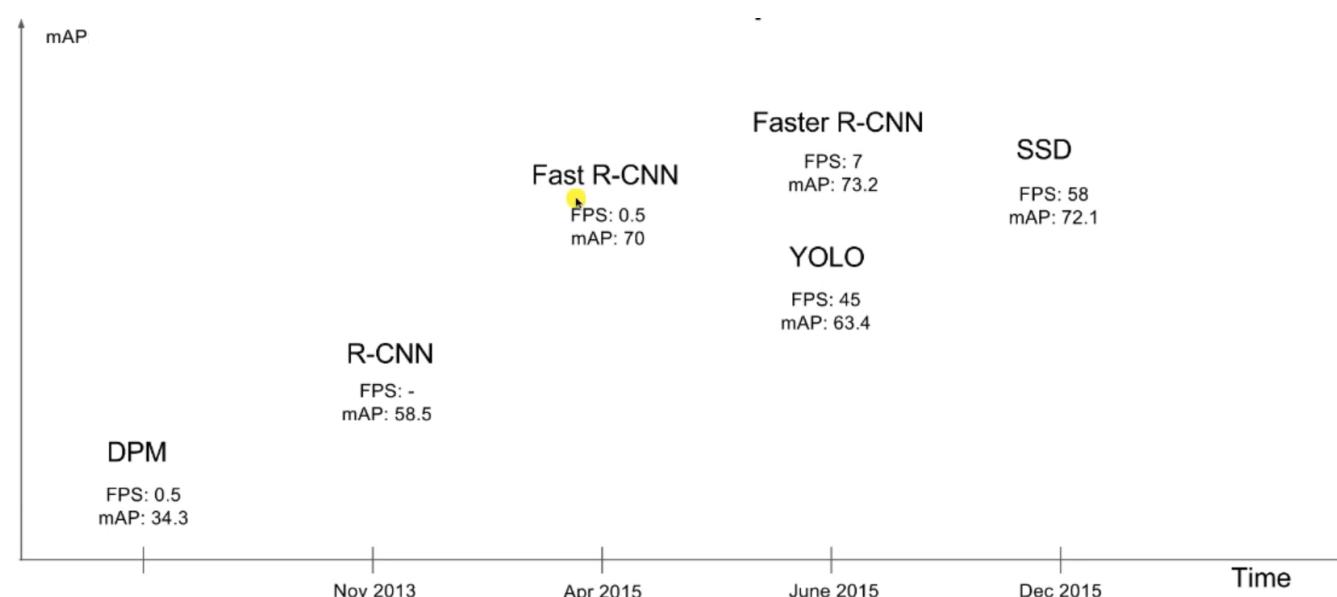
<https://deepsystems.ai>

Детектирование объектов

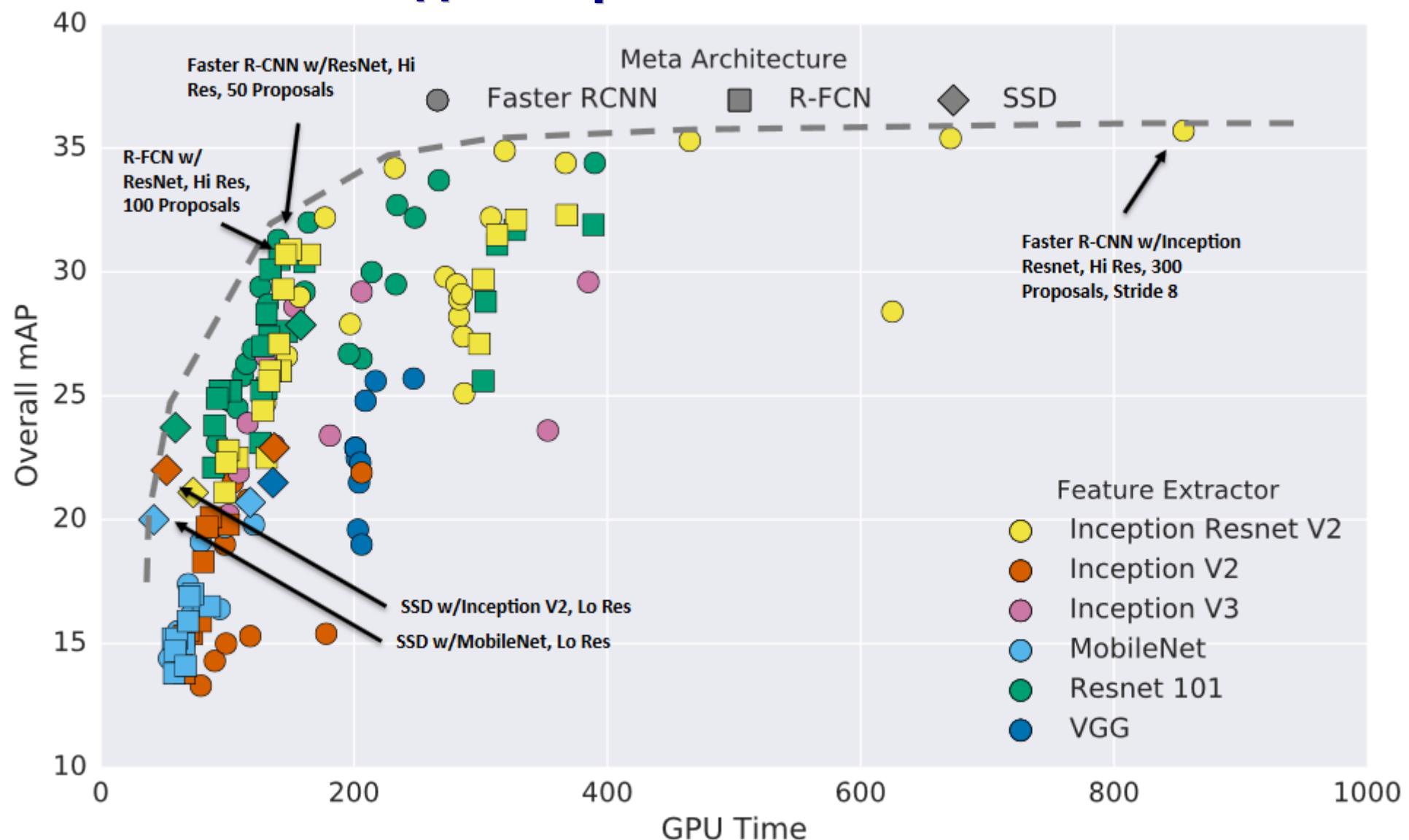


<https://deepsystems.ai>

Детектирование объектов



Детектирование объектов



Speed/accuracy trade-offs for modern convolutional object detectors
[Jonathan Huang и др. 2017 <https://arxiv.org/pdf/1611.10012.pdf>]

Детектирование объектов: выводы

- **локализация объектов не обязательно должна быть классоориентированной (class-specific)!**
- **в одном регионе можно одновременно детектировать несколько объектов!**

Семантическая сегментация

Расстановка меток классов для пикселей



segmented

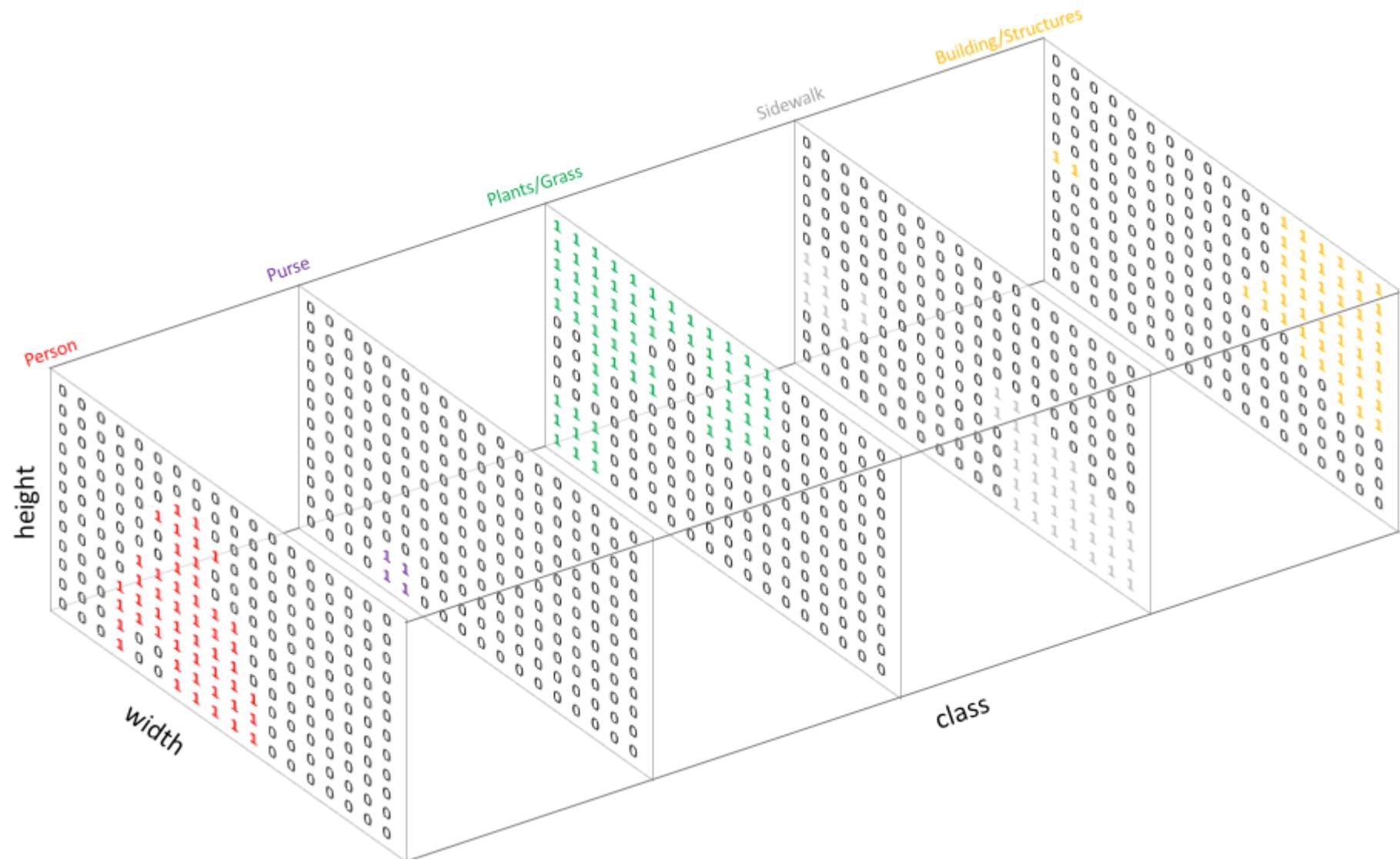
- 1: Person
- 2: Purse
- 3: Plants/Grass
- 4: Sidewalk
- 5: Building/Structures

3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	1	1	3	3	3	3	3	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	1	1	1	1	3	3	3	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	1	1	3	3	3	3	3	5	5	5	5	5
5	5	3	3	3	3	3	3	1	1	3	3	3	5	5	5	5	5	5	5	5	5
4	4	3	4	1	1	1	1	1	1	1	1	1	4	4	4	4	5	5	5	5	5
4	4	3	4	1	1	1	1	1	1	1	1	1	4	4	4	4	4	5	5	5	5
4	4	4	1	1	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4	4	4
3	3	3	1	1	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	1	1	4	4	4	4	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	1	1	4	4	4	4	4	4	4	4	4

<https://www.jeremyjordan.me/semantic-segmentation/>

**Мера качества часто: segmentation accuracy
(для класса – обычная точность по пикселям)**

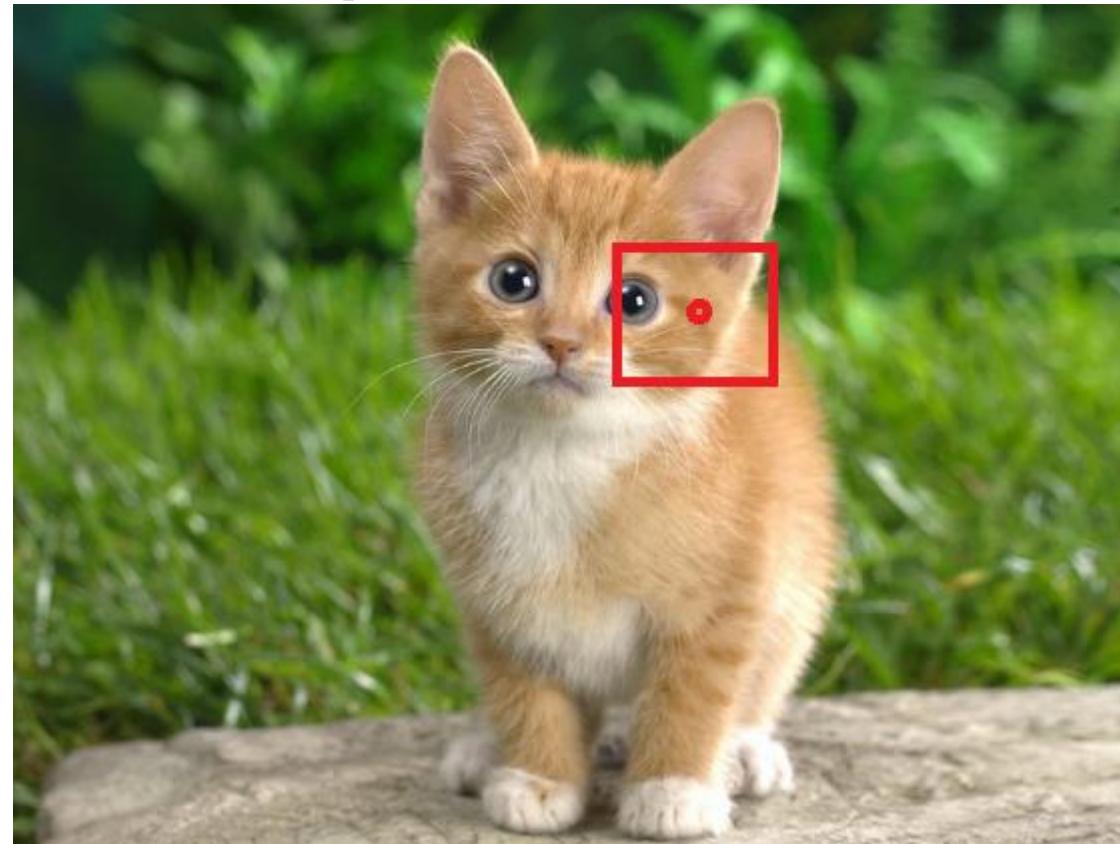
Семантическая сегментация ОНЕ-кодирование целевого вектора



Семантическая сегментация

Надо расставить метки классов пикселям...

