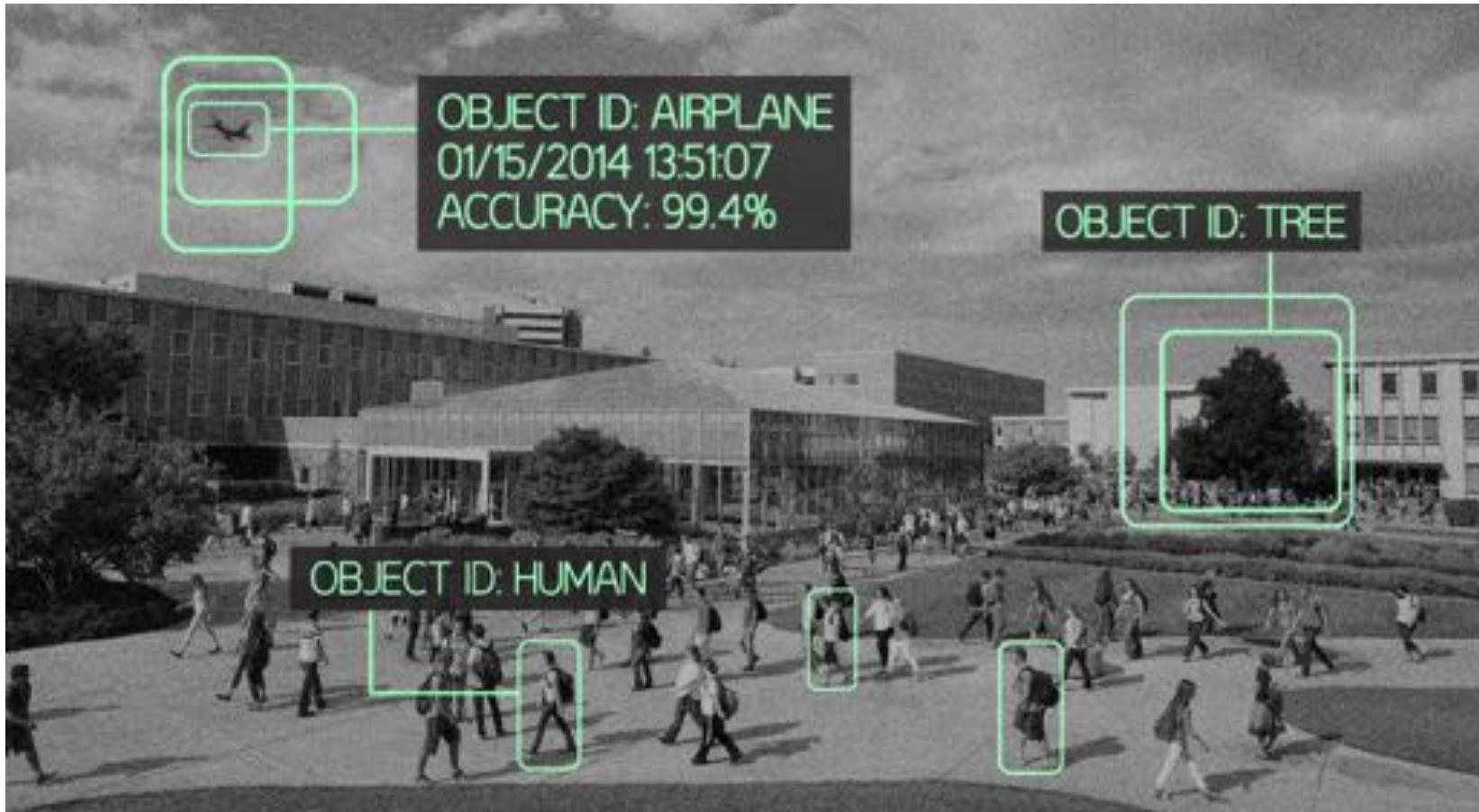


# Задачи компьютерного зрения

- Детекция/Распознавание
- Сегментация изображений
- Восстановление изображений
- Восстановление 3D формы по 2D изображениям
- Идентификация
- Оценка движения
- Оптическое распознавание символов

# Детекция



# Сегментация



Classification



CAT

Classification  
+ Localization



CAT

Object Detection



CAT, DOG

Instance Segmentation

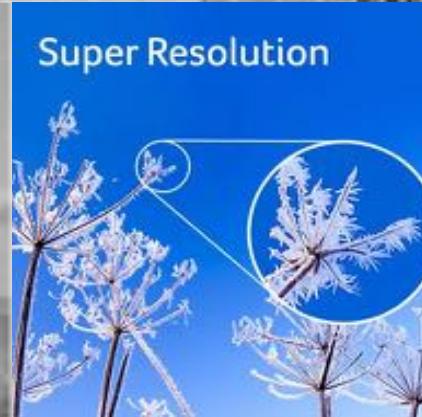


CAT, DOG

Single object

Multiple objects

# Восстановление изображений



$$l_x = D(l_y; \sigma)$$

# Метрики качества

PSNR: Peak Signal to Noise Ratio / Пиковое соотношение сигнал/шум

$L$  — максимально возможное значение пикселя  
(255 для 8-битного изображения)

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (I(i) - \hat{I}(i))^2,$$

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{L^2}{\text{MSE}}\right).$$

## Метрики качества

SSIM - structural similarity index measure / Индекс структурного сходства

$\mu$  - среднее,  $\sigma$  - дисперсия,  $c$  - корреляция обоих изображений

$L$  — максимально возможное значение пикселя  
(255 для 8-битного изображения)

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

$$c1 = (k1 * L)^2, k1 = 0.01$$

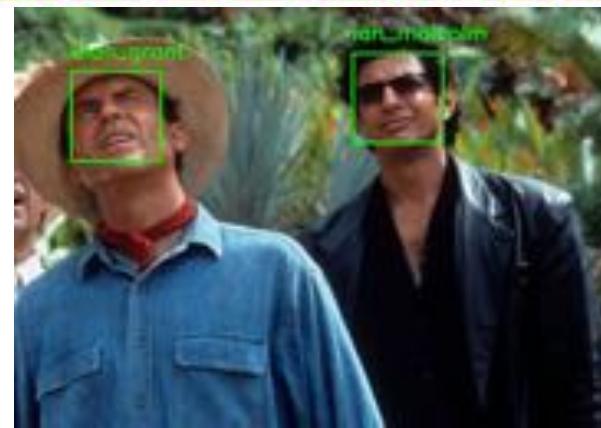
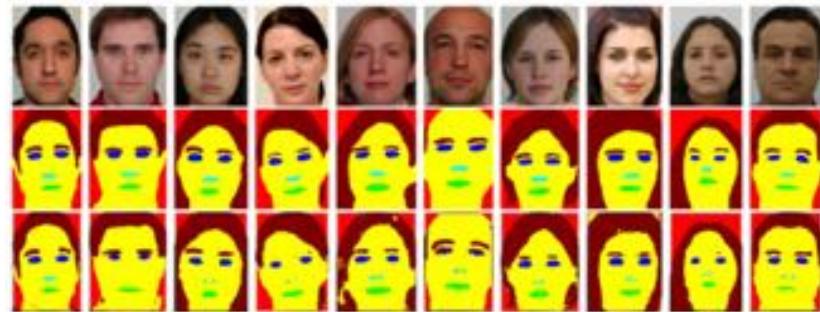
$$c2 = (k2 * L)^2, k2 = 0.03$$

# Восстановление 3D форм

Input	Ground truth	AtlasNet	OGN	Matryoshka	Clustering	Retrieval	Oracle NN
							
							
							
							

# Распознавание лиц (пример идентификации)

- Face Detection
- Features Segmentation
- Face Recognition



# Основные проблемы

- Проблема освещенности



- Проблема положения головы

## Метрики качества

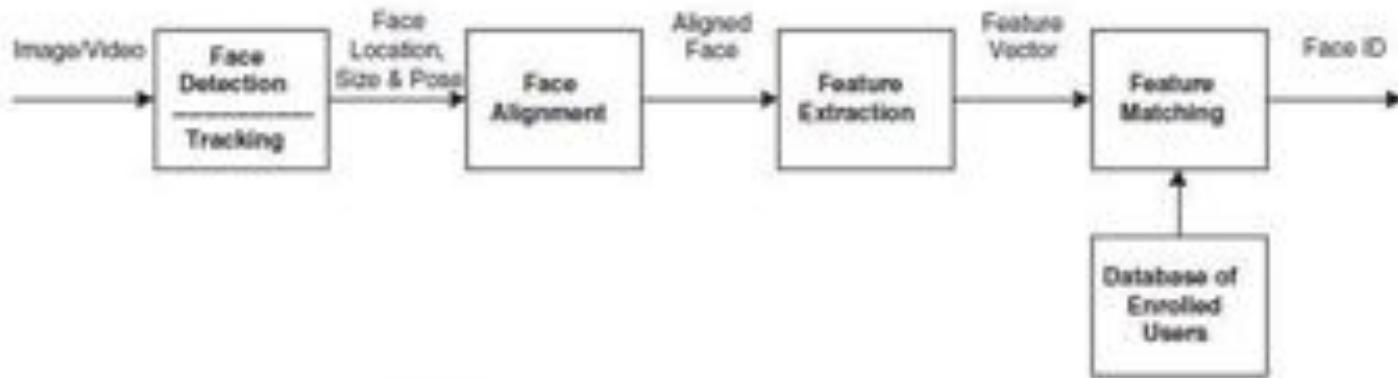
HTER (Half-Total Error Rate) – половина полной ошибки

FAR (False Acceptance Rate) – ошибочно разрешенных идентификаций

FRR (False Rejection Rate) – ошибочно запрещенных идентификаций

$$\text{HTER} = \frac{1}{2} (\text{FAR} + \text{FRR})$$

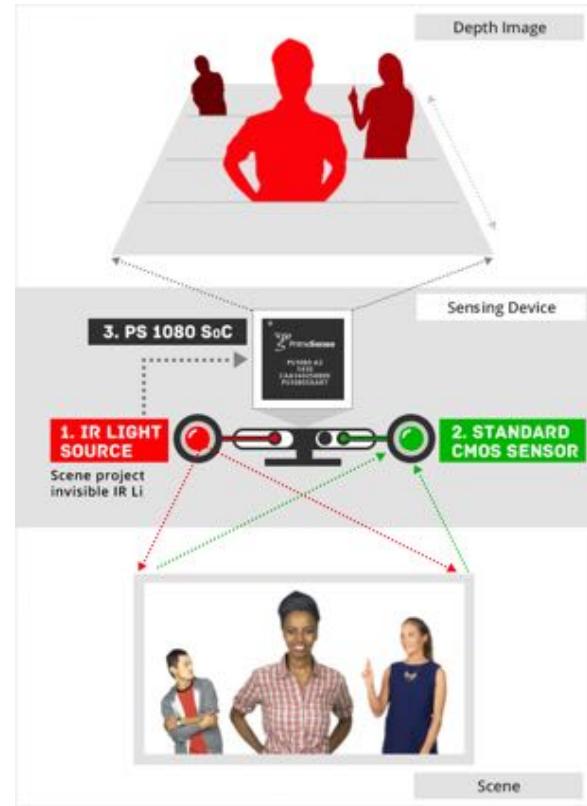
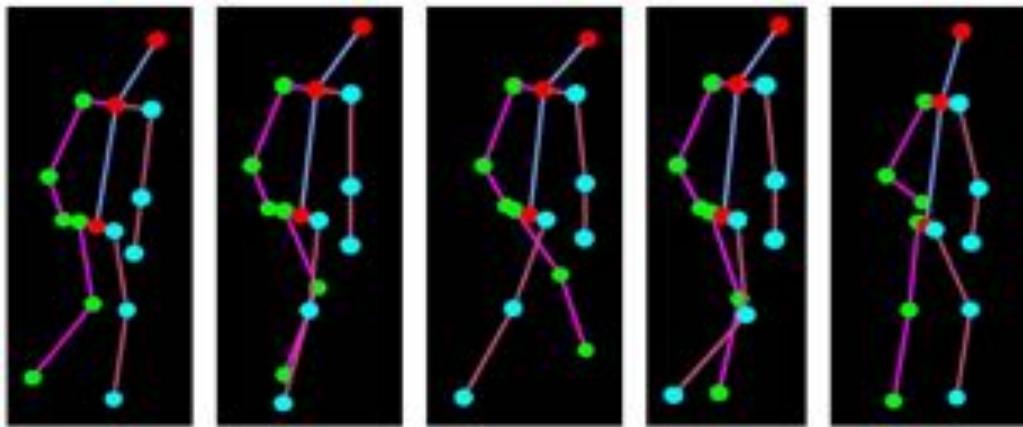
# Методы



Face recognition processing flow.

## Определение позы (пример оценки движения)

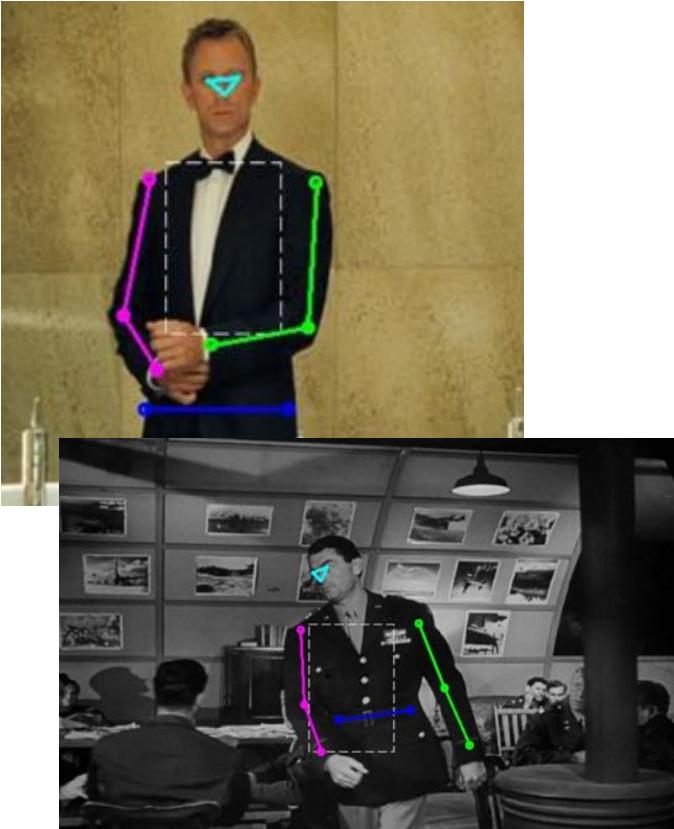
# Применения



# Скелет позы



# Датасеты



*Frames Labeled In Cinema (FLIC)*

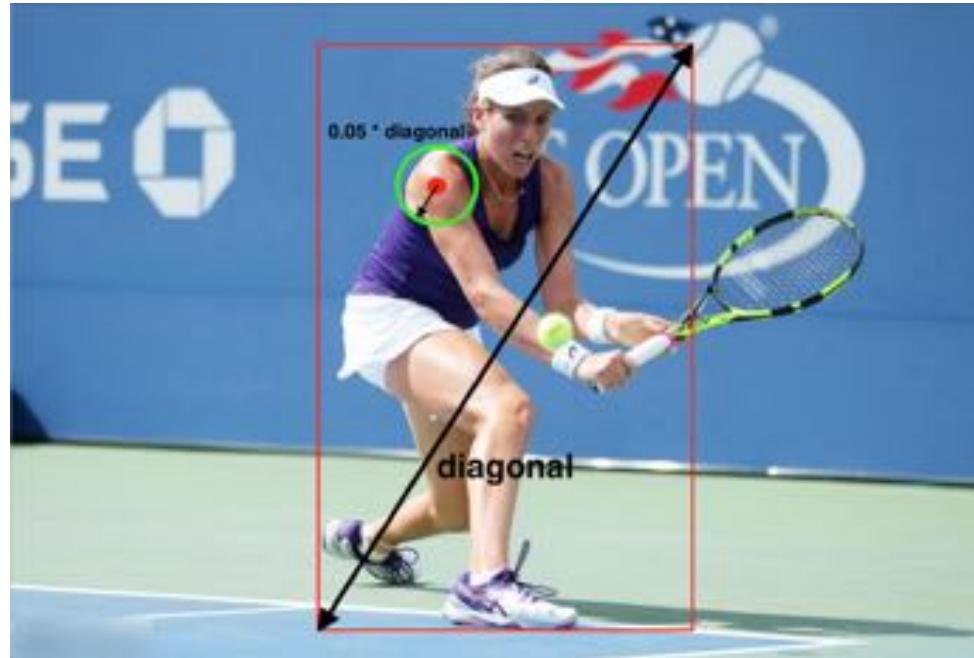


*Leeds Sports Pose Dataset (LSP)*

# Метрики

*Percent of Detected Joints (PDJ)*

$$PDJ = \frac{\sum_{i=1}^n \text{bool}(d_i < 0.05 * \text{diagonal})}{n}$$



# Метрики

*Object Keypoint Similarity (OKS)*

$$OKS = \exp\left(-\frac{d_i^2}{2s^2k_i^2}\right)$$

# Техники



*Вверху: Top-down подход.  
Снизу: Bottom-up подход.*

# Отслеживание жестов

[air-drawing](#)

