Основы глубинного обучения

Лекция 6

Задачи компьютерного зрения

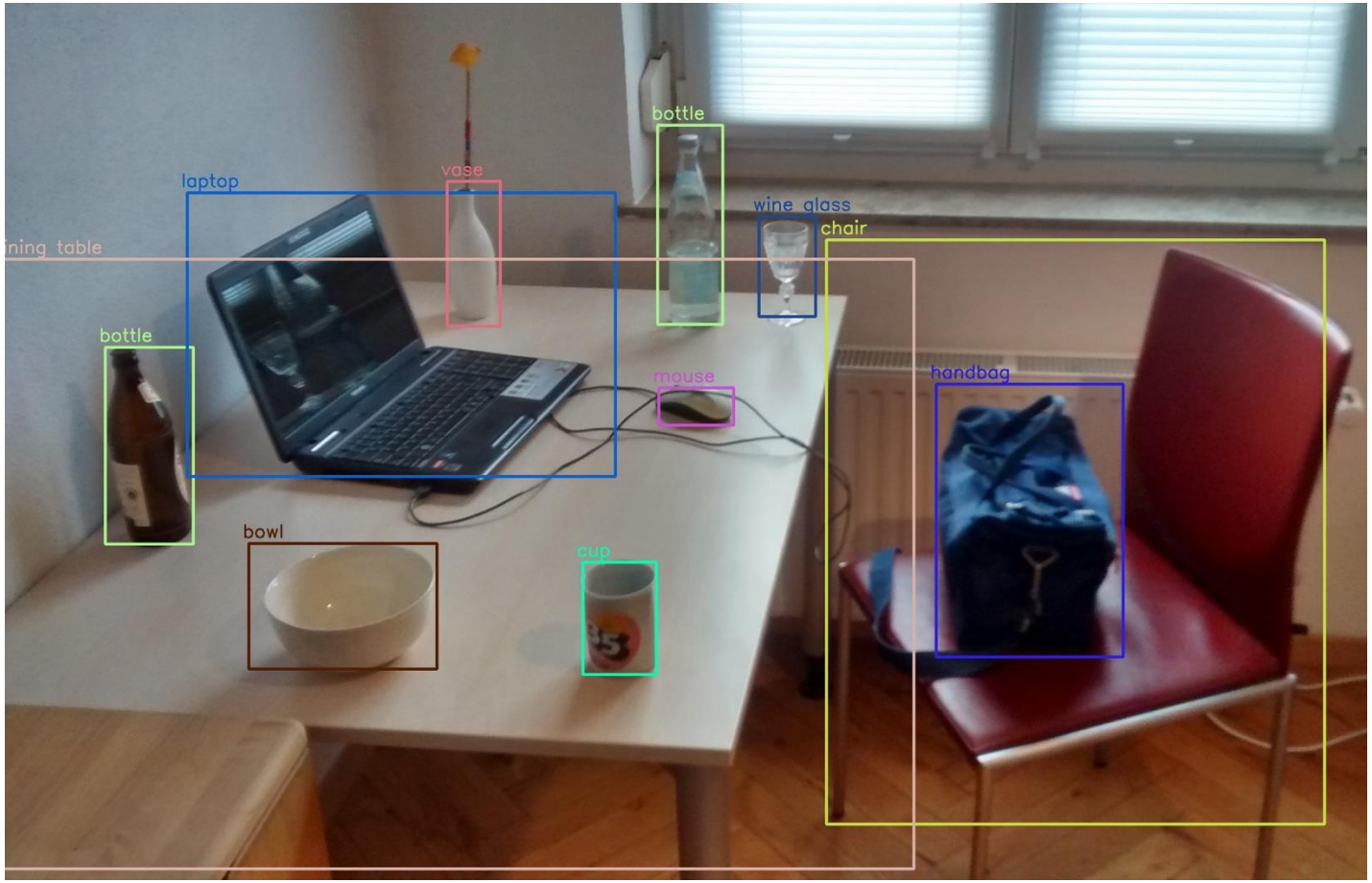
Евгений Соколов

esokolov@hse.ru

НИУ ВШЭ, 2025

Детекция объектов

Задача детекции



Задача детекции

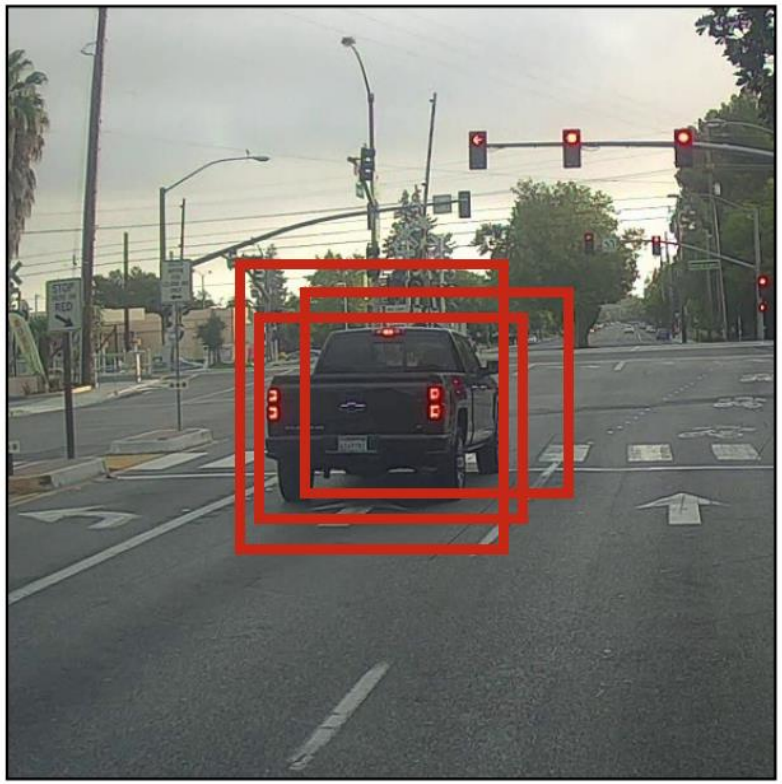
- Перебирать все прямоугольники слишком долго
- Не очень понятно, что такое хороший прямоугольник
- Прямоугольники разного размера, усложняется классификация

