

# Deep learning CV part2

Detection, localization, masking, death

# CV tasks

## Classification



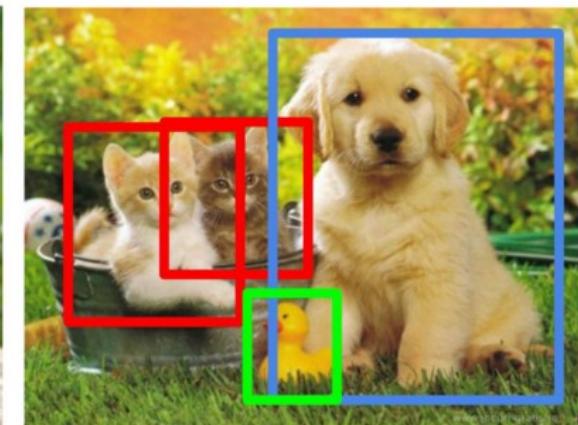
CAT

## Classification + Localization



CAT

## Object Detection



CAT, DOG, DUCK

## Instance Segmentation

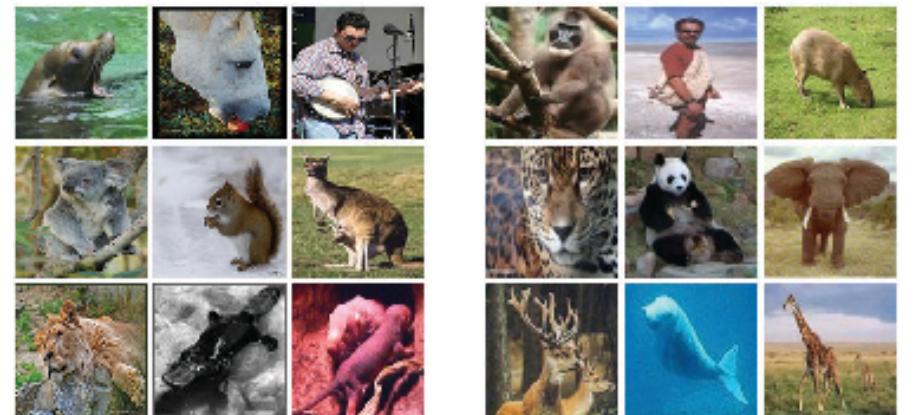
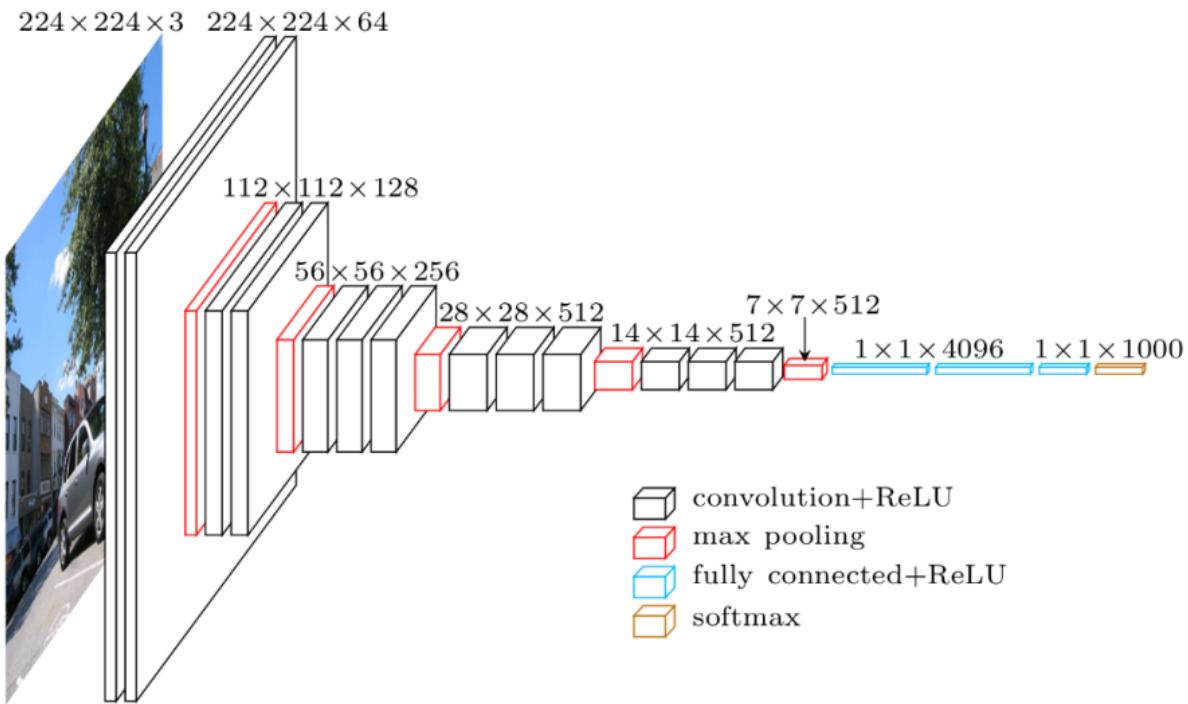