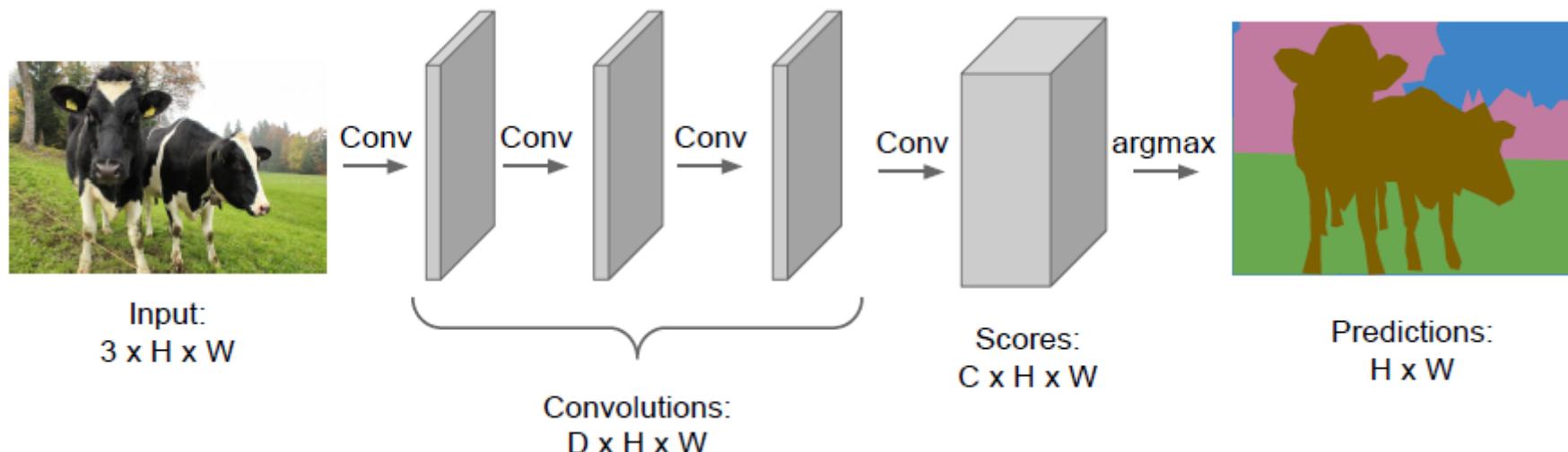
Примитивная идея



**Метка по классификации окна с центром в этом пикселе
Очень долго...**

Семантическая сегментация

**Идея: полностью свёрточная структура сети –
Fully Convolutional Network (FCN)**



Проблемы: высокое разрешение исходного изображения

- надо его понизить (**downsampling**)
 - **pooling**
 - **strided convolution**
- потом надо восстановить (**upsampling**)

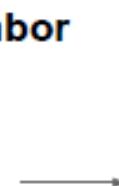
<http://cs231n.stanford.edu>

Семантическая сегментация

Как восстанавливать изображение?

Nearest Neighbor

1	2
3	4



1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Input: 2 x 2

Output: 4 x 4

“Bed of Nails”

1	2
3	4



1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Input: 2 x 2

Output: 4 x 4

Max Pooling

Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8



Rest of the network

Input: 4 x 4

Output: 2 x 2

Max UnpoolingUse positions from
pooling layer

1	2
3	4



0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

Input: 2 x 2

Output: 4 x 4

Обратная свёртка (upconvolution / conv-transpose)

Напомним...

$$\begin{aligned}
 & \begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{pmatrix} * \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix} = \\
 & = \underbrace{\begin{pmatrix} k_{11} & k_{12} & 0 & k_{21} & k_{22} & 0 & 0 & 0 & 0 \\ 0 & k_{11} & k_{12} & 0 & k_{21} & k_{22} & 0 & 0 & 0 \\ 0 & 0 & 0 & k_{11} & k_{12} & 0 & k_{21} & k_{22} & 0 \\ 0 & 0 & 0 & 0 & k_{11} & k_{12} & 0 & k_{21} & k_{22} \end{pmatrix}}_H \cdot \begin{pmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{31} \\ x_{32} \\ x_{33} \end{pmatrix}.
 \end{aligned}$$

Обратная свёртка (upconvolution / conv-transpose)

Можно...

$$\begin{pmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{pmatrix} *^T \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix} = H^T \cdot \begin{pmatrix} z_{11} \\ z_{21} \\ z_{12} \\ z_{22} \end{pmatrix}$$

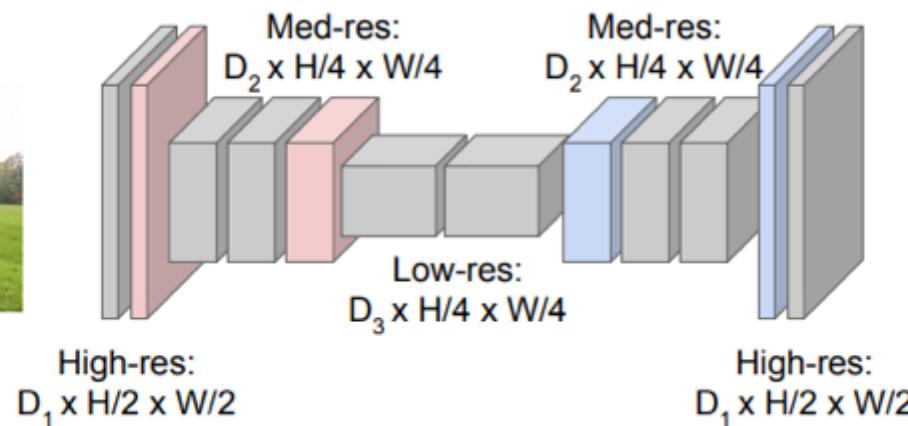
Обратная свёртка увеличивает пространственное разрешение...

Семантическая сегментация

В итоге что-то такое...



Input:
 $3 \times H \times W$



High-res:
 $D_1 \times H/2 \times W/2$

High-res:
 $D_1 \times H/2 \times W/2$

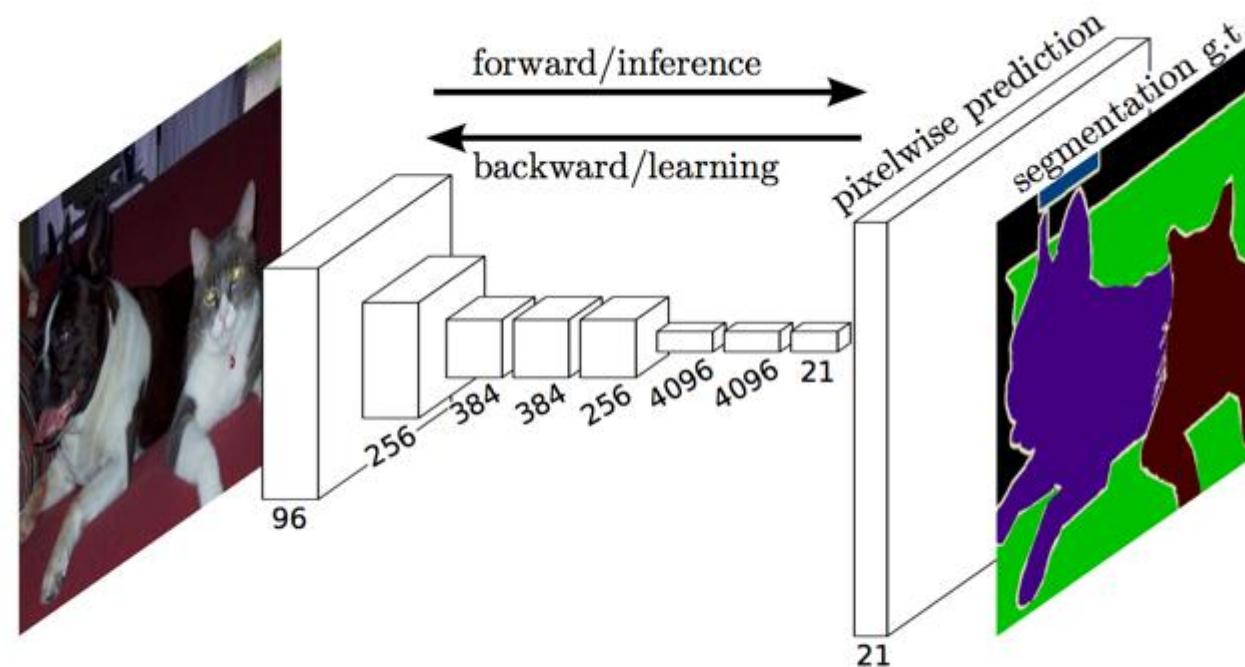


Predictions:
 $H \times W$

уменьшаются размеры, но увеличивается число каналов!

Fully Convolutional Networks (FCN)

Первая успешная попытка..



«Fully Convolutional Networks for Semantic Segmentation» [Long et al., 2015]

https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

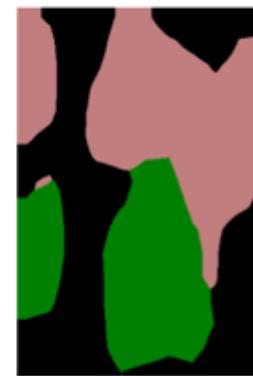
Fully Convolutional Networks (FCN)

Проблема

Ground truth target



Predicted segmentation

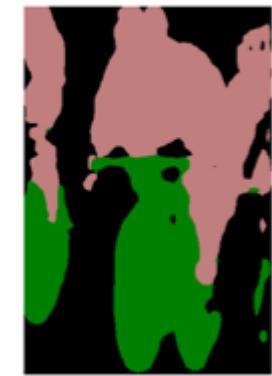


Сейчас сделаем так...