Non-maximum supression

- Модель выдаёт для класса k список прямоугольников с уверенностями
- Проходим в порядке уменьшения уверенности
- Для каждого прямоугольника удаляем все последующие, с которыми Intersection over Union (IoU) > 0.5

Non-maximum supression

Before non-max suppression



Non-Max Suppression

After non-max suppression

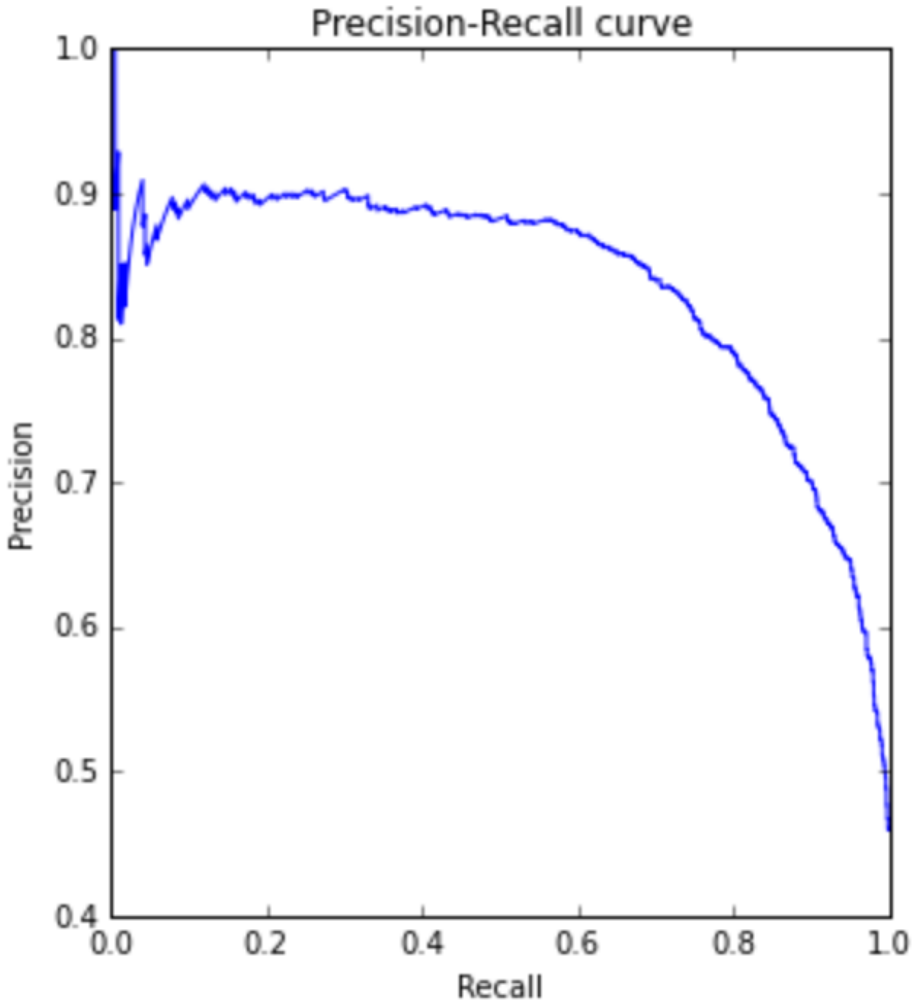
https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c

Метрики качества

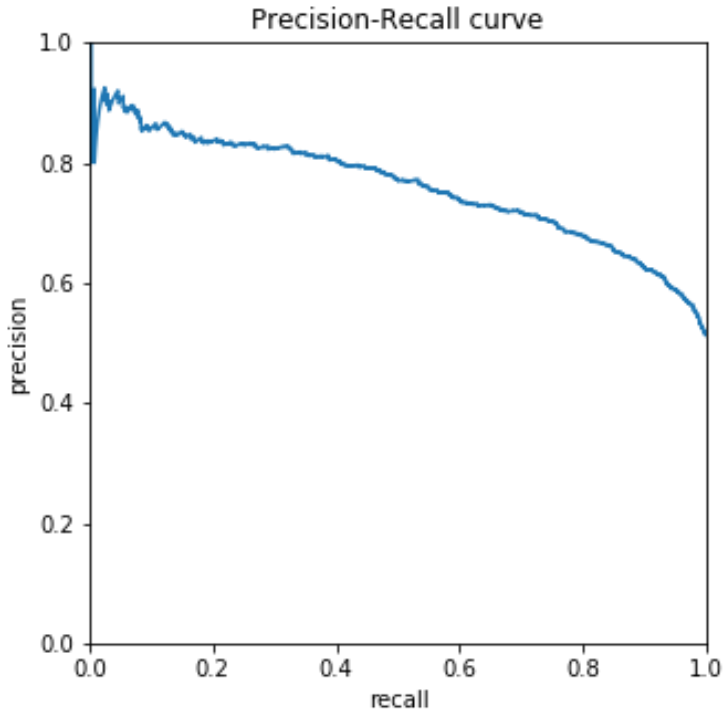
- Модель выдаёт для класса k список прямоугольников с уверенностями
- Считаем прямоугольник корректным, если
$$\mathrm{IoU}(y, z) = \mathrm{Jaccard}(y, z) > t$$
- Строим PR-кривую, считаем под ней площадь, получаем Average Precision
- Усредняем по всем классам, получаем mAP (mean AP)
- Раньше $t = 0.5$, сейчас скорее $t = 0.75$