CAT, DOG, DUCK

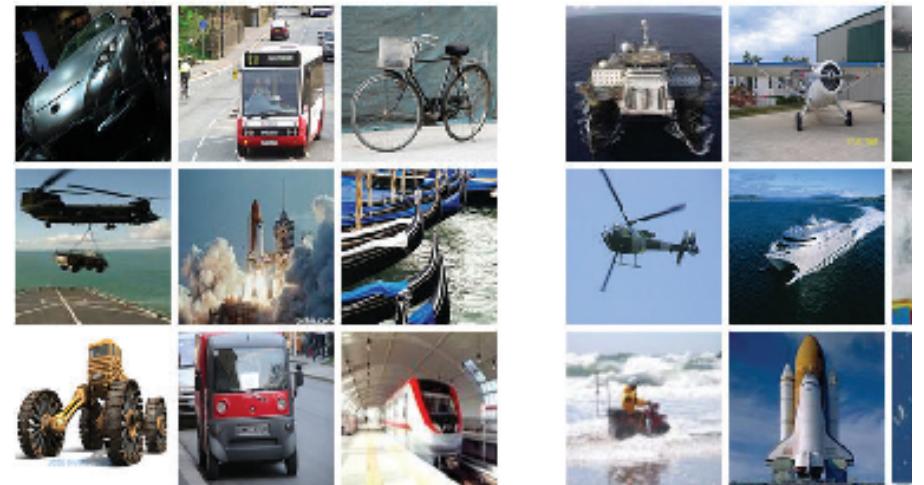
Single object

Multiple objects

# Classification

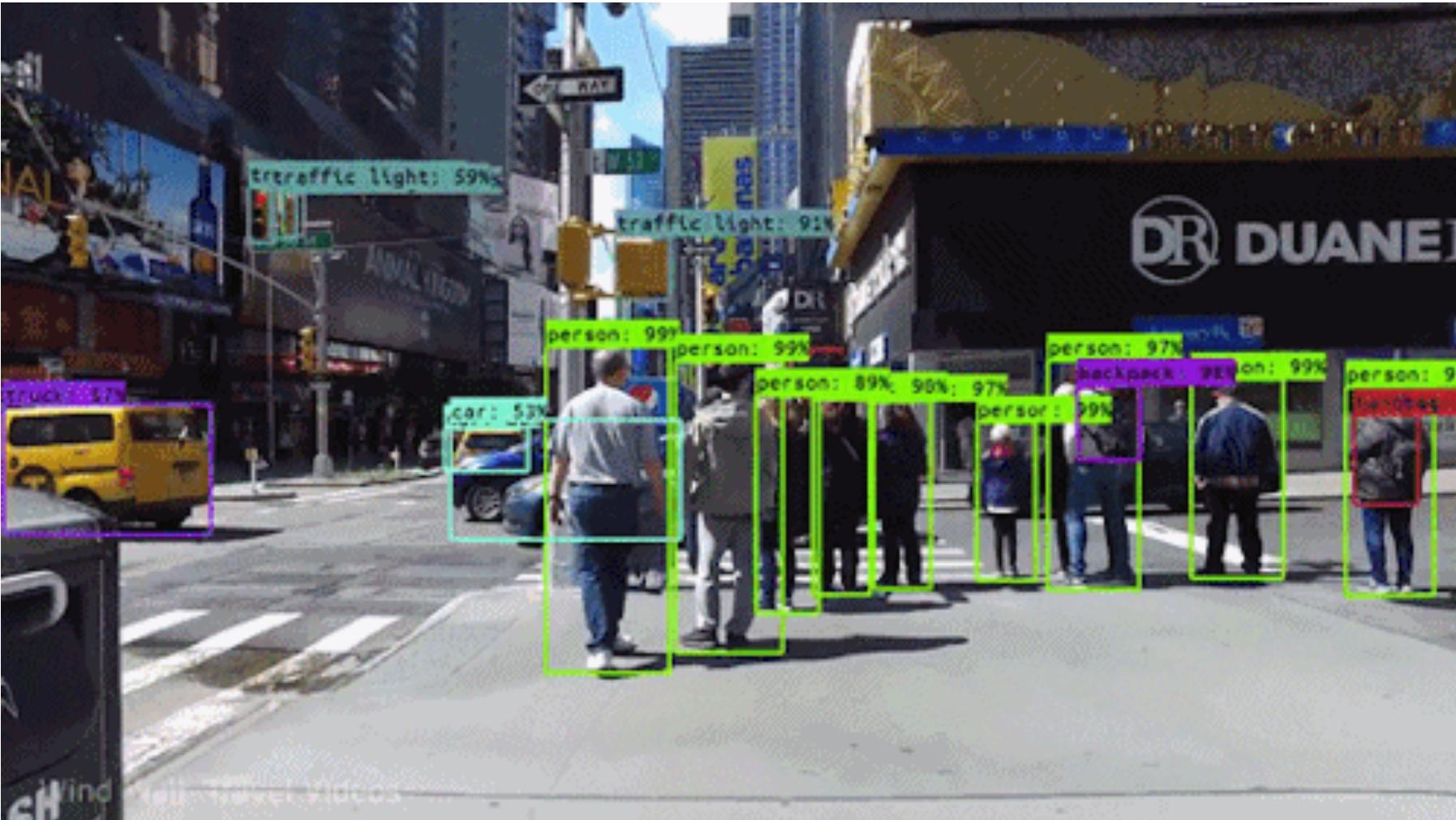


mammal → placental —



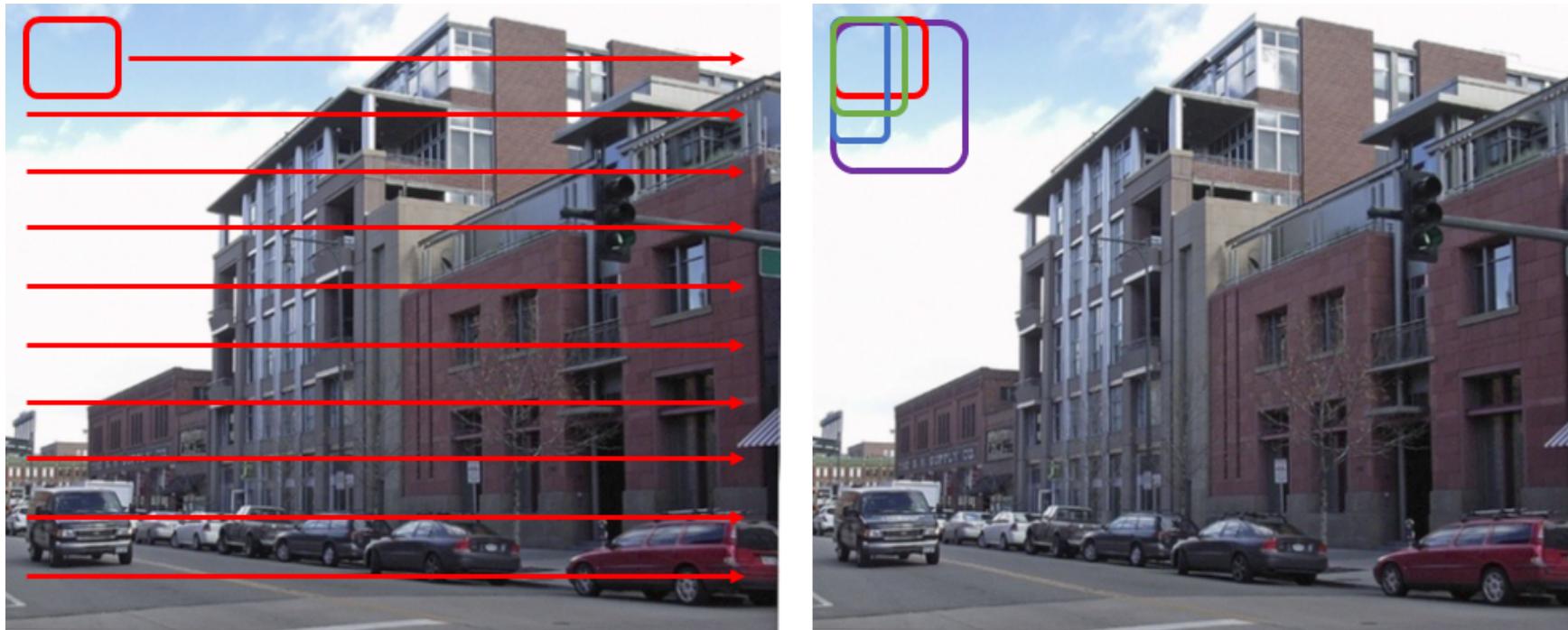
— vehicle → craft —

# Detection



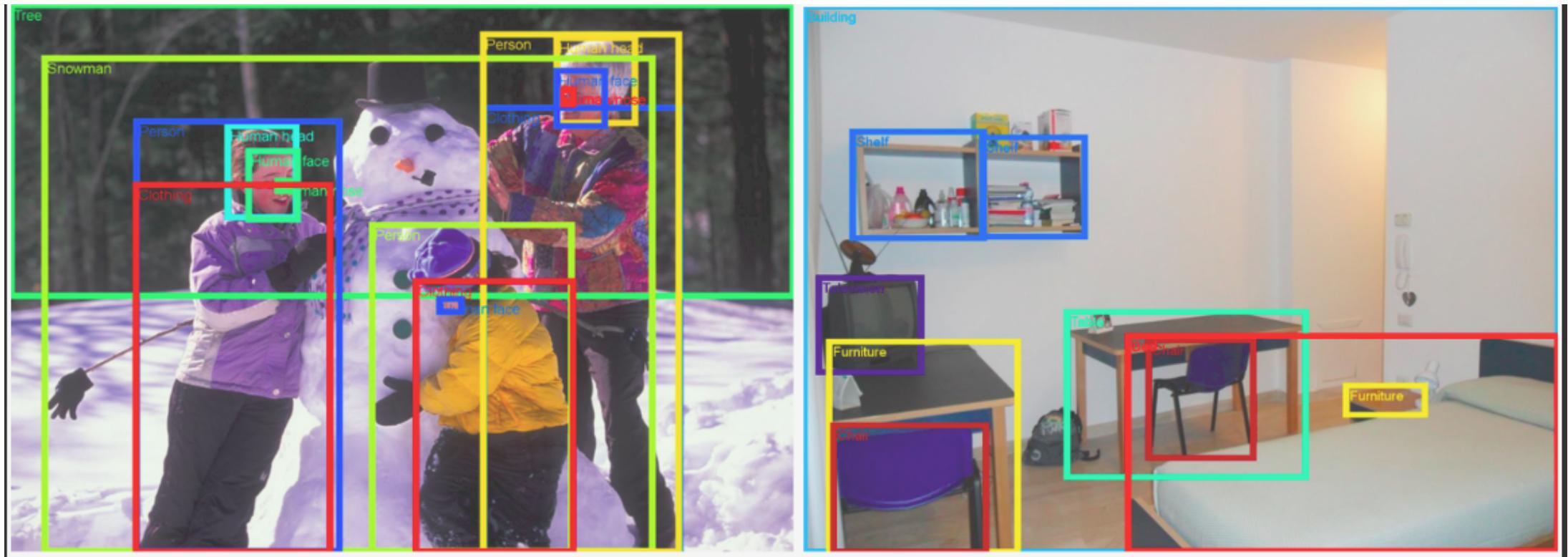
# Naive method

- Запускать CNN для классификации на небольших участках
- Запускать несколько раз на разных размерах окна
- Собирать результаты вместе
- Минус: невероятно долго

