

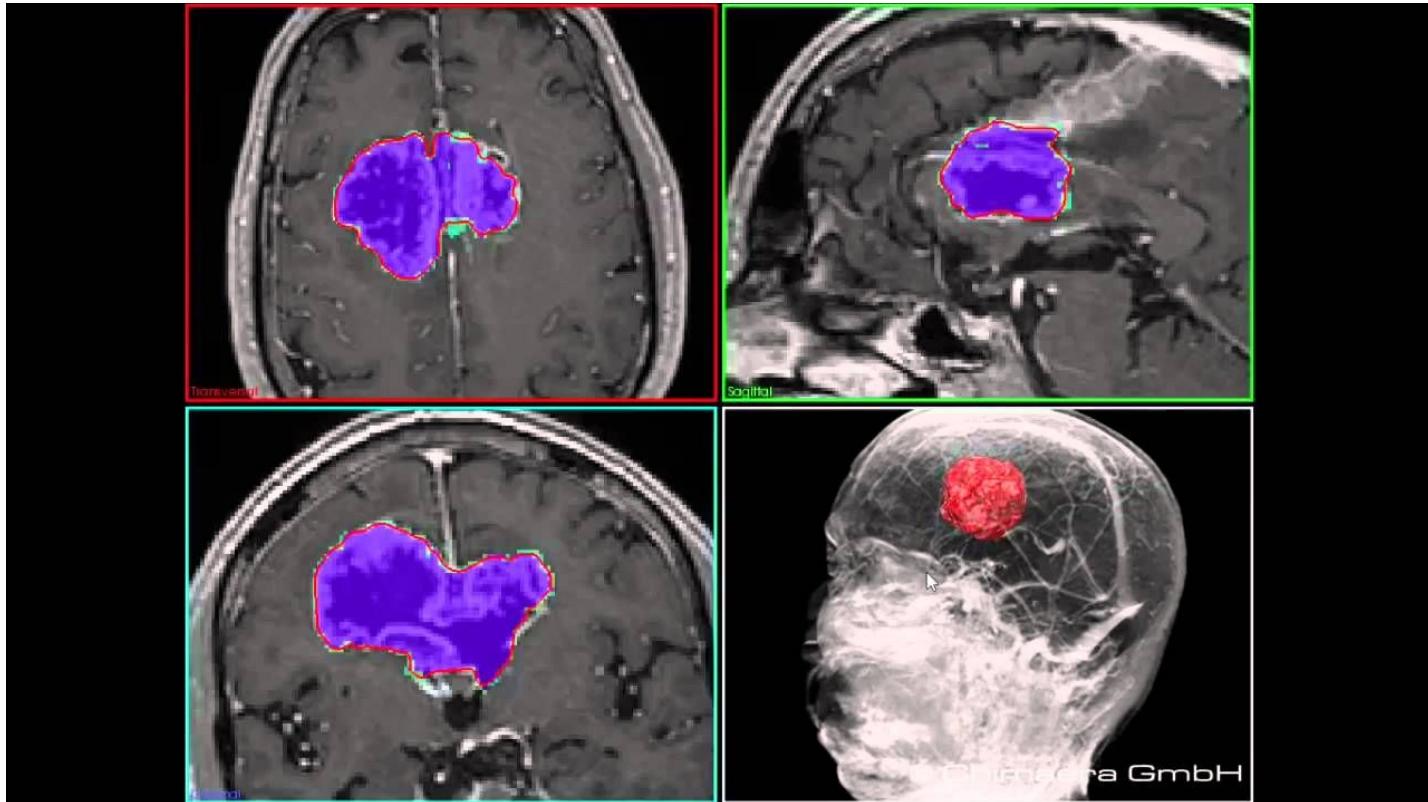
Цифровая обработка изображения

7. Применение сверточных сетей для задач
сегментации и детекции

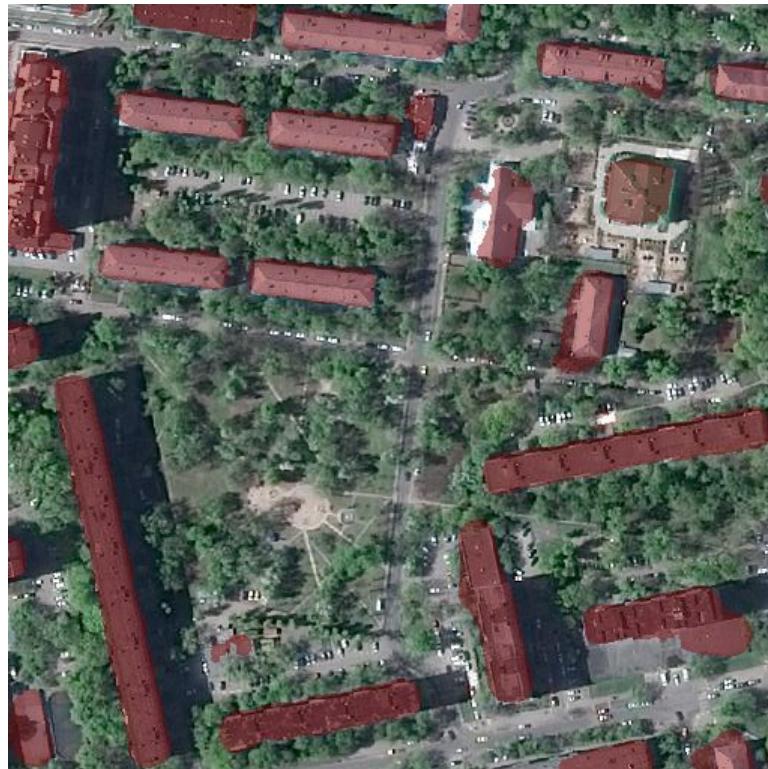
План занятия

- Детекция
- Сегментация

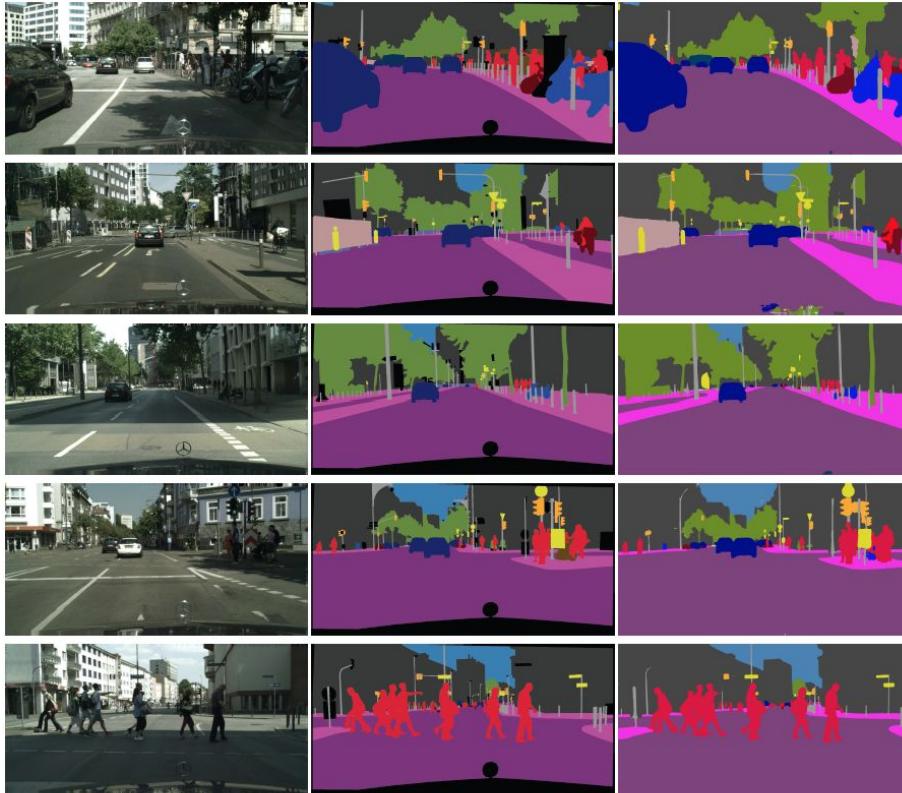
Детекция аномалий на медицинских снимках



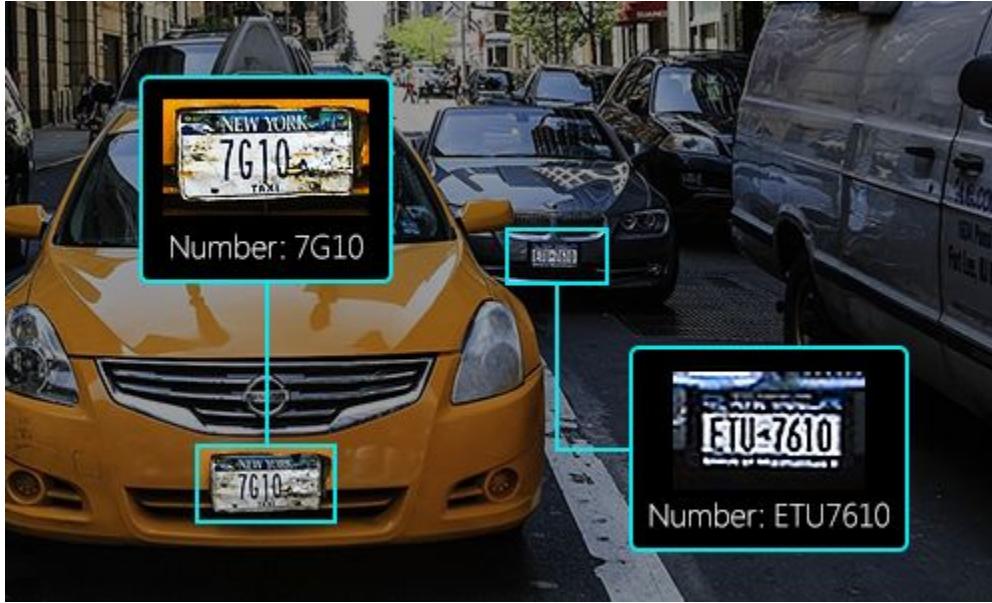
Детекция изменений на спутниковых снимках



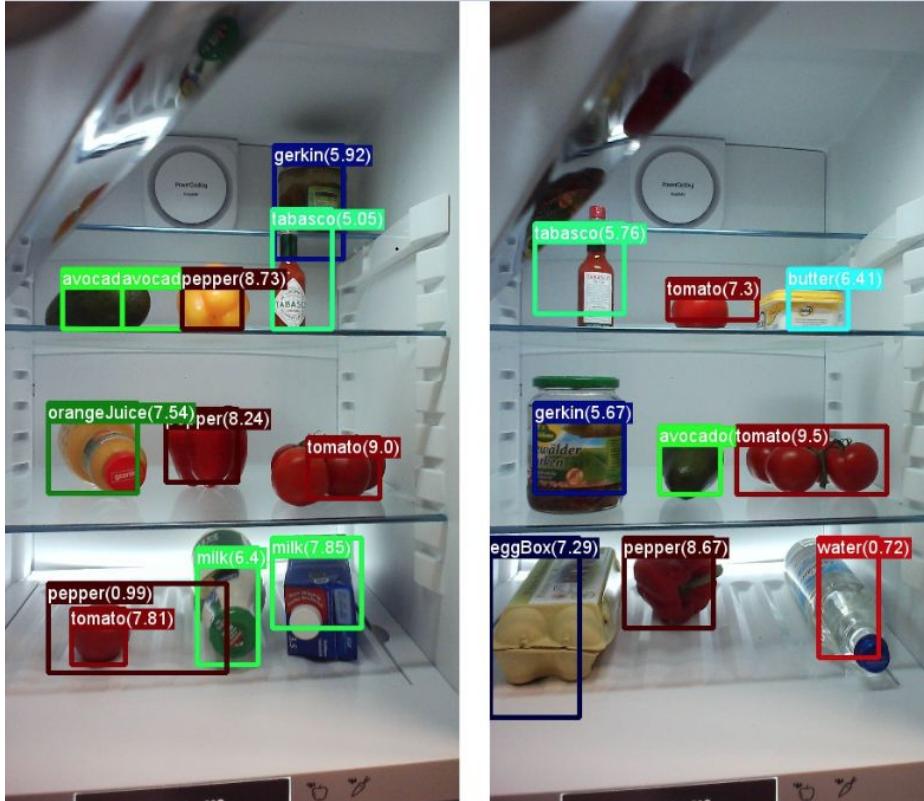
Анализ сцены



Детектор номерных знаков



УМНЫХ ХОЛОДИЛЬНИК



Сегментация

Постановка задачи

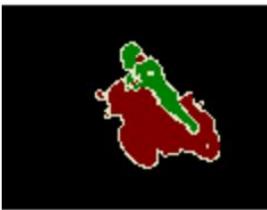
- задача сегментации - определение положения и формы объекта на изображении
- сегментация классов - каждой точке изображения присваиваем номер соответствующего класса
- сегментация объектов - каждой точке на изображении присваиваем идентификатор объекта
- фон изображения выделяется отдельным классом