[YoHa | handtracking.io](#)



## Оптическое распознавание символов

- Чтение штрих-кодов
- Чтение капчи



- Распознавание текста
- Распознавание номеров машин
- Распознавание дорожных знаков



# Перенос стиля



# Попробовать

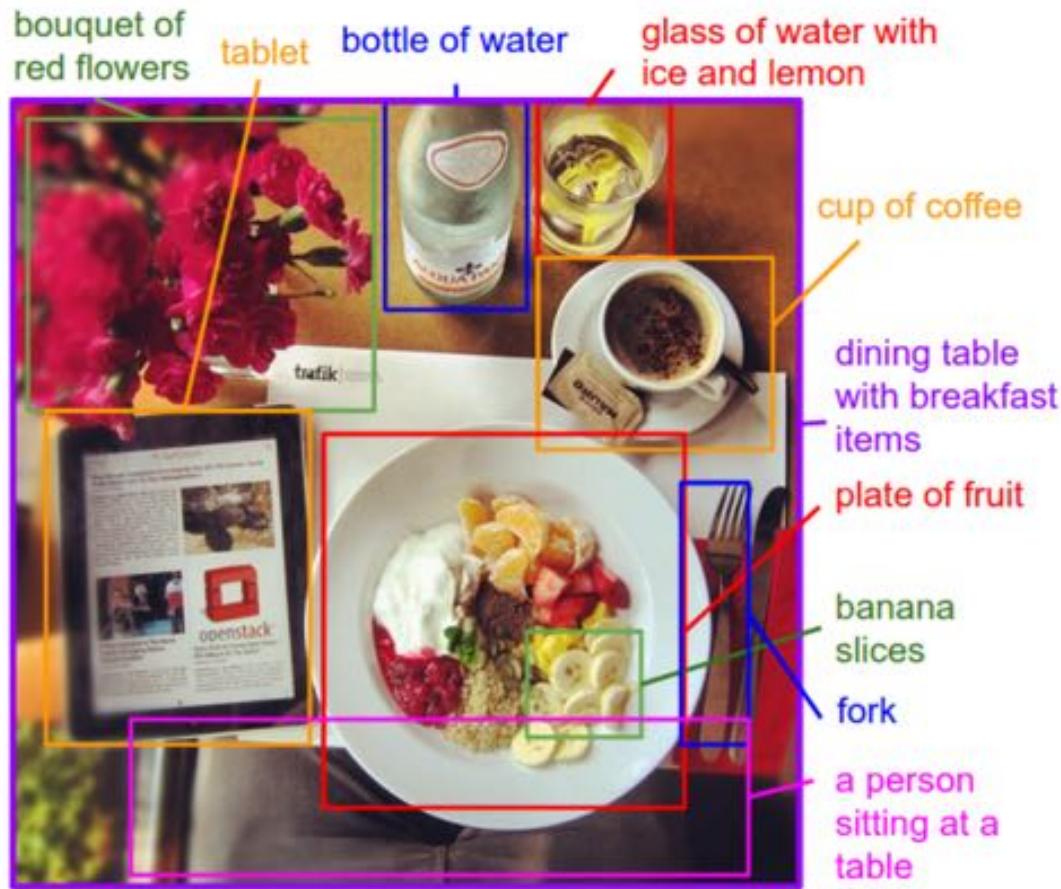
[Stylegan-Nada](#)



[StyleCariGAN](#)



# Текстовое описание изображений



Или наоборот, изображение по тексту

[Text2Animation](#)



+

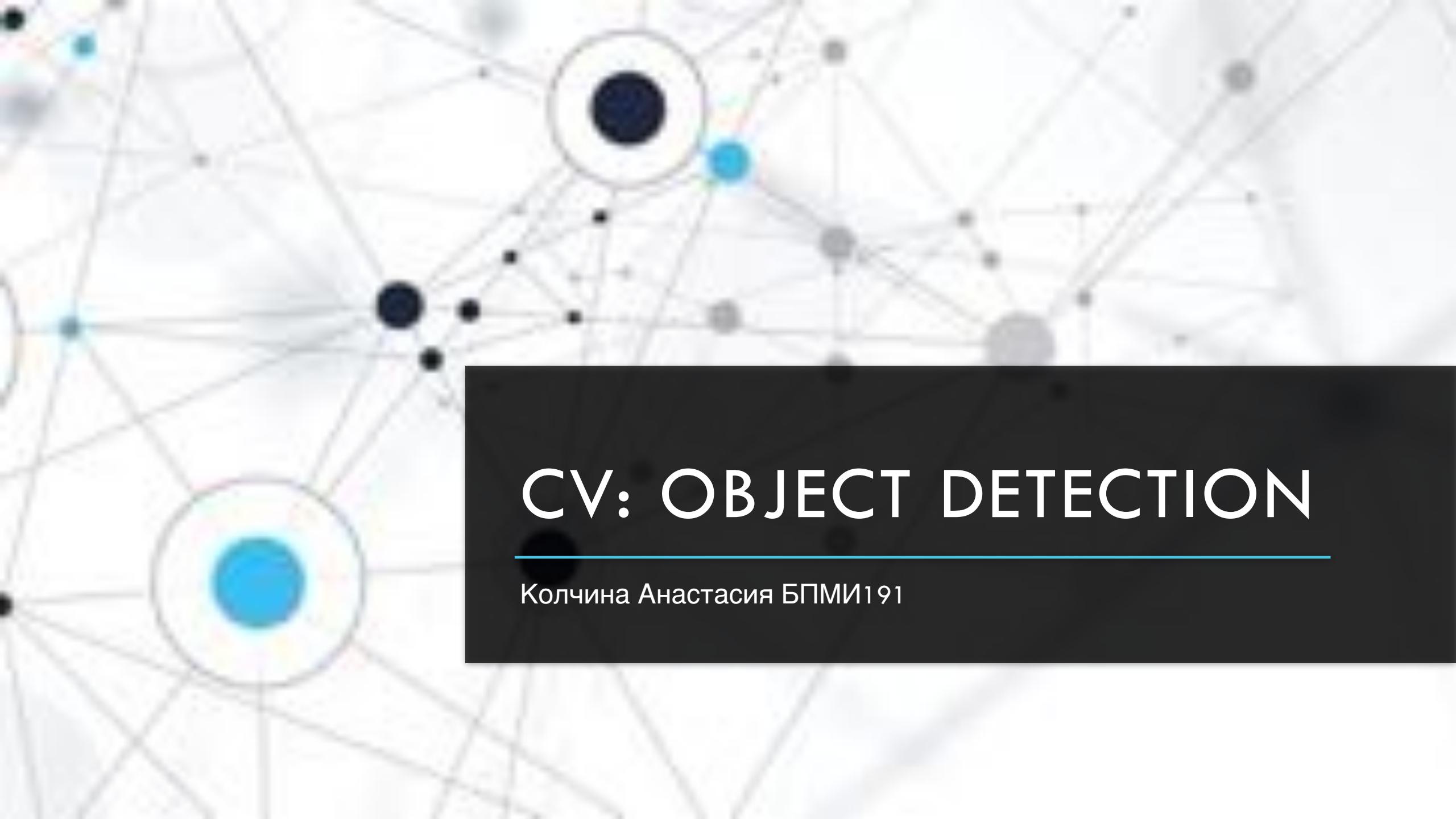


= ?

[neuralblender](#)



[Процедурно сгенерированный город](#)

A complex network graph serves as the background, featuring numerous small black dots connected by thin grey lines. Several larger, semi-transparent circular nodes are overlaid: one large blue node in the bottom left, one large dark blue node at the top center, and several smaller cyan and yellow nodes scattered across the frame.

# CV: OBJECT DETECTION

---

Колчина Анастасия БПМИ191

# CLASSIFICATION – DETECTION - SEGMENTATION

Is this a dog?



Image Classification

What is there in image  
and where?



Object Detection

Which pixels belong to  
which object?

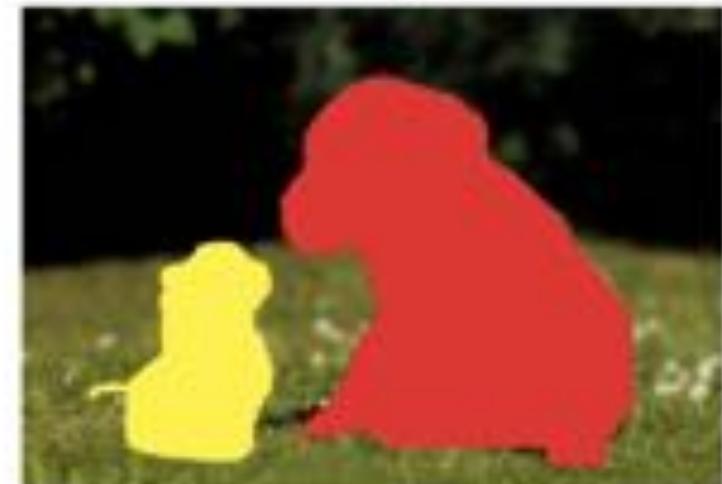


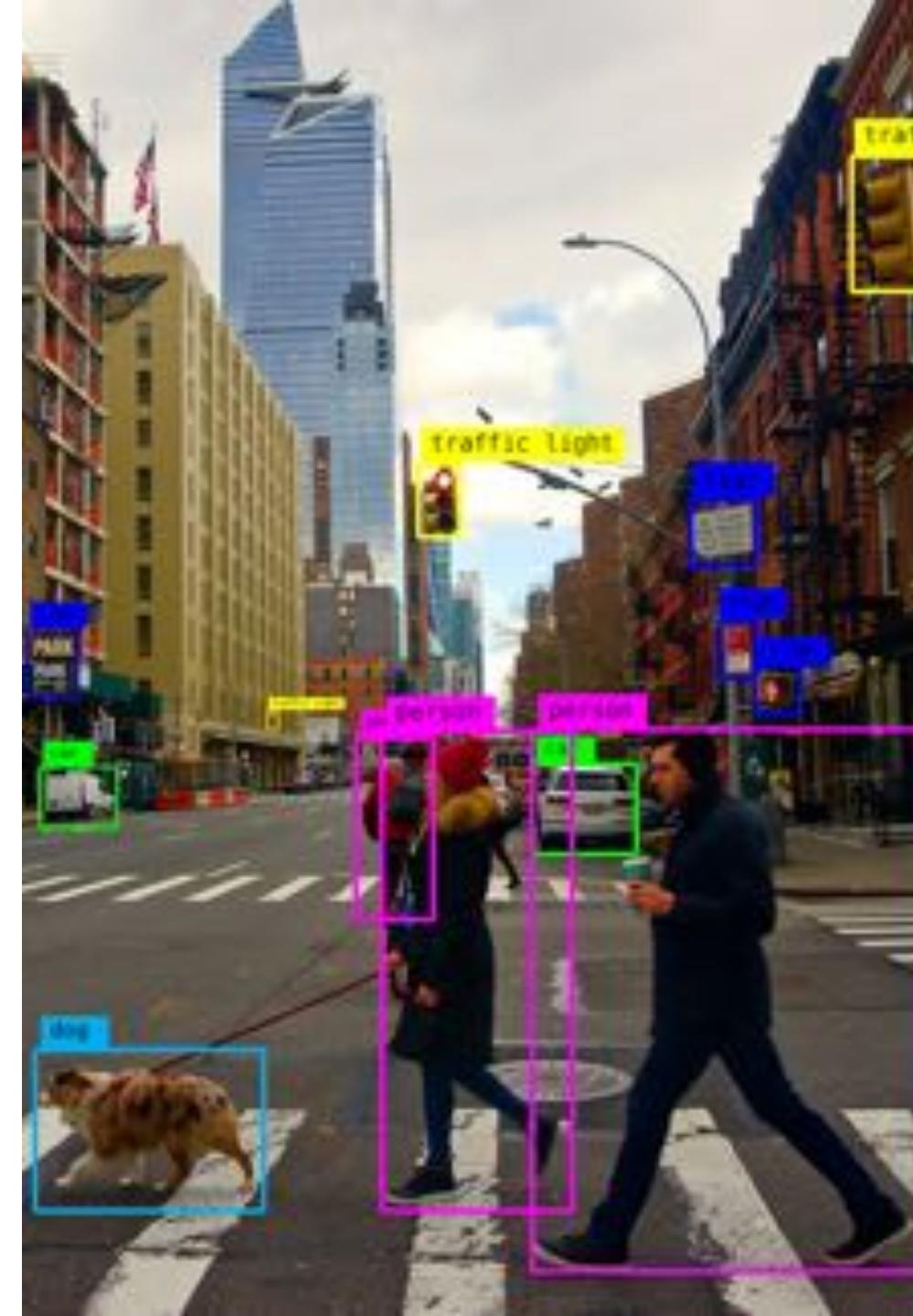
Image Segmentation

# ОБЪЕКТ DETECTION

Обнаружение объекта =  
локализация (рамка-граница) +  
классификация

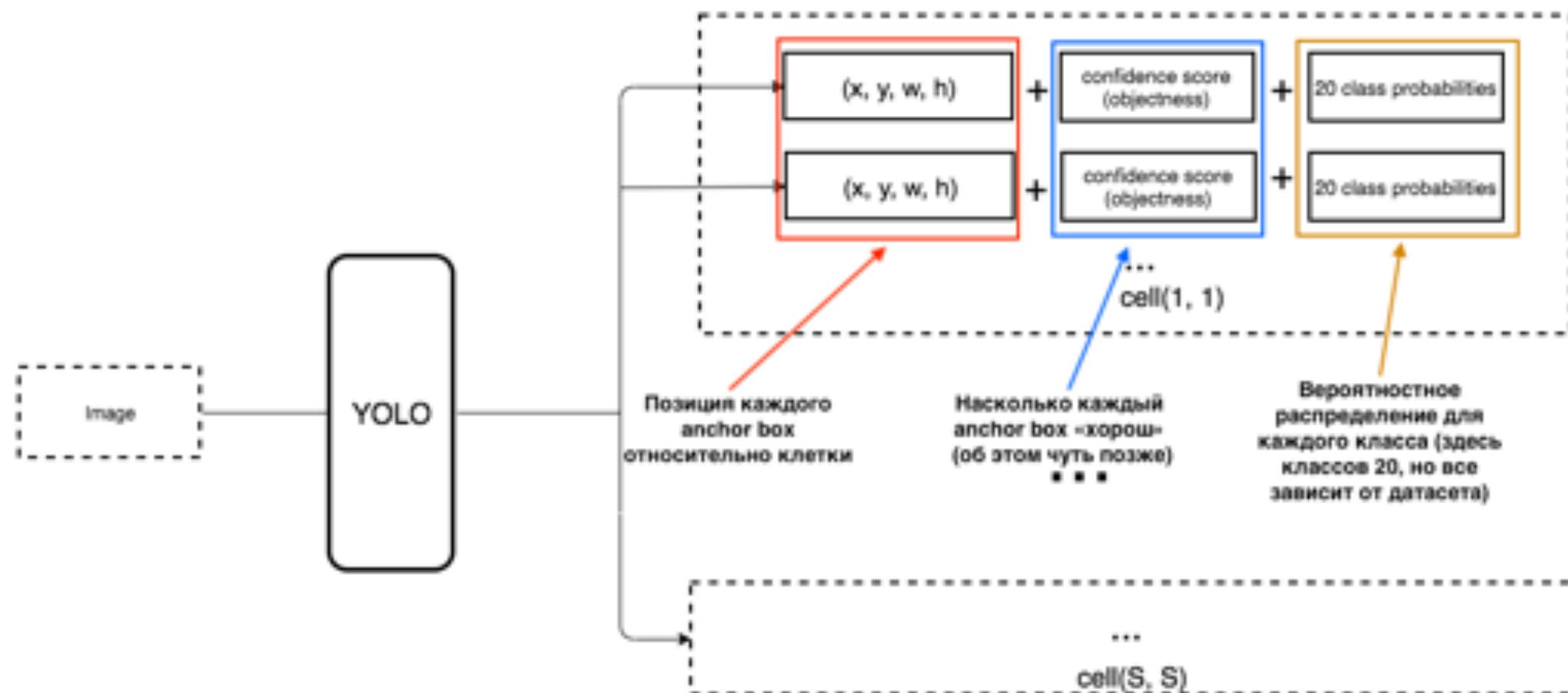
**Input:** изображение с некоторым  
количеством объектов