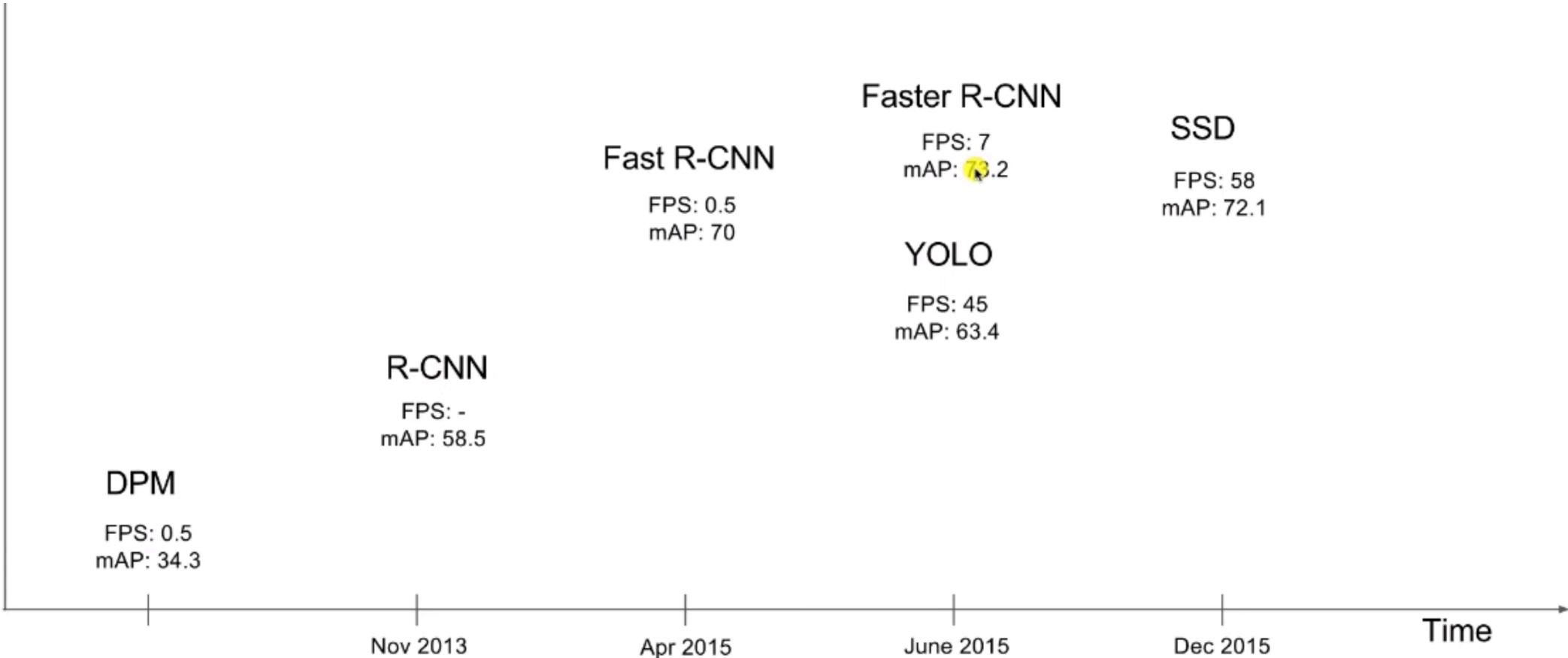


# Datasets for detection

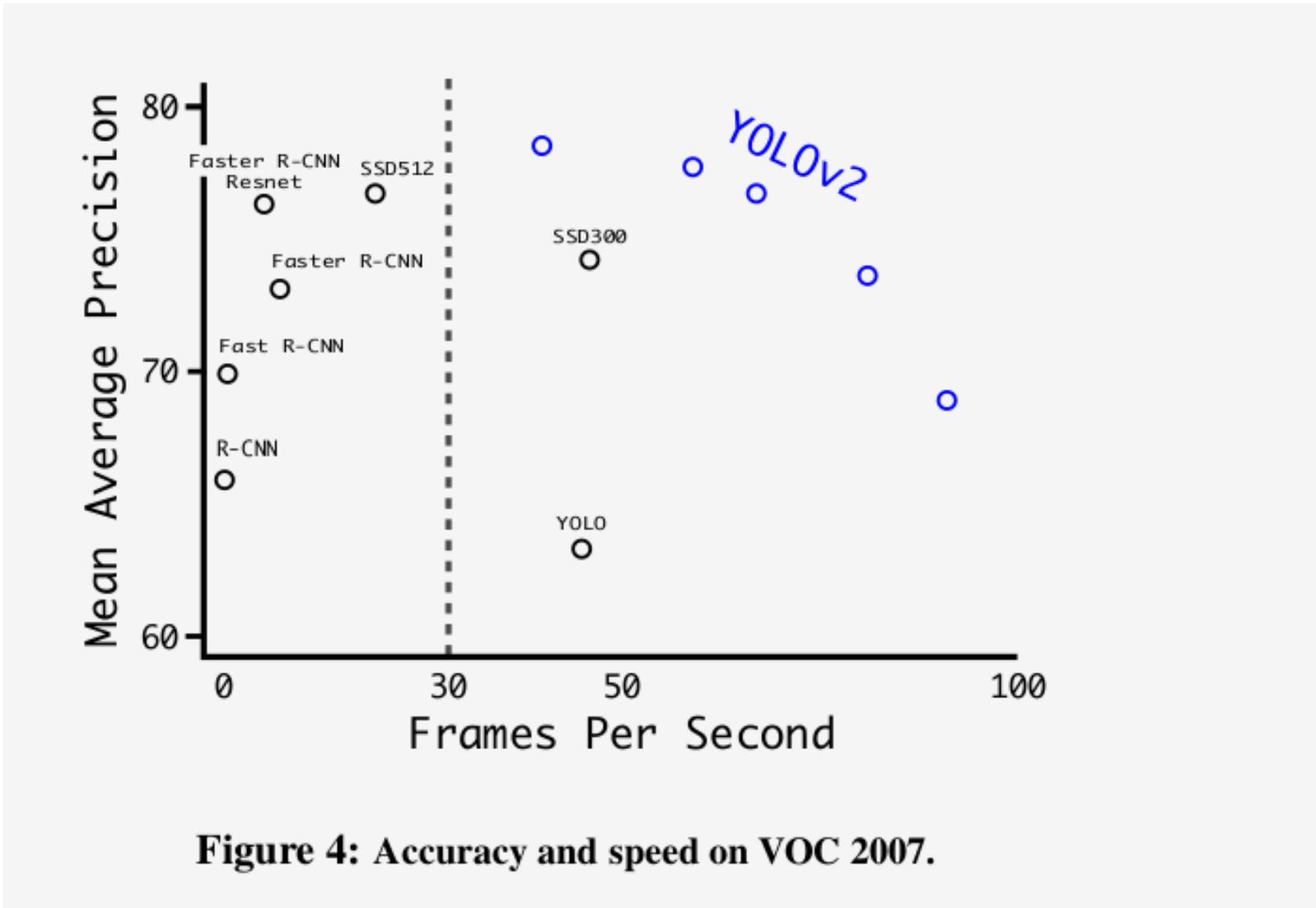
- Open Images by Google - 9,011,219 images