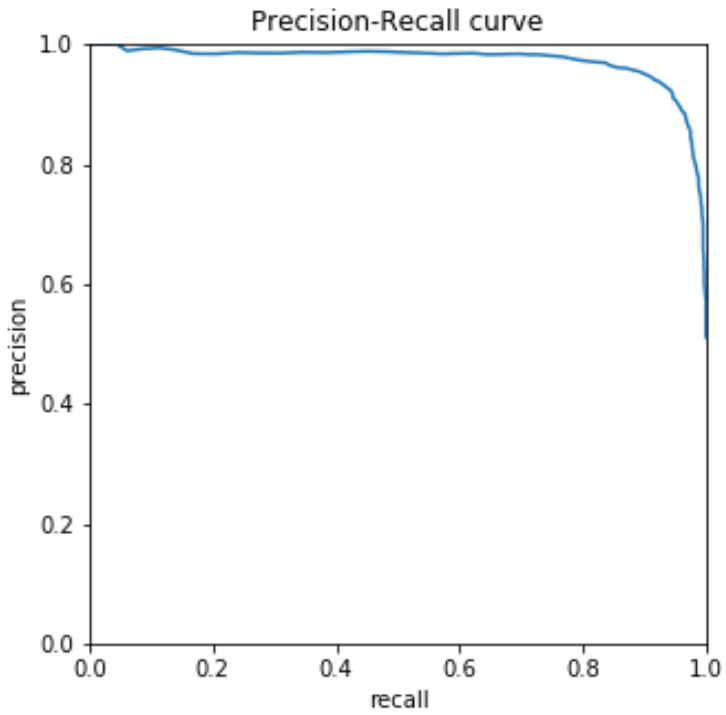
PR-кривая

- Левая точка: (0, 1)
- Правая точка: $(1, r)$, r — доля положительных объектов
- Для идеального классификатора проходит через (1, 1)
- AUC-PRC — площадь под PR-кривой



PR-кривая


Precision-Recall curve
precision
recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

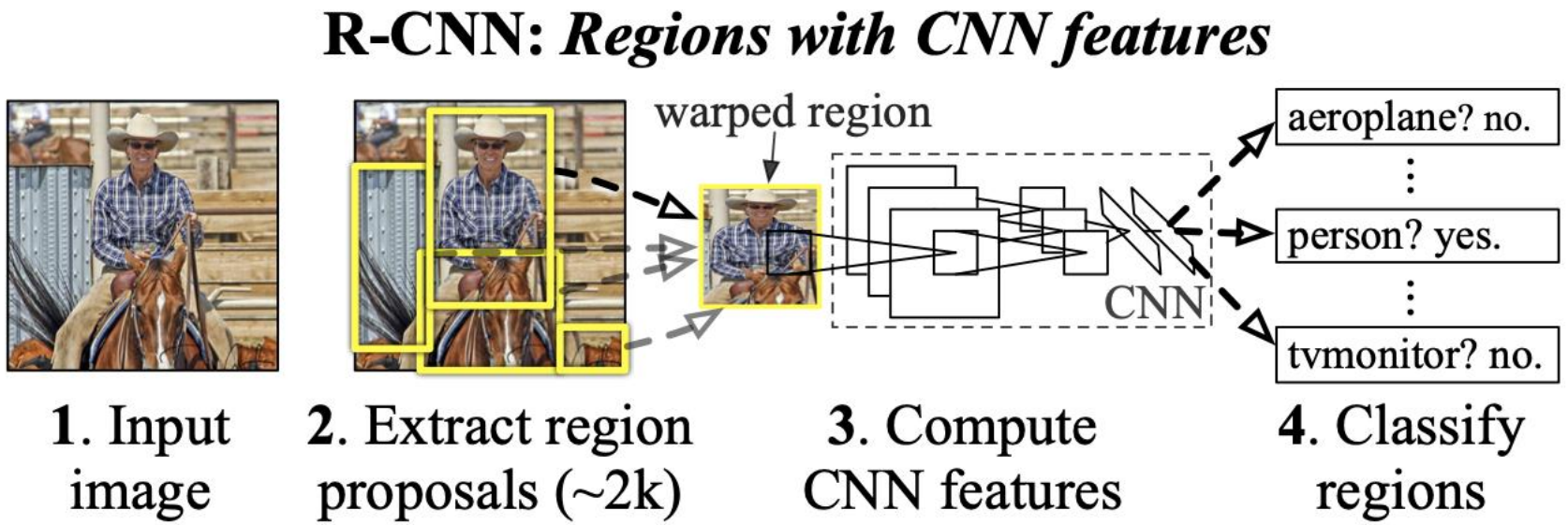

Precision-Recall curve
precision
recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

Two-shot detection

Будем решать задачу в два этапа:

1. Находим «кандидатов» — прямоугольники, где скорее всего что-то есть
2. Классифицируем эти прямоугольники

R-CNN

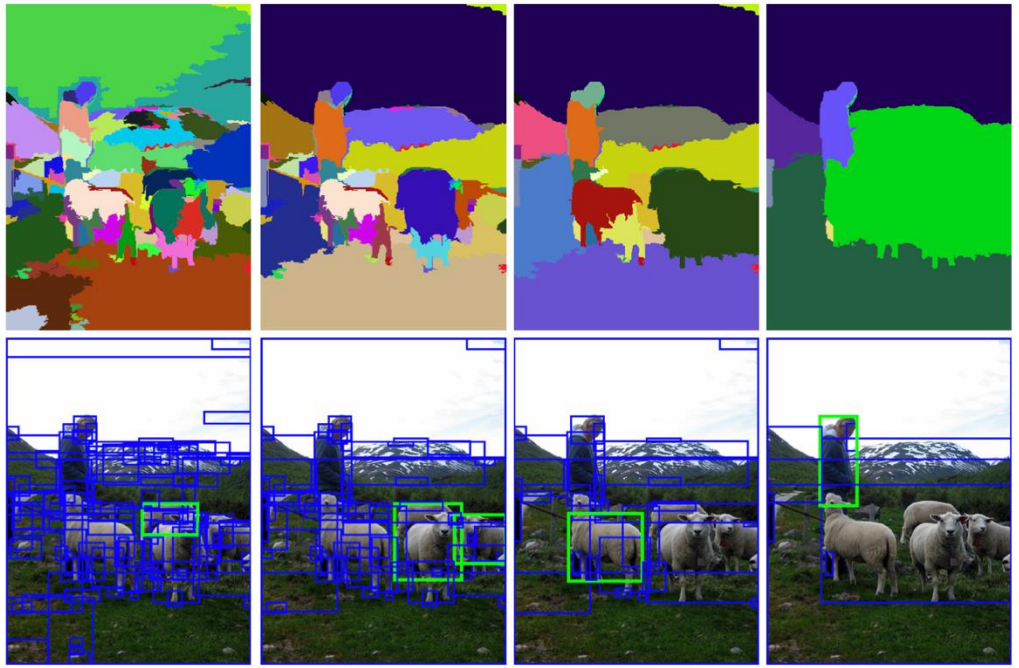


https://arxiv.org/abs/1311.2524

R-CNN

Генерация кандидатов:

- «Внешние» методы из классического компьютерного зрения
- Выбирается около 2000 прямоугольников



https://ivi.fnwi.uva.nl/isis/publications/2013/UijlingsIJCV2013/UijlingsIJCV2013.pdf

R-CNN

Извлечение признаков:

- Выходы предпоследнего слоя AlexNet (4096 чисел)

https://arxiv.org/abs/1311.2524

R-CNN

Классификация прямоугольников:

- SVM
- One-vs-all

https://arxiv.org/abs/1311.2524

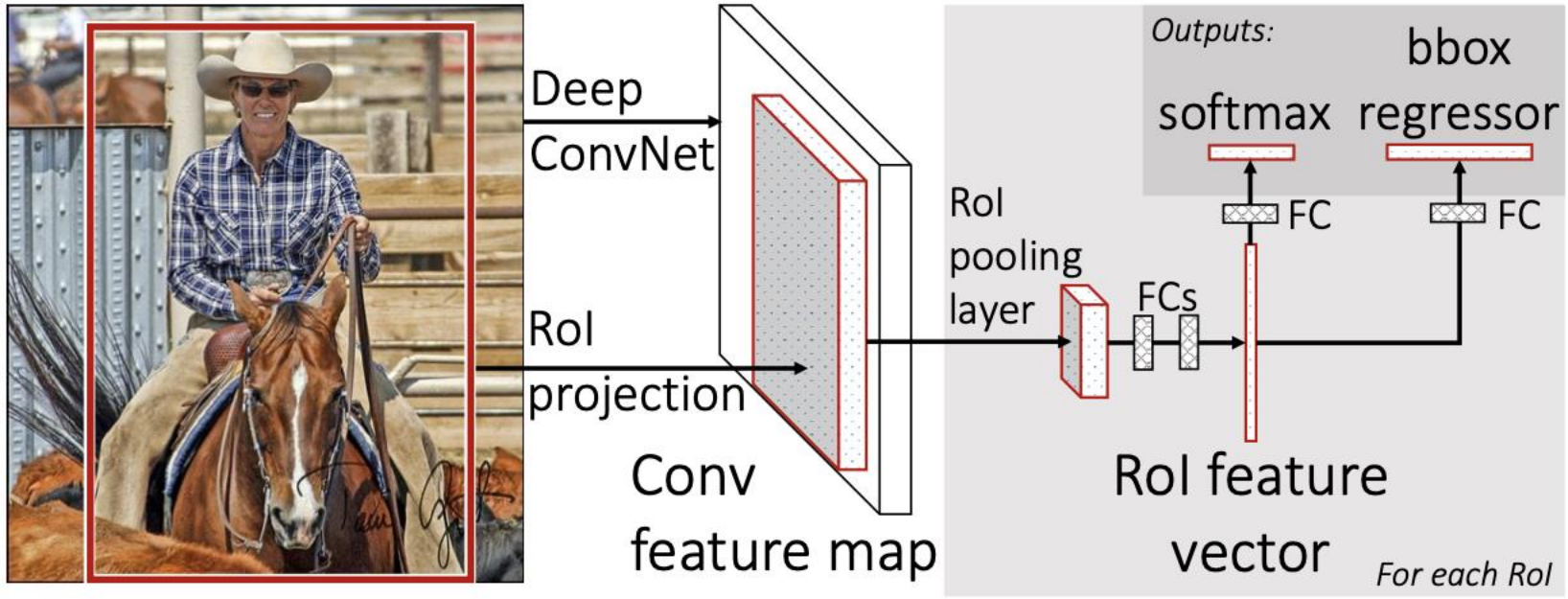
R-CNN

Проблемы:

- Не end-to-end
- Генерация кандидатов может быть очень сложной
- Свёрточная сеть практически не настраивается под данные
- Долго (много признаков, много классификаторов)