**Output:** рамки объектов (координаты  
пикселей) + классы объектов



# OUTPUT МОДЕЛИ

Детекция – это классификация и регрессия



# НАБОРЫ ДАННЫХ

- Google Open Images
- ImageNet
- PASCAL VOC 2012
- MS COCO 2017

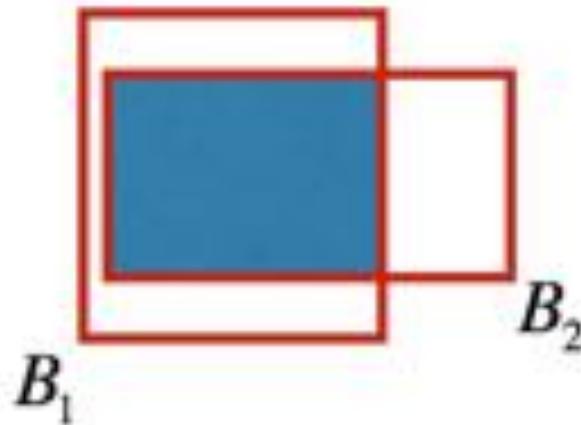


Google Open Source

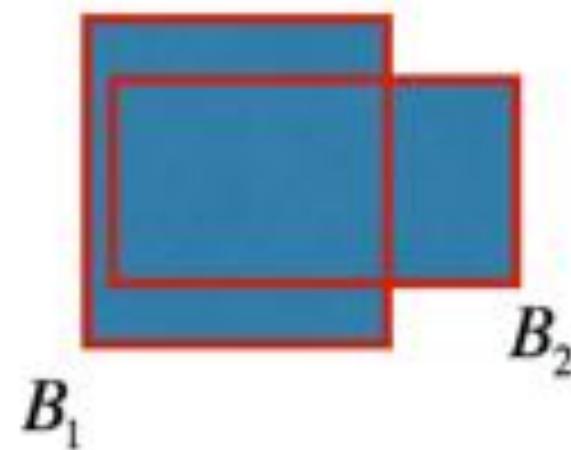


# МЕТРИКА IOU – INTERSECTION OVER UNION

Intersection



Union

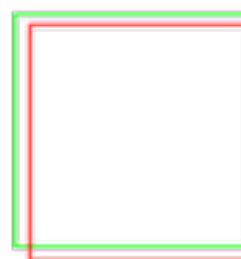


Intersection over Union

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2} = \frac{\text{Area of intersection}}{\text{Area of union}}$$

A diagram illustrating the formula for IoU. It shows two overlapping rectangles, \$B\_1\$ and \$B\_2\$, with their intersection highlighted in blue and their union highlighted in orange. Below this, three examples show different overlap scenarios with corresponding IoU values and quality labels.

IoU=0.92



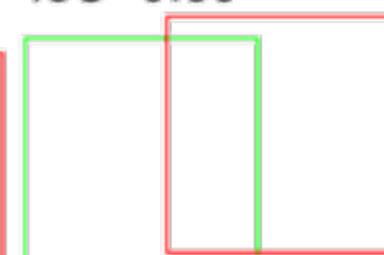
Excellent

IoU=0.71



Good

IoU=0.39



Poor

# ПОДХОДЫ К РЕШЕНИЮ ЗАДАЧИ

## Two-stage methods (region-based)

- Находим “регионы интереса” изображения
- Предсказываем класс
- Определяем местоположение

Точно, но очень медленно.

## One-stage methods

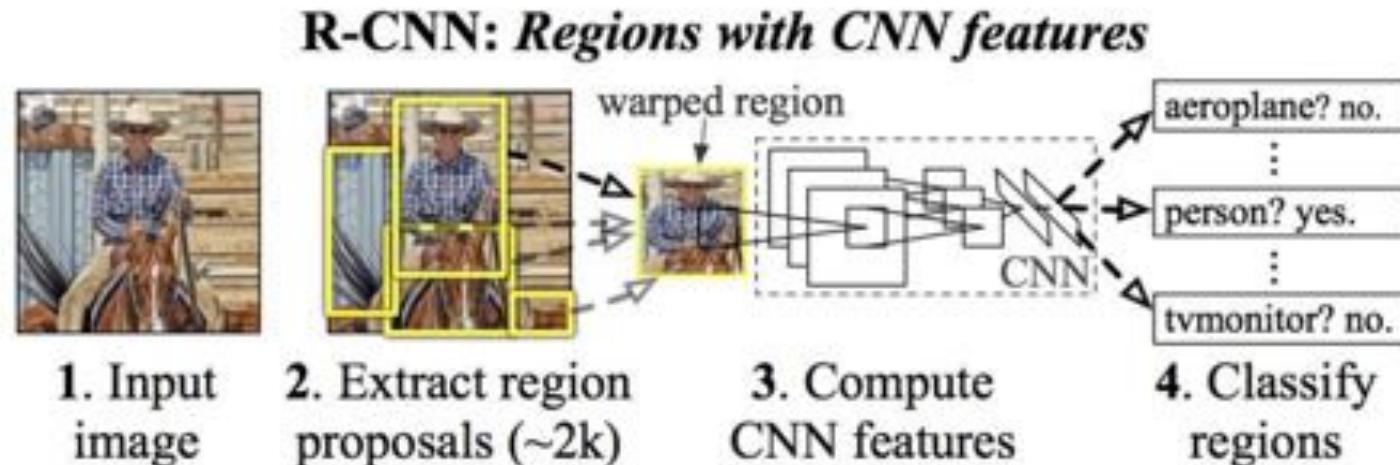
Предсказываем координаты ограничивающих рамок с разными характеристиками (степень уверенности и т.д.), корректируя их локацию

Real-time detection – быстро, но неточно.

Поговорим о Two-stage methods

# RCNN

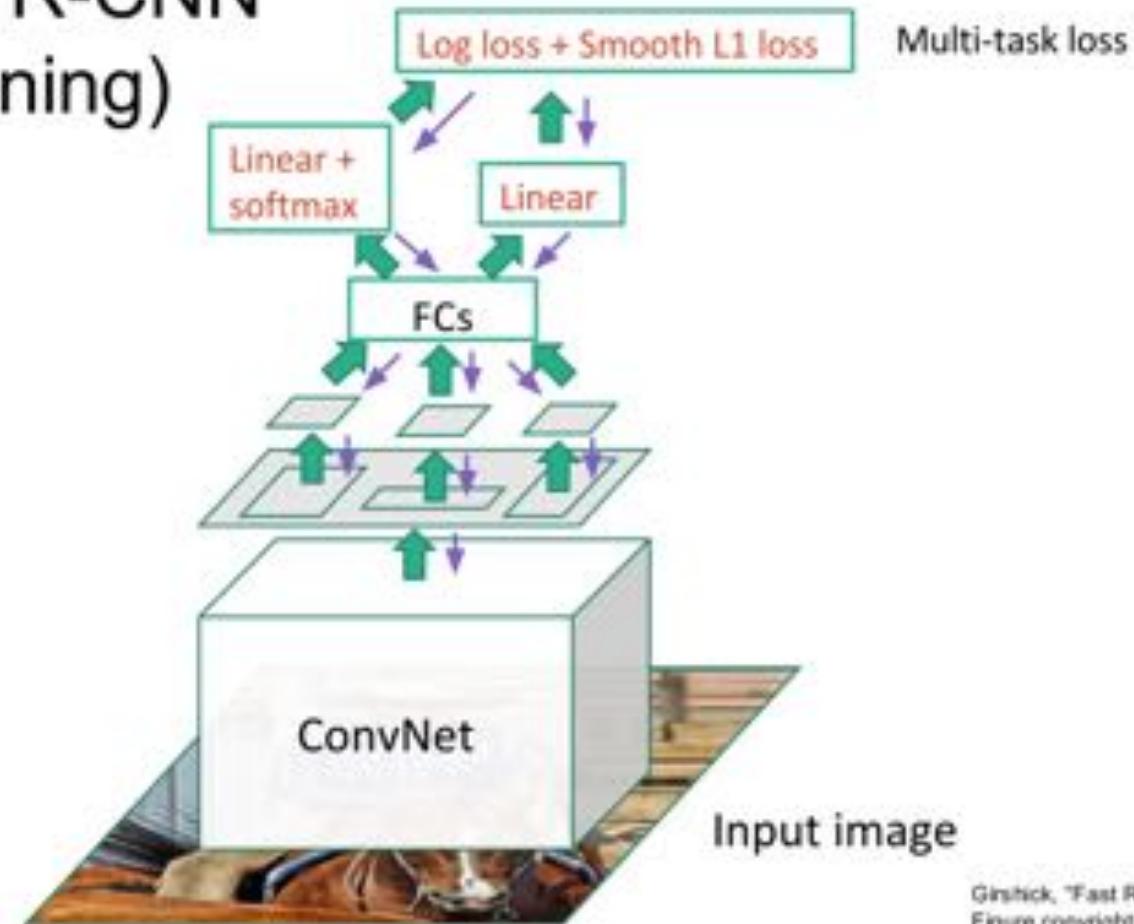
1. Определение набора гипотез.
2. Извлечение из предполагаемых регионов признаков с помощью CNN и их кодирование в вектор.
3. Классификация объекта внутри региона.
4. Улучшение (корректировка) координат региона.



# FAST RCNN

- Нейронная сеть запускается только один раз на изображение – все гипотезы проверяются на основе единой карты признаков.
- «Умная» обработка гипотез разного размера за счет RoI слоя.
- Multi-task loss – функция потерь решает задачу и классификации, и регрессии

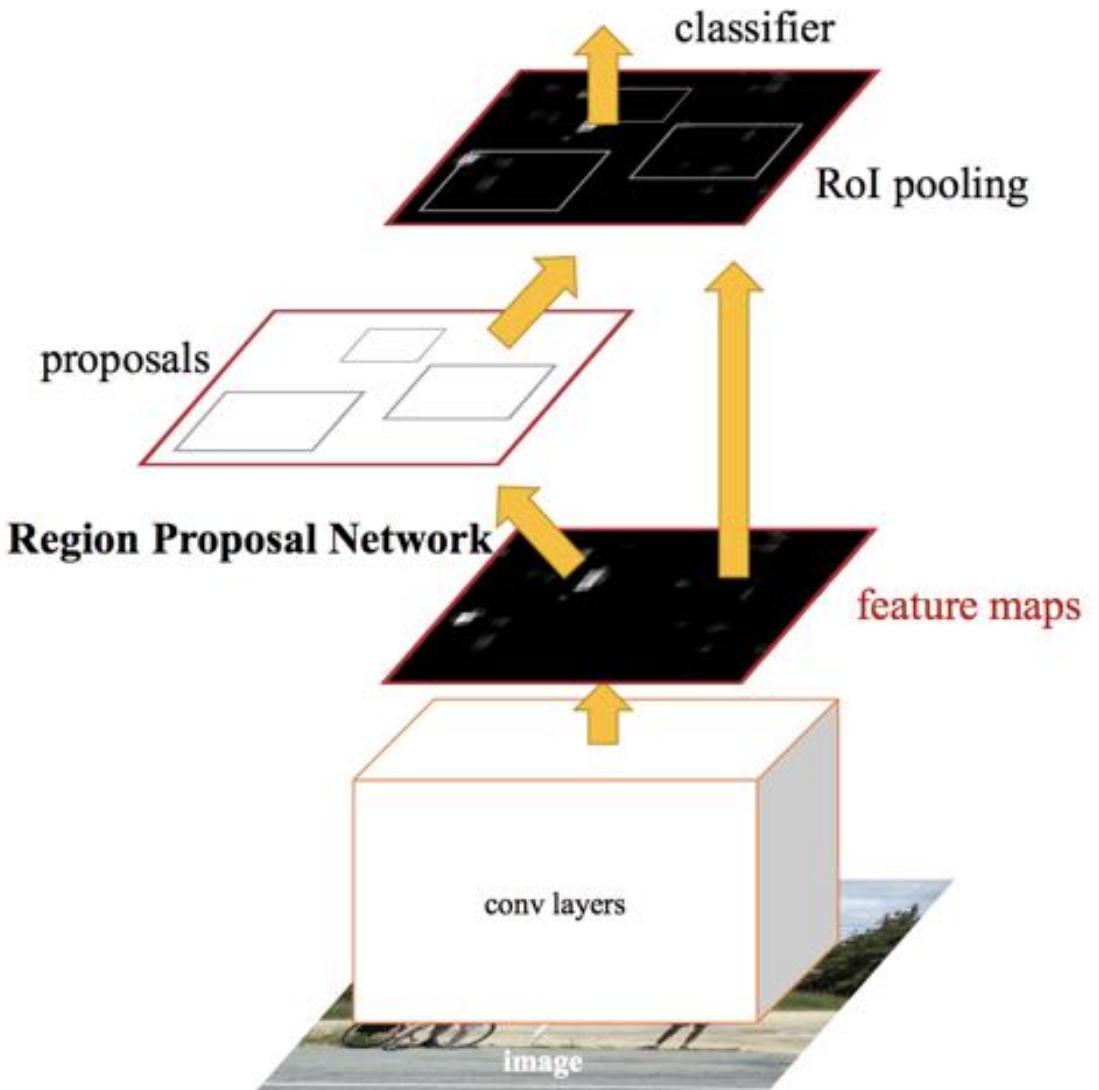
## Fast R-CNN (Training)



Girshick, "Fast R-CNN".  
Figure copyright © 2014 Microsoft Corporation.

# FASTER RCNN

1. Подается изображение на вход
2. Картинка прогоняется через CNN для формирования feature maps
3. Отдельной нейронной сетью определяются регионы с высокой вероятностью нахождения в них объектов
4. Эти регионы с помощью RoI pooling сжимаются и подаются в нейронную сеть – классификатор
  - Самая быстрая из этих трех моделей.
  - Является одной из самых точных и по сей день.