# Architectures

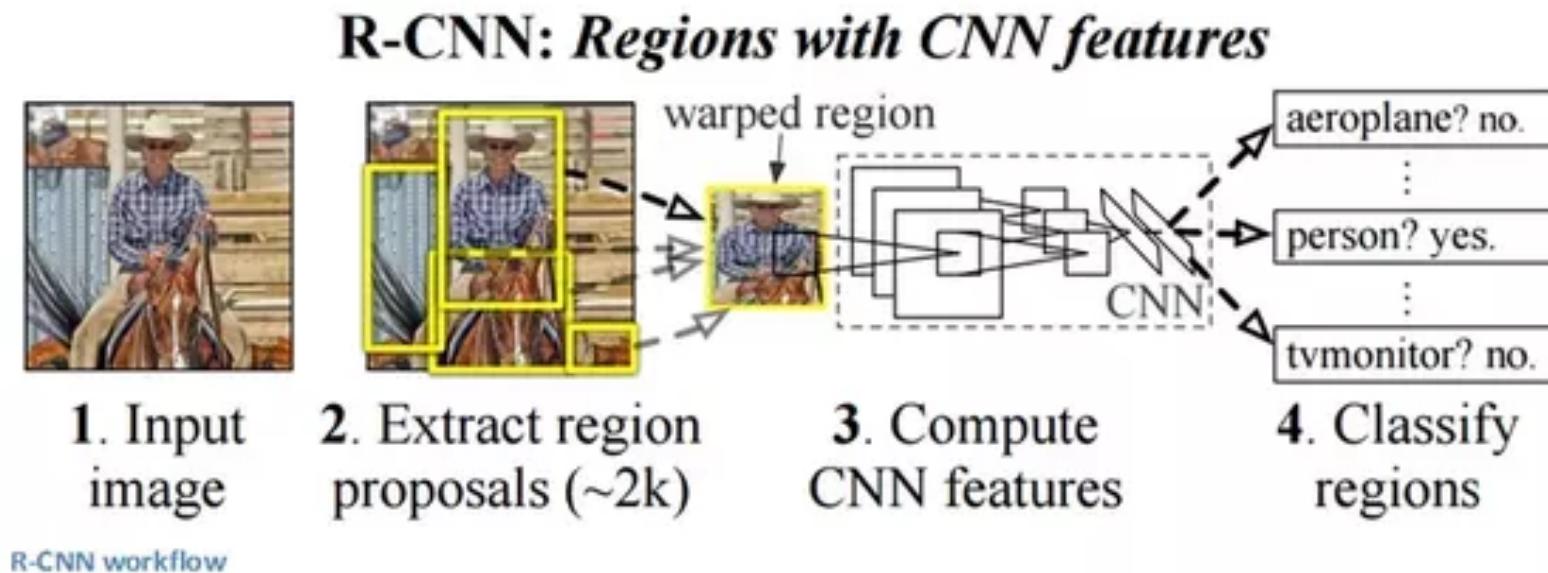


# Architecture options for this task

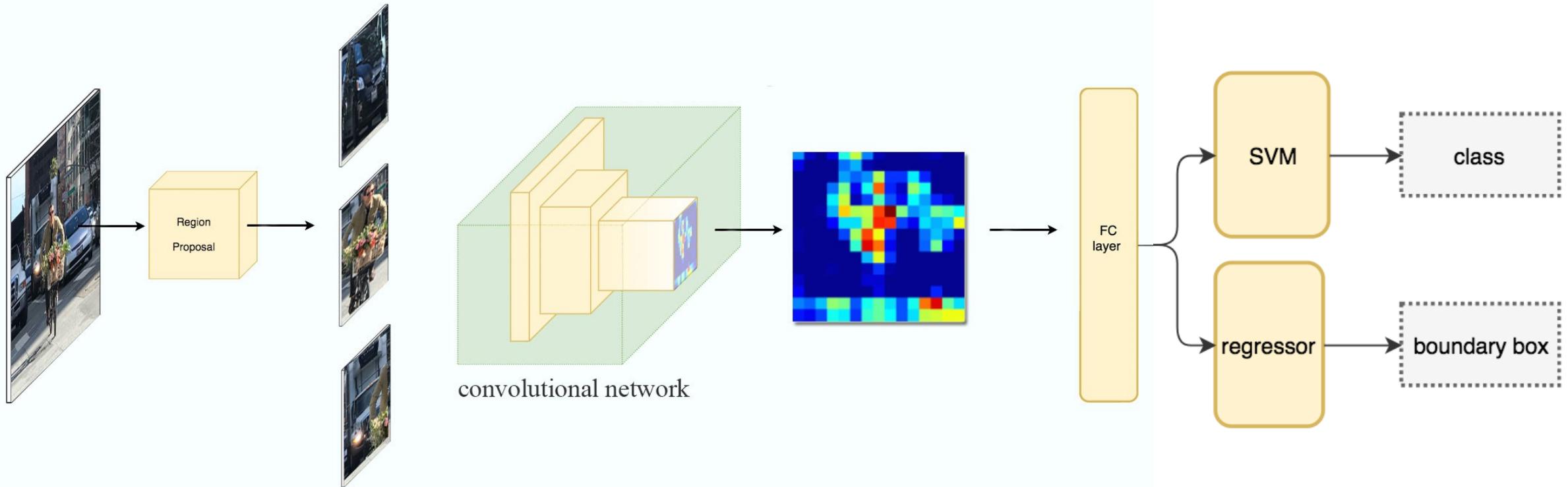


# R-CNN (Region Based Convolution Neural Network)

- Использует **Selective Search** для выделения около 2000 регионов
- Каждый регион прогоняем через CNN (AlexNet, , 4096 фич)
- Подаем фичи на вход SVM:
  - Классификация регионов
  - регрессия по координатам боксов



# R-CNN



# R-CNN: Selective Search

- Иерархическая кластеризация на основе цвета и текстуры
- На первой итерации находит много, потом объединяет

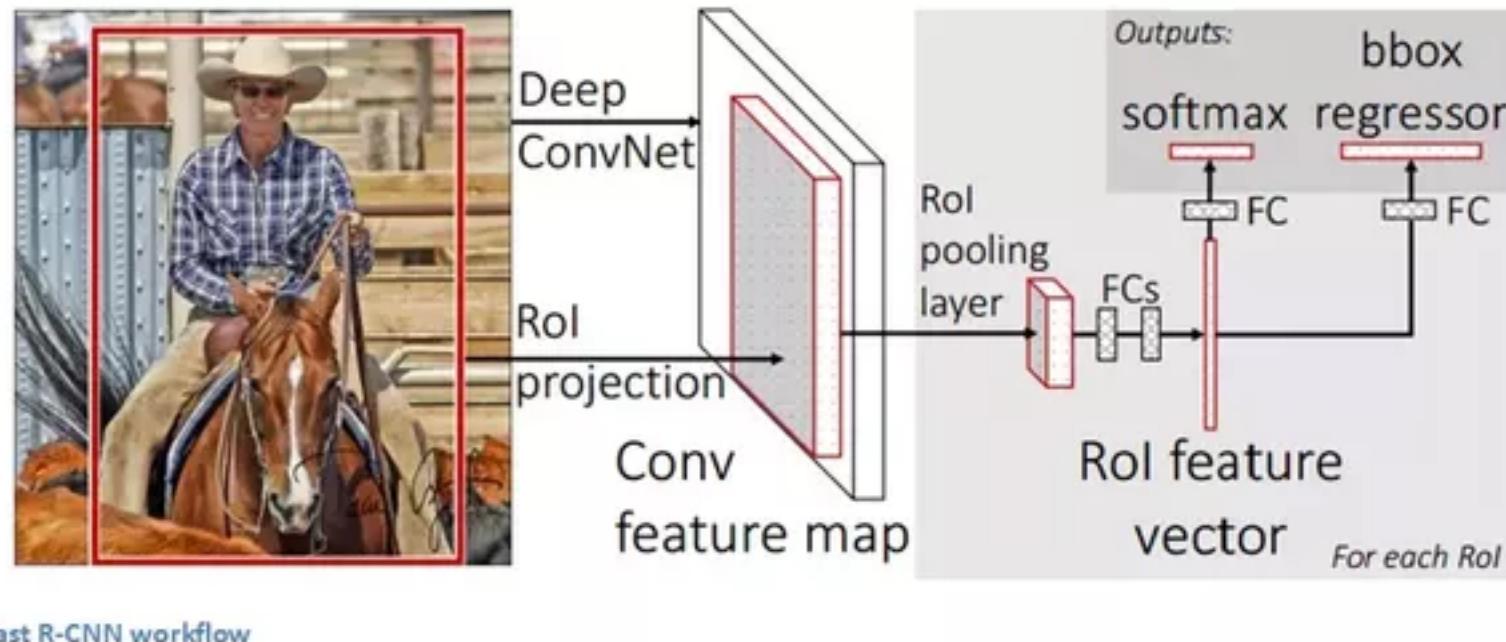


# R-CNN

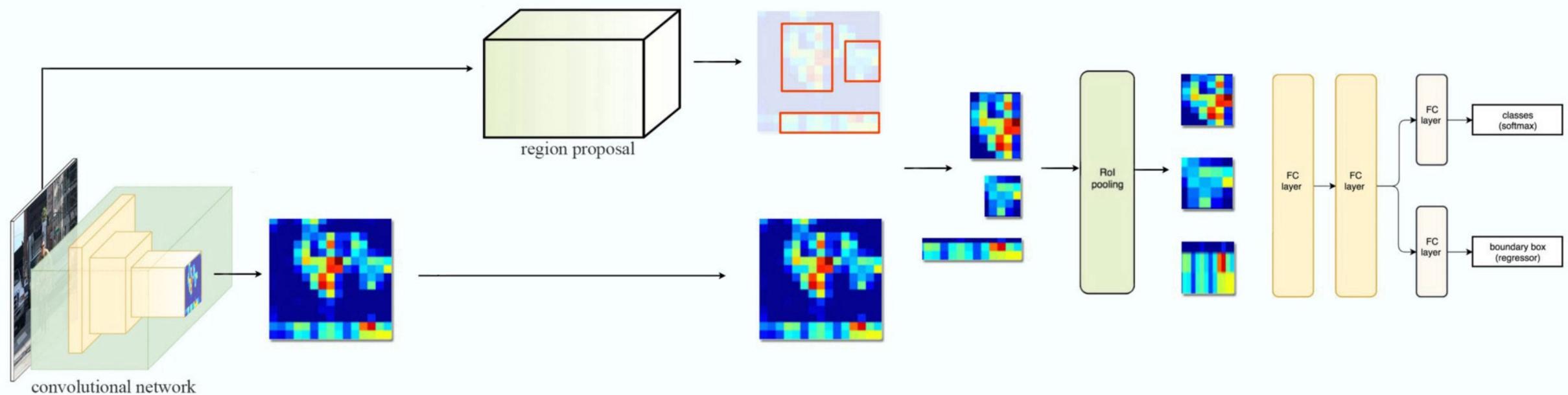
- Плюсы:
  - Достаточно точная
  - Быстрее, чем наивный подход
  - Может быть построена на основе любой архитектуры для классификации
- Минусы:
  - Состоит из трех независимых моделей (Selective Search, CNN, SVM), нельзя потренировать все сразу, ошибки накладываются
  - Тренировка очень долгая - порядка 84 часов
  - Инференс (детекшн) долгий – порядка 47 секунд на картинку если брать VGG19

# Fast R-CNN

- Также использует Selective Search
- Но прогоняет изображение через CNN только один раз целиком
- Выкинули SVM, теперь классификация происходит внутри CNN



# Fast R-CNN



# Fast R-CNN

- Плюсы:

- Работает значительно быстрее (запуск CNN 2000 было наибольшей проблемой)
- CNN и классификатор обучаются вместе

	Fast R-CNN	R-CNN
Train time (h)	<b>9.5</b>	84
- Speedup	<b>8.8x</b>	1x
Test time / image	<b>0.32s</b>	47.0s
Test speedup	<b>146x</b>	1x
mAP	<b>66.9%</b>	66.0%

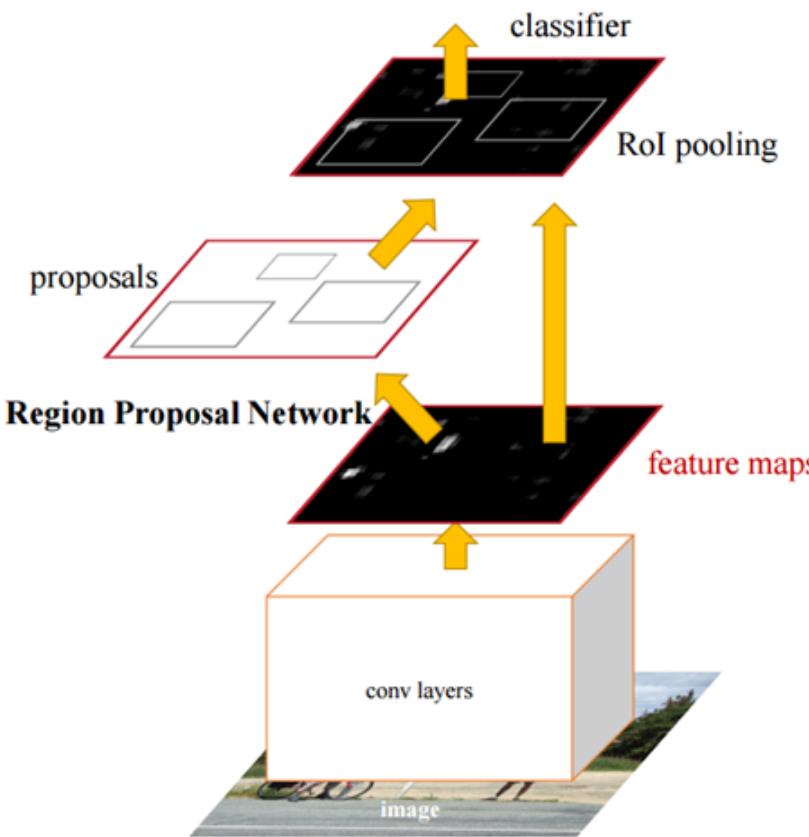
- Минусы:

- Все еще зависит от Selective Search

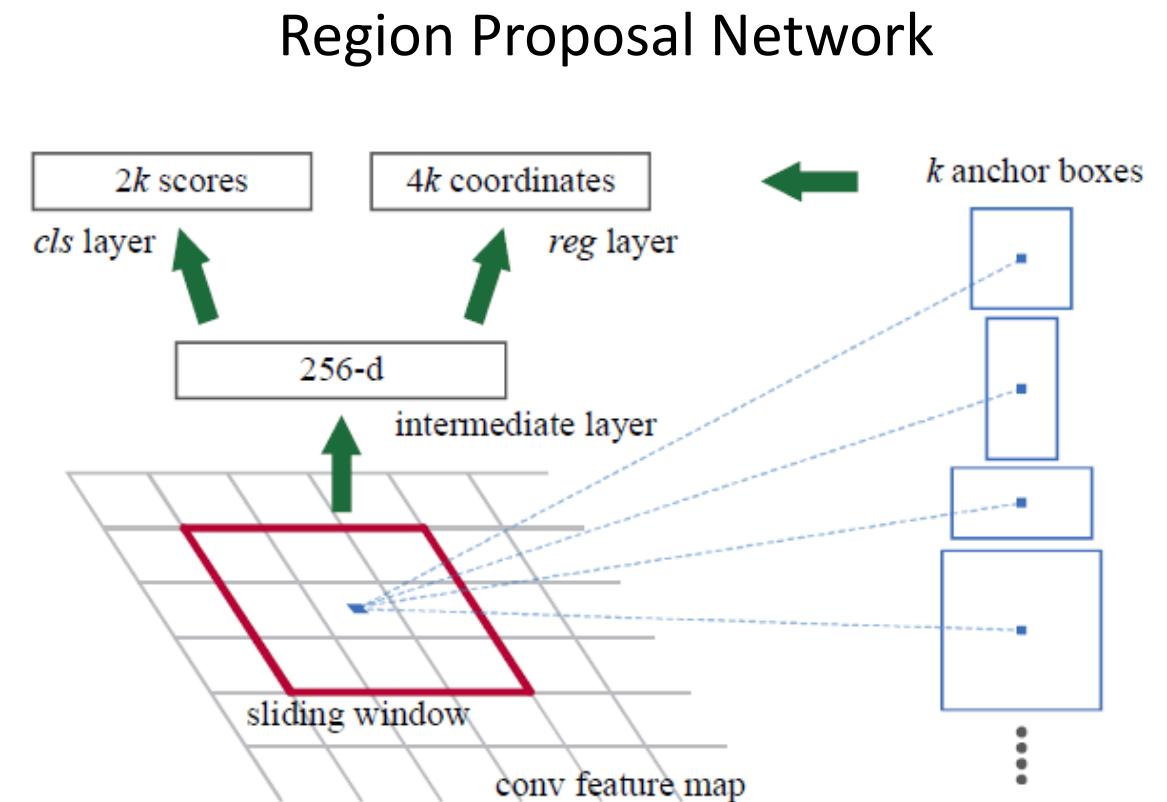
# Faster R-CNN

- Избавляет нас от Selective Search
- Изображение прогоняется через CNN 1 раз
- Содержит внутри еще одну подсеть – RPN (region proposal network)
- Она предсказывает bounding box по feature map

# Faster R-CNN

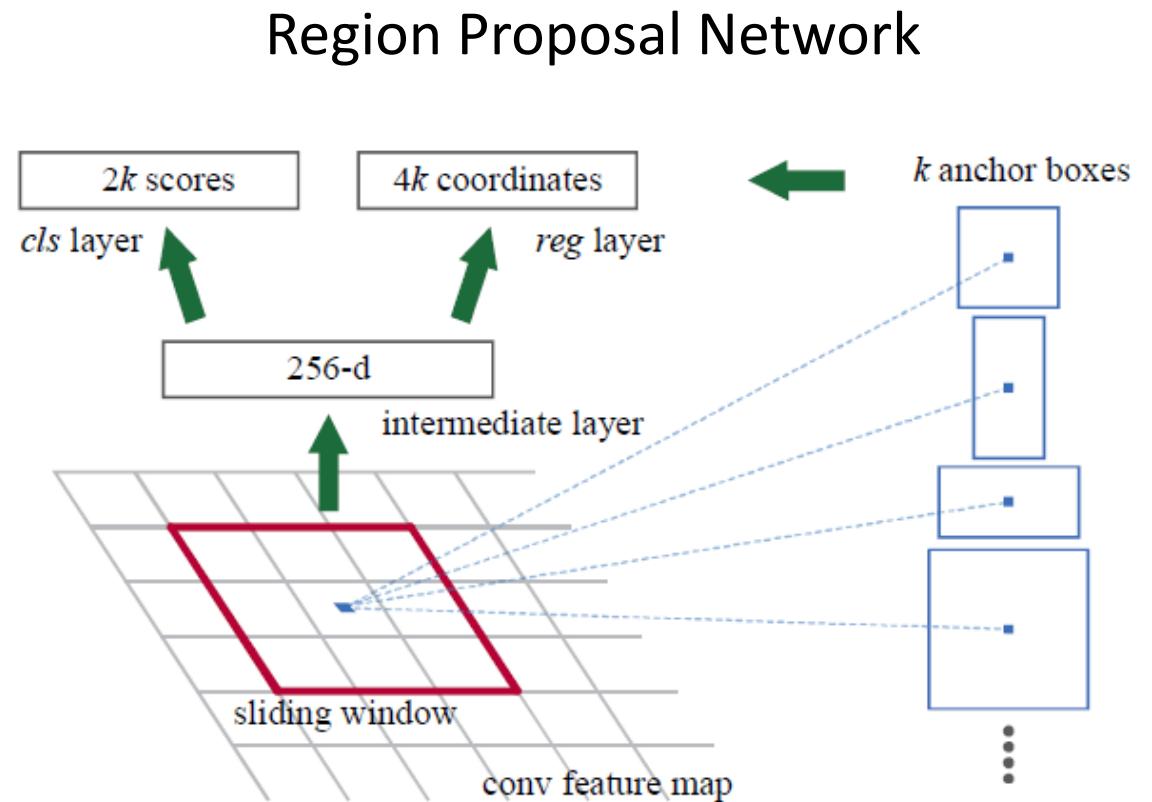


Faster R-CNN workflow

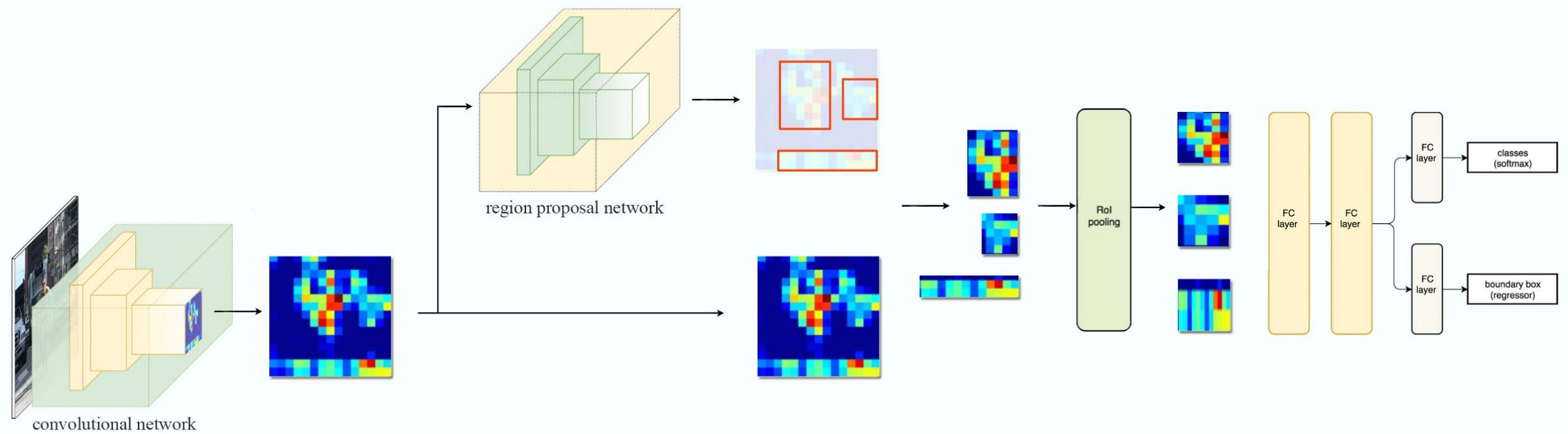


# Faster R-CNN: RPN

- Then a **sliding window** is used in RPN for each location over the feature map.
- For each location, **k (k=9) anchor boxes** are used (**3 scales of 128, 256 and 512, and 3 aspect ratios of 1:1, 1:2, 2:1**) for generating region proposals.