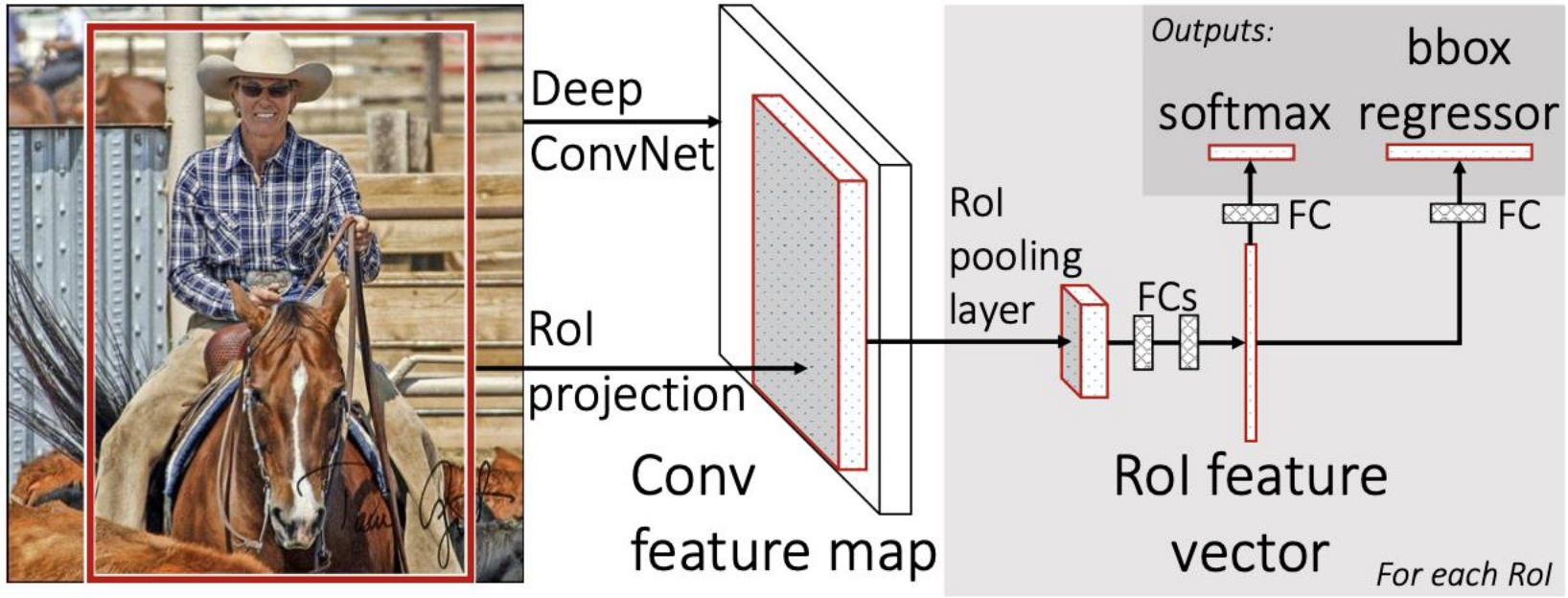
https://arxiv.org/abs/1311.2524

Fast R-CNN



https://arxiv.org/abs/1504.08083

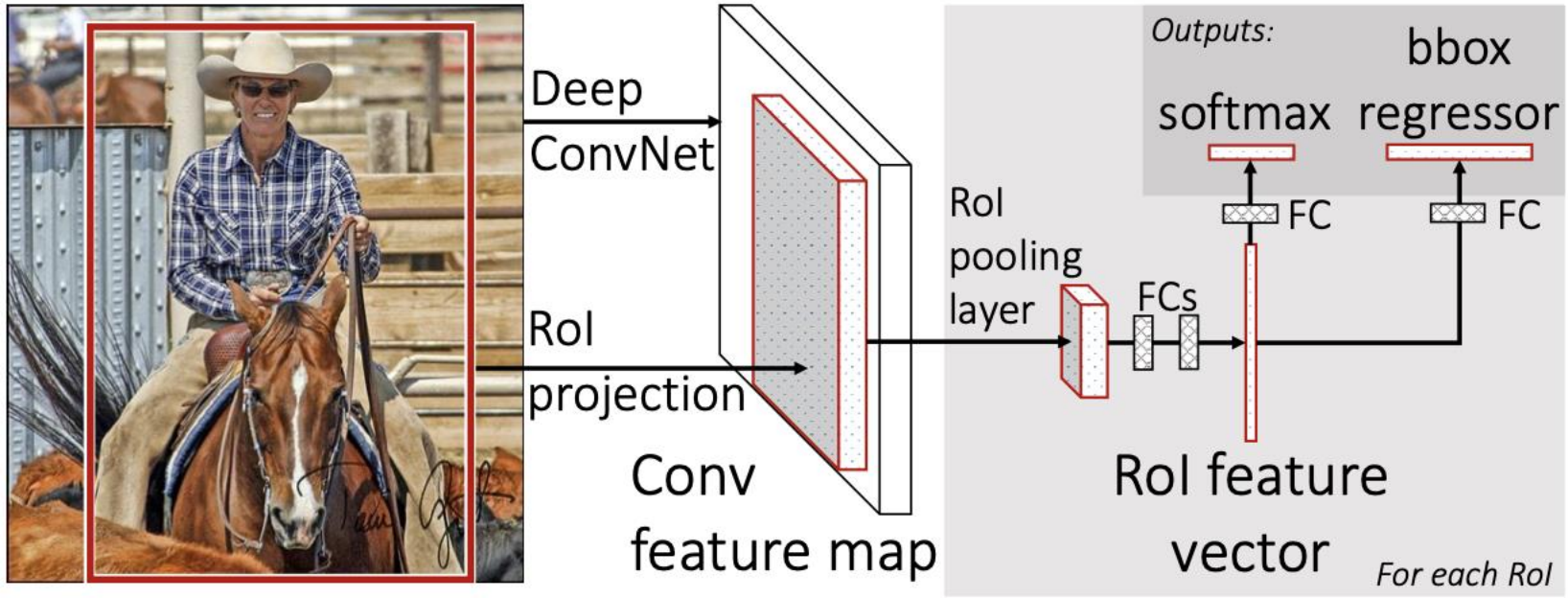
Fast R-CNN



- Кандидаты всё ещё генерируются внешним методом

https://arxiv.org/abs/1504.08083

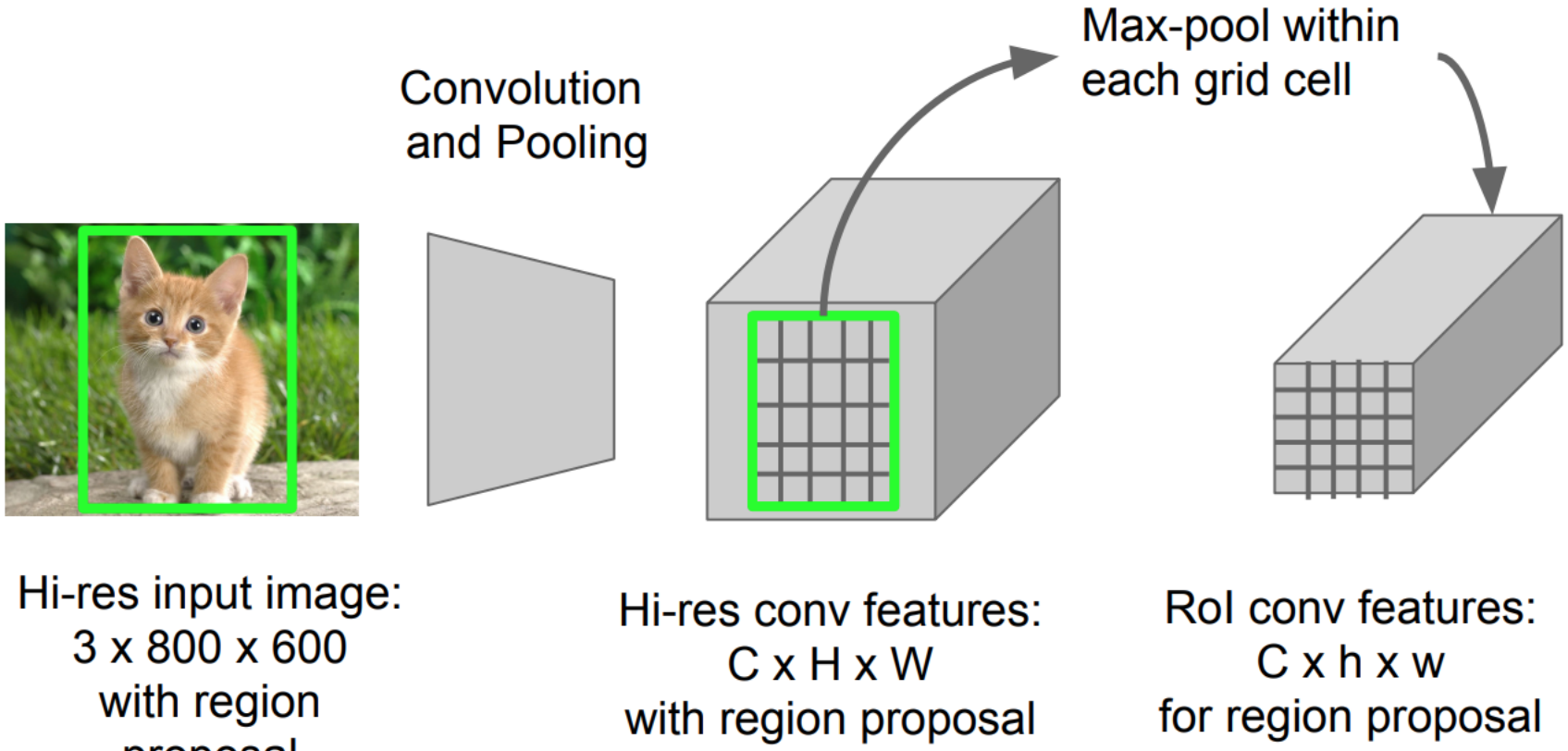
Fast R-CNN