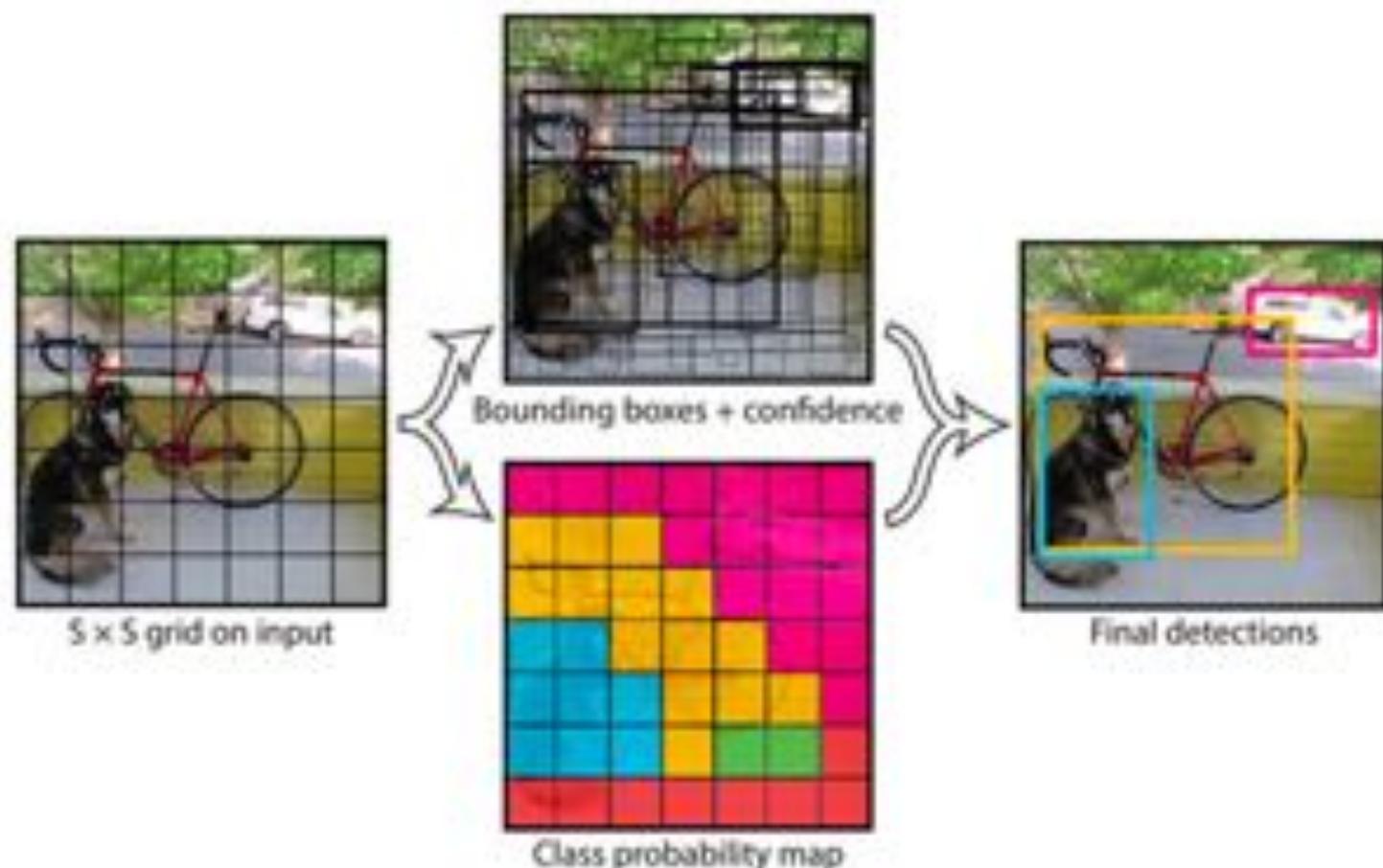


Теперь о самом интересном: one-stage methods

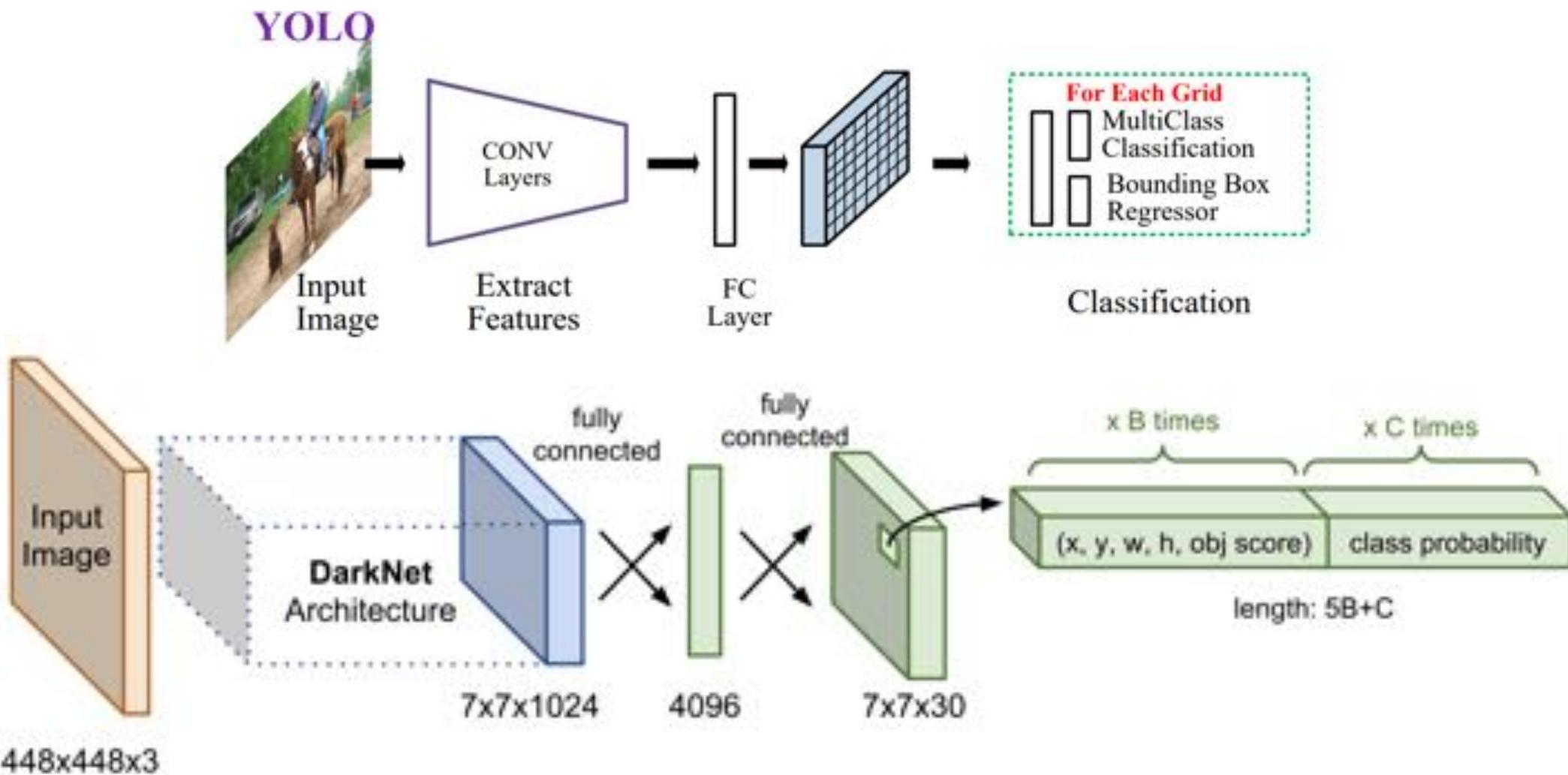
# YOLO (YOU ONLY LOOK ONCE)

YOLO (You Only Look Once) – смотрим на картинку один раз, и за этот просмотр (т.е. один прогон картинки через одну нейросеть) делаем все необходимые определения объектов.

By Joseph Redmon, 2015



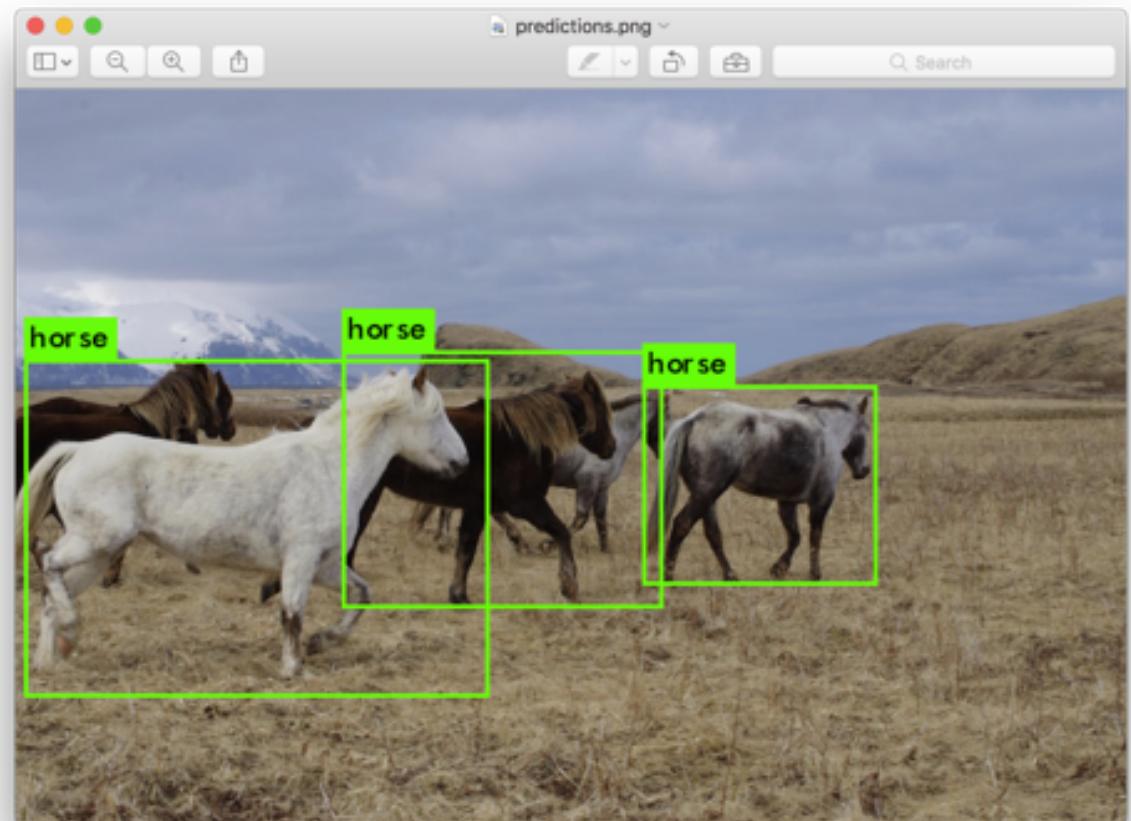
# АРХИТЕКТУРА YOLO



# BACKBONE – FEATURE MAPPING

	Type	Filters	Size	Output
1x	Convolutional	32	$3 \times 3$	$256 \times 256$
	Convolutional	64	$3 \times 3 / 2$	$128 \times 128$
	Convolutional	32	$1 \times 1$	
	Convolutional	64	$3 \times 3$	
	Residual			$128 \times 128$
2x	Convolutional	128	$3 \times 3 / 2$	$64 \times 64$
	Convolutional	64	$1 \times 1$	
	Convolutional	128	$3 \times 3$	
	Residual			$64 \times 64$
8x	Convolutional	256	$3 \times 3 / 2$	$32 \times 32$
	Convolutional	128	$1 \times 1$	
	Convolutional	256	$3 \times 3$	
	Residual			$32 \times 32$
8x	Convolutional	512	$3 \times 3 / 2$	$16 \times 16$
	Convolutional	256	$1 \times 1$	
	Convolutional	512	$3 \times 3$	
	Residual			$16 \times 16$
4x	Convolutional	1024	$3 \times 3 / 2$	$8 \times 8$
	Convolutional	512	$1 \times 1$	
	Convolutional	1024	$3 \times 3$	
	Residual			$8 \times 8$
	Avgpool		Global	
	Connected		1000	
	Softmax			

Table 1. Darknet-53.



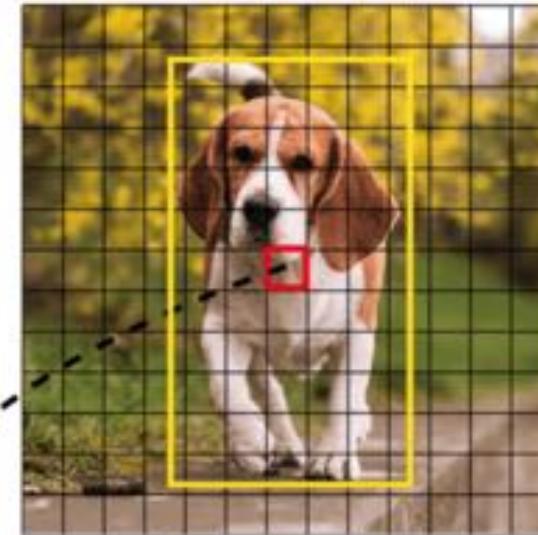
<https://pjreddie.com/darknet/yolo/>

# PREDICT

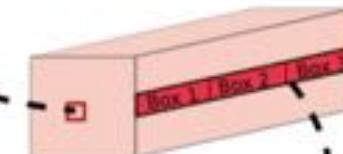
Ограничивающая рамка (bounding box) – координаты, ограничивающие определенную область изображения

$B$  – число bounding box'ов

Свертка  $1 \times 1$  предсказывает вероятности



Prediction Feature Map



Attributes of a bounding box

$t_x$	$t_y$	$t_w$	$t_h$	$p_o$	$p_1$	$p_2$	....	$p_c$	$\times B$
Box Co-ordinates				Objectness Score				Class Scores	

# NON-MAX SUPPRESSION

Есть несколько box-ов с достаточно высокими вероятностями. Как выбрать нужный?

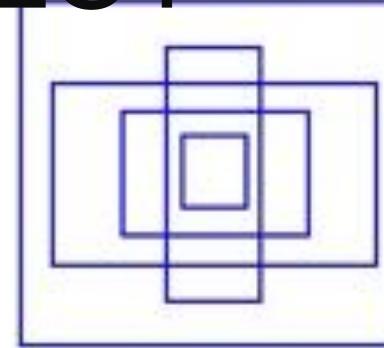
Алгоритм **Non max suppression**:

1. Ищем bounding box с наибольшей вероятностью принадлежности к объекту.
2. Пробегаем по всем bounding boxам объекта.
3. Удаляем их, если Intersection over Union (IoU) с первым bounding box-ом больше заданного порога.



# КАК ОБУЧАЕТСЯ YOLO?

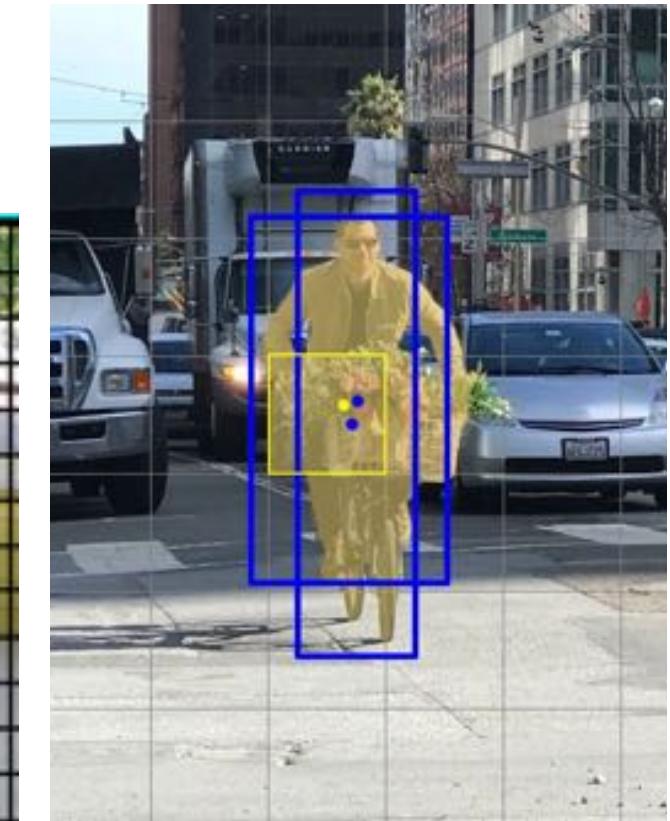
Делим картинку на  
клетки размером  $n \times n$   
(Yolo 3-4: 13x13)