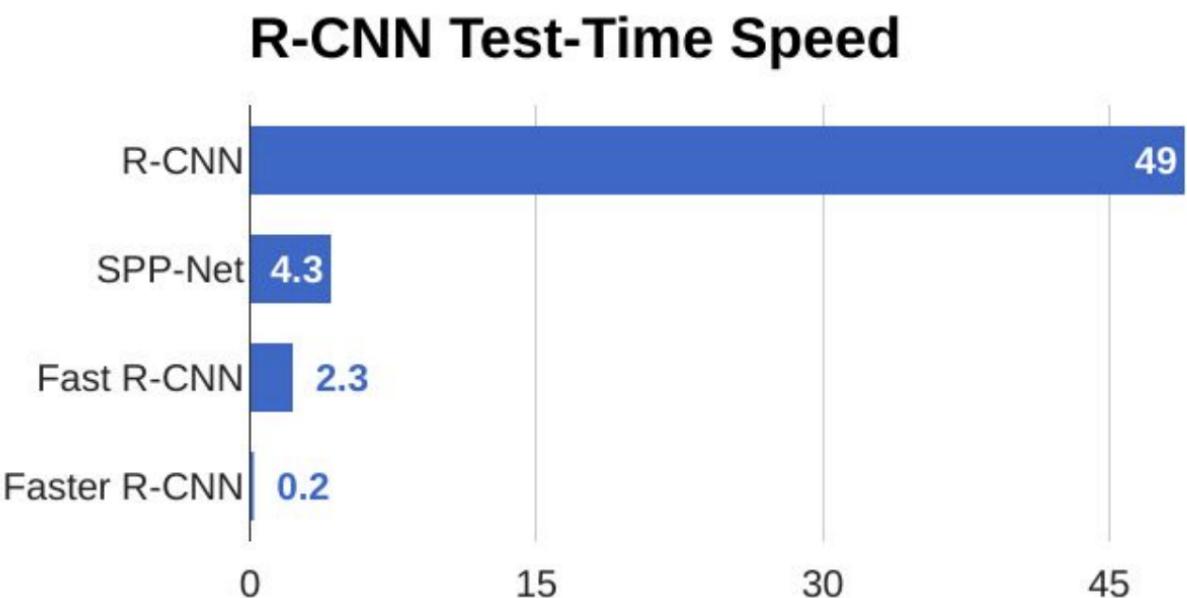


# Faster R-CNN



# Faster R-CNN

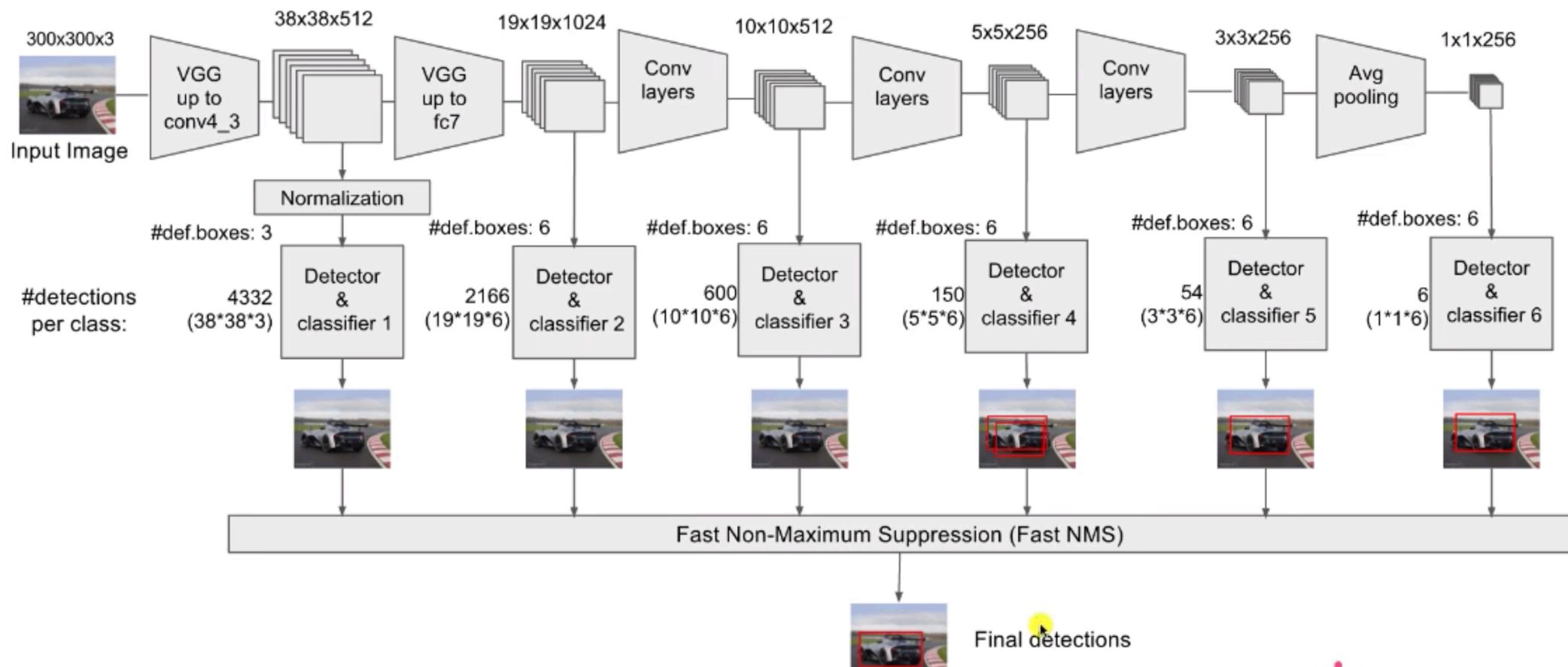
- Самая поздняя, самая быстрая и точная архитектура
- Все компоненты внутри одной сети и обучаются вместе
- Минусы:
  - Работает 7 кадров в секунду, все еще недостаточно для Real time