Ground truth target



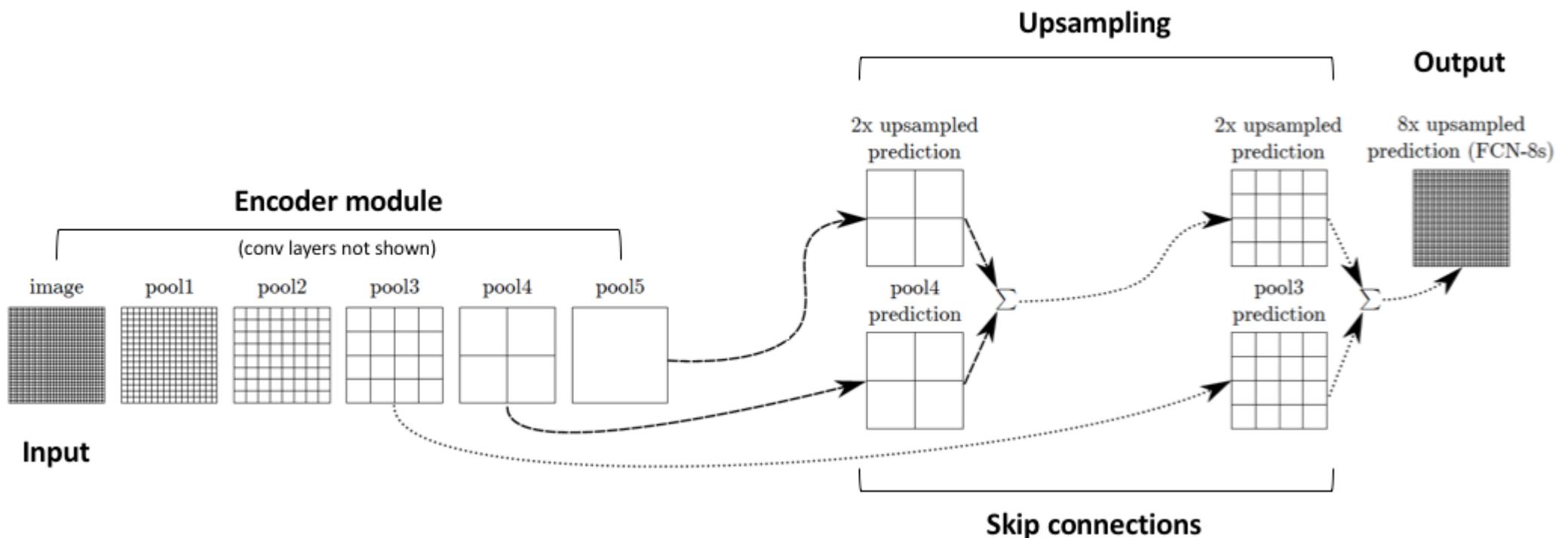
Predicted segmentation



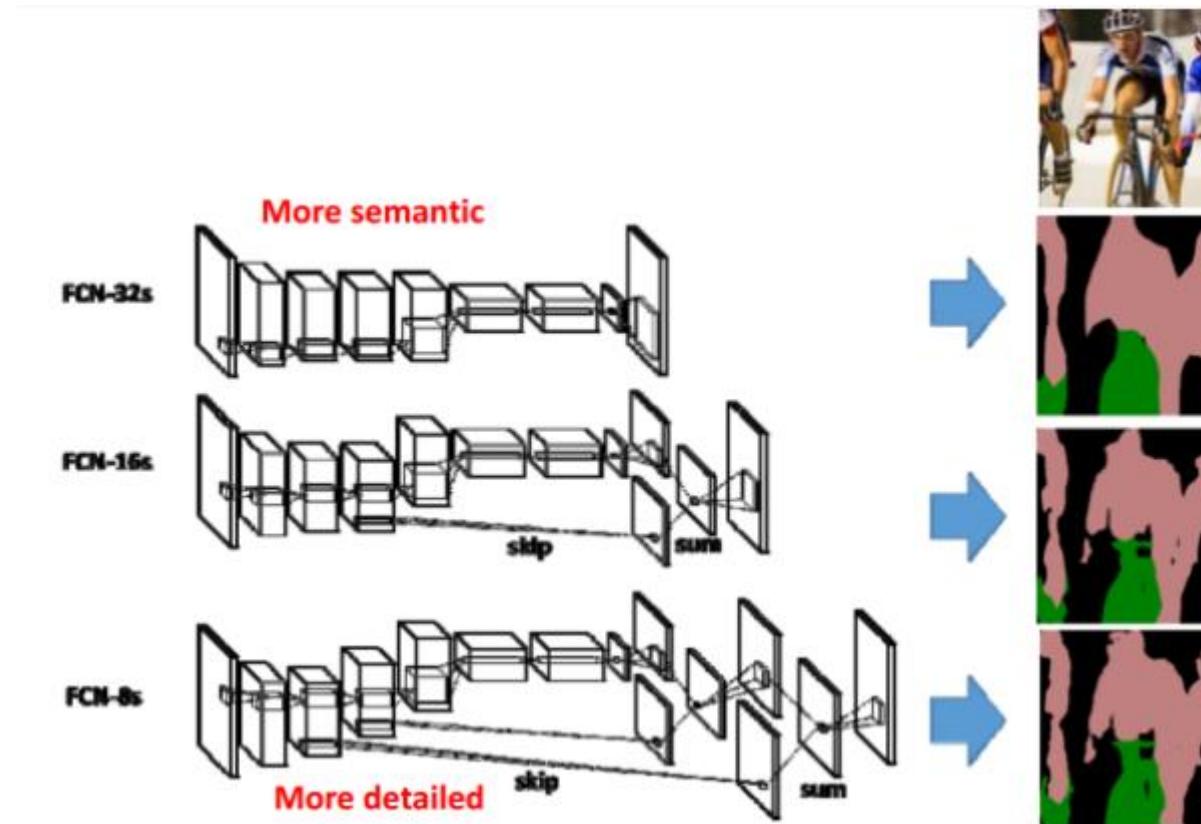
**Понимаем «что» (на последних слоях),
но забываем «где» (было на первых)**

Fully Convolutional Networks (FCN)

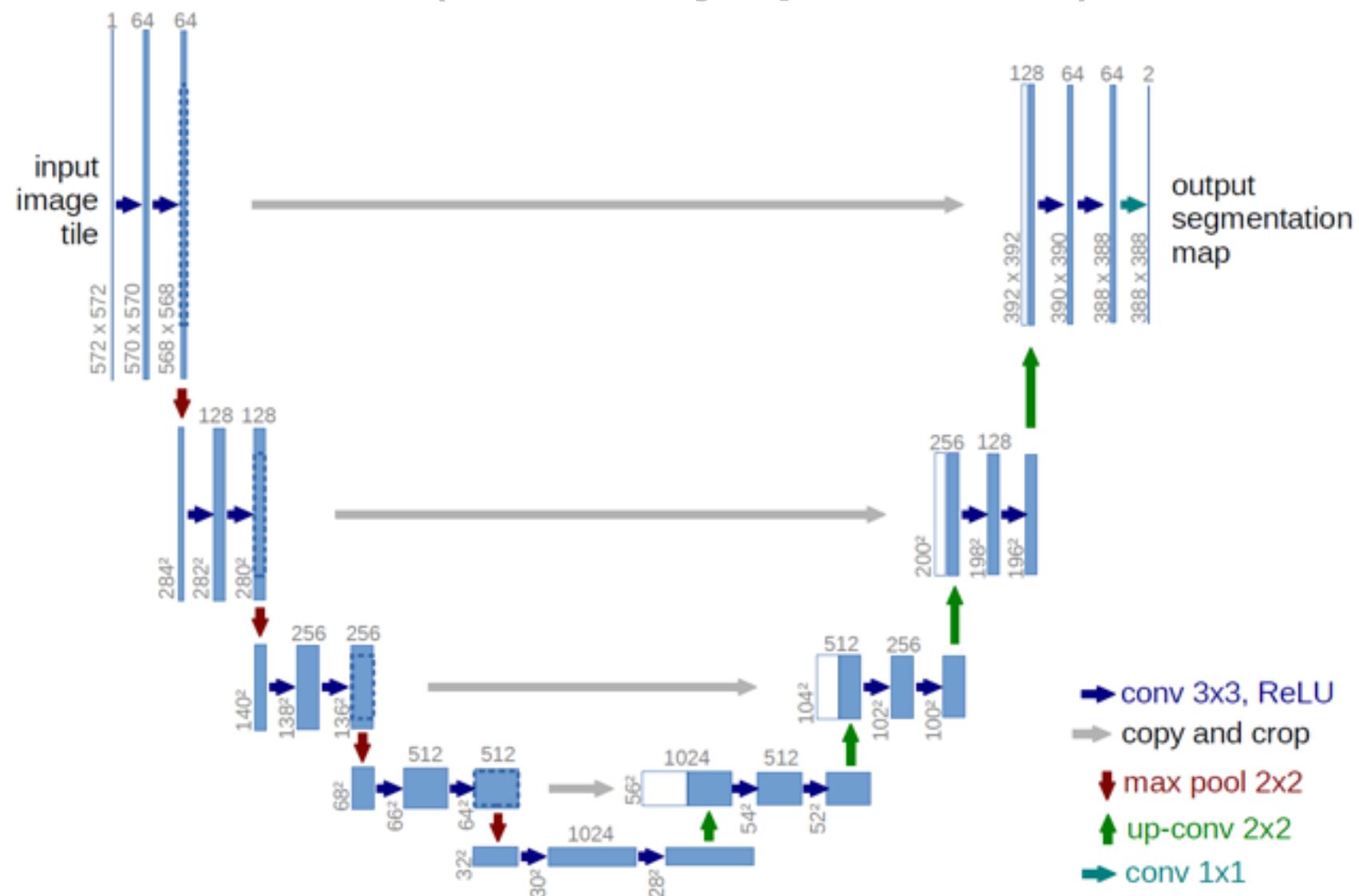
Решение – прокидывание связей (skip connections)



Эффект прокидывания связей



U-Net (самая популярная сейчас)



«U-Net: Convolutional Networks for Biomedical Image Segmentation» [Ronneberger O. и др., 2015
<https://arxiv.org/abs/1505.04597>】

U-Net (самая популярная сейчас)

Левая часть

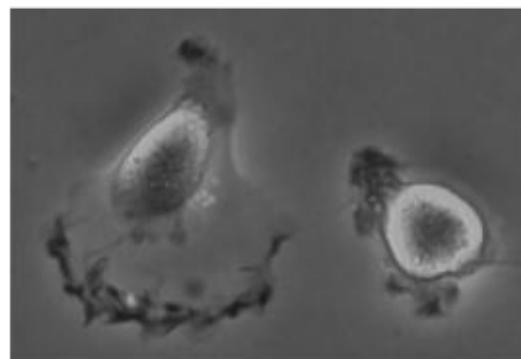
3×3-свёртки, ReLU, 2×2-пулинг (шаг=2)

Правая часть

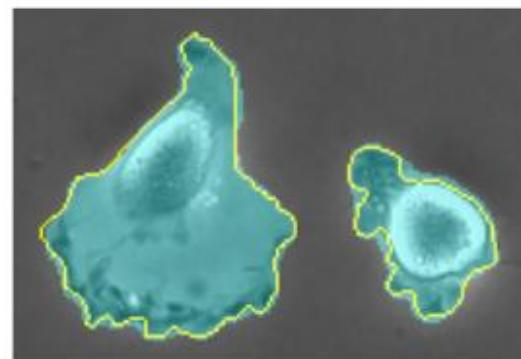
2×2 обратная свёртка (половинит число каналов), конкатенация с обрезанными признаками из левой части (там больше H×W), 3×3-свёртки, ReLU

всего 23 свёрточных слоя, реализация – Caffe

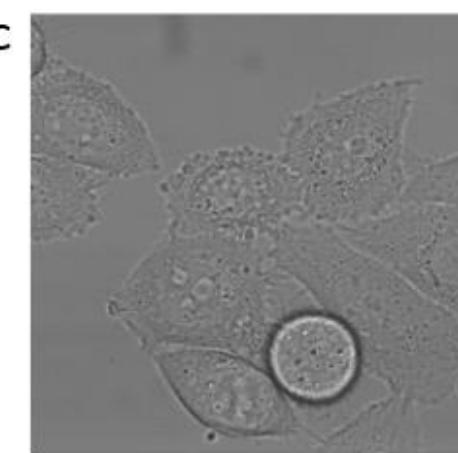
a



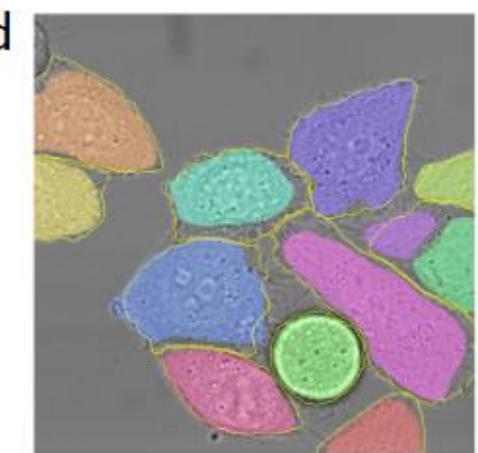
b



c

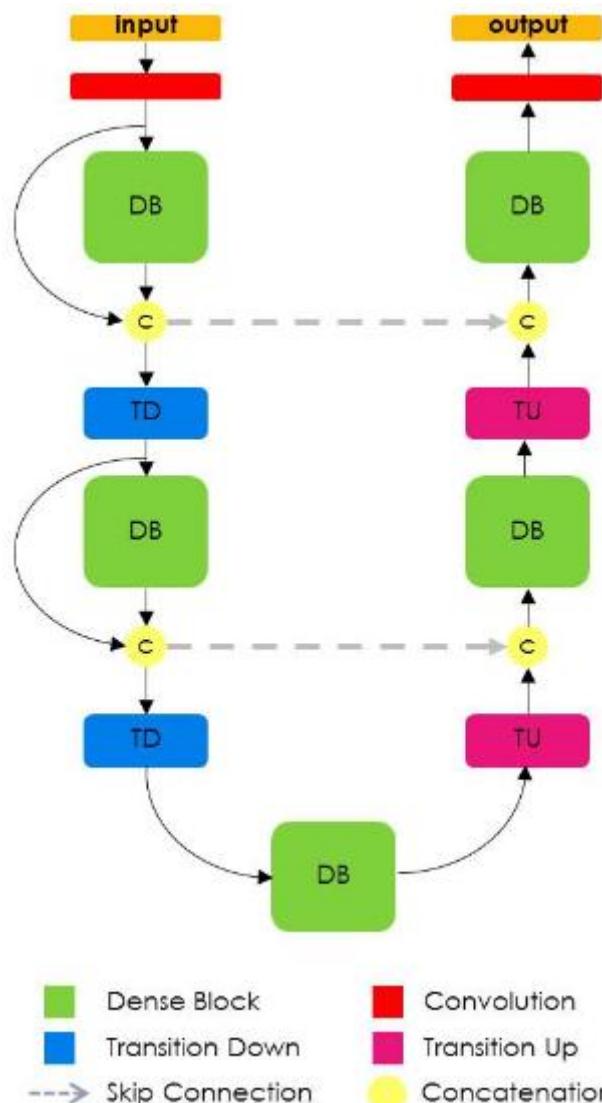


d



Польза аугментации в некоторых задачах!