Pascal VOC'12 Dataset

Image

Object
segmentation

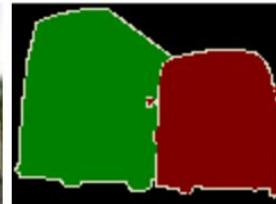
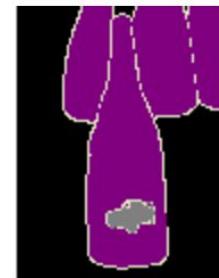
Class
segmentation



Image

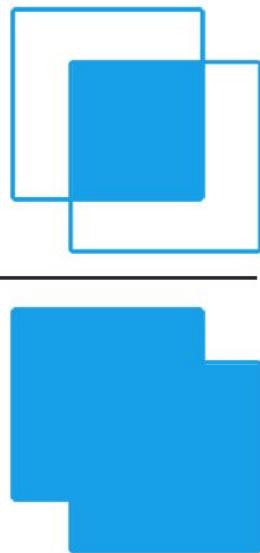
Object
segmentation

Class
segmentation

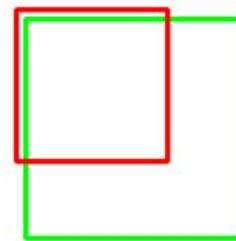


Метрики

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

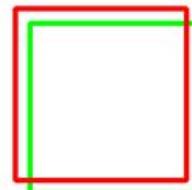


IoU: 0.4034



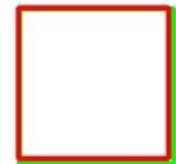
Poor

IoU: 0.7330



Good

IoU: 0.9264



Excellent

Подходы

- в основе лежит архитектура классифицирующей сверточной сети
- архитектура состоит из двух частей
 - *энкодер*: последовательность сверток генерирующая сжатое представление изображения
 - *декодер*: классификатор пикселей на основе сжатого представления
- на выходе размер сегментированного изображения либо совпадает, либо меньше исходного

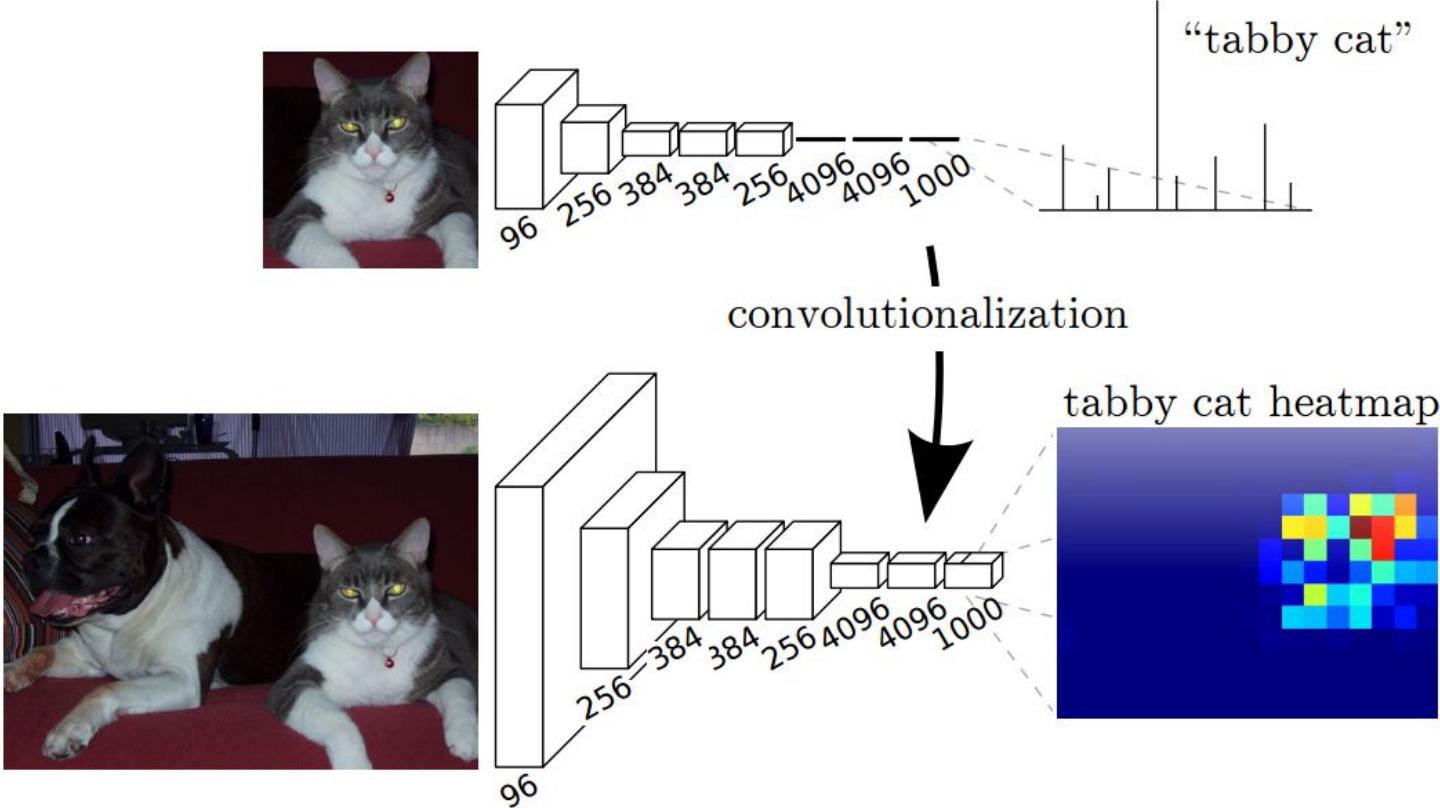
FCN - Fully Convolution Network (2015)

[Fully Convolutional Networks for Semantic Segmentation](#)

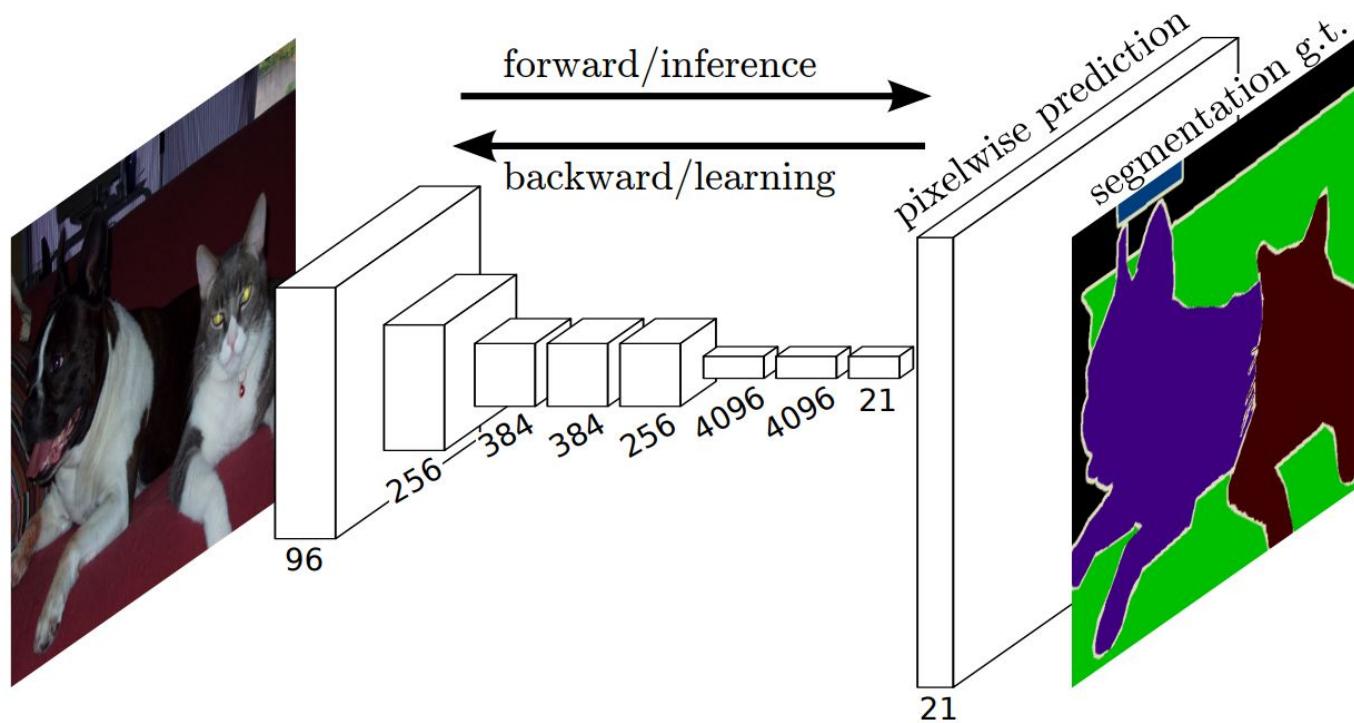
FCN - Fully Convolution Network

- адаптирована классифицирующая сверточная сеть для задачи сегментации
- для уточнения границ объекта используются признаки сверточных слоев на разных уровнях
- для объединения признаков с разным размером применяется линейная интерполяция
- реализация в Caffe: <https://github.com/shelhamer/fcn.berkeleyvision.org>