Вокруг каждой клетки  
рисуются несколько  
bounding boxes



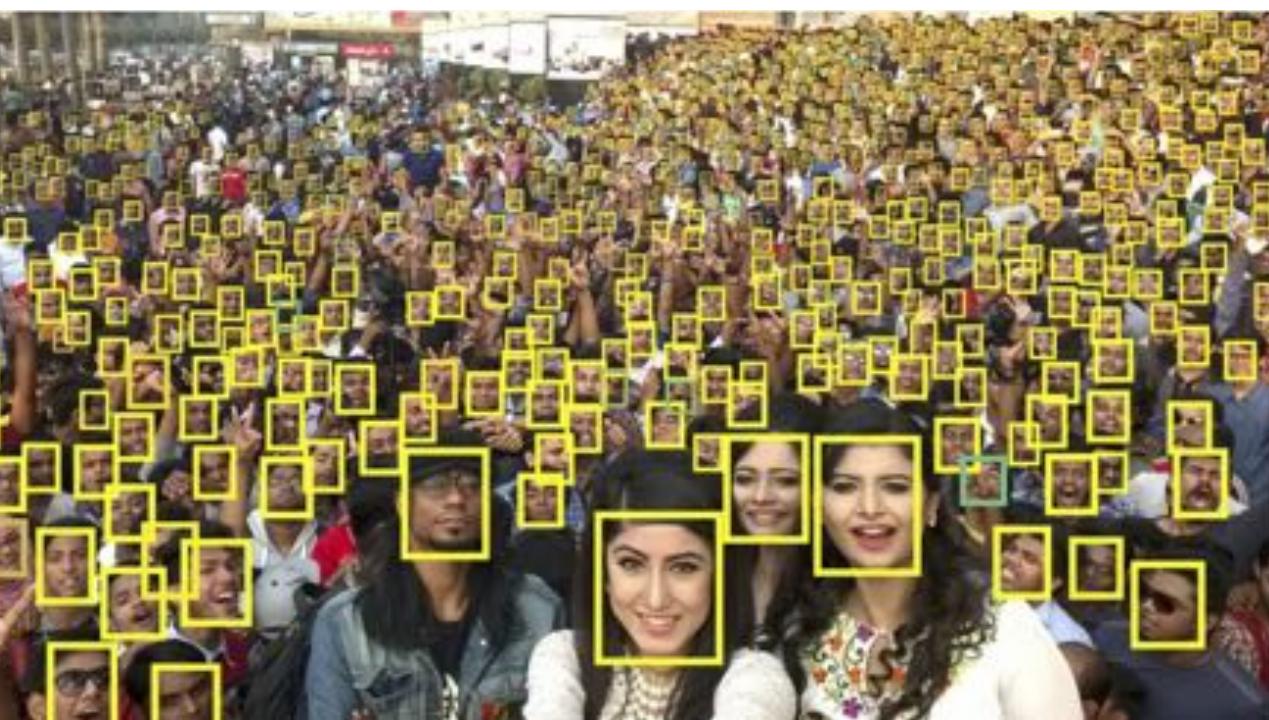
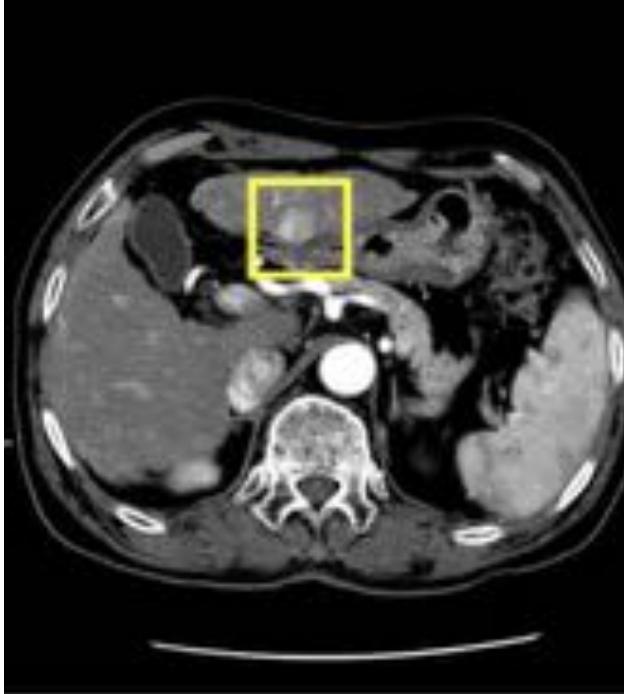
Картина прогоняется  
через нейросеть (в train  
должны быть позиции и  
размеры настоящих  
bounding boxes)



# ПОЧЕМУ YOLO?

- + YOLO в 1000 раз быстрее чем R-CNN и около 100x быстрее чем Fast R-CNN.
- + YOLO работает быстрее алгоритмов семейства R-CNN за счёт дробления на константное количество ячеек вместо того, чтобы предлагать регионы и рассчитывать решение для каждого региона отдельно.
- Проблема YOLO: плохое качество распознавания объектов сложной формы / группы небольших объектов из-за ограниченного числа кандидатов для ограничивающих рамок.





# ПРИМЕНЕНИЕ ДЕТЕКЦИИ

- Анализ спутниковых снимков – обнаружение географических объектов, лесных пожаров, мест крушения самолетов и т.д.
- Обнаружение опухолей, инородных тел в органах человека
- На дорогах: отслеживание потоков людей, детекция автомобилей, нарушающих ПДД + использование нейросетей в беспилотных автомобилях

# IMAGE SEGMENTATION

Maria  
Poklonskaya

# СЕГМЕНТАЦИЯ ИЗОБРАЖЕНИЯ

## ЦЕЛЬ

процесс присвоения метки каждому пикслю в изображении таким образом, чтобы пиксели с одинаковой меткой обладали определенными характеристиками.

## РЕЗУЛЬТАТ

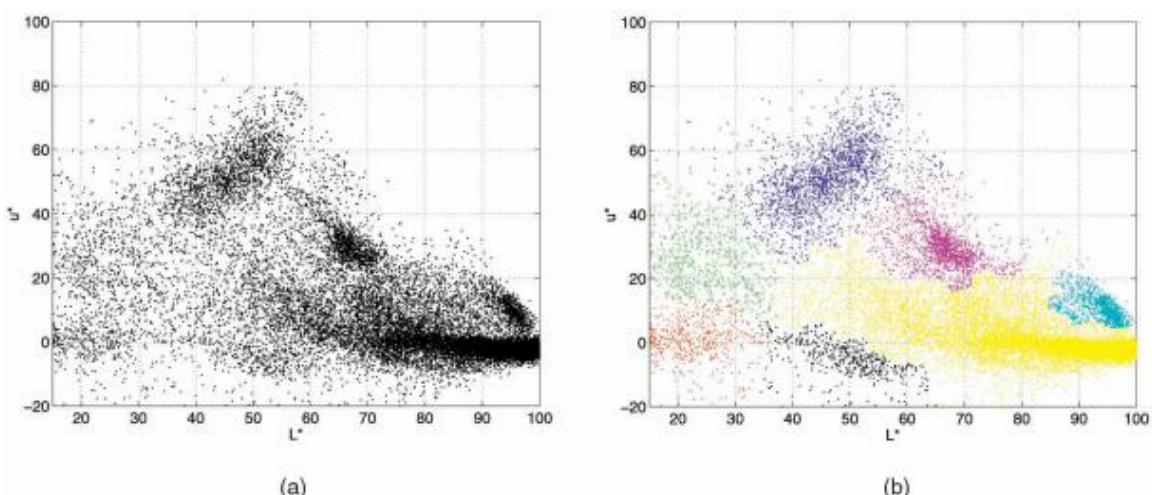
изменить представление изображения на что-то более значимое и более простое для анализа.

набор сегментов, которые вместе покрывают все изображение

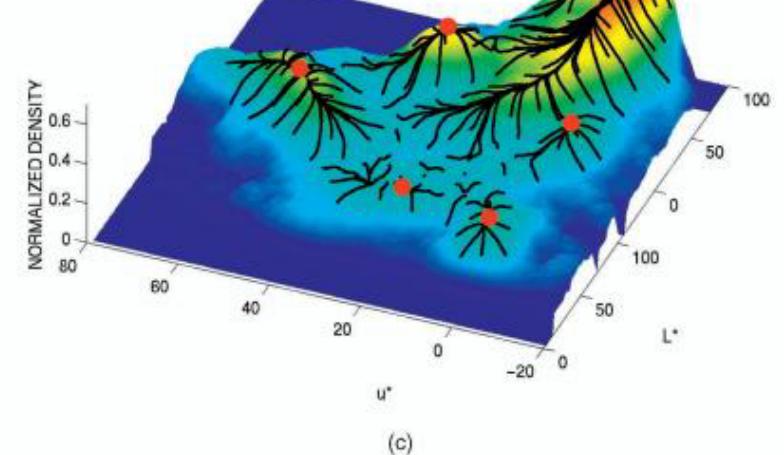
набор из контуров, извлеченных из изображения.