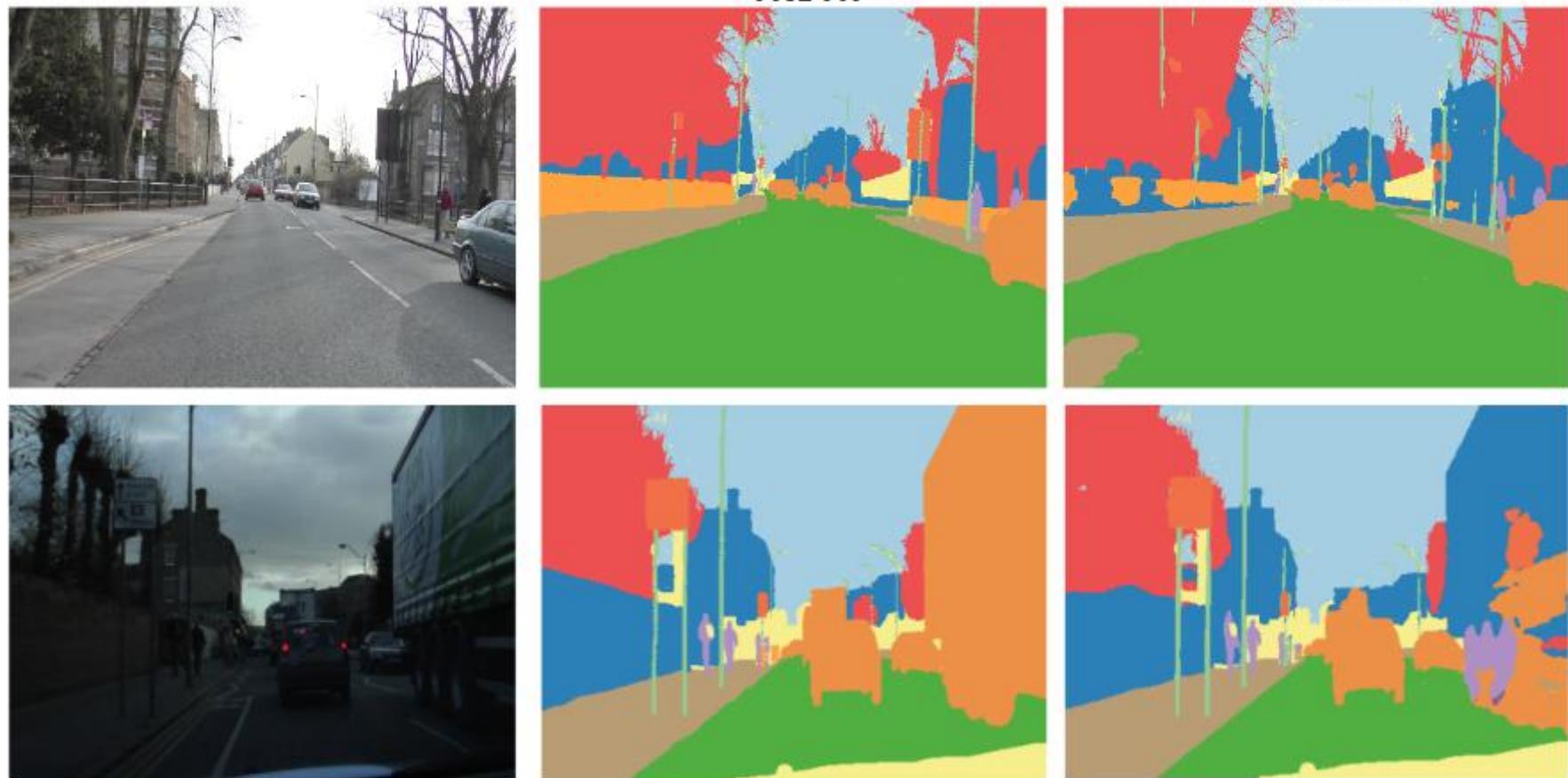
Это была фишка статьи – данных можно сделать много!

DenseNet + U-Net

**HeUniform + RMSprop + weight
Dropout + BN**

Input, $m = 3$
3×3 Convolution, $m = 48$
DB (4 layers) + TD, $m = 112$
DB (5 layers) + TD, $m = 192$
DB (7 layers) + TD, $m = 304$
DB (10 layers) + TD, $m = 464$
DB (12 layers) + TD, $m = 656$
DB (15 layers), $m = 896$
TU + DB (12 layers), $m = 1088$
TU + DB (10 layers), $m = 816$
TU + DB (7 layers), $m = 578$
TU + DB (5 layers), $m = 384$
TU + DB (4 layers), $m = 256$
1×1 Convolution, $m = c$
Softmax

«The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation» [Jégou S. и др., 2017 <https://arxiv.org/abs/1611.09326>]

DenseNet + U-Net

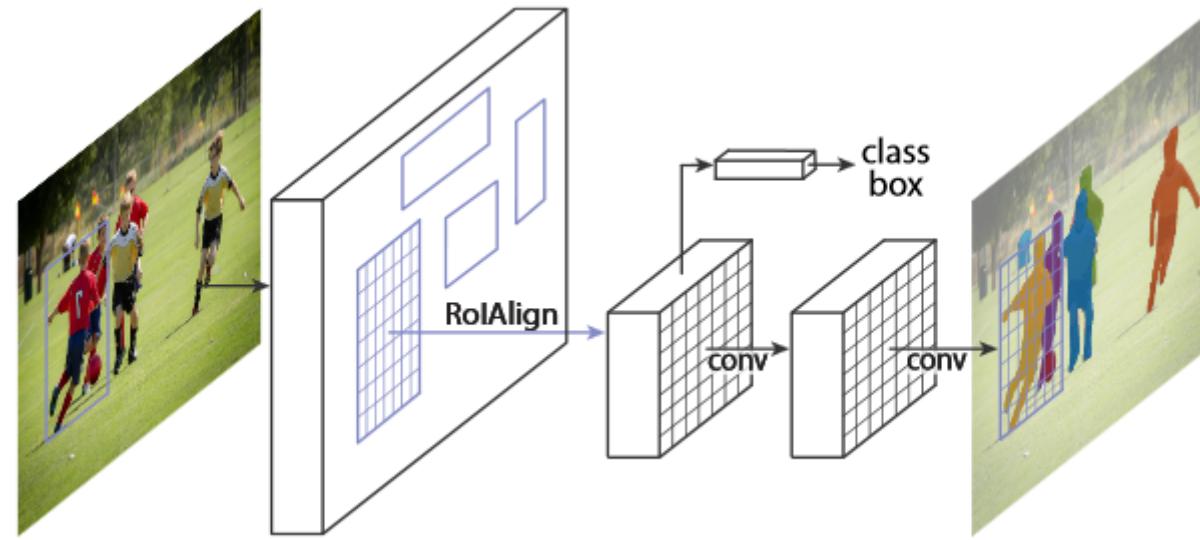
Сегментация объектов

не просто найти рамку, а выделить объект!



Mask R-CNN <https://arxiv.org/pdf/1703.06870.pdf>

Mask R-CNN



**Faster R-CNN + определение сегментационной маски
(маленькая FCN)**

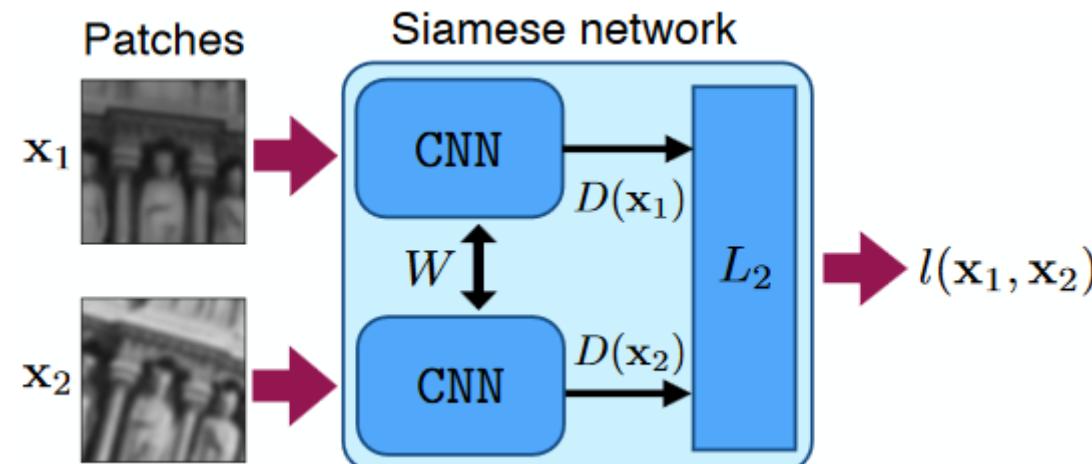
+ борьба за более точное определение границ

Поиск изображений

Найти похожие изображения

Идея: изображение → вектор, поиск по векторам

Сиамские сети

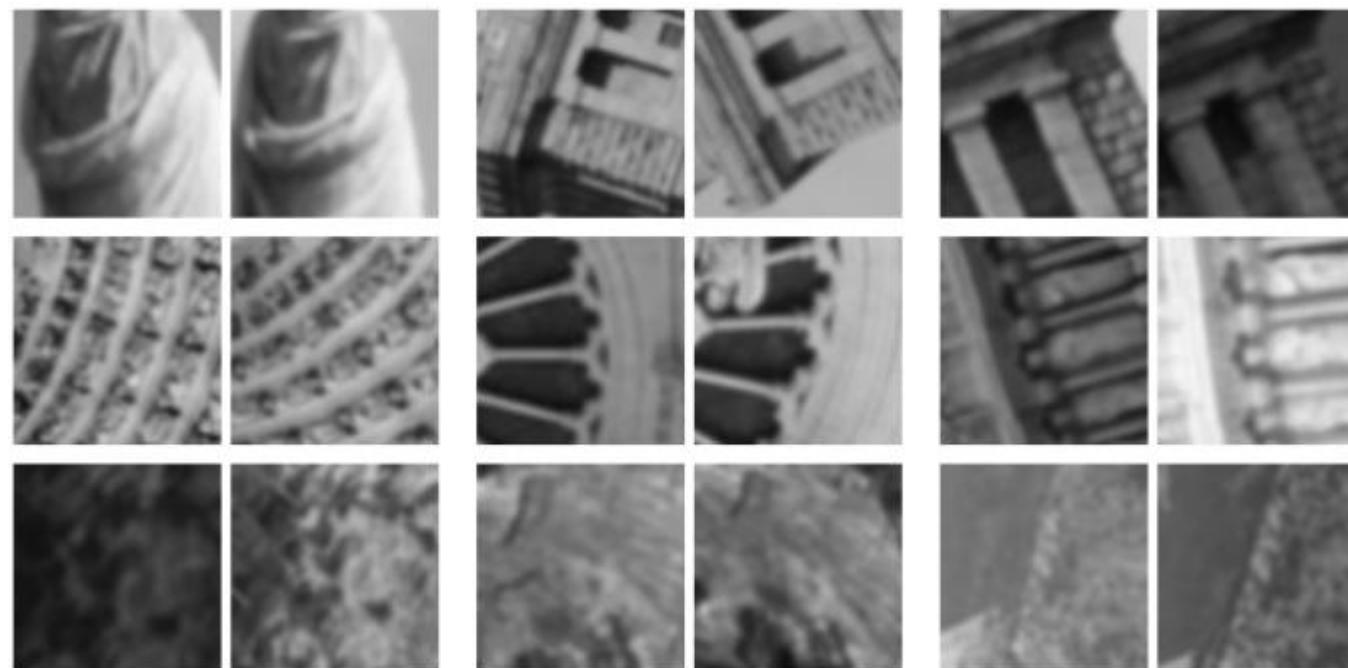


Simo-Serra et al. «Discriminative Learning of Deep Convolutional Feature Point Descriptors», 2015 //
<http://icwww.epfl.ch/~trulls/pdf/iccv-2015-deepdesc.pdf>

**Учим дескрипторы: L2-расстояние было на них адекватно
Раньше дескрипторы строились вручную
Hinge Embedding Loss**

$$l(x_1, x_2) = \begin{cases} \|D(x_1) - D(x_2)\|_2, & p_1 = p_2 \\ \max(0, C - \|D(x_1) - D(x_2)\|_2), & p_1 \neq p_2 \end{cases}$$

Сиамские сети



Пары похожих изображений для обучения

Multi-View Stereo Dataset (MVS)