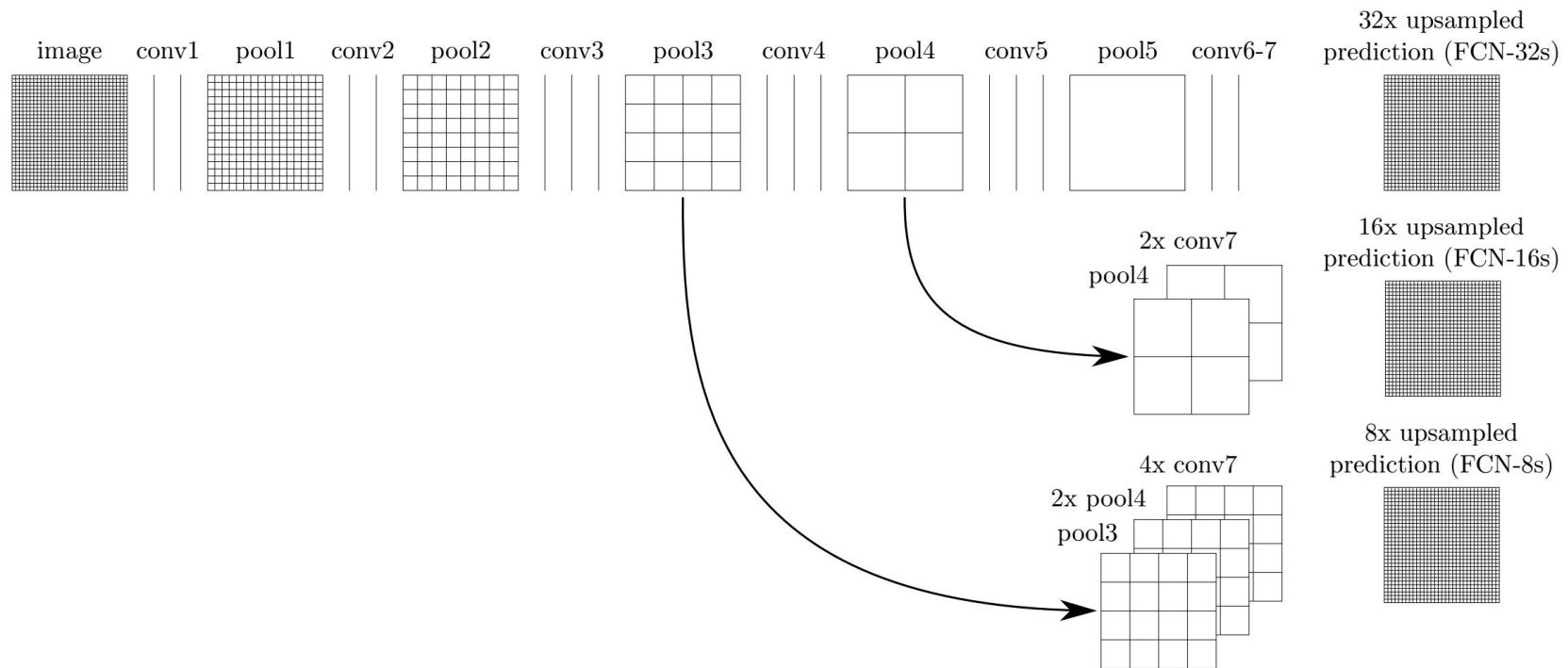
FCN - Fully Convolution Network



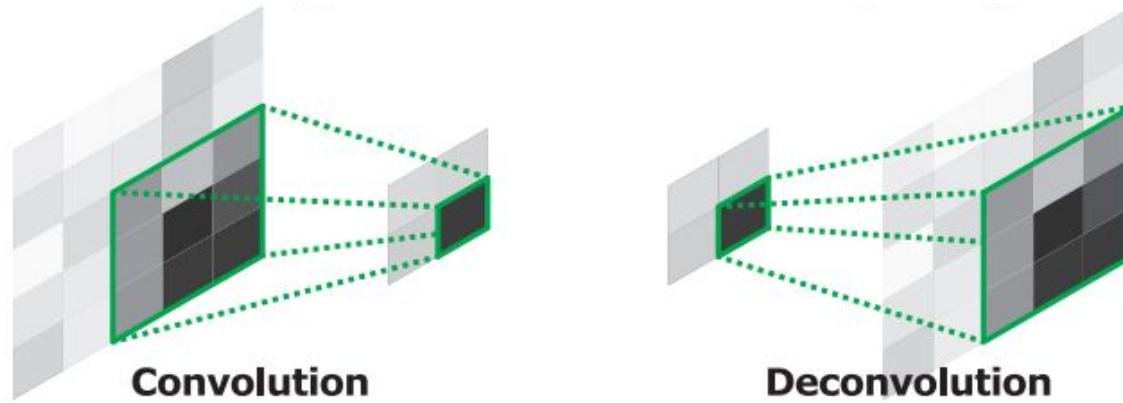
FCN - Fully Convolution Network



FCN - Fully Convolution Network

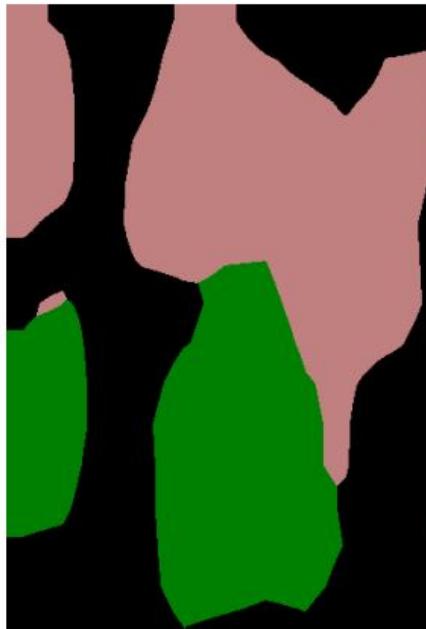


FCN - Fully Convolution Network



FCN - Fully Convolution Network

FCN-32s



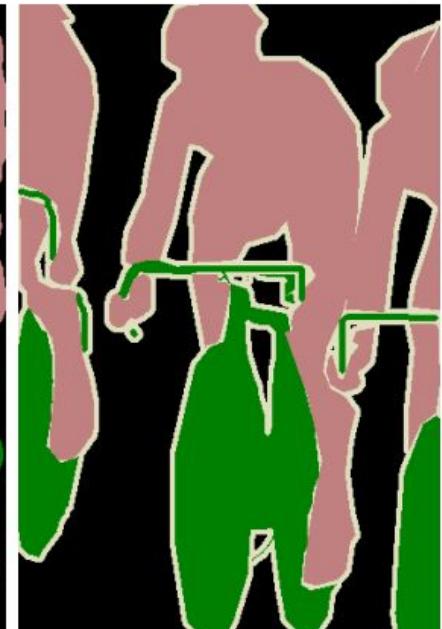
FCN-16s



FCN-8s



Ground truth



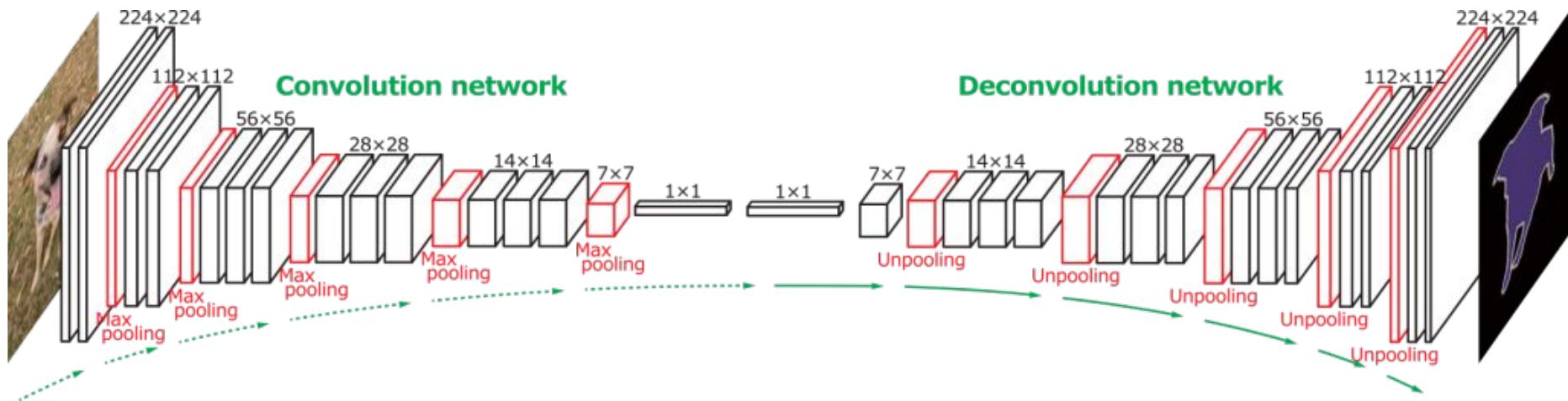
Deconvolution Network (2015)

[Learning Deconvolution Network for Semantic Segmentation](#)

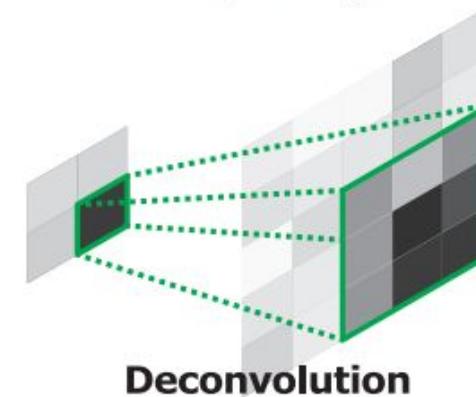
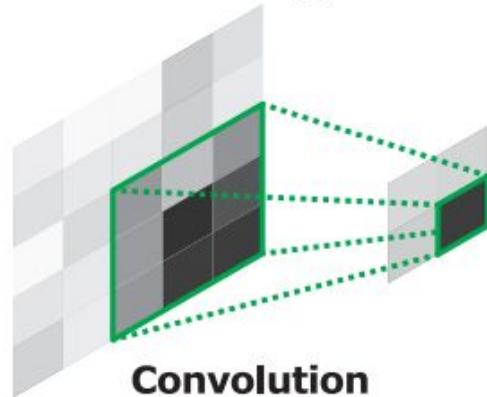
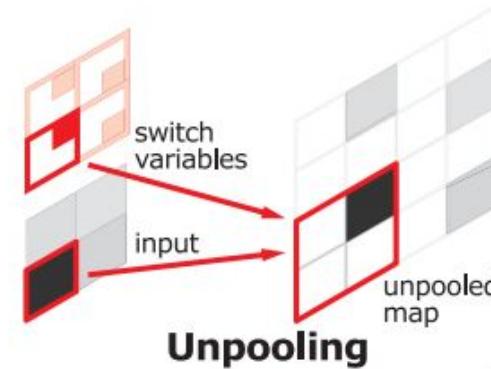
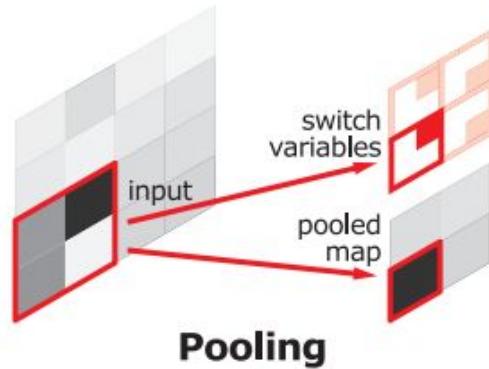
Deconvolution Network

- декодер имеет сверточную структуру
- для декодирования размера применяется операции Unpooling и Deconvolution
- веса операции Deconvolution обучаются

Deconvolution Network



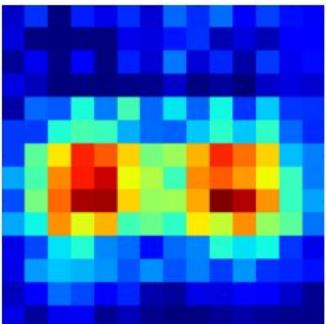
Deconvolution Network



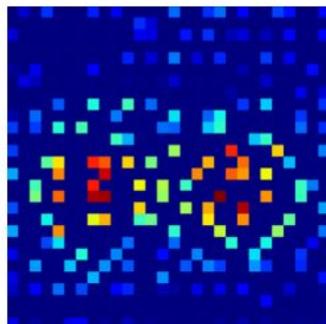
Deconvolution Network



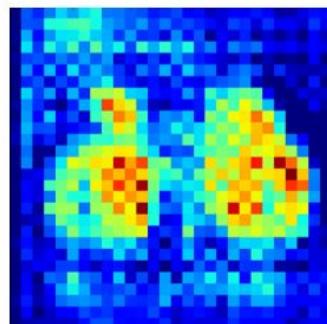
(a)