# Single shot detectors

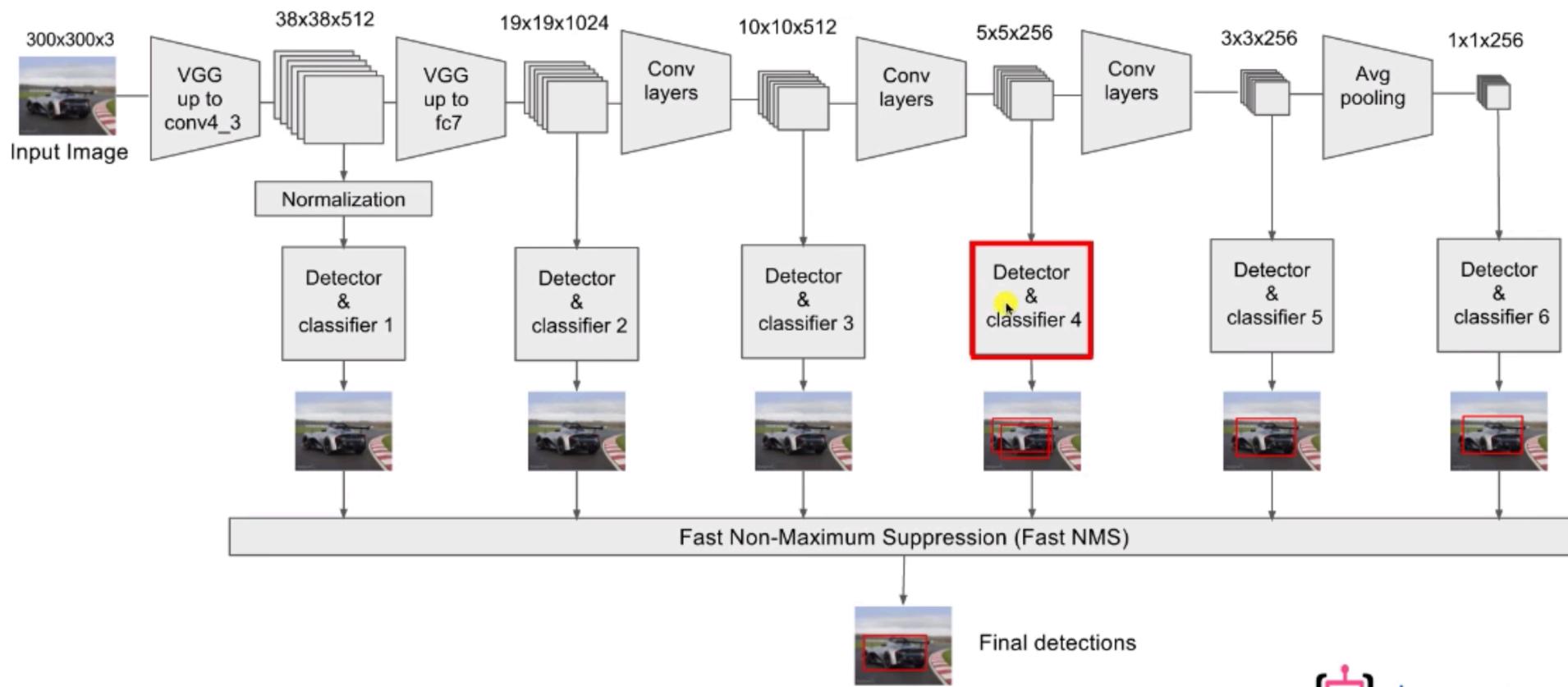
# SSD

## Архитектура SSD 300



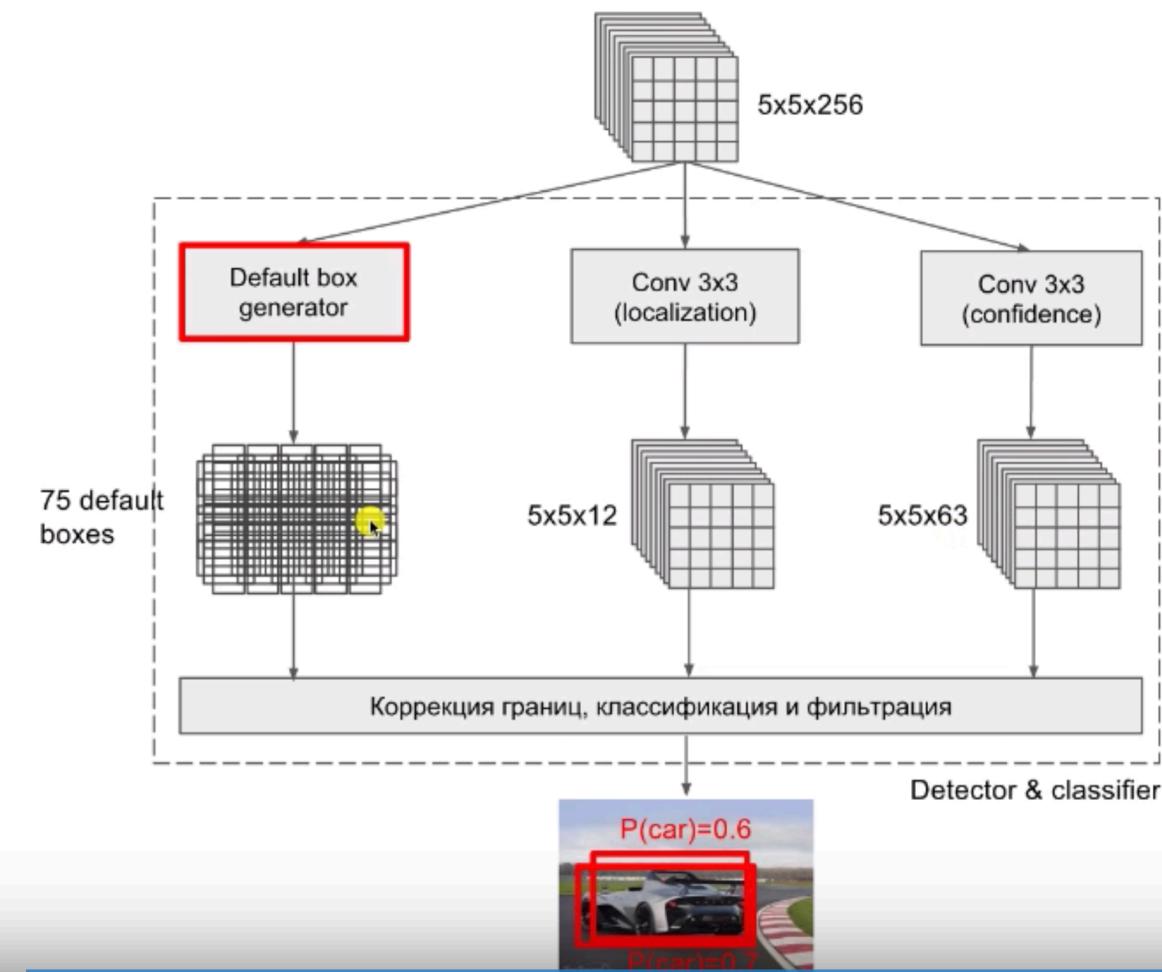
# SSD

## Архитектура SSD 300



# SSD

## Архитектура SSD 300. Detector & classifier



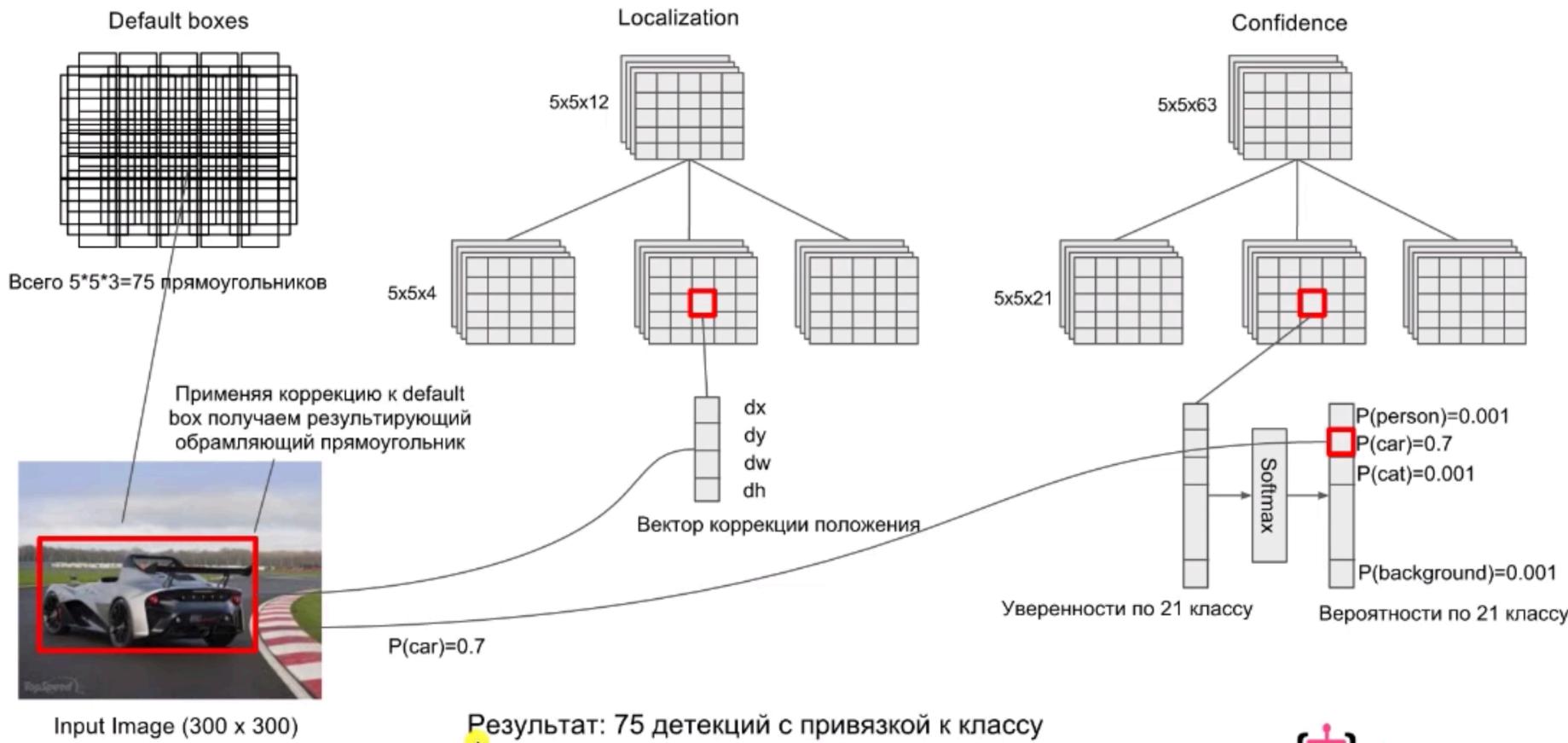
Пусть заданы следующие параметры:

- Размер исходного изображения ( $300 \times 300$ )
- Размерность feature maps ( $5 \times 5 \times 256$ )
- #default boxes = 3



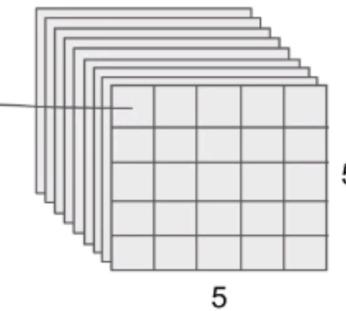
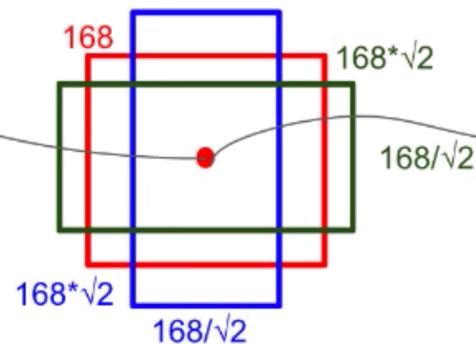
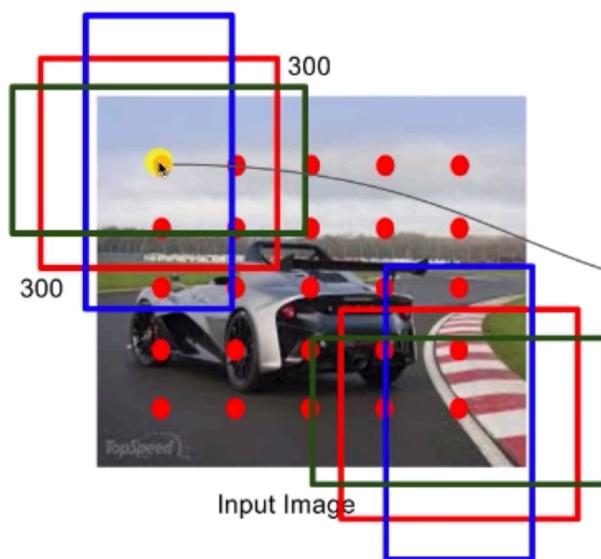
# SSD