> 1.5M чб 64×64 с разных ракурсов

3D-реконструкций разных архитектурных сооружений

Сиамские сети

Layer	1	2	3
Input size	64×64	29×29	8×8
Filter size	7×7	6×6	5×5
Output channels	32	64	128
Pooling & Norm.tion	2×2	3×3	4×4
Nonlinearity	Tanh	Tanh	Tanh
Stride	2	3	4

Tanh > ReLU

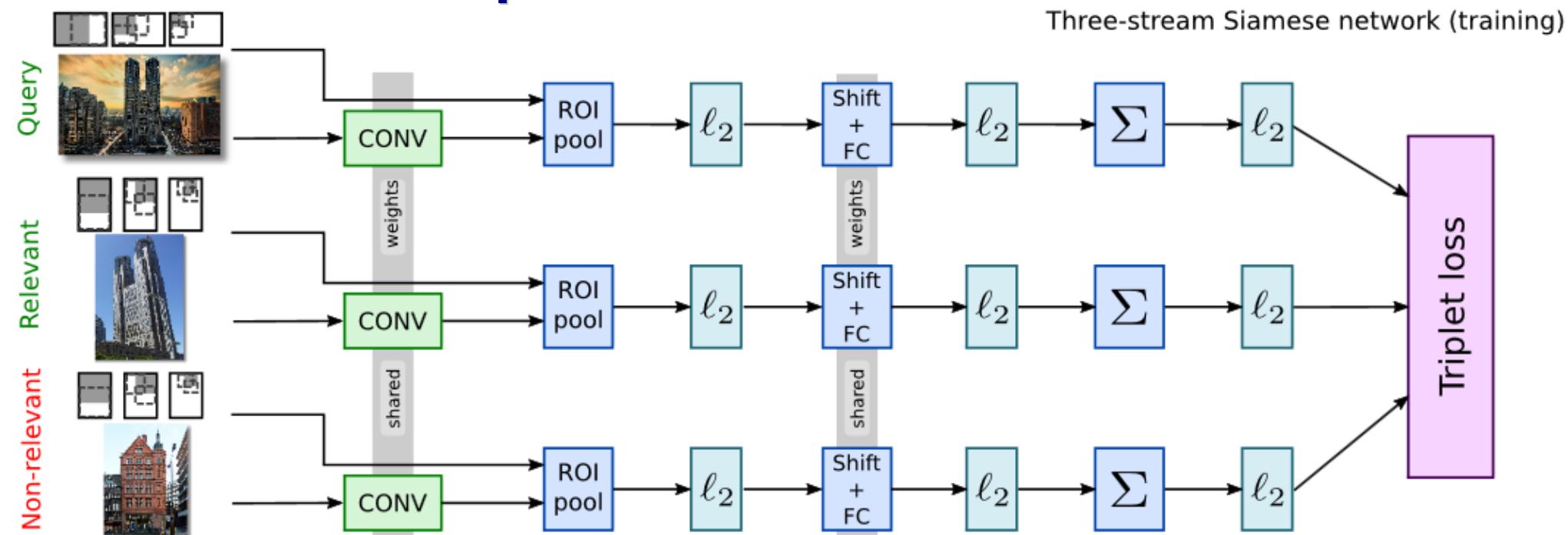
L2-pooling > max-pooling

Нормировки!!!

Subtractive normalization

(вычитаем взвешенное среднее по 5×5 -ядру)

Triplet Loss в Сиамских сетях



Все три сети одинаковые!!! Нас интересуют признаки на выходе

Свёрточные слои и натренированной VGG16 → локальные

признаки, не зависящие от размеров

МАХ-pooling в разных регионах

PCA (сдвиг + FC-слой)

L2-нормализация и сумма

(поэтому нет зависимости числа признаков от числа регионов)

Triplet Loss в Сиамских сетях

В итоге

скалярное произведение пр-ов ~ many-to-many region matching

Все операции дифференцируемые!

Обучение – Triplet Loss

запрос(q) – релевантный (+) – нерелевантный (–) ответ

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

Gordo et al. «Deep Image Retrieval», 2016 <https://arxiv.org/pdf/1604.01325.pdf>

Поиск изображений

Before
training



After
training

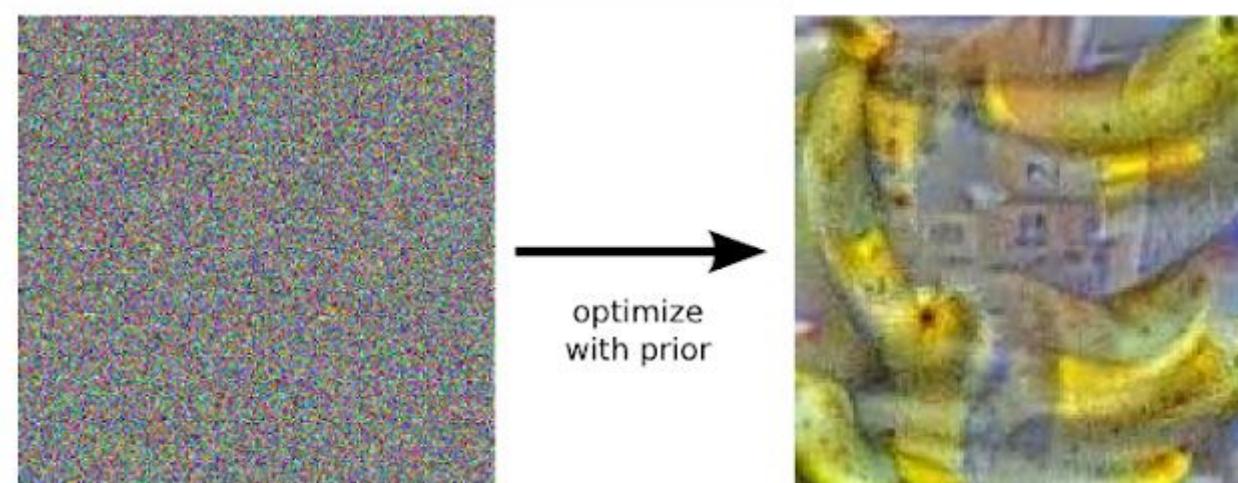


Генерация изображений

Натренировали НС-классификатор

Дали на вход случайный шум и стали градиентно менять изображение: увеличить уверенность, что это «банан»

Есть хак: соседние пиксели коррелированы



Генерация изображений



Hartebeest



Measuring Cup



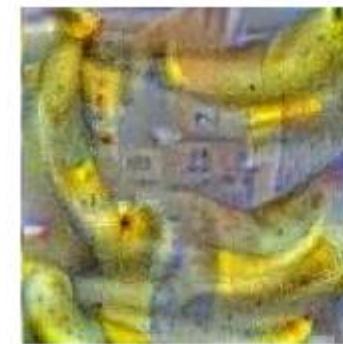
Ant



Starfish



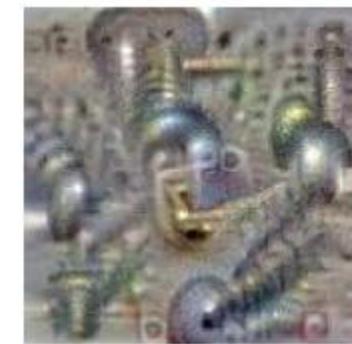
Anemone Fish



Banana



Parachute



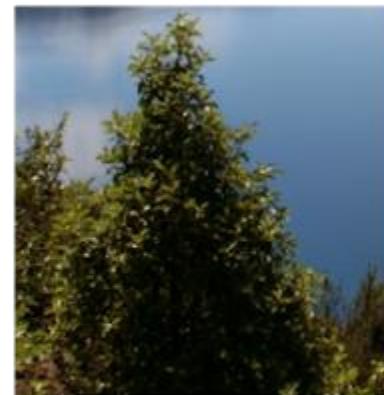
Screw

**Можно максимизировать не вероятность класса,
а активацию какого-то нейрона**

Генерация изображений



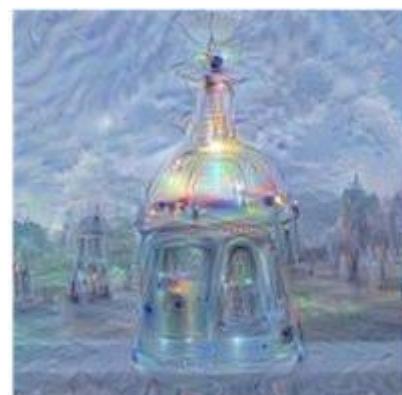
Horizon



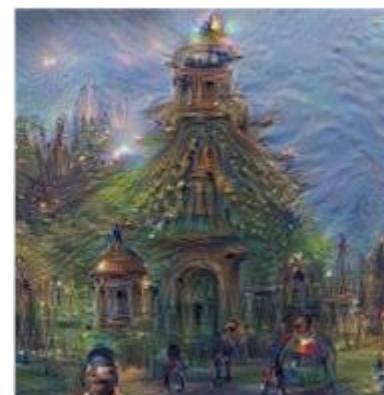
Trees



Leaves



Towers & Pagodas



Buildings



Birds & Insects

Можем превращать изображения в изображения другого класса!

Генерация изображений

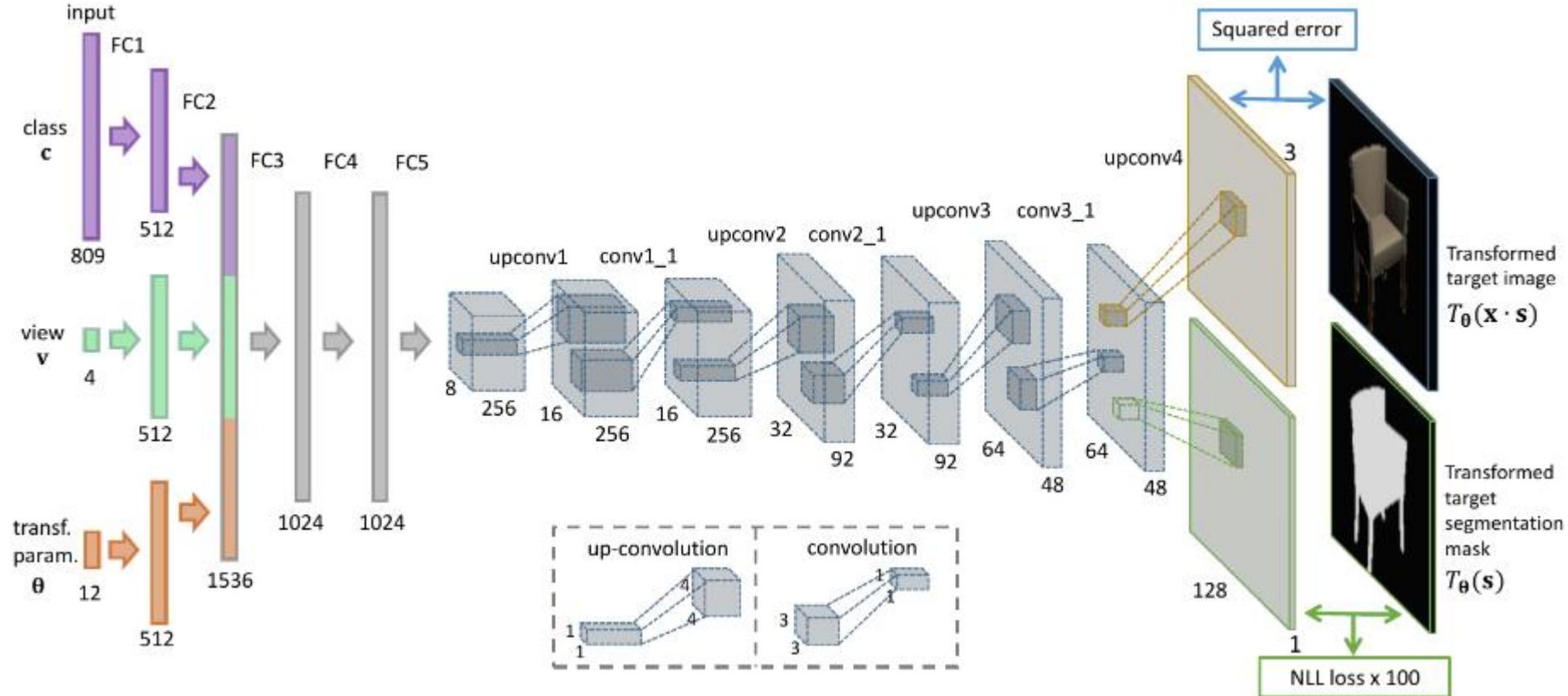
Зачем: чтобы понять, что выучились тому...



Класс «гантель» – всегда генерируется гантель с рукой!

<https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>

Генерация изображений по стилю, ориентации (относительно камеры), цвету, яркости и т.п.