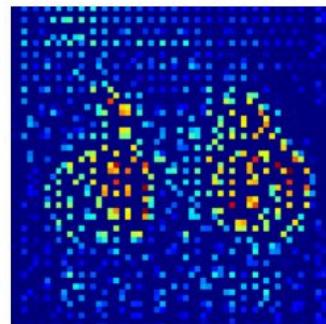
(b)



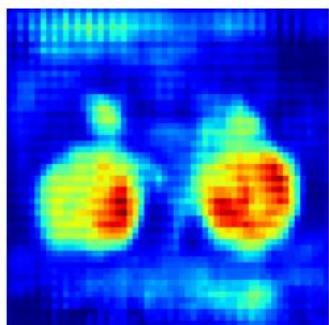
(c)



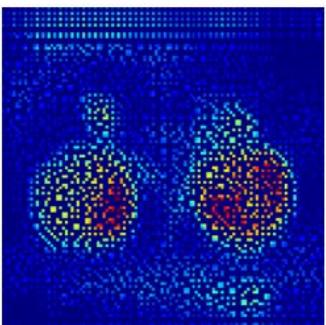
(d)



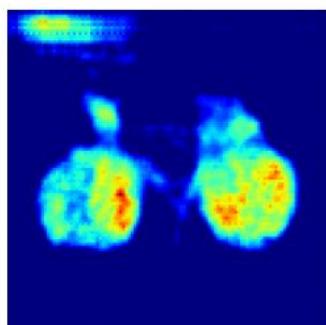
(e)



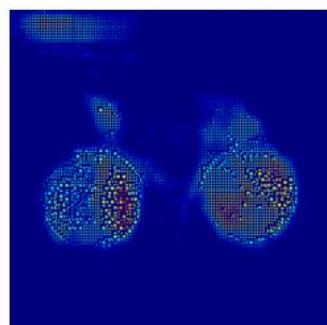
(f)



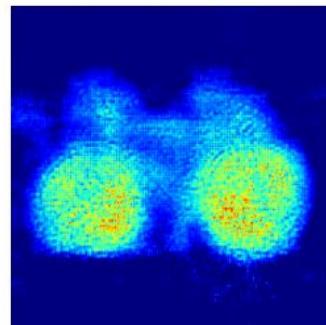
(g)



(h)



(i)



(j)

Deconvolution Network



(a) Input image

(b) FCN-8s

(c) Ours

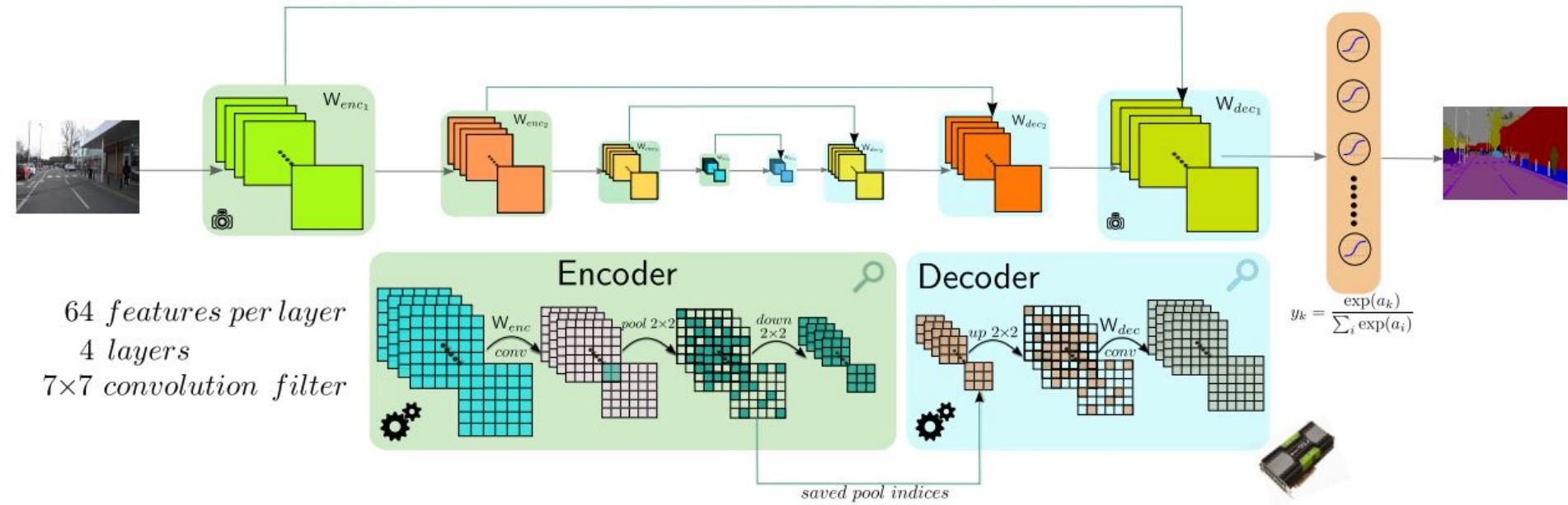
SegNet (2015)

[SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation](#)

SegNet

- при декодировании использует индексы соответствующего слоя MaxPooling энкодера
- к полученной разреженной матрице применяют операцию свертки
- такой подход позволяет существенно сократить число параметров
- доступна реализация в Caffe: <https://github.com/alexgkendall/caffe-segnet>

SegNet



SegNet vs FCN

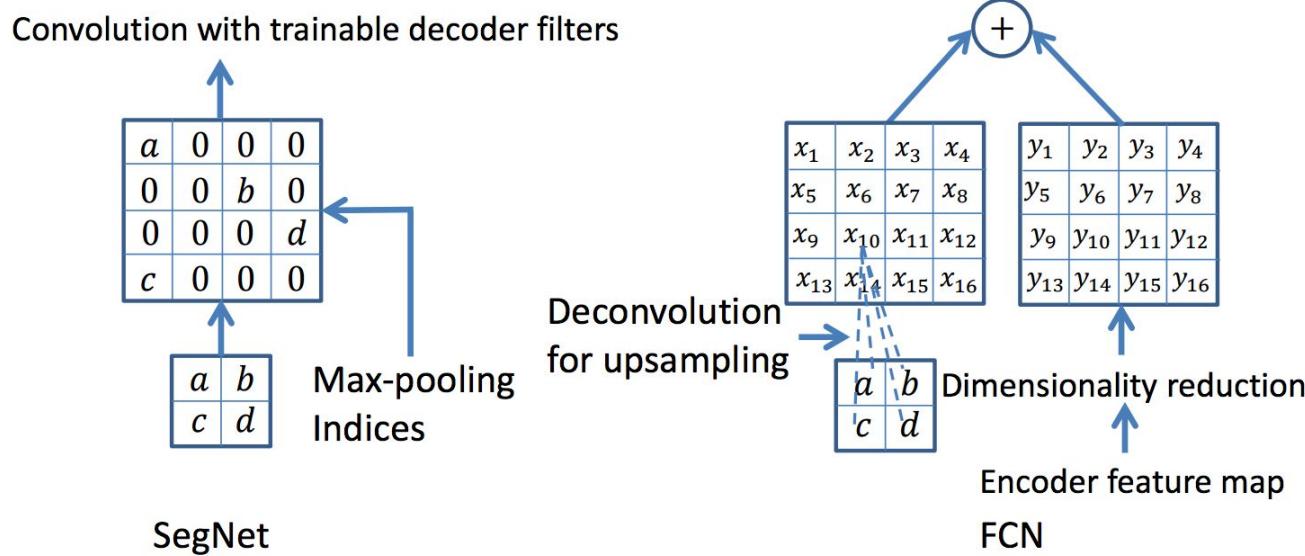


Fig. 3. An illustration of SegNet and FCN [2] decoders. a, b, c, d correspond to values in a feature map. SegNet uses the max pooling indices to upsample (without learning) the feature map(s) and convolves with a trainable decoder filter bank. FCN upsamples by learning to deconvolve the input feature map and adds the corresponding encoder feature map to produce the decoder output. This feature map is the output of the max-pooling layer (includes sub-sampling) in the corresponding encoder. Note that there are no trainable decoder filters in FCN.

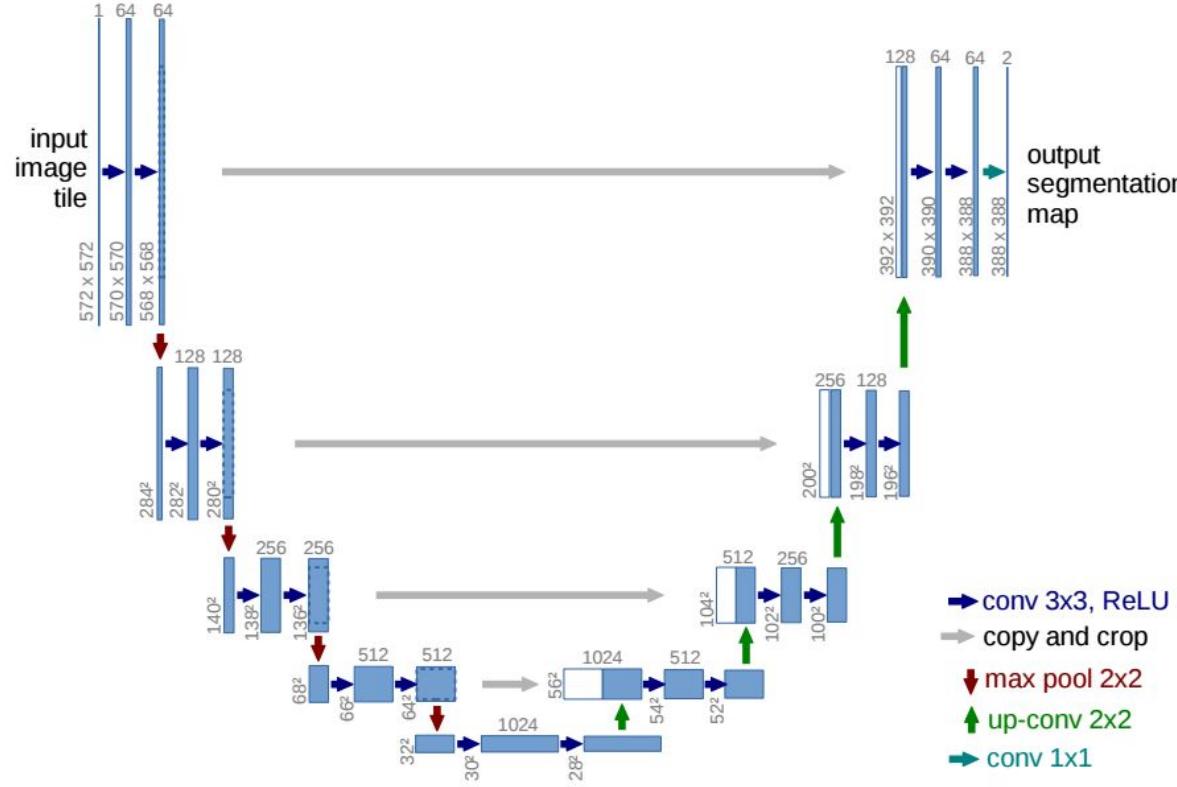
U-Net

[U-Net: Convolutional Networks for Biomedical Image Segmentation](#)