# АЛГОРИТМЫ СЕГМЕНТАЦИИ ИЗОБРАЖЕНИЙ

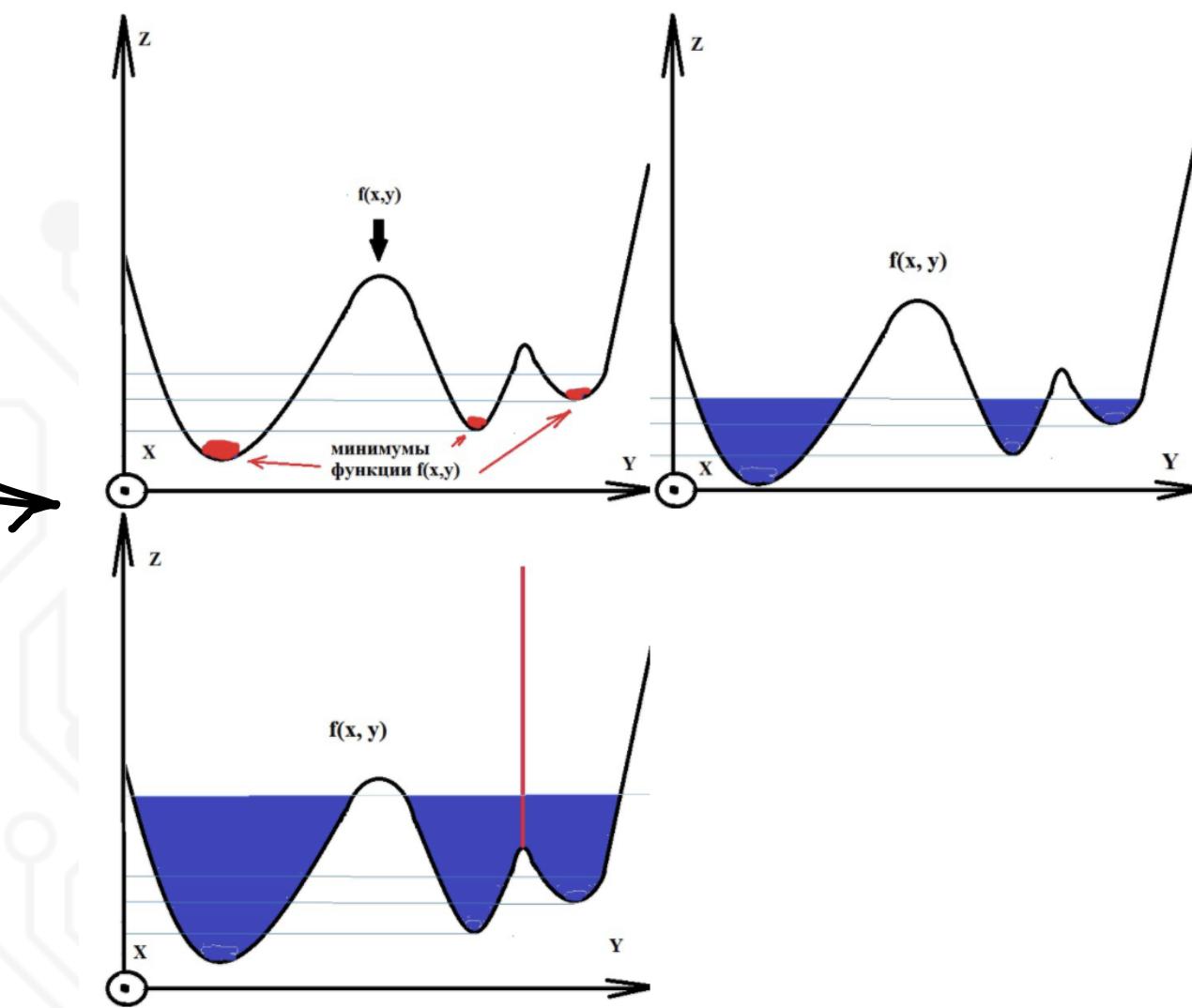
без использования нейронных сетей



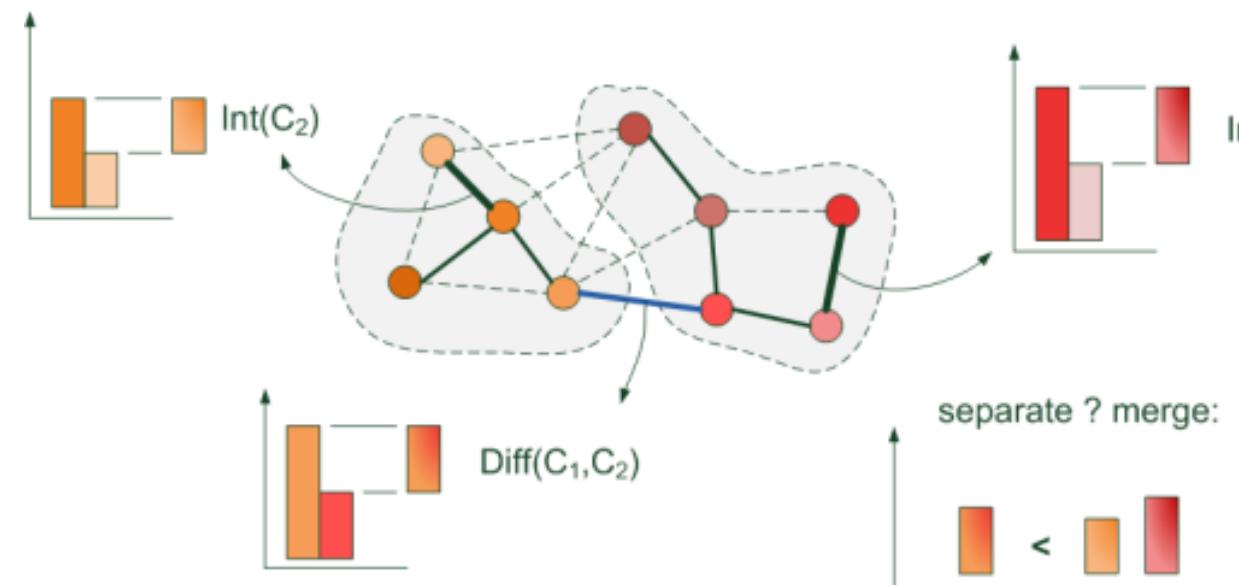
MEAN-SHIFT SEGMENTATION



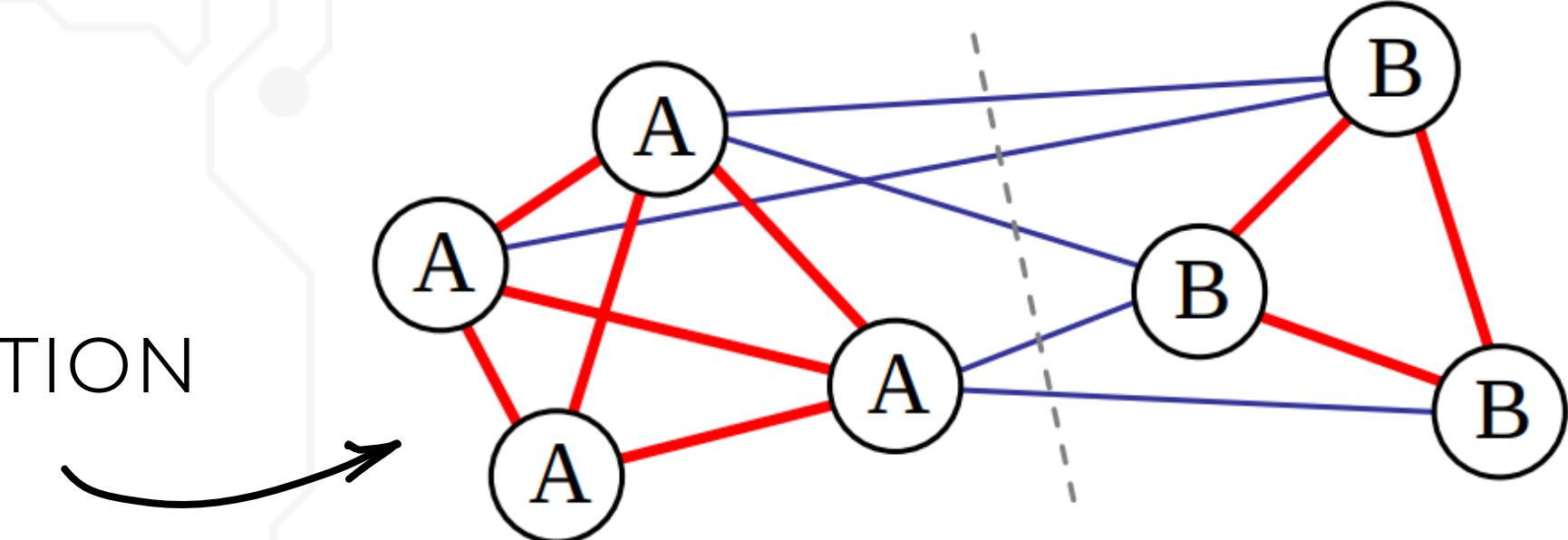
WATERSHED SEGMENTATION



GRAPH-BASED SEGMENTATION



NORMALIZED CUTS SEGMENTATION



# АЛГОРИТМЫ СЕГМЕНТАЦИИ ИЗОБРАЖЕНИЙ

без использования нейронных сетей



MEAN-SHIFT SEGMENTATION



WATERSHED SEGMENTATION



GRAPH-BASED SEGMENTATION



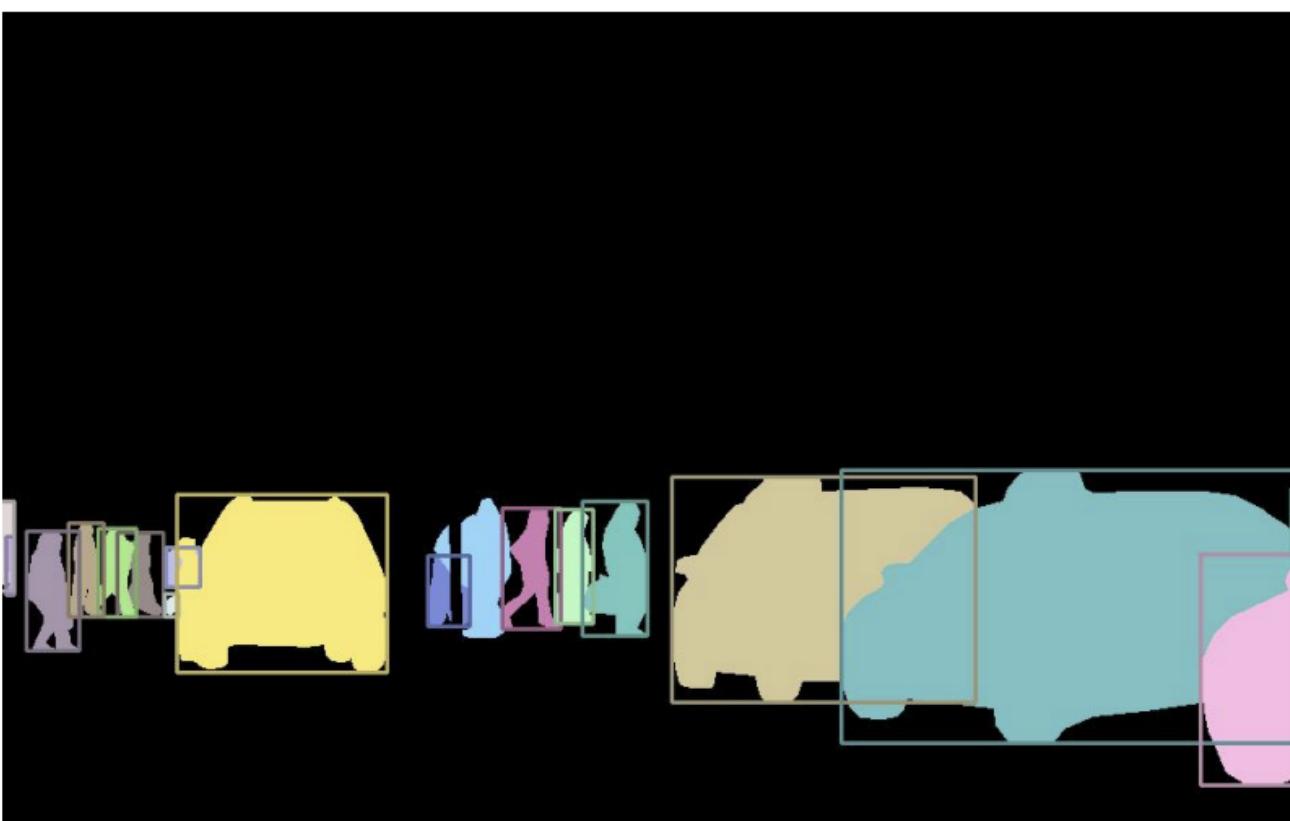
NORMALIZED CUTS SEGMENTATION

# Semantic vs Instance vs Panoptic segmentation

Исходное изображение



Сегментация экземпляра (Instance)



Семантическая сегментация (Semantic)



Паноптическая сегментация (Panoptic)



# SEMANTIC SEGMENTATION

Определяет принадлежность наборов пикселей на изображении к определенным классам объектов (собаки, люди, цветы, автомобили)



[https://blog.csdn.net/weixin\\_41697507](https://blog.csdn.net/weixin_41697507)

# SEMANTIC SEGMENTATION

## ОСНОВНЫЕ ПОДХОДЫ

- 
- 1 Областная семантическая сегментация

Из изображения извлекаются области свободной формы с их описанием, затем они классифицируются. Во время тестирования предсказания преобразуются в пиксели.
  - 2 Семантическая сегментация на основе полностью сверточной сети

Изображение обрабатывается от пикселя к пикселю без извлечения объектов по определенной области.
  - 3 Семантическая сегментация со слабым контролем

Слабо контролируемые методы, предназначенные для достижения семантической сегментации с использованием аннотированных ограничивающих рамок.

# INSTANCE SEGMENTATION

Задача обнаружения и выделения пикселей каждого отдельного интересующего объекта, появляющегося на изображении.

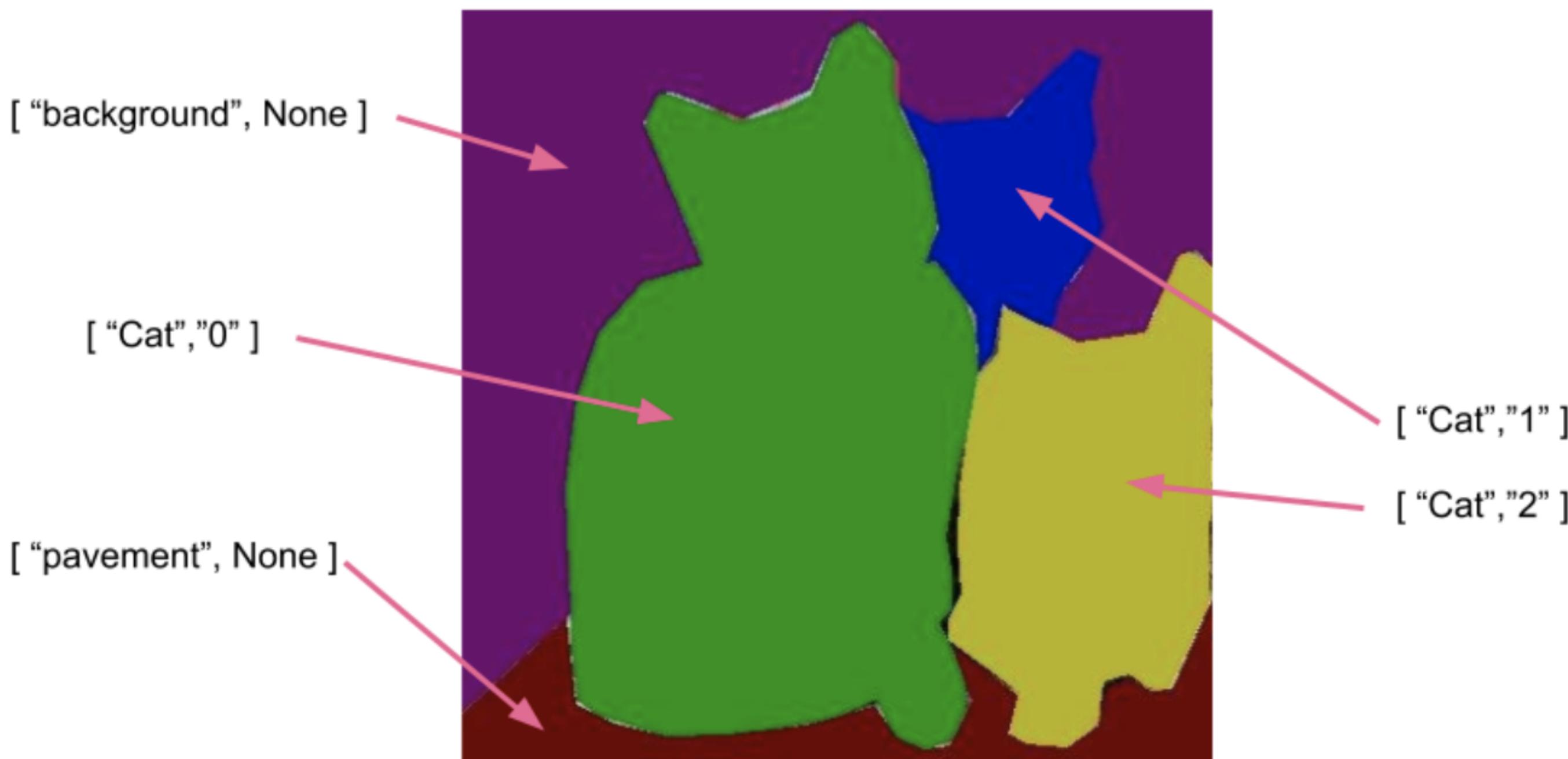


Применяется в задачах, где необходим подсчет количества объектов  
(например количество людей на фото)

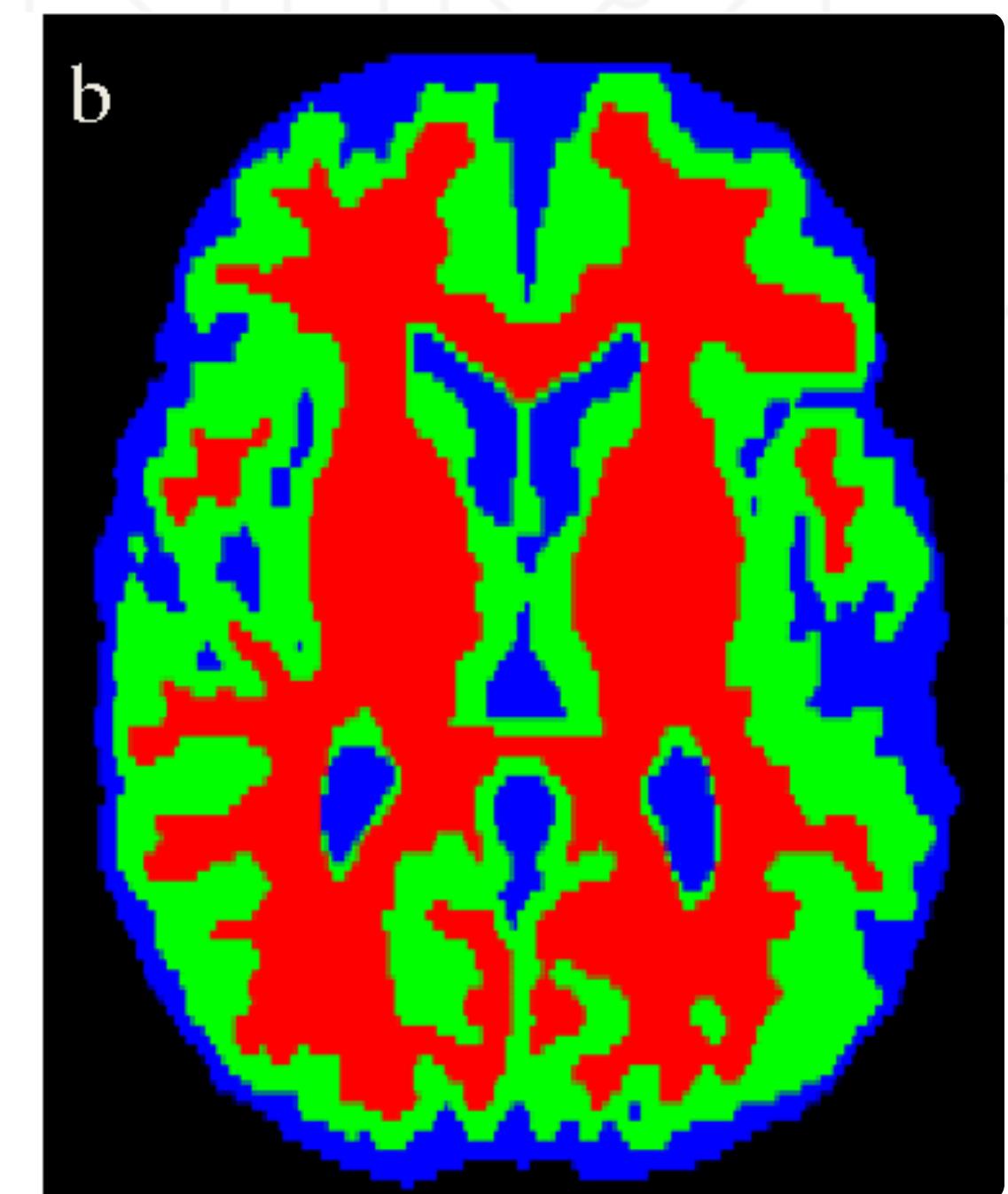
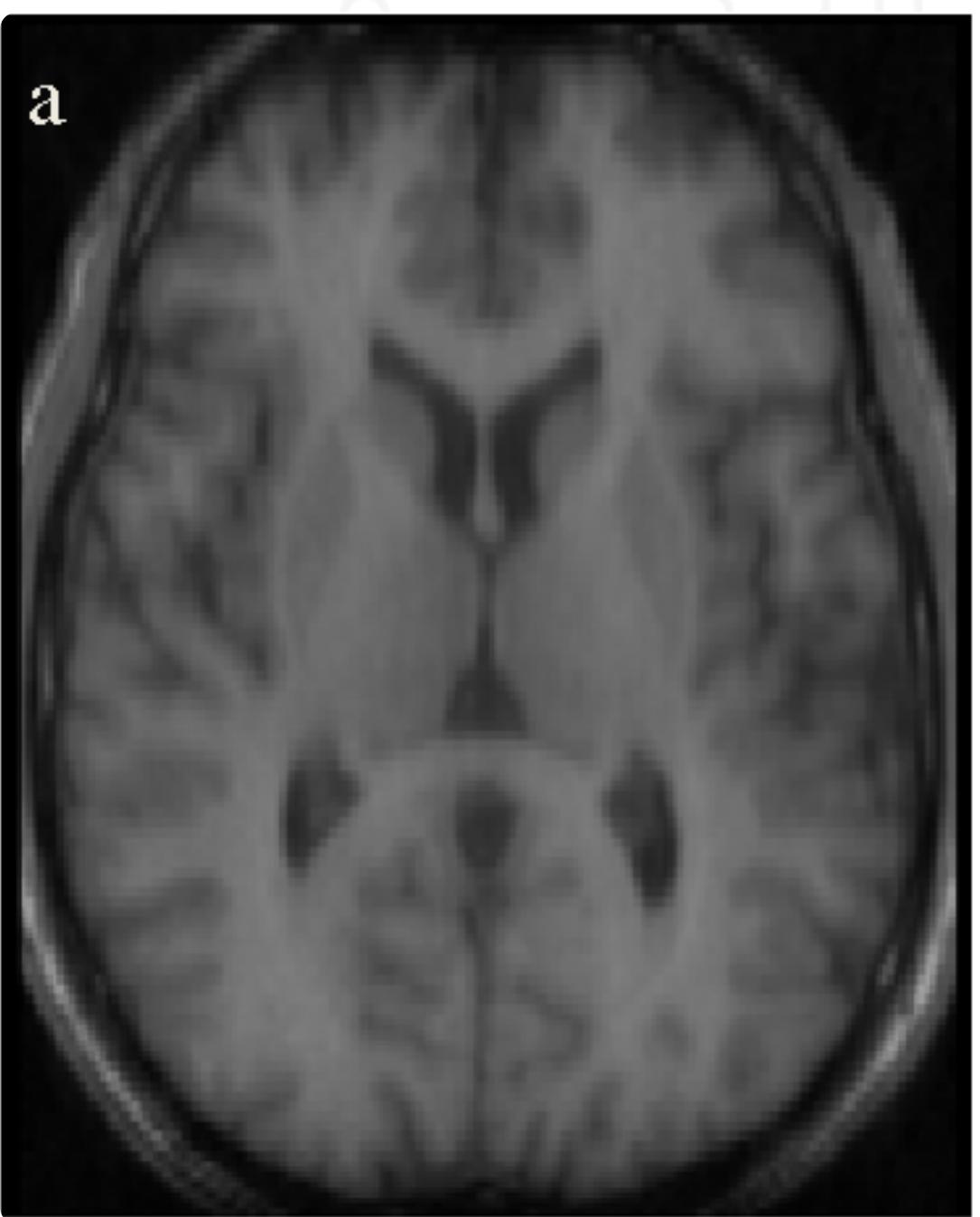
# PANOPTIC SEGMENTATION