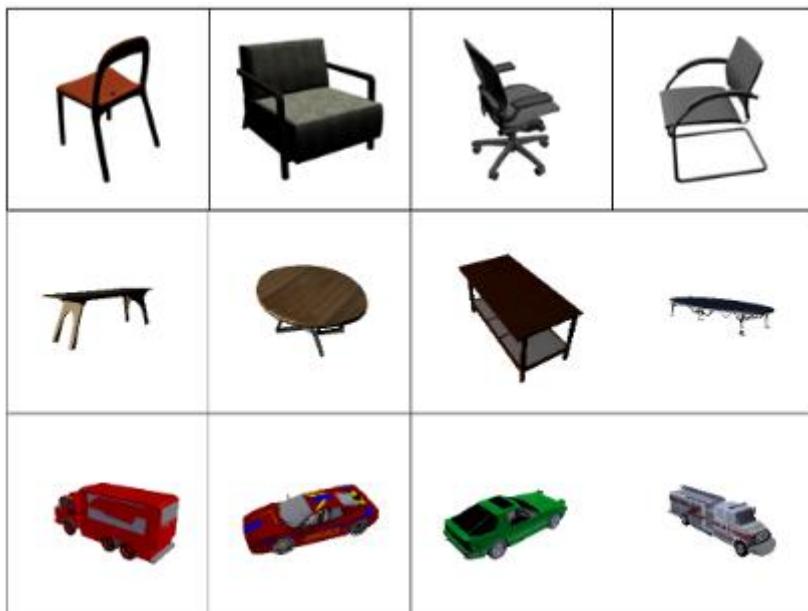


как бы развёрнутая CNN

параметры → 1024 признака → изображение и маска сегментации

Dosovitskiy A. и др. «Learning to Generate Chairs, Tables and Cars with Convolutional Networks», 2017 // <https://arxiv.org/pdf/1411.5928.pdf>

Генерация изображений



Выборка:

**(ОНЕ-стиль, угол камеры, доп.
параметры)**

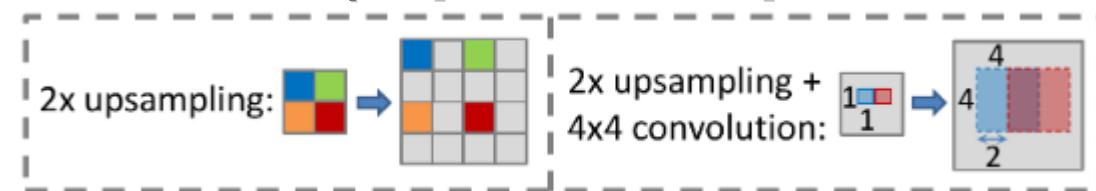
→

**(изображение,
маска сегментации)**

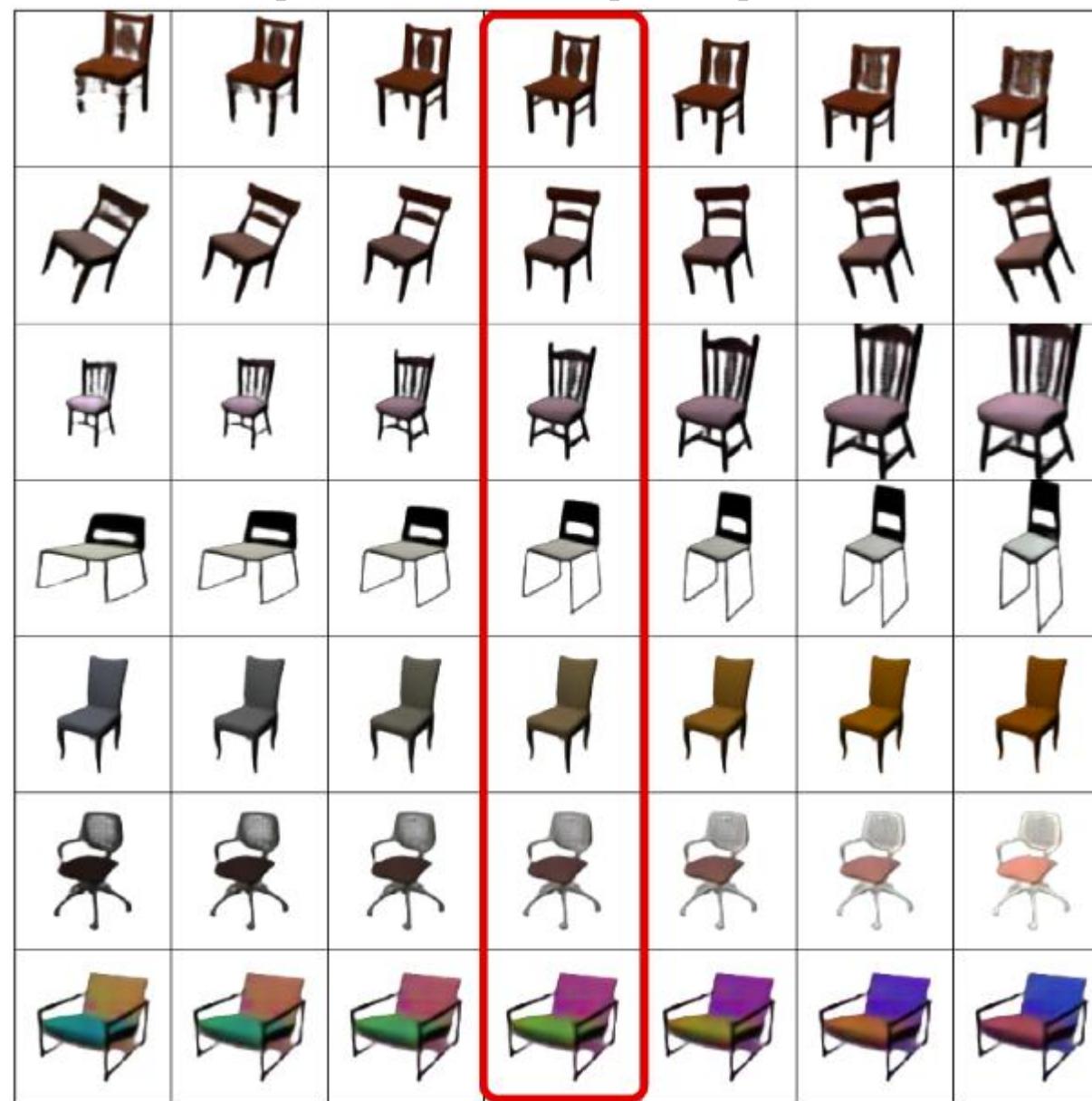
**Маска для простоты (можно
забелить фон)**

Деконволюционные слои (deconvolutional layers)

Деконволюция (обратная операция к свёртке)



Генерация изображений – преобразование эталонов



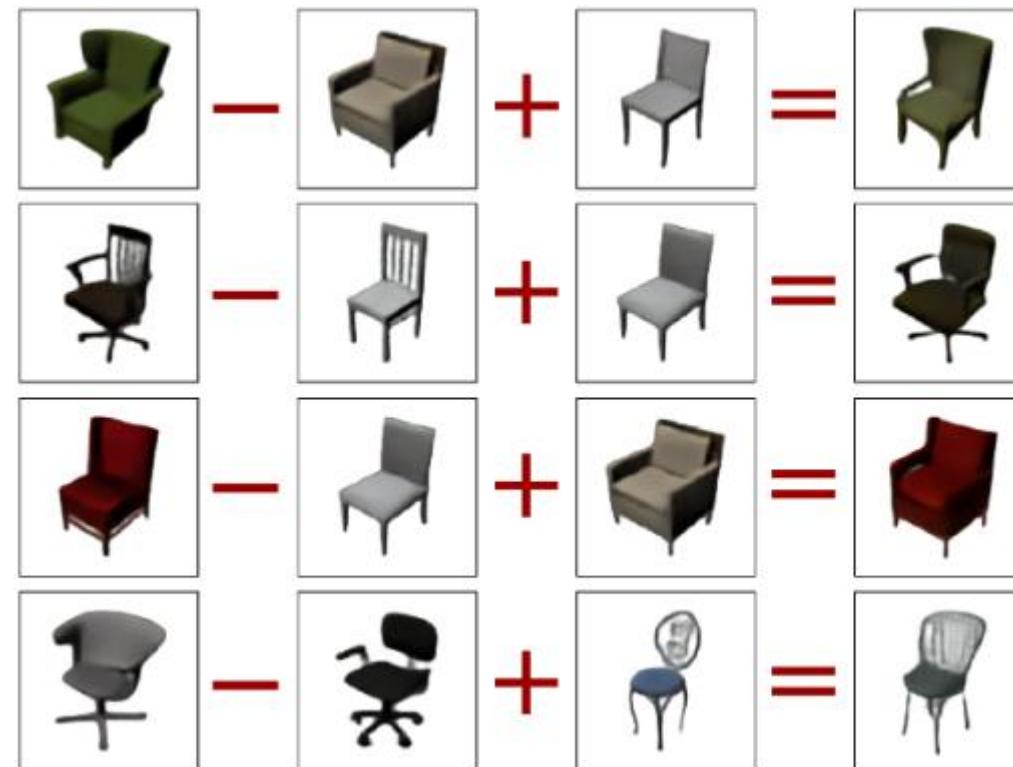
Генерация изображений

Линейная комбинация стилей



Генерация изображений

Арифметика над признаками

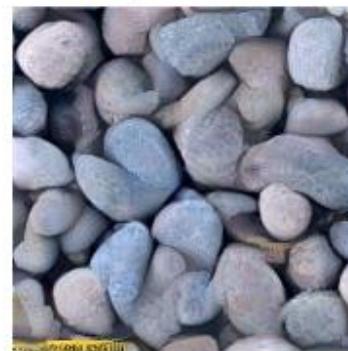


Генерация текстур

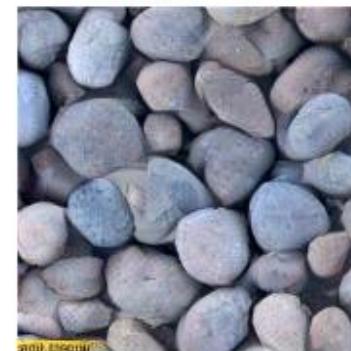
A ~1k parameters



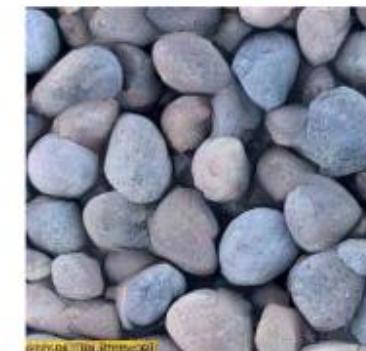
~10k parameters



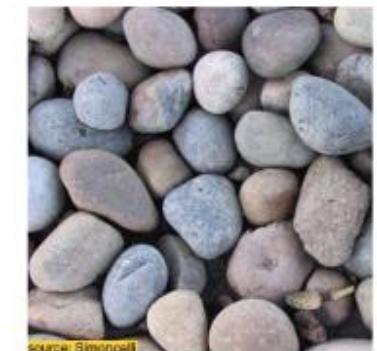
~177k parameters



~852k parameters

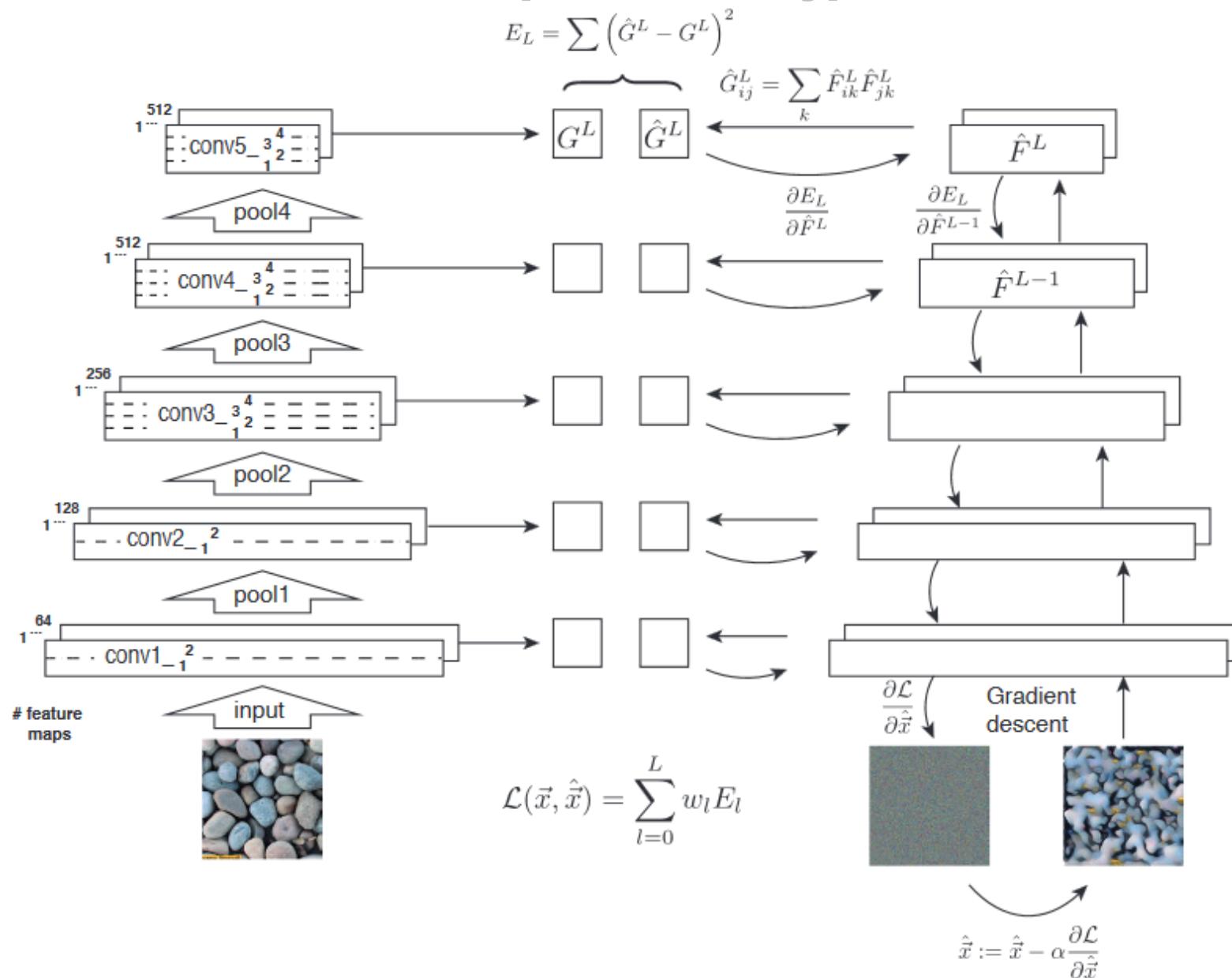


original



Leon A. Gatys «Texture Synthesis Using Convolutional Neural Networks» <https://arxiv.org/pdf/1505.07376.pdf>