## Коррекция границ, классификация и фильтрация (1)



# SSD

## Генерация default boxes



Пусть заданы следующие параметры:

- Размер исходного изображения ( $300 \times 300$ )
- Пространственная размерность Feature maps ( $5 \times 5$ )
- #default boxes = 3, (на одну точку в feature maps приходится 3 прямоугольника)
- min\_size=168
- aspect\_ratio=2

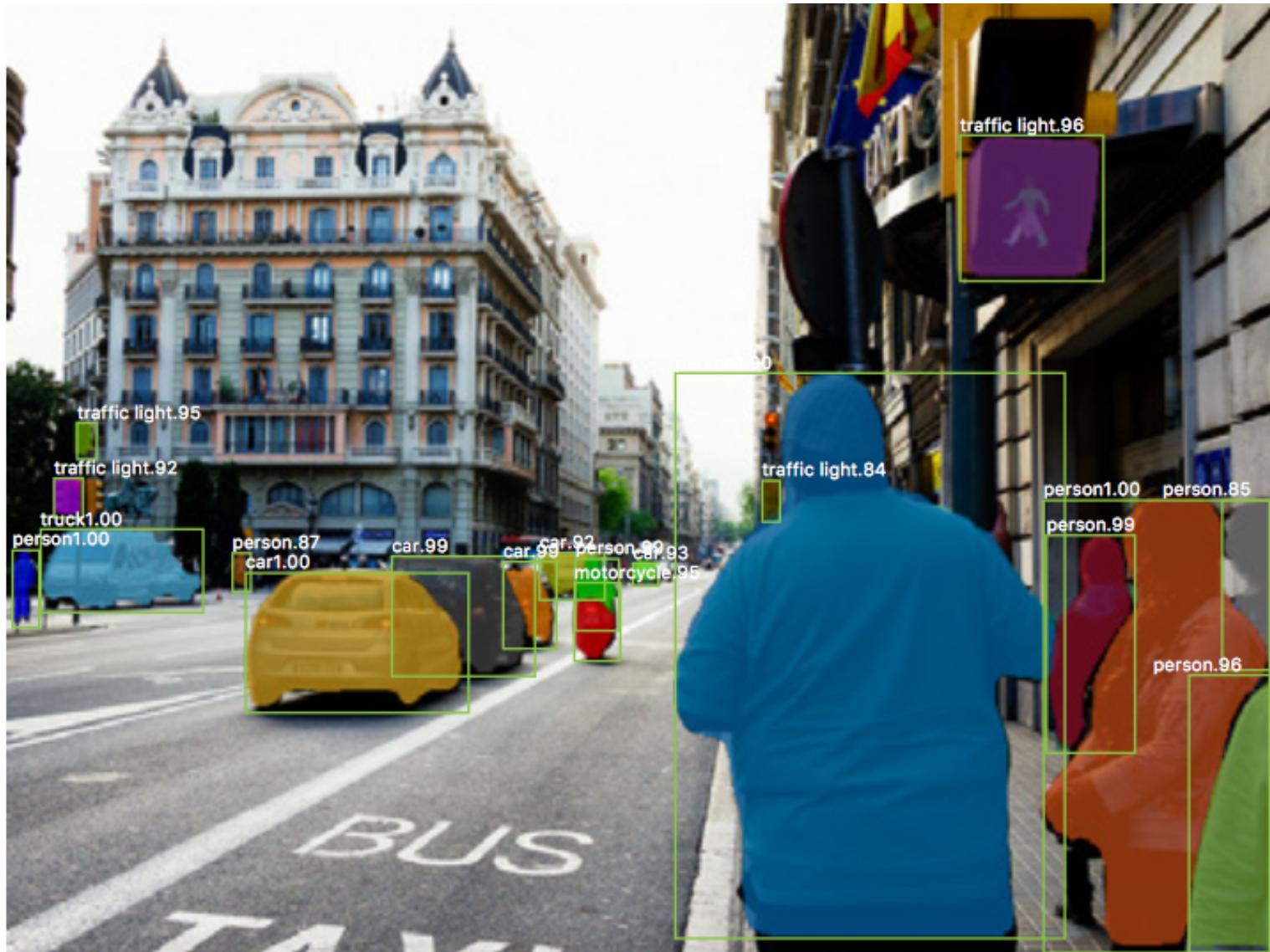
Default box задается следующими величинами:

- $x_c$ , центр прямоугольника по x
- $y_c$  - центра прямоугольника по y
- $w$  - ширина прямоугольника
- $h$  - высота прямоугольника

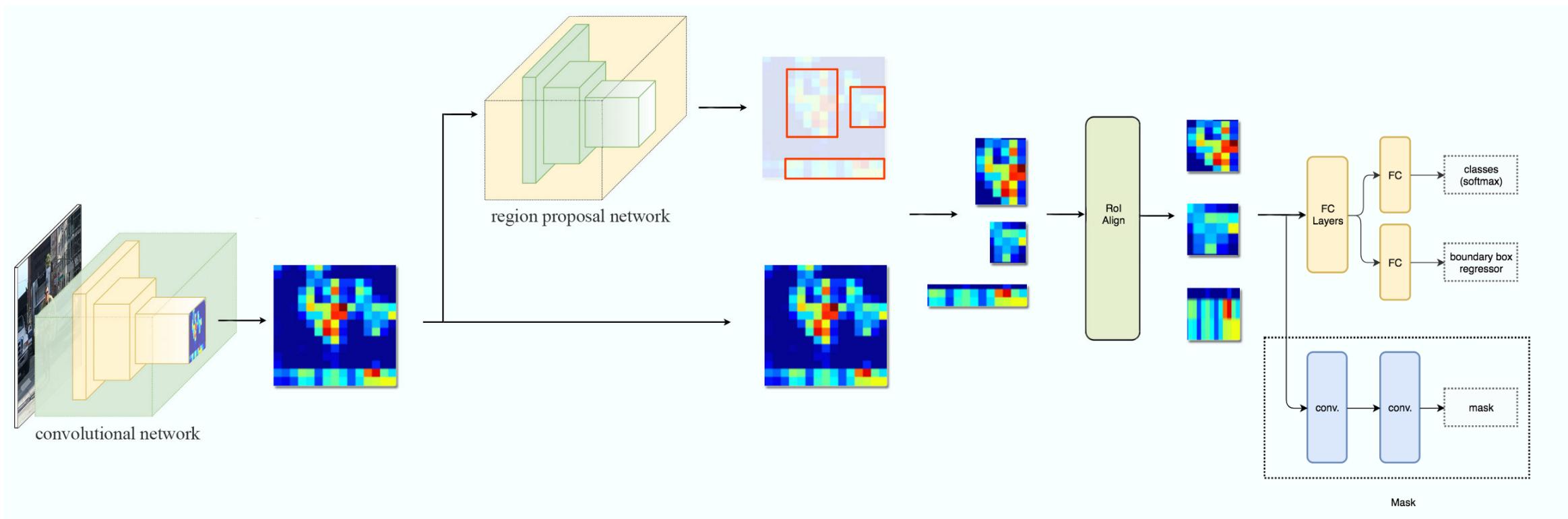
# YOLO

- YOLO делит входное изображение на сетку  $S \times S$ .
- Каждая ячейка сетки предсказывает только **один** объект
- [https://medium.com/@jonathan\\_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088](https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088)

# Masking



# Mask R-CNN



# Roi align

0.1	0.3	0.2	0.3	0.2	0.6	0.8	0.9
0.4	0.5	0.1	0.4	0.7	0.1	0.4	0.3
0.2	0.1	0.3	0.8	0.6	0.2	0.1	0.1
0.4	0.6	0.2	0.1	0.3	0.6	0.1	0.2
0.1	0.8	0.3	0.3	0.5	0.3	0.3	0.3
0.2	0.9	0.4	0.5	0.1	0.1	0.1	0.2
0.3	0.1	0.8	0.6	0.3	0.3	0.6	0.5
0.5	0.5	0.2	0.1	0.1	0.2	0.1	0.2

0.1	0.3	0.2	0.3	0.2	0.6	0.8	0.9
0.4	0.5	0.1	0.4	0.7	0.1	0.4	0.3
0.2	0.1	0.3	0.8	0.6	0.2	0.1	0.1
0.4	0.6	0.2	0.1	0.3	0.6	0.1	0.2
0.1	0.8	0.3	0.3	0.5	0.3	0.3	0.3
0.2	0.9	0.4	0.5	0.1	0.1	0.1	0.2
0.3	0.1	0.8	0.6	0.3	0.3	0.6	0.5
0.5	0.5	0.2	0.1	0.1	0.2	0.1	0.2

0.8	0.6
0.9	0.6

0.88	0.6
0.9	0.6

# Faster R-CNN for masking