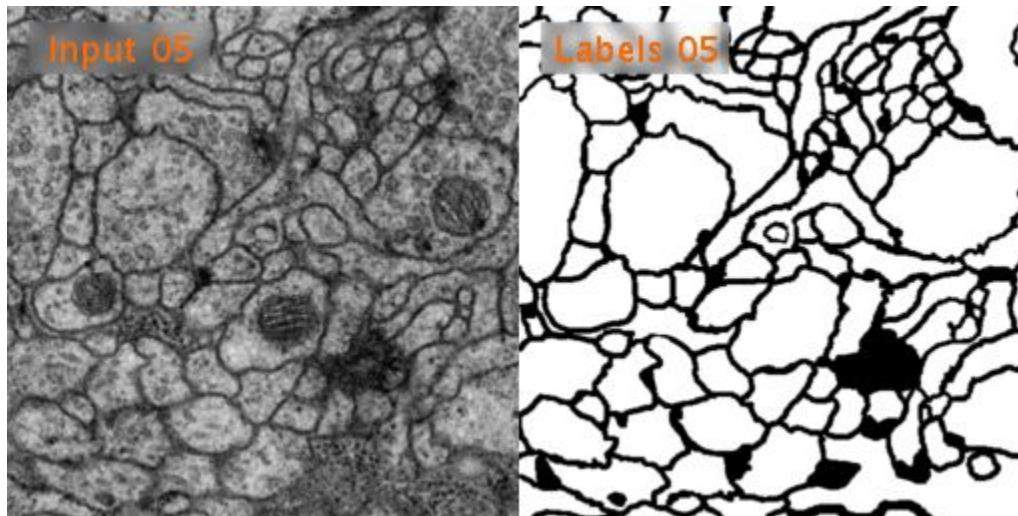
U-Net

- конкатенирует активации энкодера с расширенными (upsampling) признаками декодера
- такой подход позволяет энкодеру быстрее выучивать необходимые признаки
- получила популярность из-за простоты реализации

U-Net

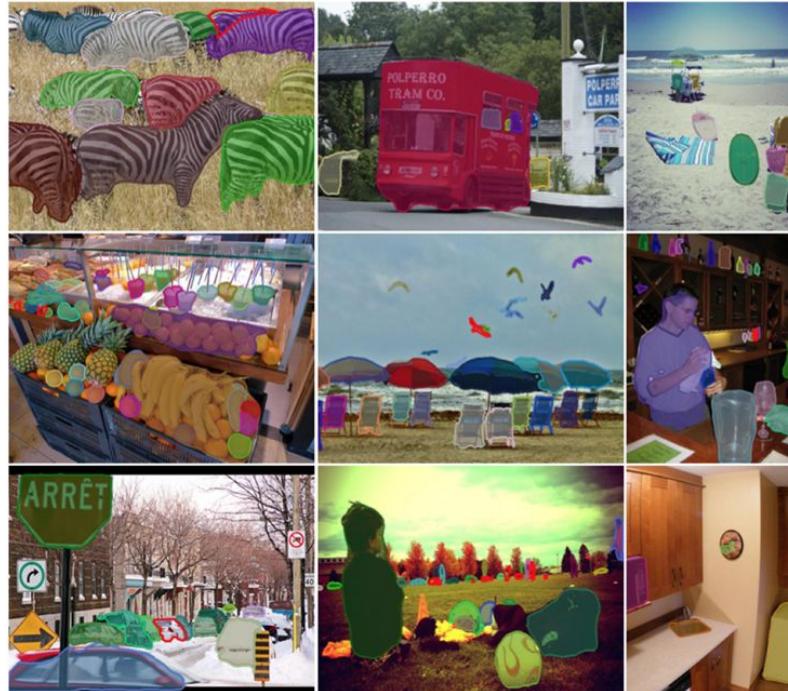


Segmentation of neuronal structures in EM stacks



Сегментация объектов

DeepMask + SharpMask



<https://github.com/facebookresearch/deepmask>

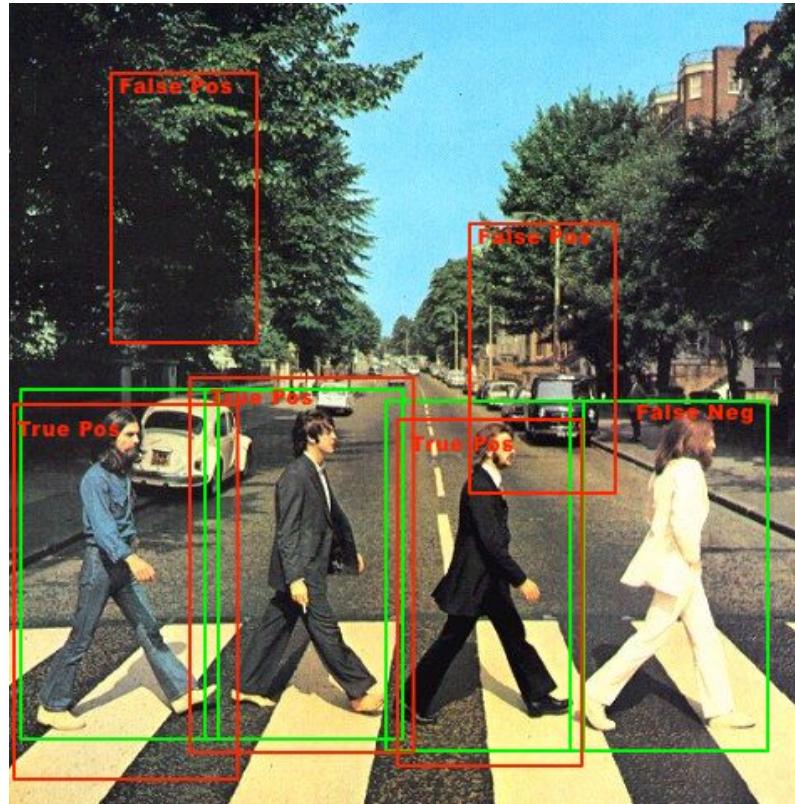
Deep Mask

Детекция

Постановка задачи

- определить область на изображении, содержащую объект
- классифицировать найденный объект
- необходимо находить и классифицировать несколько объектов на изображении

Постановка задачи



Особенности задачи

- объекты сняты под различными ракурсами
- объекты имеют различные масштабы
- на изображении объекты могут перекрывать друг-друга

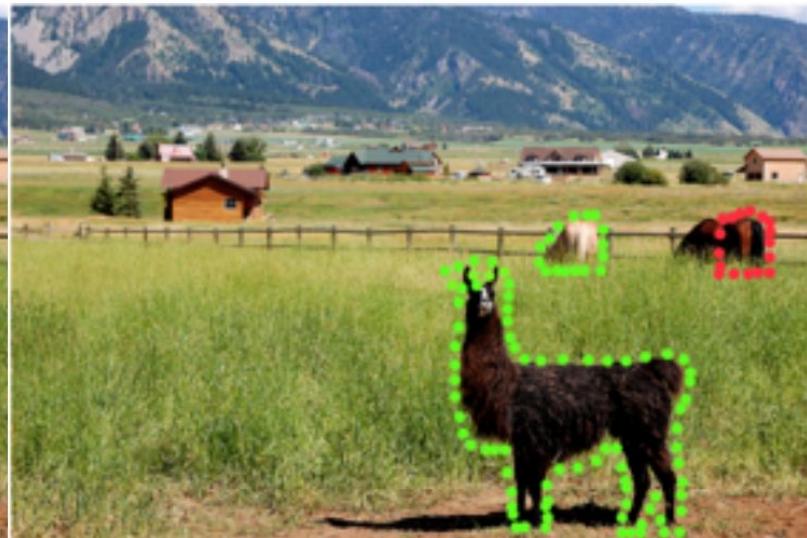
Pascal VOC 2012

- 20 классов
- 11к изображений



COCO 2015

- 80 классов
- 200к размеченных изображений



Метрики качества - Average Precision

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \dots, 1\}} p_{interp}(r)$$

$$p_{interp}(r) = \max_{\tilde{r}: \tilde{r} \geq r} p(\tilde{r})$$

p - precision

r - recall

Метрика качества - IoU

$$a_o = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})}$$

B_p - предсказанная область

B_{gt} - область разметки

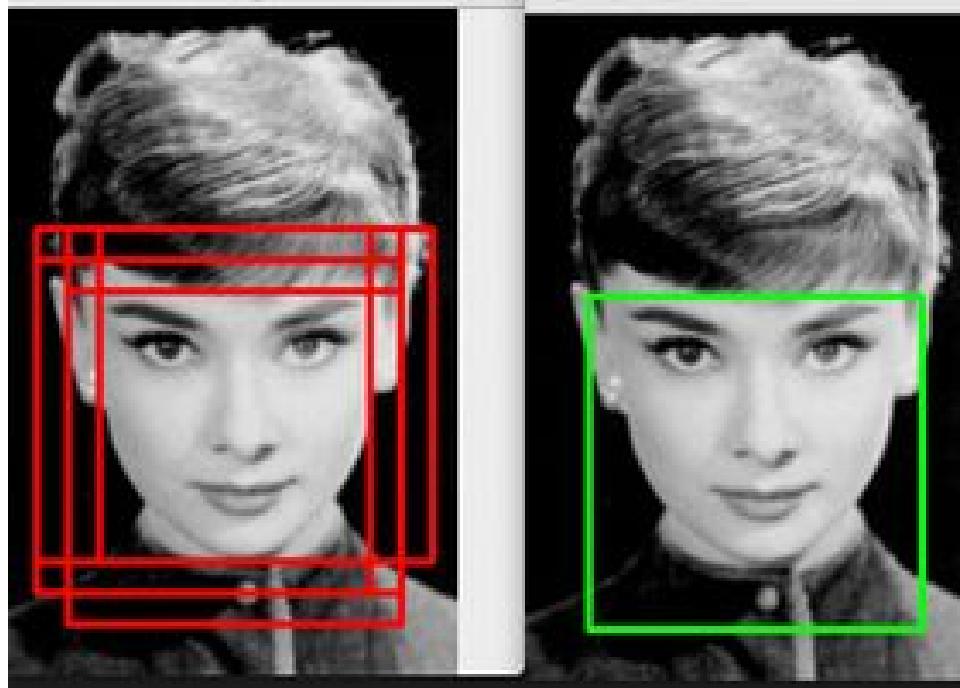
Подходы к решению задачи

- классифицируем скользящие окна разных масштабов
- выбираем области в которых может быть объект и классифицируем только их
- обучаем модель помимо классификации предсказывать размер и положение области с объектом

NMS (non maximum suppression)

- предсказания для одного объекта как правило дублируются
- для корректной детекции дубли необходимо удалить
- выделенную область считаем детекцией если в ее окрестности нет других областей с большей вероятностью детекции

NMS (non maximum suppression)



Selective Search (2012)

[Selective Search for Object Recognition](#)

Selective Search

- объекты на изображении могут располагаться в любом месте и иметь произвольный масштаб
- перебор всех возможных вариантов расположения и масштаба объектов вычислительно очень сложная задача
- необходимо сократить возможные варианты поиска

Selective Search

- предлагается сегментировать объекты на небольшие части (SuperPixel)
- затем, итеративно, двигаясь снизу вверх, объединять эти части
- части объединяются с учетом вероятности принадлежности одному объекту
- в результате объединения частей получаем область, которая наиболее вероятно содержит объект

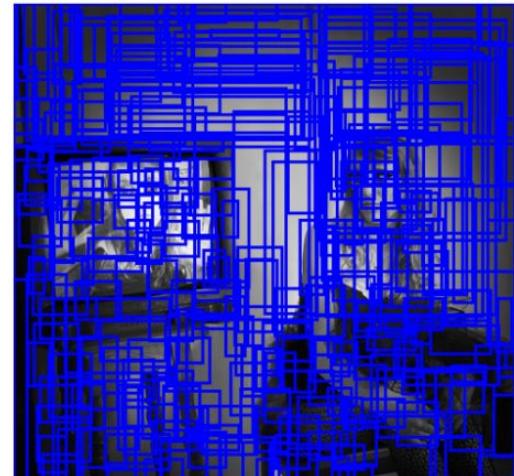
Selective Search



Input Image



Segmentation



Candidate objects

Selective search



R-CNN (2014)