- Для конкретного кандидата извлекаются признаки: вырезаются участки из последнего тензора, режутся на блоки, из каждого блока берётся максимум

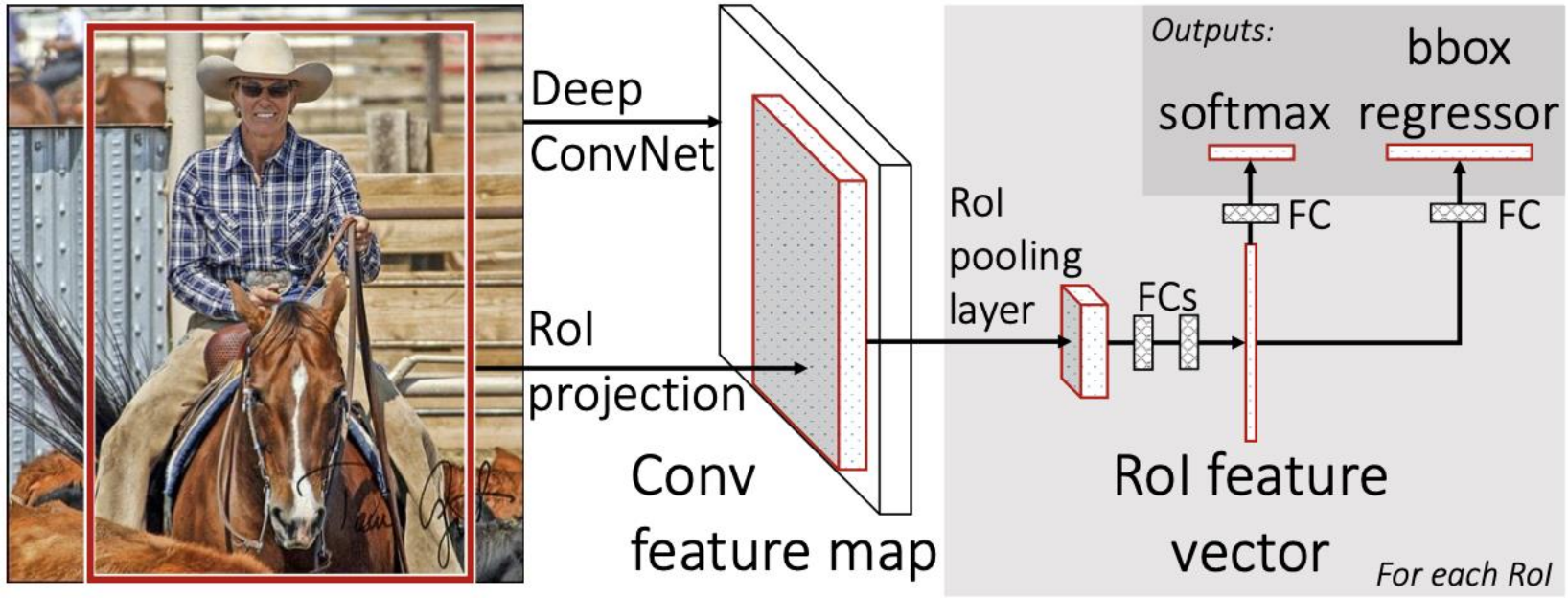
https://arxiv.org/abs/1504.08083

RoI Pooling



http://cs231n.stanford.edu/slides/2016/winter1516_lecture8.pdf

Fast R-CNN



- Для конкретного кандидата предсказываются вероятности классов и 4 поправки к размерам прямоугольника