SEMANTIC+ INSTANCE

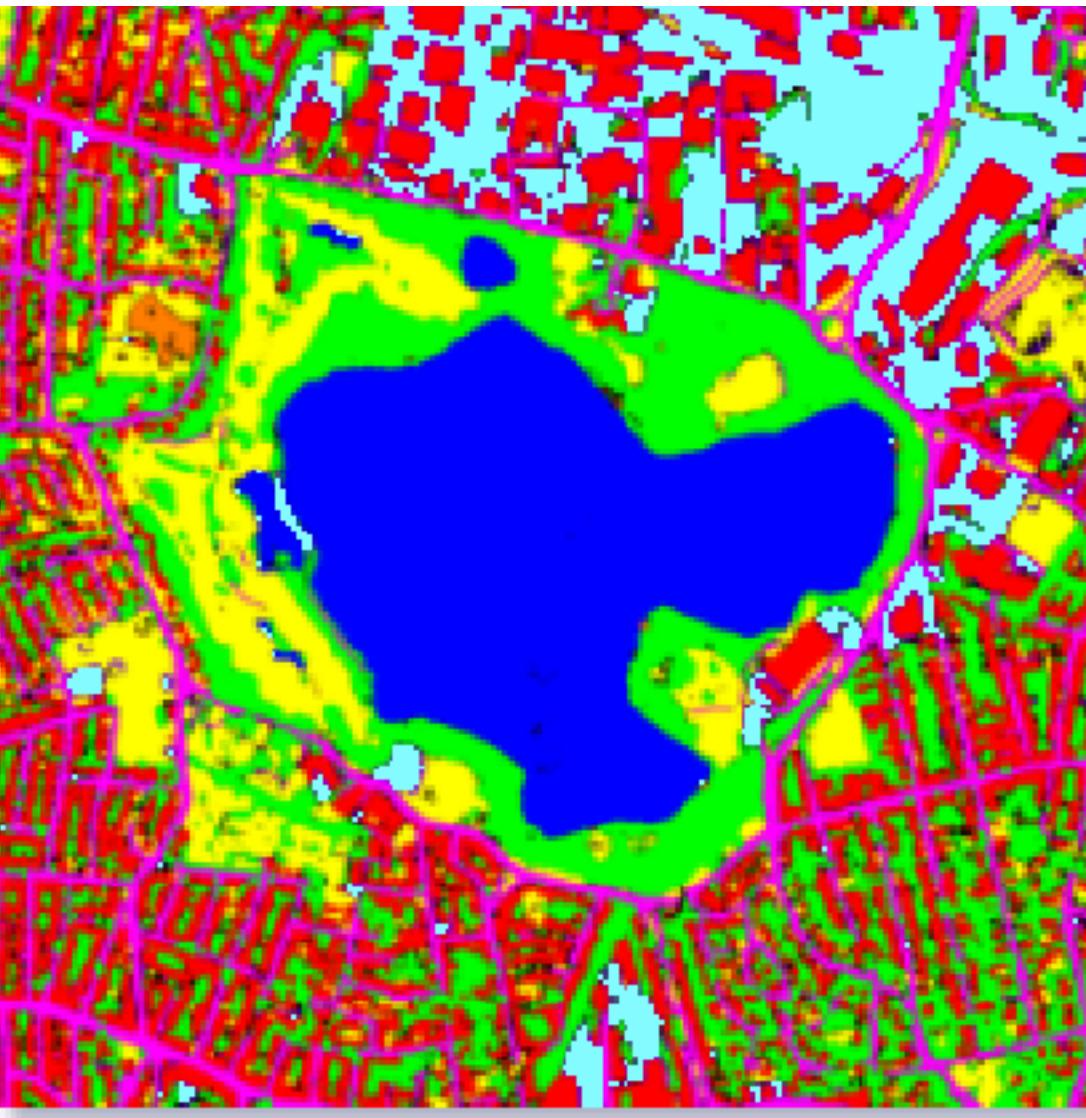
Необходимо классифицировать все пиксели изображения как принадлежащие метке класса, а также определить какому экземпляру этого класса они принадлежат.



# ОБЛАСТИ ПРИМЕНЕНИЯ

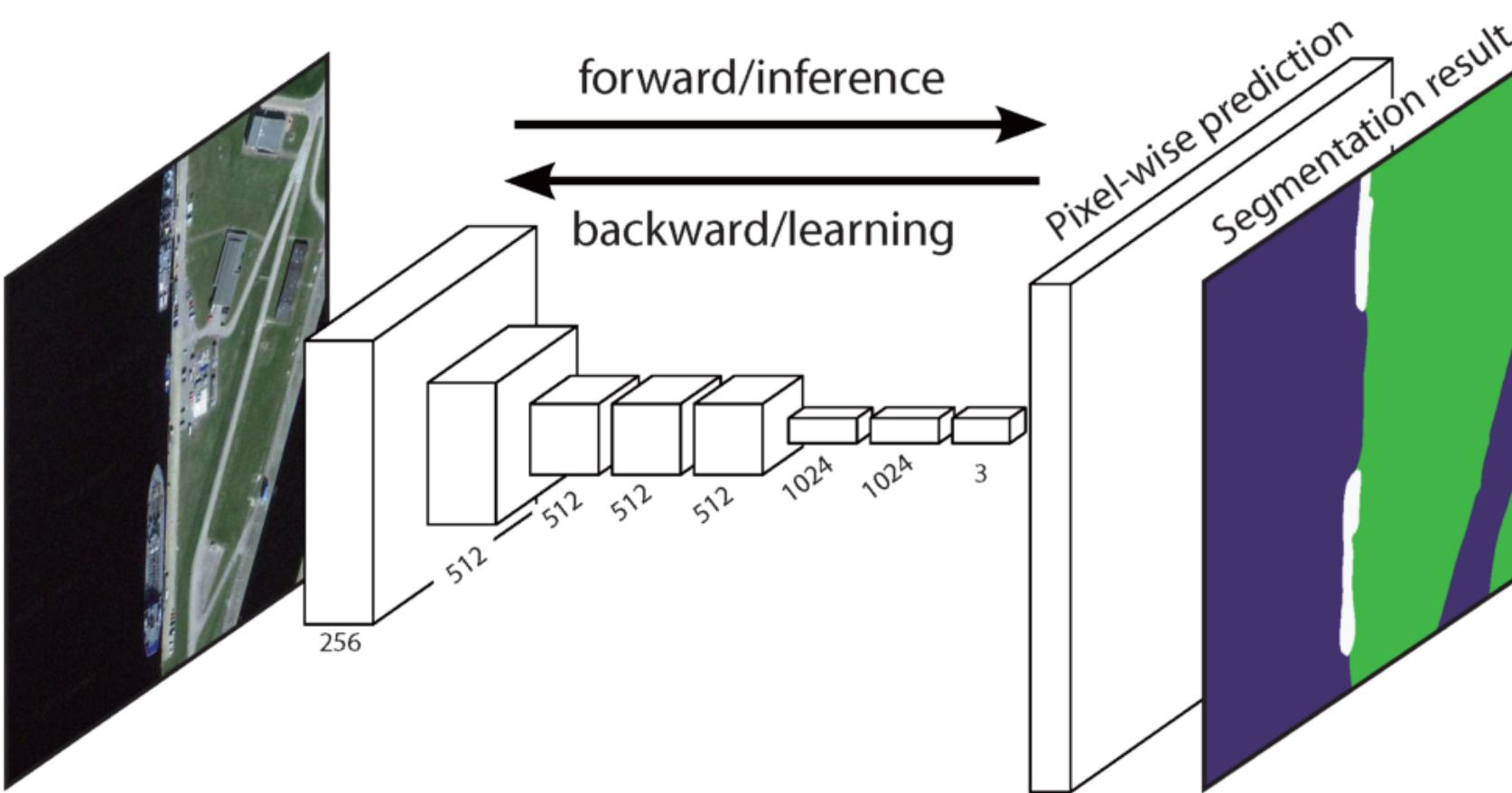


# ОБЛАСТИ ПРИМЕНЕНИЯ

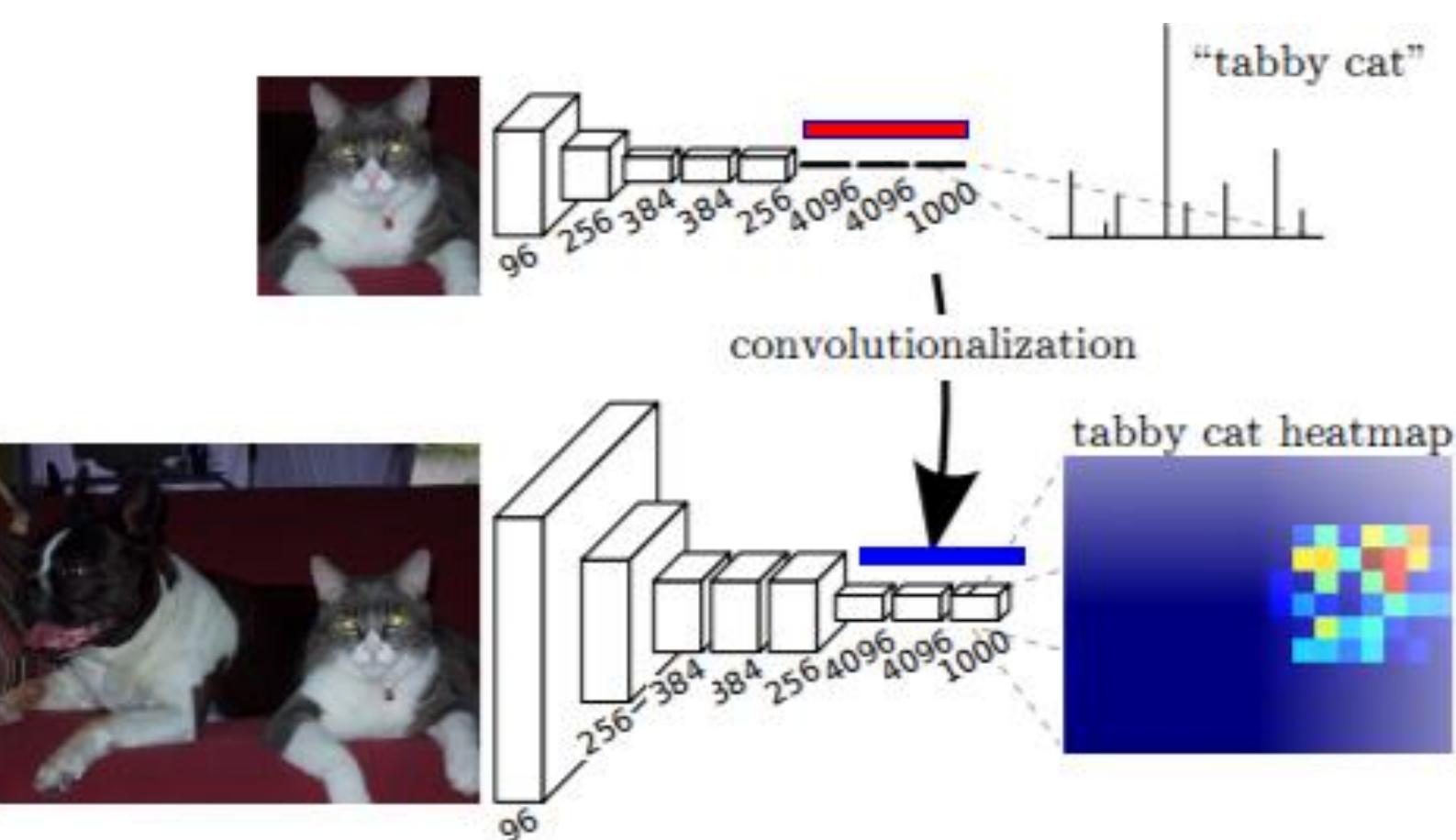


# FCN

## fully convolutional network

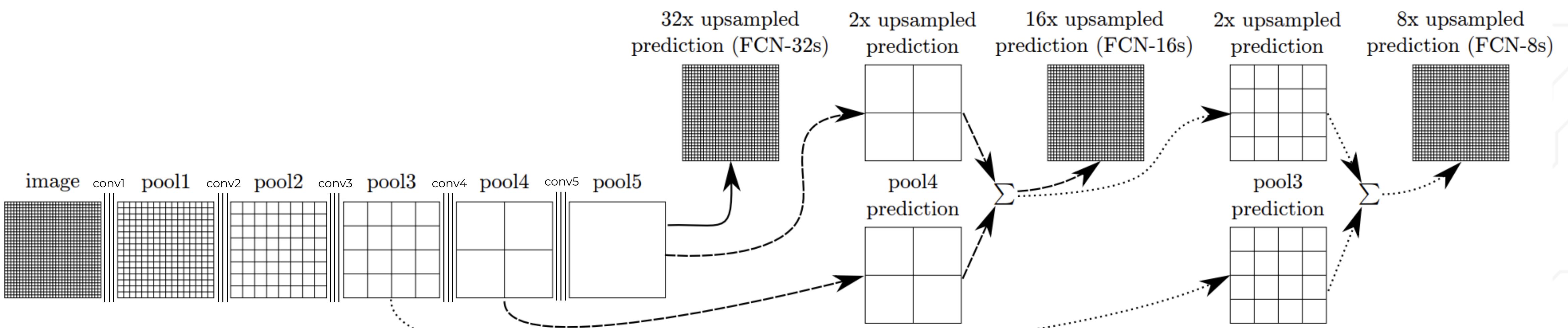


- Используют только локально связанные слои, такие как свертка, pooling и upsampling.
- Может работать с изображениями переменного размера.
- Состоит из **downsampling**, используемого для извлечения и интерпретации контекста, и **upsampling** – вместо полносвязных уровней добавляем свёртки с большим receptive field.
- Используем **skip connections** для восстановления мелкозернистой пространственной информации, потерянной в **downsampling**.

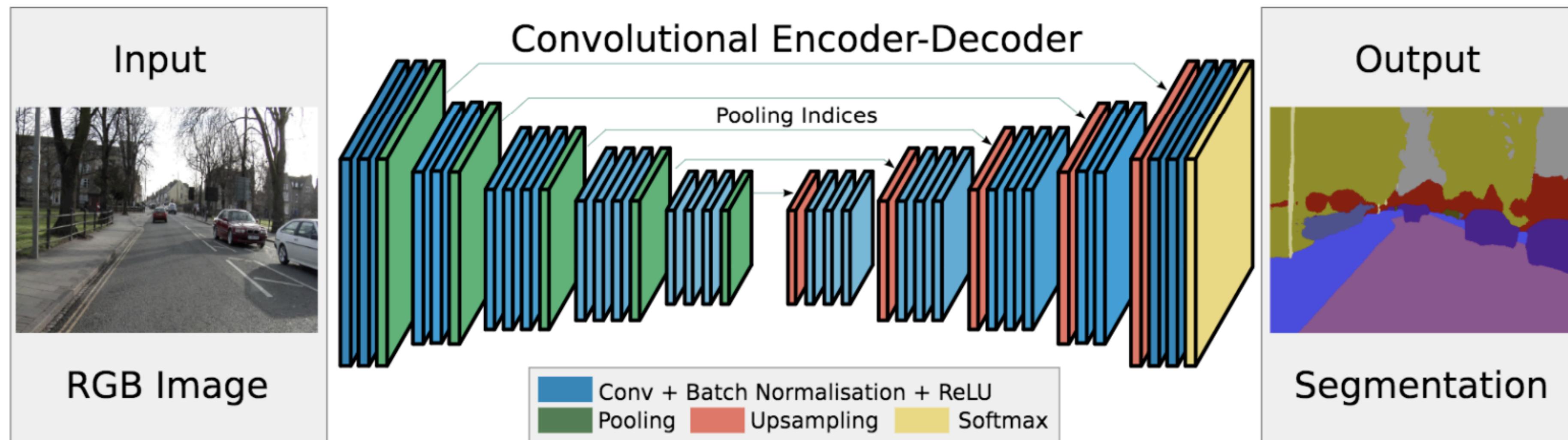


# FCN

## fully convolutional network



# SEGNET



## ENCODER

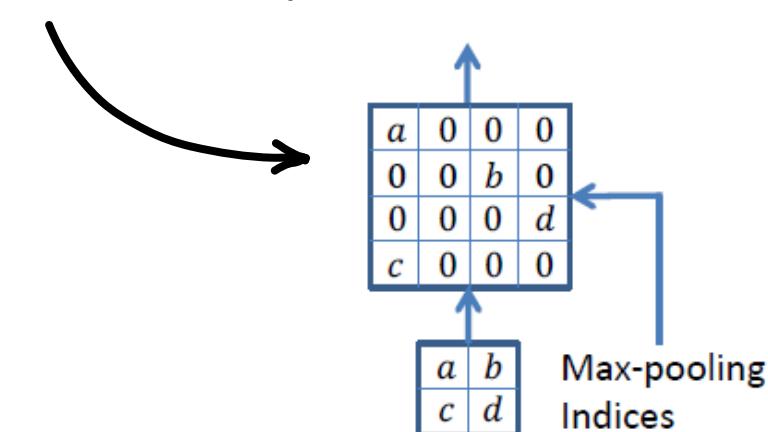
- Свертки + max pooling
  - 13 сверточных слоев из VGG-16.
  - При выполнении max pooling  $2 \times 2$  сохраняются соответствующие индексы.