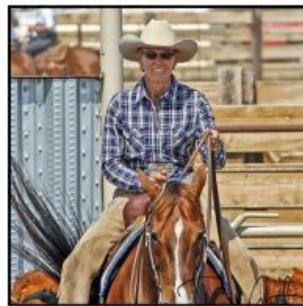
Rich feature hierarchies for accurate object detection and semantic segmentation

R-CNN

- находим области на изображении, где предположительно может находиться объект ~2000 (region proposal, region of interest, ROI)
- получаем признаки для этой области с fc слоя сети AlexNet
- т.к. AlexNet принимает на вход изображение фиксированного размера 227x227, приводим размер области в соответствие путем деформации изображения (warp)
- классифицируем область по полученным признакам
- время работы ~15сек на изображение на Tesla K40

R-CNN

R-CNN: *Regions with CNN features*

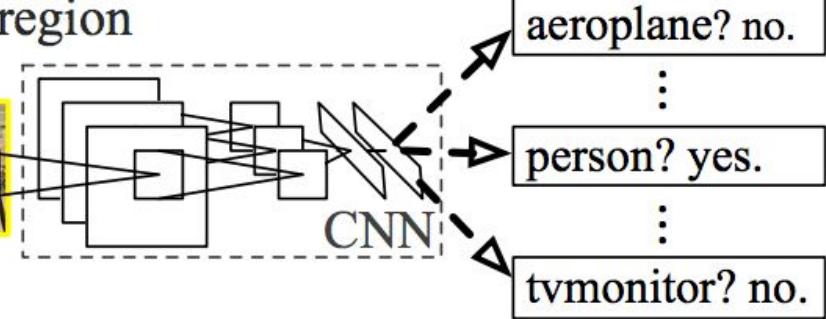


1. Input image



2. Extract region proposals (~2k)

warped region



3. Compute CNN features

4. Classify regions

Rich feature hierarchies for accurate object detection and semantic segmentation

R-CNN

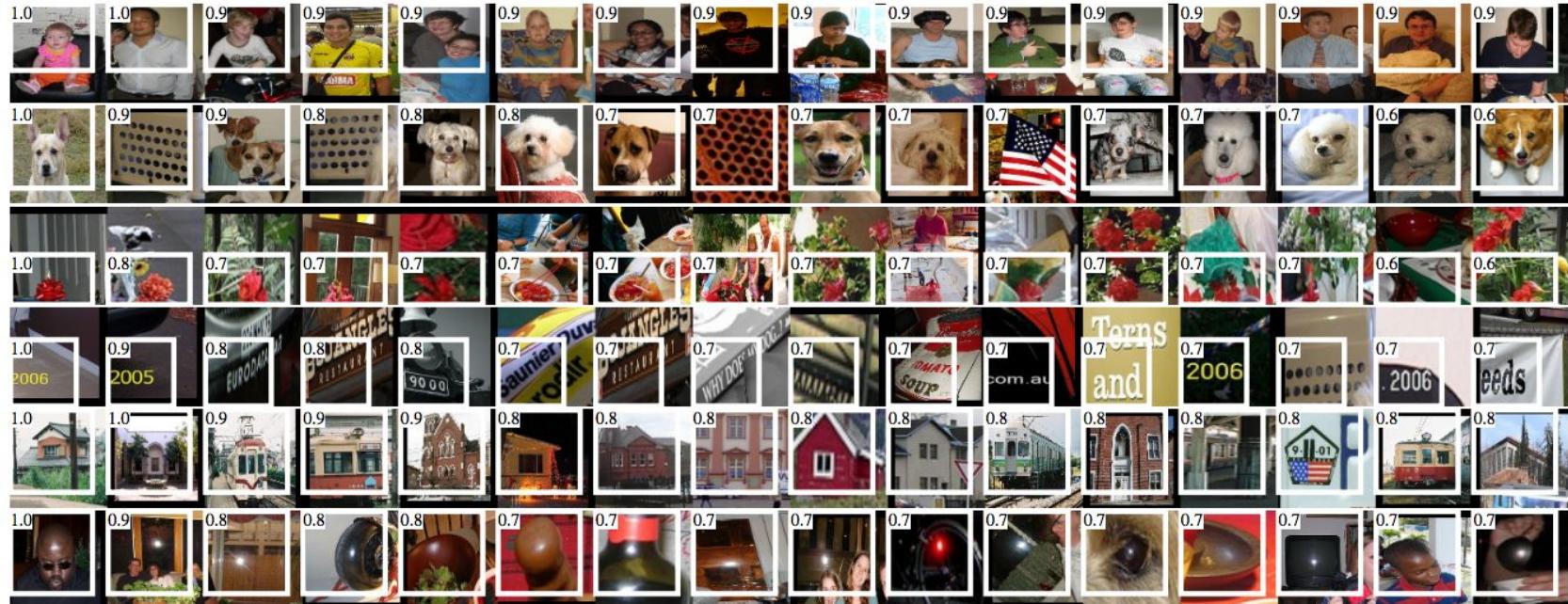
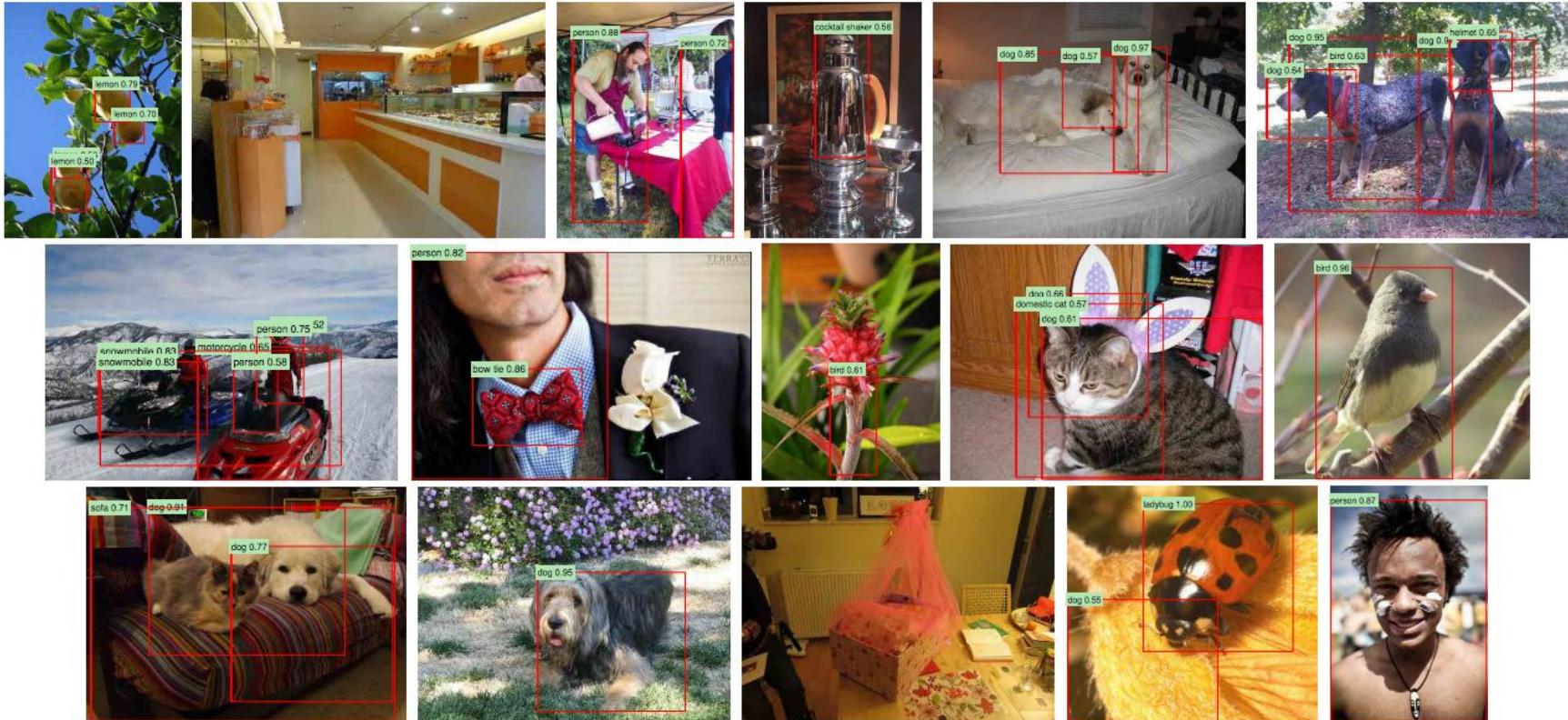


Figure 4: Top regions for six pool₅ units. Receptive fields and activation values are drawn in white. Some units are aligned to concepts, such as people (row 1) or text (4). Other units capture texture and material properties, such as dot arrays (2) and specular reflections (6).

[AlexNet implementation caffe](#)

R-CNN



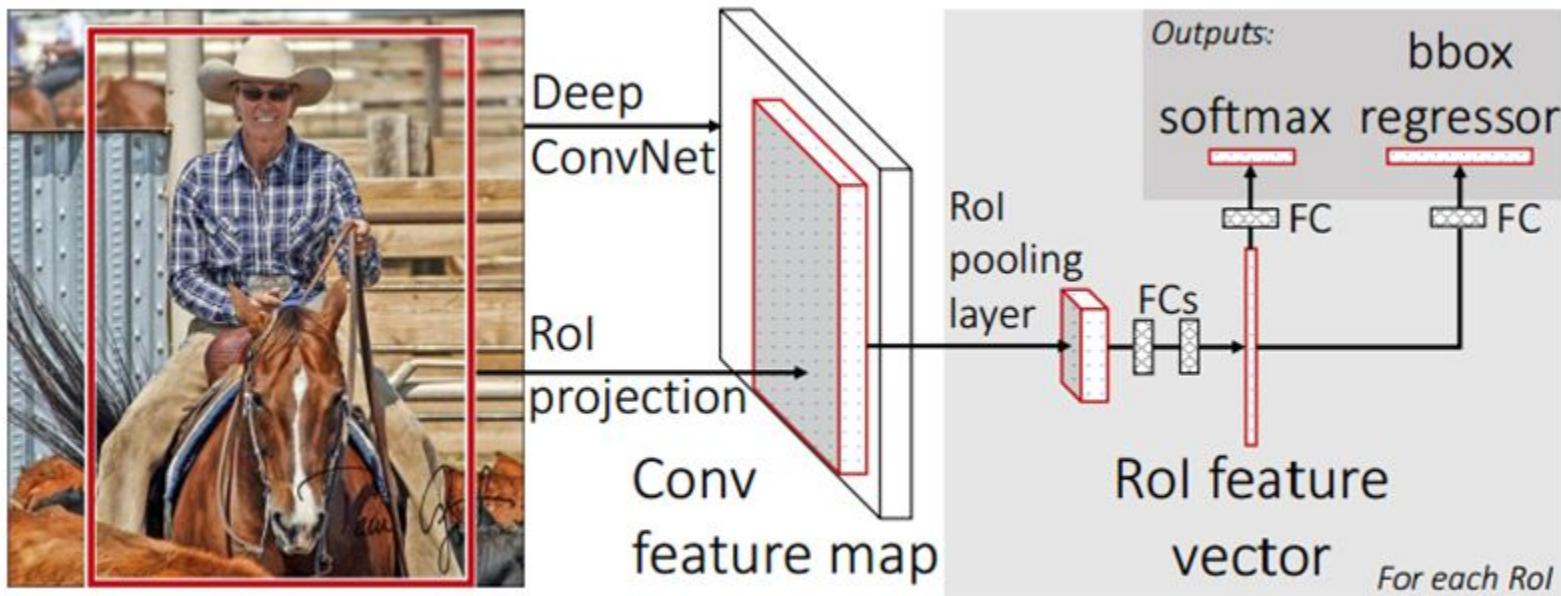
Fast R-CNN (2015)