https://arxiv.org/abs/1504.08083

Fast R-CNN

индикатор наличия объекта в области

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

CCE

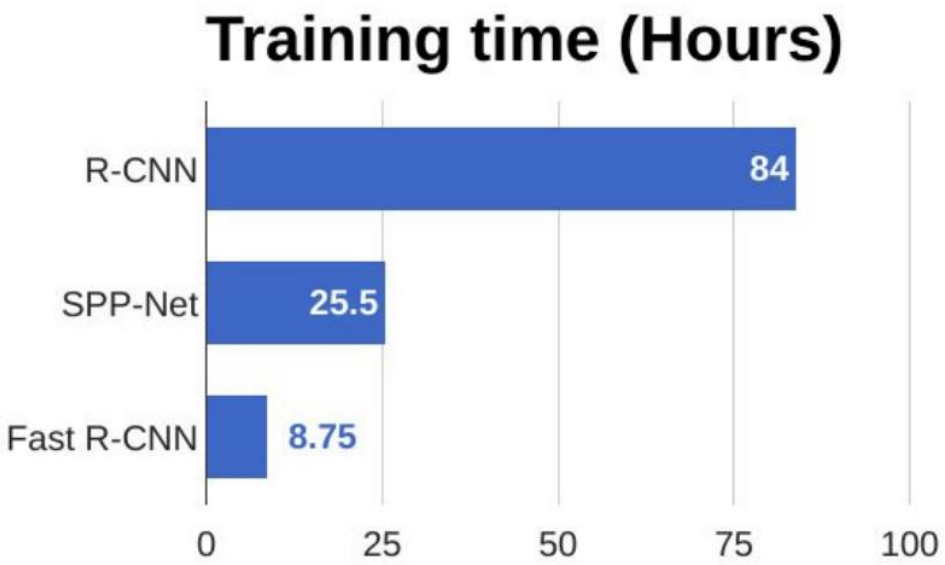
$$L_{\text{loc}}(t^u, v) = \sum_{i \in \{\text{x,y,w,h}\}} \text{smooth}_{L_1}(t_i^u - v_i),$$