## DECODER

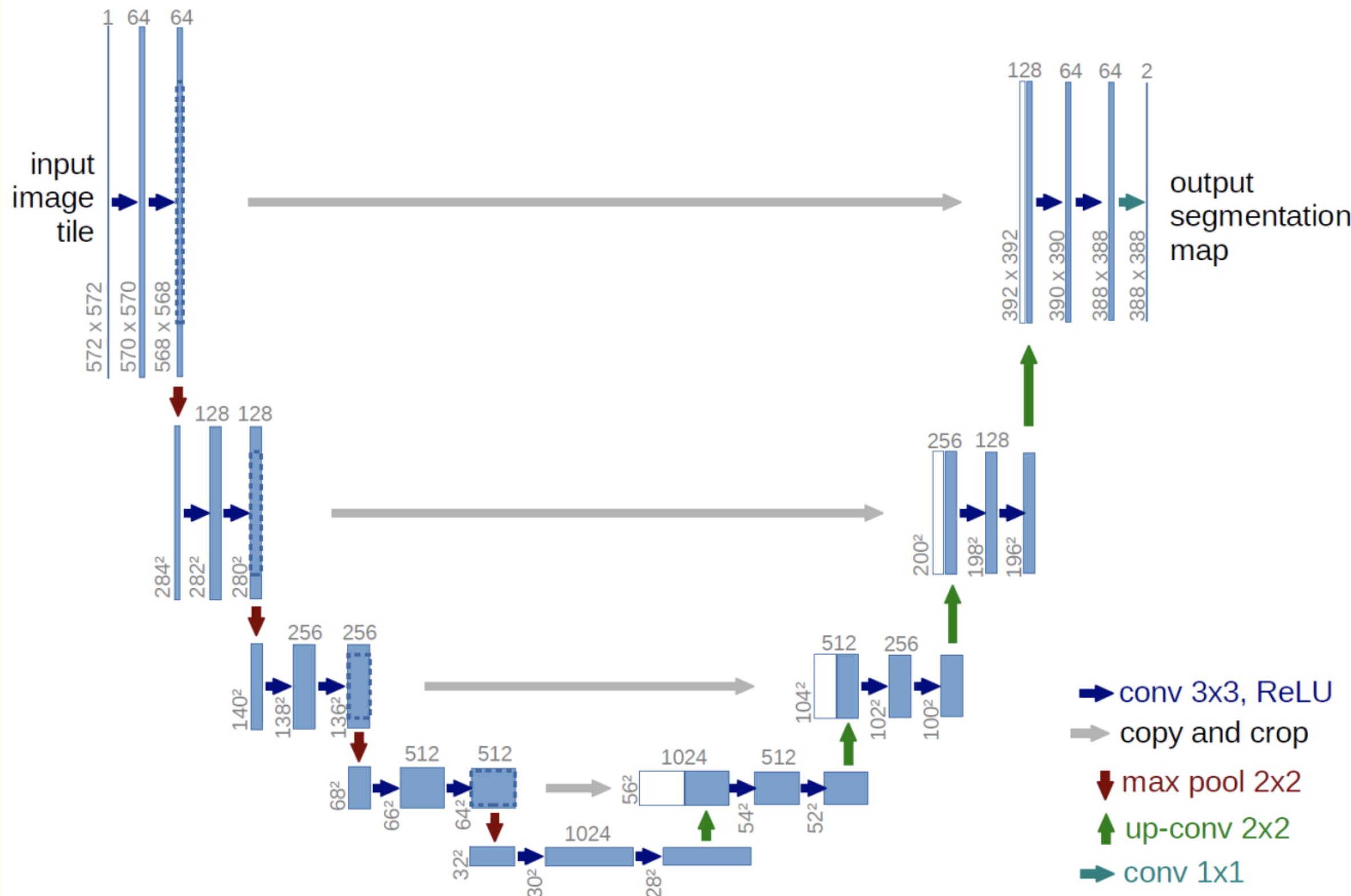
- Upsampling + свертки
  - Вызываются max pooling индексы соответствующего уровня енкодера

## SOFTMAX

Последний слой преобразует каждый пиксель выходной матрицы в целое число, показывающее класс данного пикселя.



# U-NET



Сначала использовался как сеть для сегментации биологических снимков.

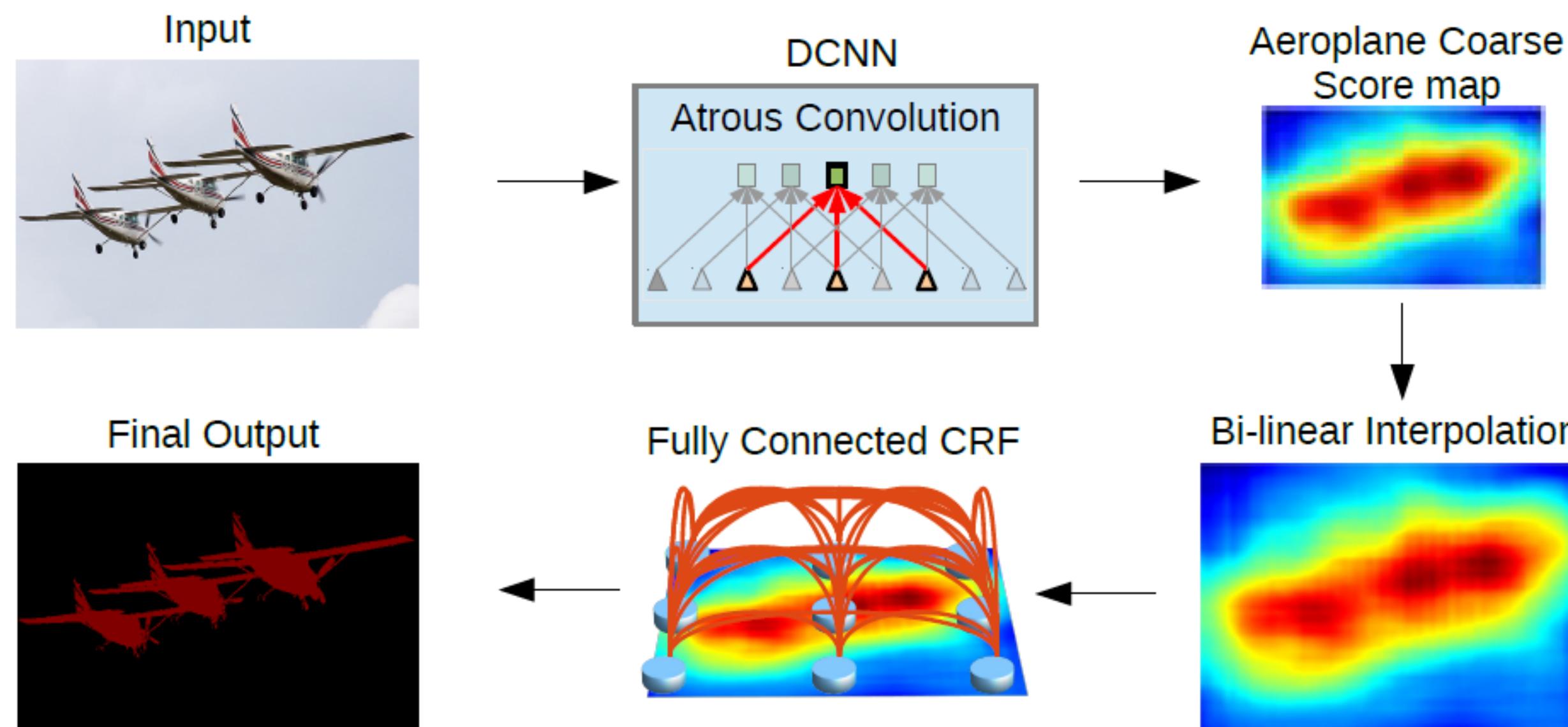
Состоит из двух частей:

- **сужающейся** (encoder) - типичная архитектура сверточной классификационной сети. Состоит из повторяющихся применений двух сверток 3x3, за которыми следуют активация (ReLU) и операция макс-пулинга 2x2 с шагом 2. На каждом шаге повышаем количество каналов вдвое.
- **расширяющейся** (decoder) - состоит из шагов обратной свертки(deconvolution) 2x2, уменьшающей количество каналов, затем конкатенация с соответствующим образом обрезанной картой признаков от соответствующей части сужающейся части, две свертки 3x3 и активация(ReLU).

На последнем уровне свертка 1x1 сопоставляет каждому 64-компонентному вектору признаков класса.

В общей сложности сеть имеет **23 сверточных слоя**.

# DEEPLAB



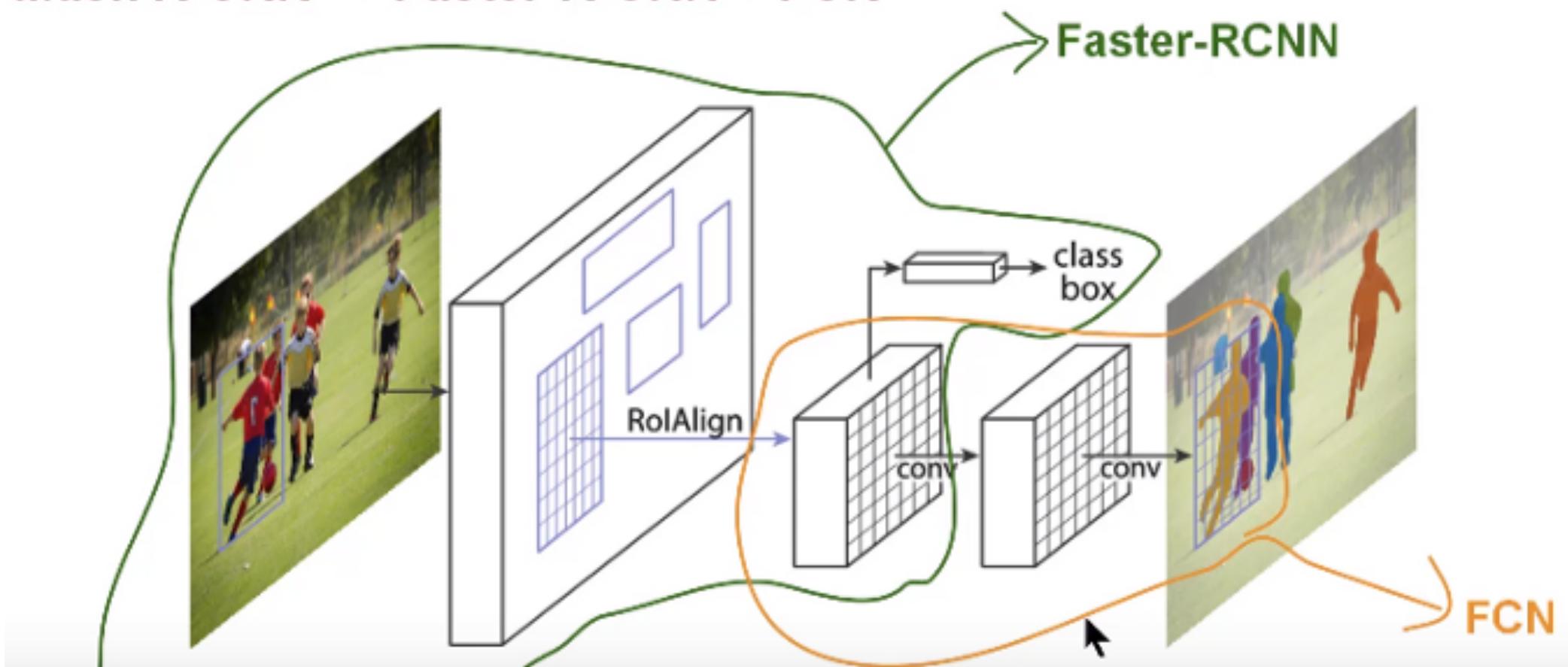
ATROUS CONVOLUTION



FULLY CONNECTED CONDITIONAL RANDOM FIELD

# MASK R-CNN

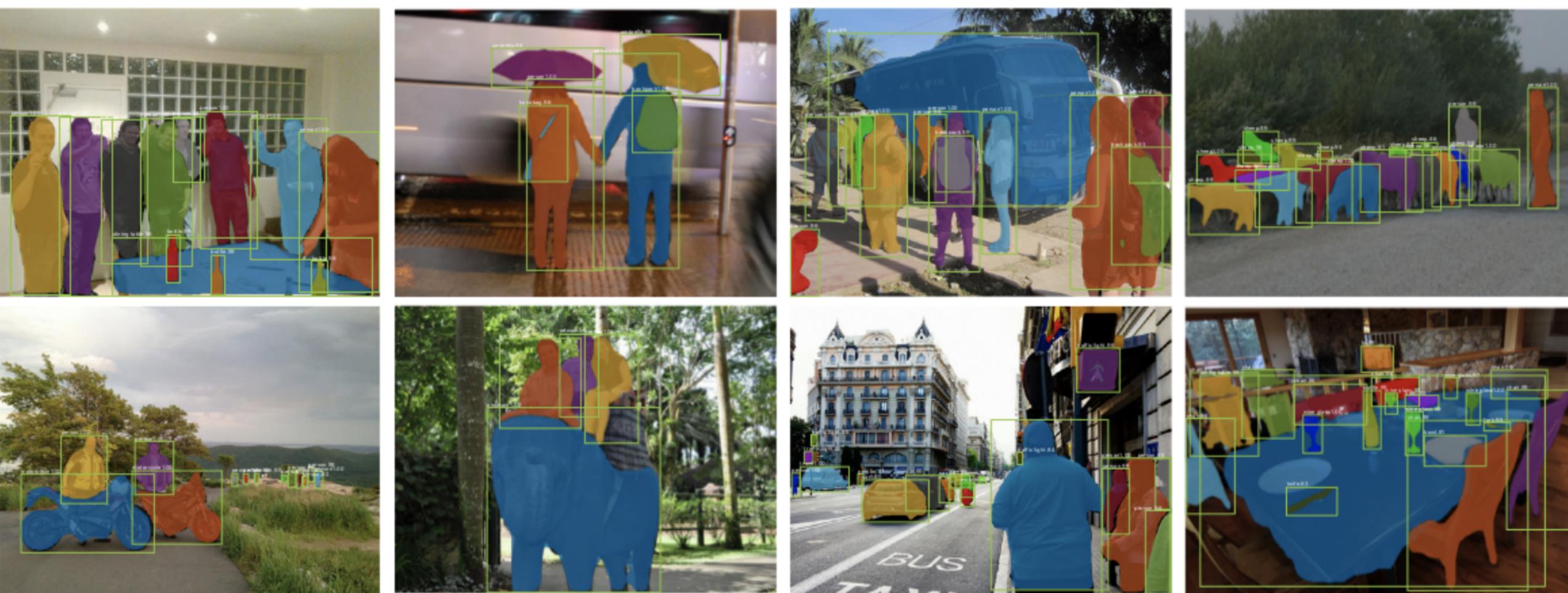
Mask R-CNN → Faster R-CNN + FCN



Faster R-CNN имеет два выхода: метка класса и сдвиг bounding box. А теперь добавляем третий выход, который попиксельно считает маску объекта.



Добавляем ошибку за маску  
(средняя кросс-энтропия по пикселям)



# СПИСОК ИСТОЧНИКОВ

- GRAPH-BASED SEGMENTATION
- SEMANTIC VS INSTANCE VS PANOPTIC SEGMENTATION
- EVOLUTION OF IMAGE SEGMENTATION
- SEMANTIC SEGMENTATION
- PANOPTIC SEGMENTATION
- FULLY CONVOLUTIONAL NETWORKS (FCN)
- SEGNET
- DEEPLAB
- MASK R-CNN