[Fast R-CNN](#)

Fast R-CNN

- в отличие от R-CNN задачи локализации и классификации объекта обучаются совместно
- необходимы начальные приближения позиций и размеров объектов
- время обучения в 9 раз быстрее R-CNN
- время применения в 213 быстрее R-CNN
- реализация на базе Caffe: <https://github.com/rbgirshick/fast-rcnn>

Fast-RCNN



Fast R-CNN workflow

Fast R-CNN - RoI Pooling Layer

- для построения модели классификации необходимо, чтобы число признаков у объектов совпадало
- тк объекты на изображении могут иметь различные размеры, то размеры соответствующих областей в Pooling слое также будут различаться
- слой RoI Pooling решает эту проблему с помощью операции Max Pooling
- причем размер окна и шаг Max Pooling зависит от размера интересующей нас области

Fast R-CNN - Функция потерь

$$L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda[u \geq 1]L_{\text{loc}}(t^u, v)$$

Lcls - log-loss классификации

Lloc - функция потерь локализации

p,u - предсказанный и настоящий классы объекта

t,v - предсказанные и действительные координаты и размер области

lambda - гиперпараметр, задает вклад локализации в функцию потерь

Fast R-CNN - Функция потерь

$$L_{\text{loc}}(t^u, v) = \sum_{i \in \{\text{x}, \text{y}, \text{w}, \text{h}\}} \text{smooth}_{L_1}(t_i^u - v_i)$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

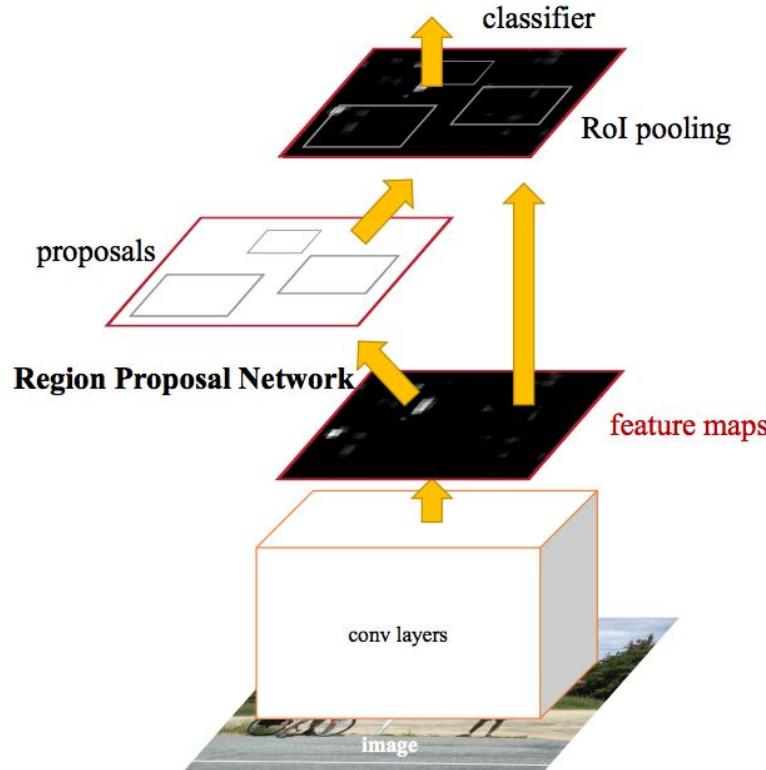
Faster R-CNN (2016)

[Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks](#)

Faster R-CNN

- в отличие от R-CNN и Fast R-CNN эта модель сама генерирует ROI
- для генерации ROI используется Region Proposal Network (RPN)
- скорость работы ~5fps
- реализация на базе Caffe: <https://github.com/rbgirshick/py-faster-rcnn>

Faster R-CNN



Faster R-CNN

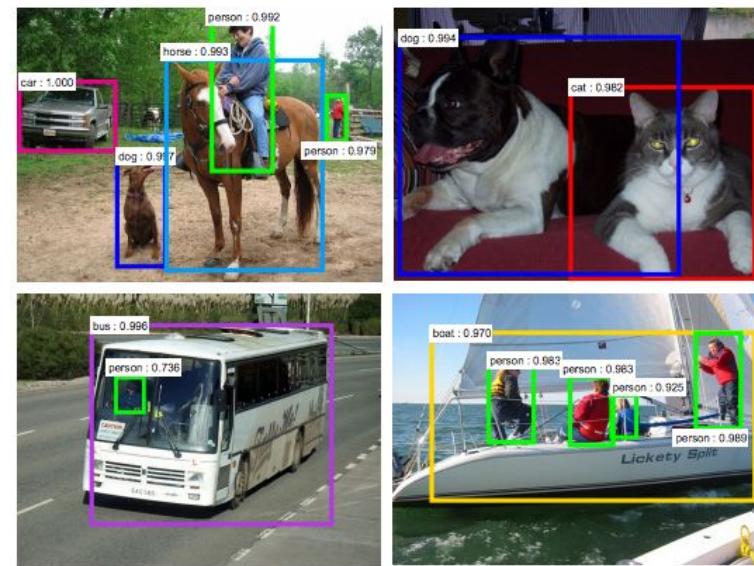
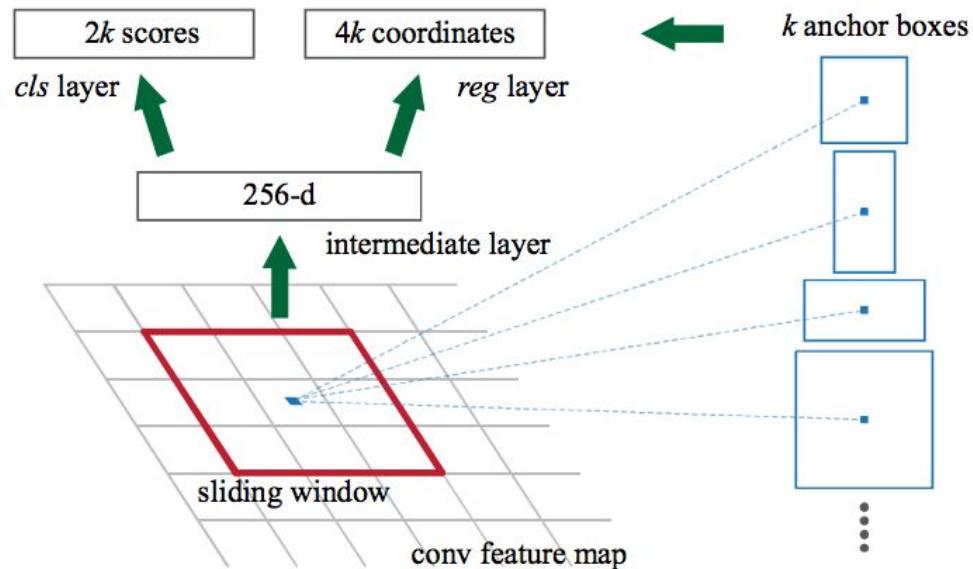


Figure 3: **Left:** Region Proposal Network (RPN). **Right:** Example detections using RPN proposals on PASCAL VOC 2007 test. Our method detects objects in a wide range of scales and aspect ratios.

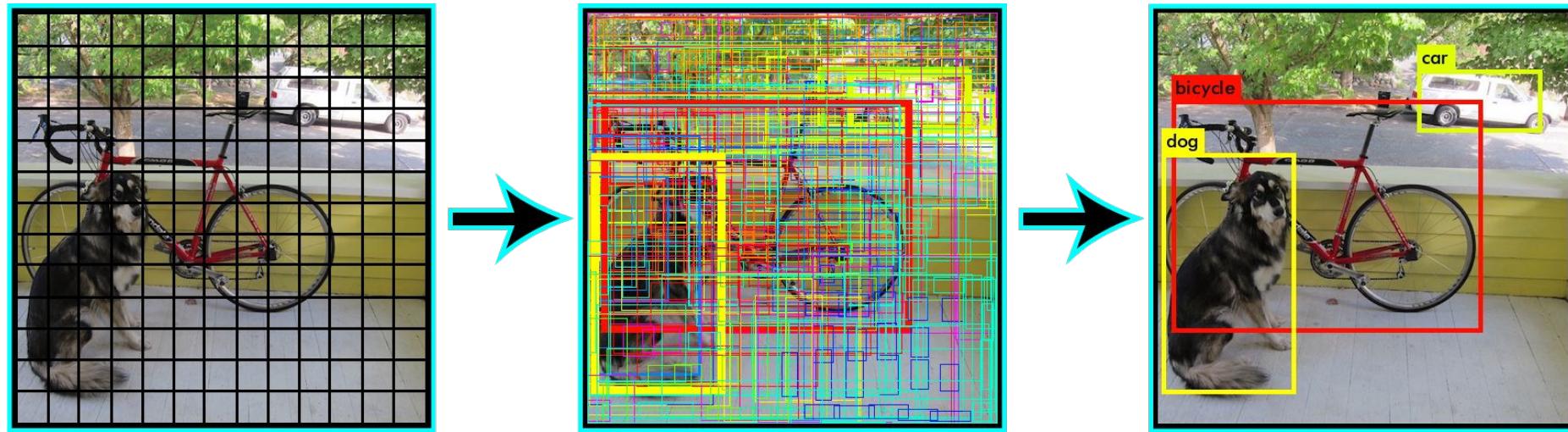
YOLO (2015)

You look only once

YOLO

- все изображение разбивается на равные части (сетку)
- для каждой ячейки предсказывается несколько объектов с размерами и классификацией
- предсказания для всех ячеек получаются за один проход
- производительность ~50fps на Titan X
- реализация доступна на сайте: <https://pjreddie.com/darknet/yolo/>