Генерация текстур



Генерация текстур

Идея:

Берём какую-нибудь CNN, например VGG-19 (её свёрточную часть)

На этой части изображение ~ тензор $w \times h \times k$

k – число каналов (м.б. 512)

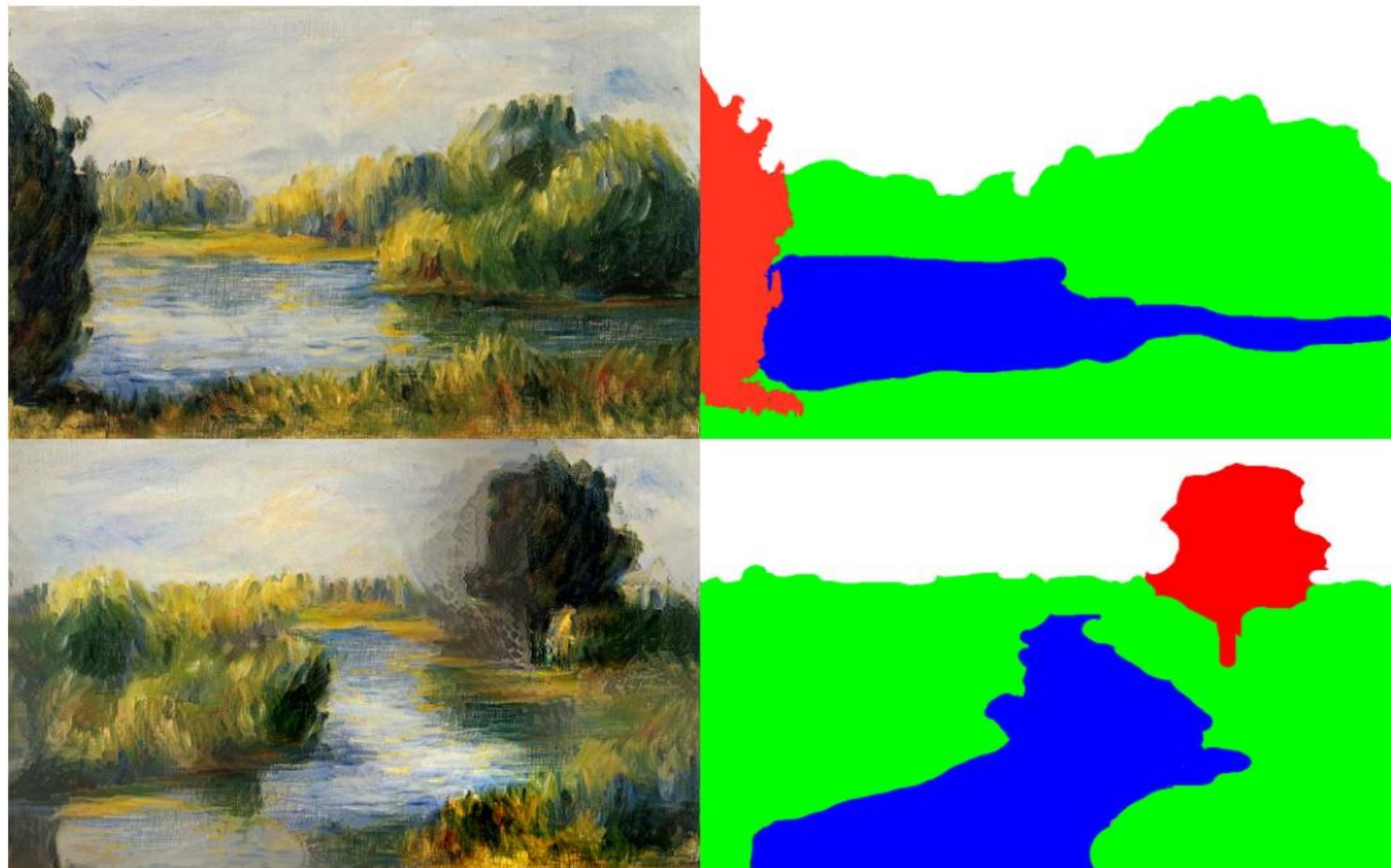
Считаем матрицу Грама $k \times k$

**ij -й элемент – корреляция $w \times h$ -мерных векторизаций
i-го канала и j-го**

**Вторая точно такая же сеть – на вход шум – тоже считаем матрицы
Грама**

**Теперь обучаем вторую НС так, чтобы её матрицы Грама были
похожи на матрицы Грама первой НС**

Генерация пейзажей



<https://github.com/DmitryUlyanov/fast-neural-doodle>

Генерация пейзажей

Вход: картина + сегментация + новая сегментация

Для каждого сегмента получаем описание стиля (матрицы Грамма разных слоёв) и генерируем новое изображение

Стилизация



Leon A. Gatys и др. «A Neural Algorithm of Artistic Style», 2015 //

<https://arxiv.org/pdf/1508.06576.pdf>

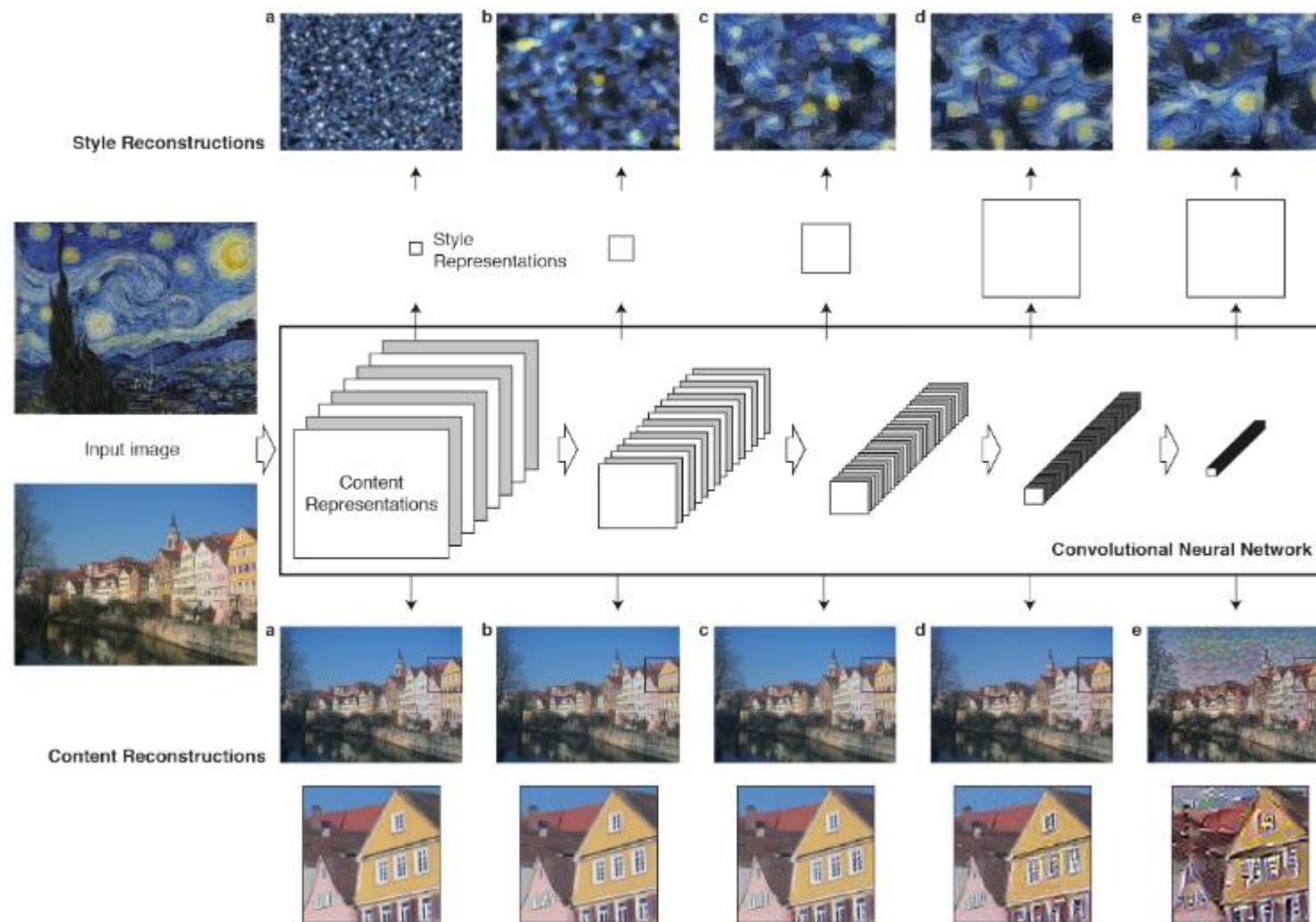
Что такое стиль?

Не должен зависеть от координат...

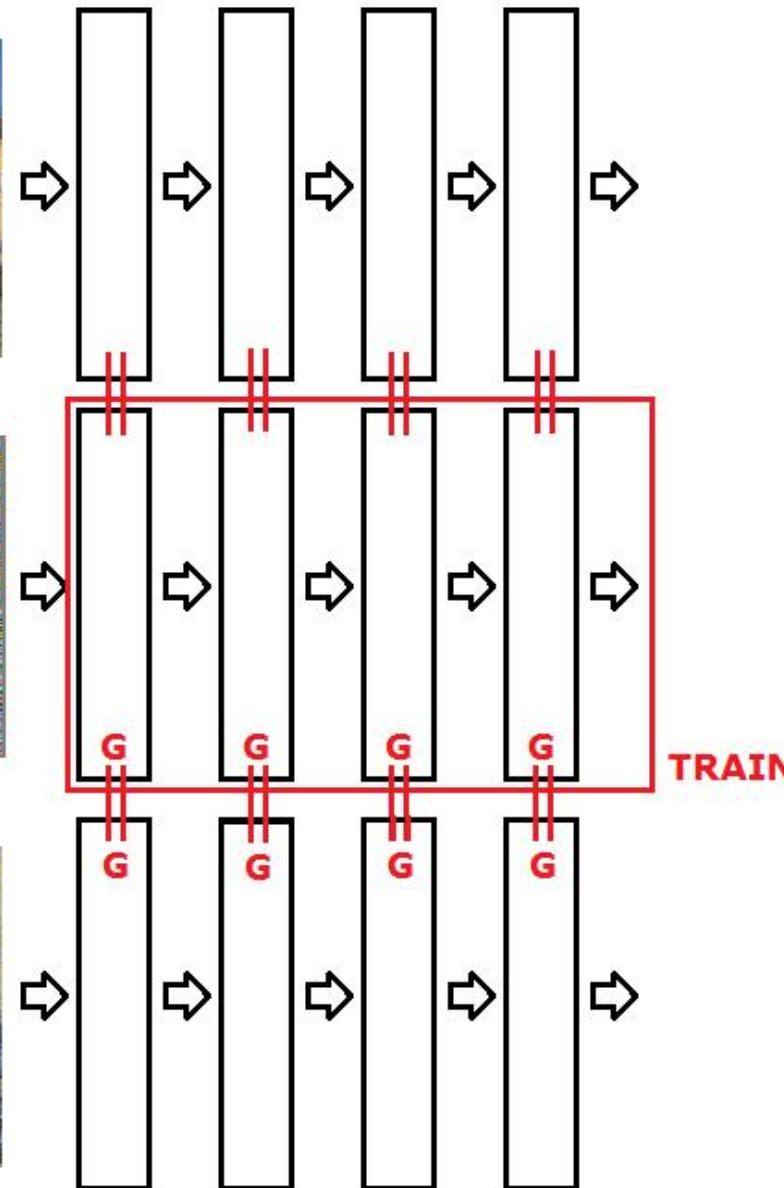
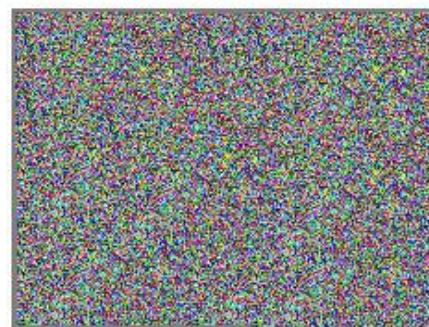
- 1) усредним признаки по пространственным измерениям;)
- 2) сложнее – посчитаем ковариации (больше информации)