YOLO



YOLO

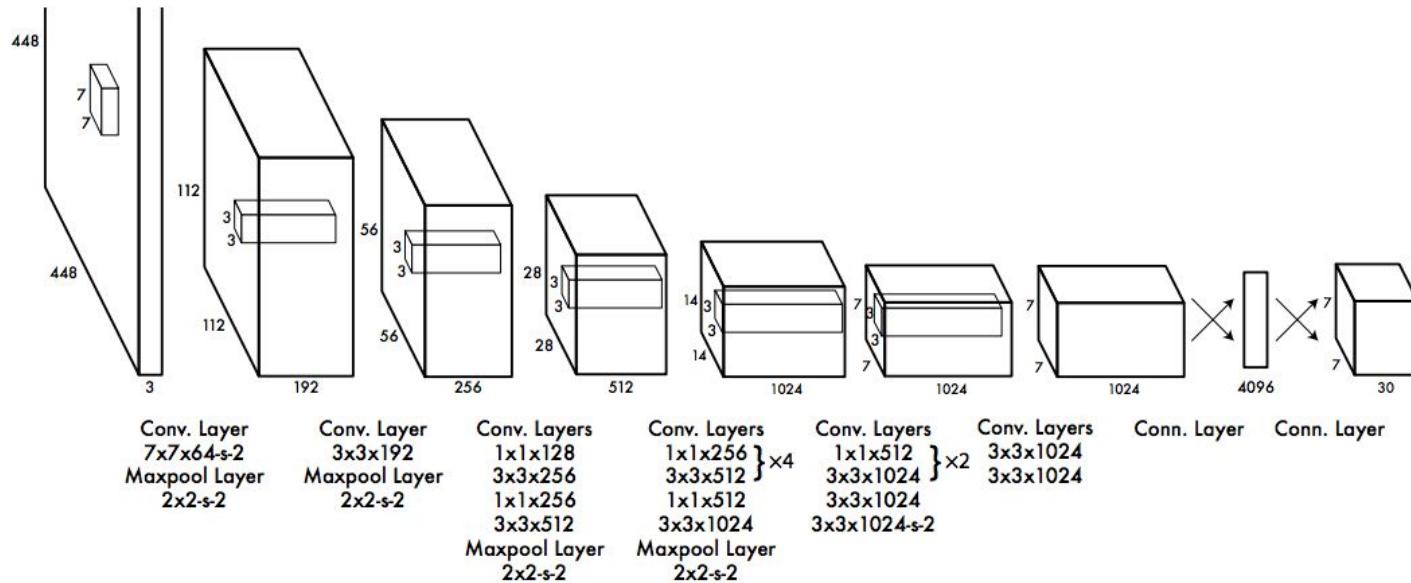


Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

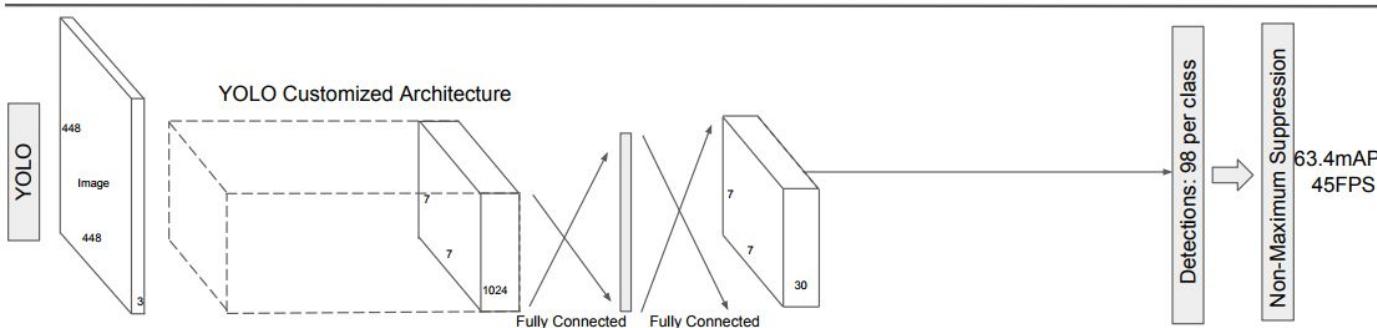
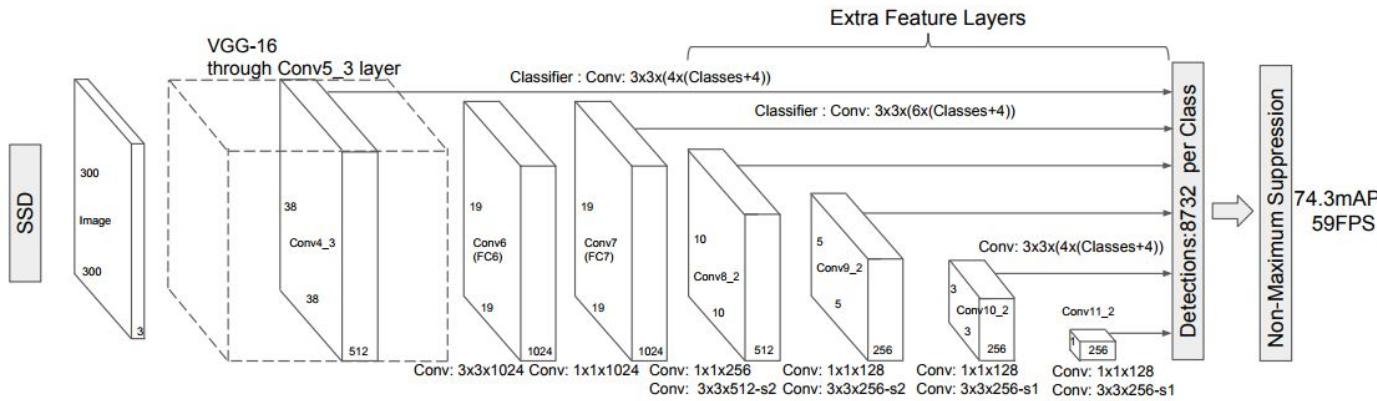
SSD Single Shot Multibox Detector (2015)

[SSD: Single Shot MultiBox Detector](#)

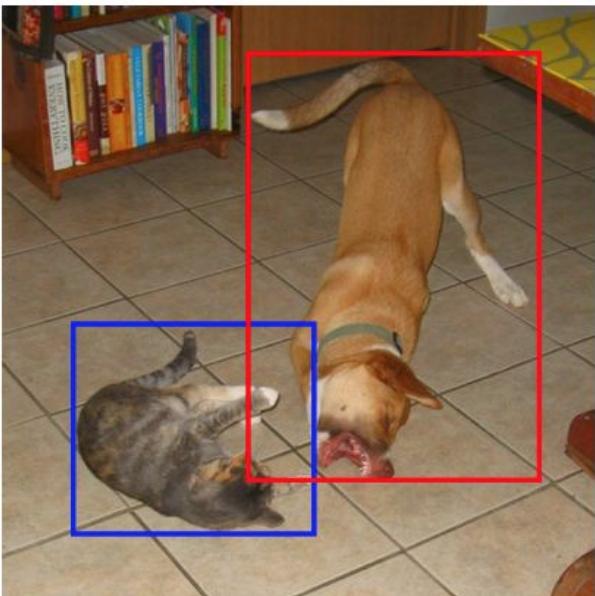
SSD

- предсказание детекции классификации генерируются за один проход
- не требуется предварительная генерация RoI
- производительность 59fps на Titan X
- реализация на базе Caffe: <https://github.com/weiliu89/caffe/tree/ssd>

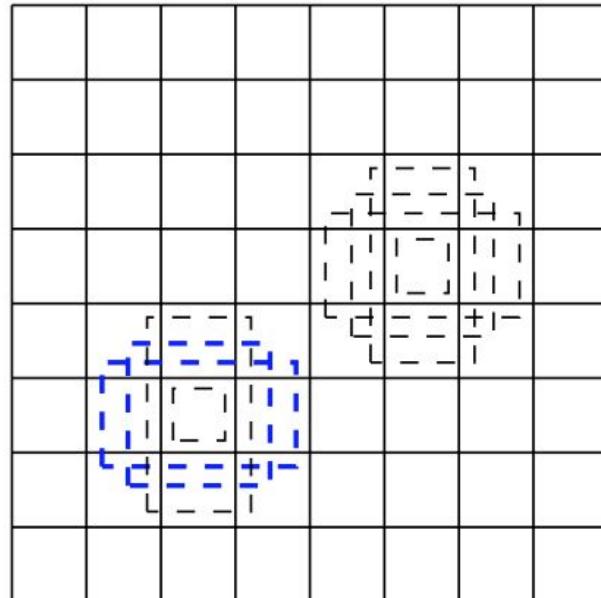
SSD



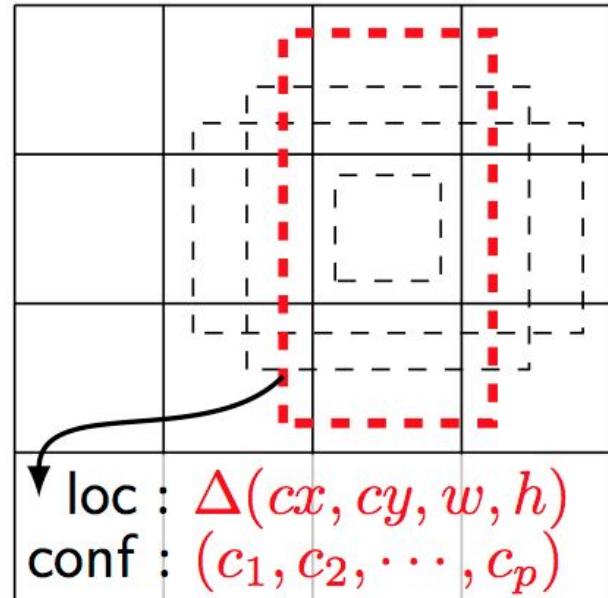
SSD



(a) Image with GT boxes



(b) 8×8 feature map



(c) 4×4 feature map

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

Сравнение качества моделей

Method	data	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Fast R-CNN [5]	07++12	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	87.5	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
Faster R-CNN [15]	07++12	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
YOLO [14]	07++12	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
SSD300 [11]	07++12	72.4	85.6	80.1	70.5	57.6	46.2	79.4	76.1	89.2	53.0	77.0	60.8	87.0	83.1	82.3	79.4	45.9	75.9	69.5	81.9	67.5
SSD512 [11]	07++12	74.9	87.4	82.3	75.8	59.0	52.6	81.7	81.5	90.0	55.4	79.0	59.8	88.4	84.3	84.7	83.3	50.2	78.0	66.3	86.3	72.0
ResNet [6]	07++12	73.8	86.5	81.6	77.2	58.0	51.0	78.6	76.6	93.2	48.6	80.4	59.0	92.1	85.3	84.8	80.7	48.1	77.3	66.5	84.7	65.6
YOLOv2 544	07++12	73.4	86.3	82.0	74.8	59.2	51.8	79.8	76.5	90.6	52.1	78.2	58.5	89.3	82.5	83.4	81.3	49.1	77.2	62.4	83.8	68.7

Table 4: PASCAL VOC2012 test detection results. YOLOv2 performs on par with state-of-the-art detectors like Faster R-CNN with ResNet and SSD512 and is $2 - 10 \times$ faster.

Сравнение качества моделей

		0.5:0.95	0.5	0.75	S	M	L	1	10	100	S	M	L
Fast R-CNN [5]	train	19.7	35.9	-	-	-	-	-	-	-	-	-	-
Fast R-CNN[1]	train	20.5	39.9	19.4	4.1	20.0	35.8	21.3	29.5	30.1	7.3	32.1	52.0
Faster R-CNN[15]	trainval	21.9	42.7	-	-	-	-	-	-	-	-	-	-
ION [1]	train	23.6	43.2	23.6	6.4	24.1	38.3	23.2	32.7	33.5	10.1	37.7	53.6
Faster R-CNN[10]	trainval	24.2	45.3	23.5	7.7	26.4	37.1	23.8	34.0	34.6	12.0	38.5	54.4
SSD300 [11]	trainval35k	23.2	41.2	23.4	5.3	23.2	39.6	22.5	33.2	35.3	9.6	37.6	56.5
SSD512 [11]	trainval35k	26.8	46.5	27.8	9.0	28.9	41.9	24.8	37.5	39.8	14.0	43.5	59.0
YOLOv2 [11]	trainval35k	21.6	44.0	19.2	5.0	22.4	35.5	20.7	31.6	33.3	9.8	36.5	54.4

Table 5: Results on COCO test-dev2015. Table adapted from [11]

Резюме

- большинство сегментационных моделей представляют собой последовательно соединенные энкодер-декодер
- проброс сигнала с энкодера помогает улучшить качество сегментации
- детектор состоит из двух основных частей: локализация и классификация
- современные модели детекторов объединяют в себе обе части
- добавление признаков сверточных слоев разной глубины позволяет учесть различие в масштабе объектов

Полезные материалы

- [Semantic Segmentation using Fully Convolutional Networks over the years](#)
- [Deconvolutional Networks](#)
- [Deep Learning Trends @ ICLR 2016](#)