**Идея: похожесть на фотографию + похожесть на стиль рисунка
= похожесть признакового описания + похожесть матриц Грама**

Стилизация

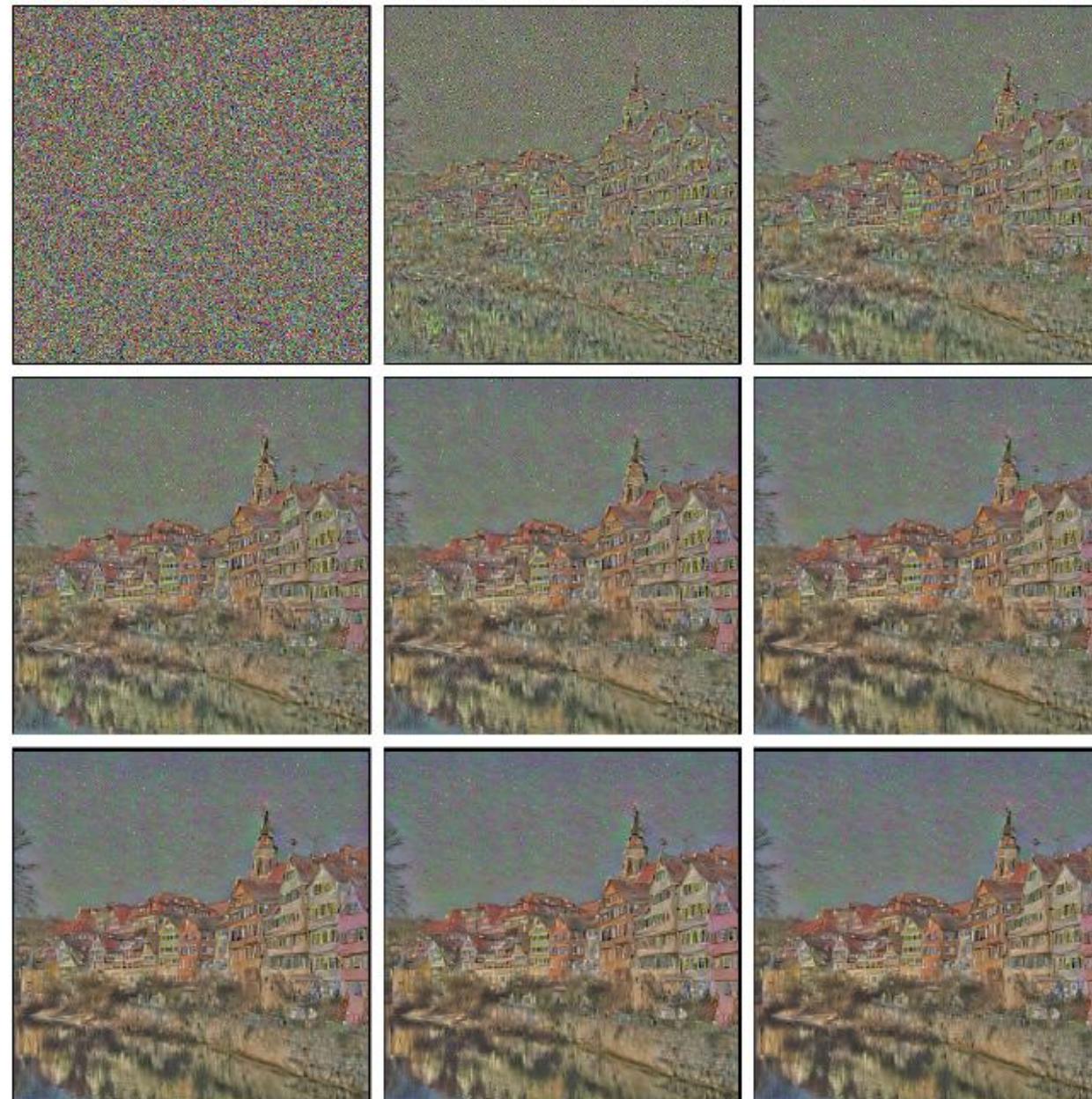


Стилизация

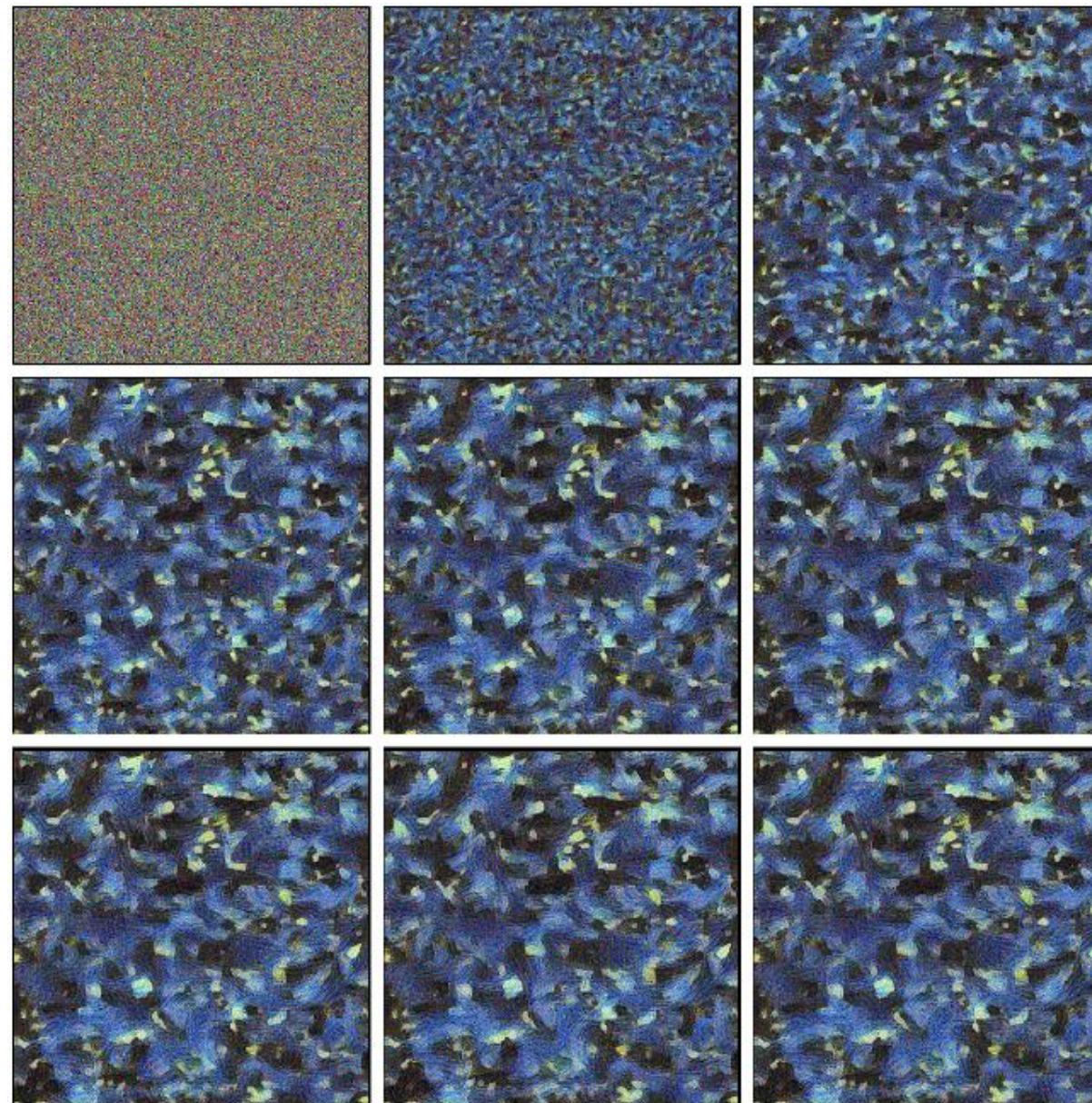


на самом деле с равенствами не совсем так...

Оптимизация по содержанию

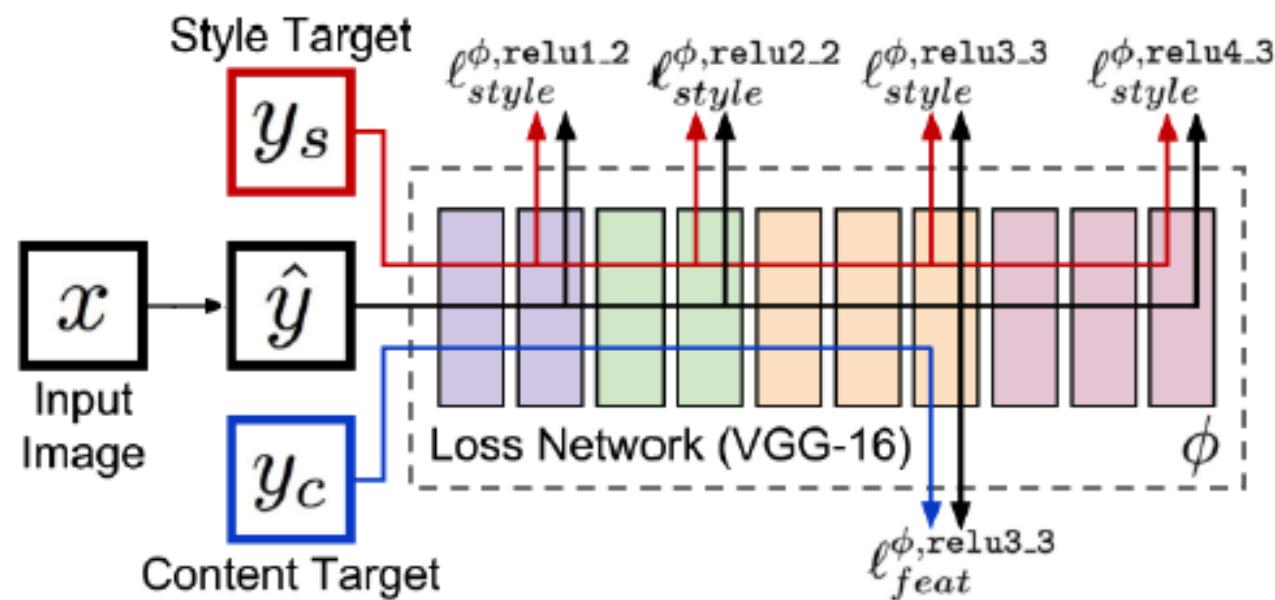


Оптимизация по стилю

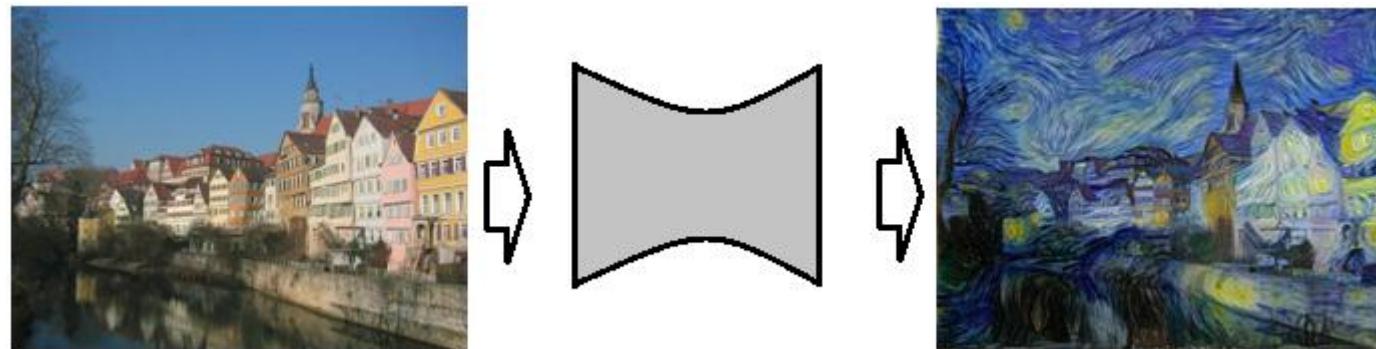


Стилизация

$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x})$$



Быстрая стилизация



**пусть будет всего одна сеть,
но она умеет делать конкретную стилизацию!**

Кстати,

- зашумлять изображения, чтобы была устойчива к шуму**

[Дмитрий Ульянов]

Дальше – GAN

Курсы

Convolutional Neural Networks for Visual Recognition

<http://cs231n.stanford.edu/>