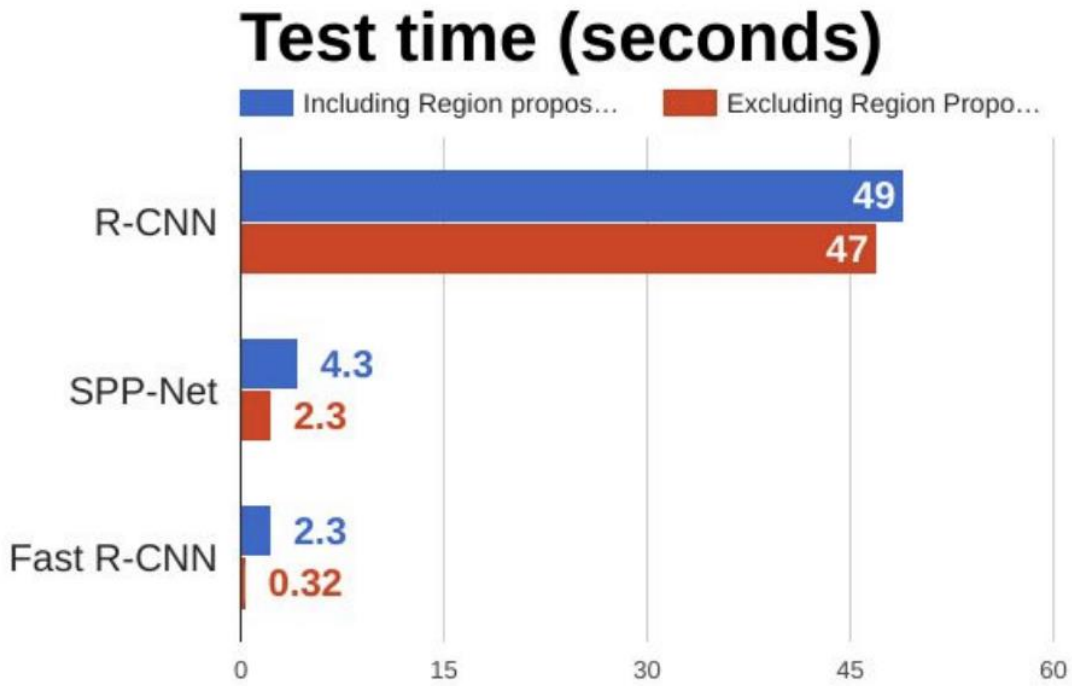
in which

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

https://arxiv.org/abs/1504.08083

Fast R-CNN





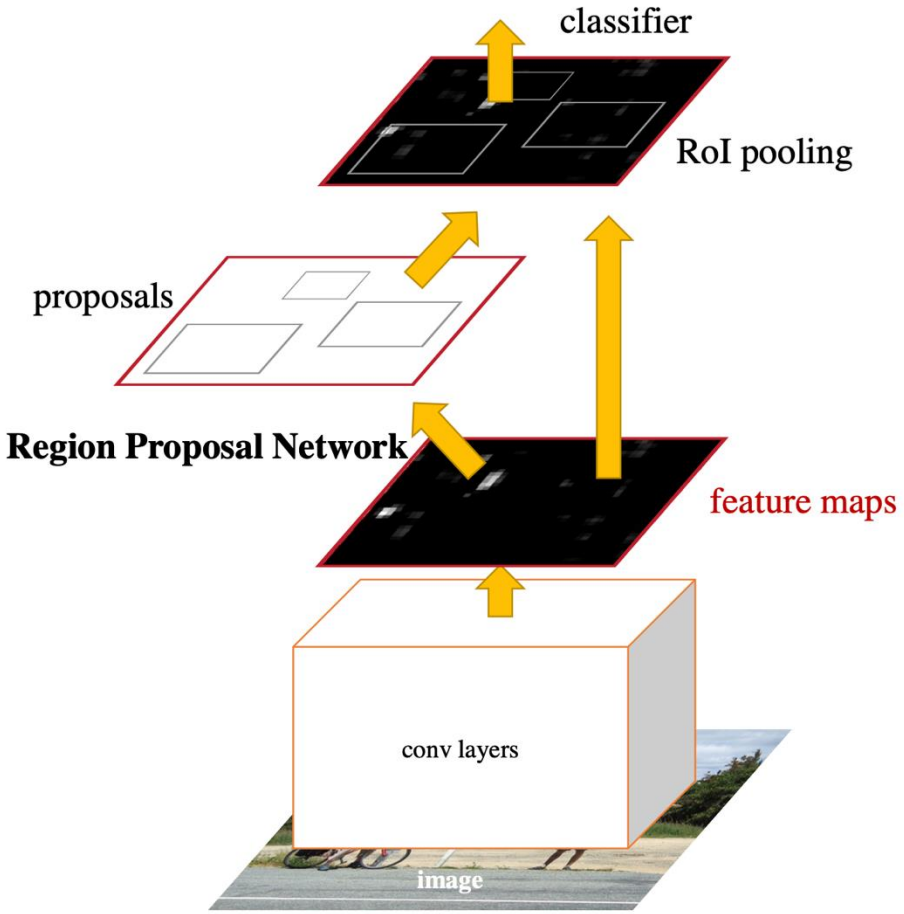
https://arxiv.org/abs/1504.08083

Fast R-CNN

- Есть дообучение всей сети
- Исправили почти все проблемы, но теперь генерация кандидатов занимает больше всего времени

https://arxiv.org/abs/1504.08083

Faster R-CNN

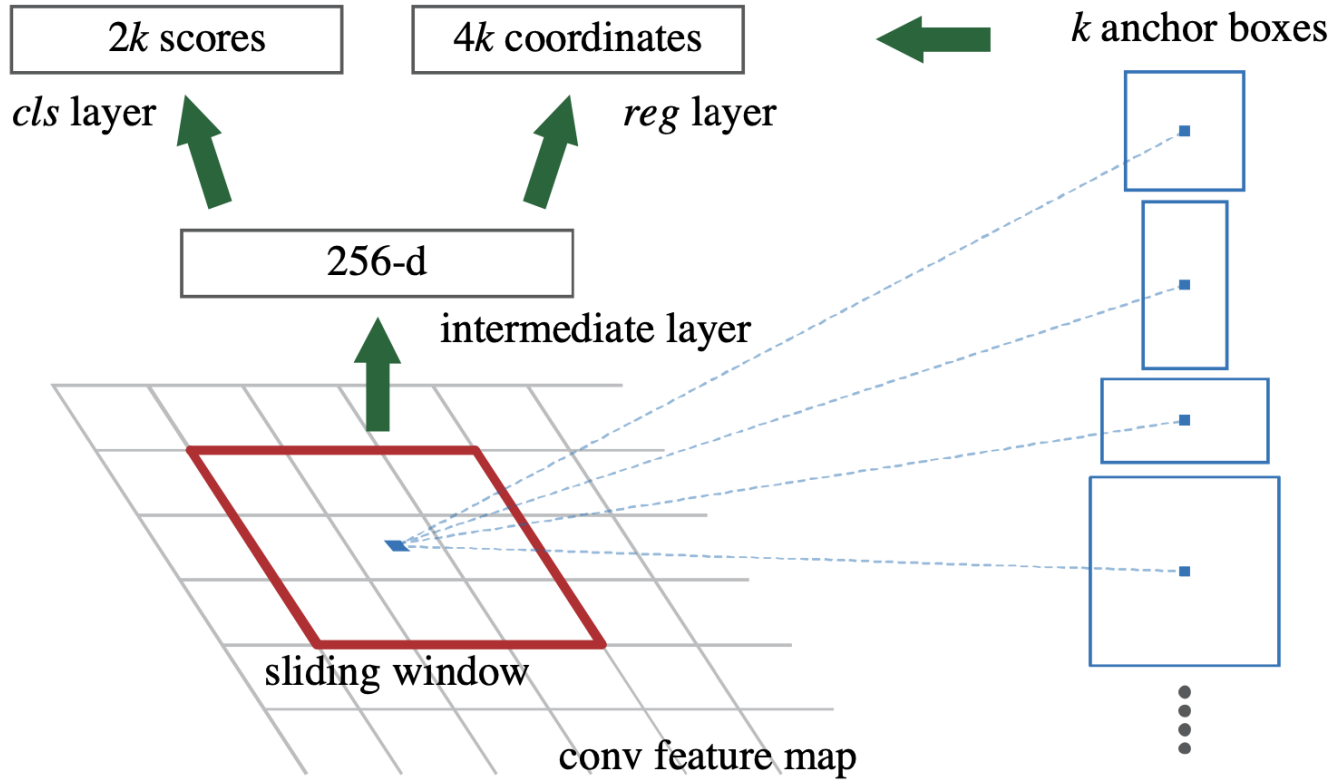


https://arxiv.org/abs/1506.01497

Region Proposal Network

- Берём последний тензор из свёрточной сети
- Применяем свёрточный слой 3x3 с большим количеством каналов (256 или 512)
- Для каждой точки берём 9 прямоугольников с разными длинами сторон и соотношениями сторон
- Для каждого предсказываем:
  - Есть ли там объект
  - 4 поправки к сторонам

Region Proposal Network



https://arxiv.org/abs/1506.01497

Faster R-CNN

Всё обучается совместно, оптимизируем сумму 4 функций потерь:

- Классификация «есть ли объект» в RPN
- Регрессия для корректировки прямоугольника в RPN
- Классификация на K классов в основной модели
- Регрессия для корректировки прямоугольника в основной модели

https://arxiv.org/abs/1506.01497

Faster R-CNN

На этапе применения:

- Выделяем самые уверенные регионы из RPN
- Подаём в основную модель для детекции

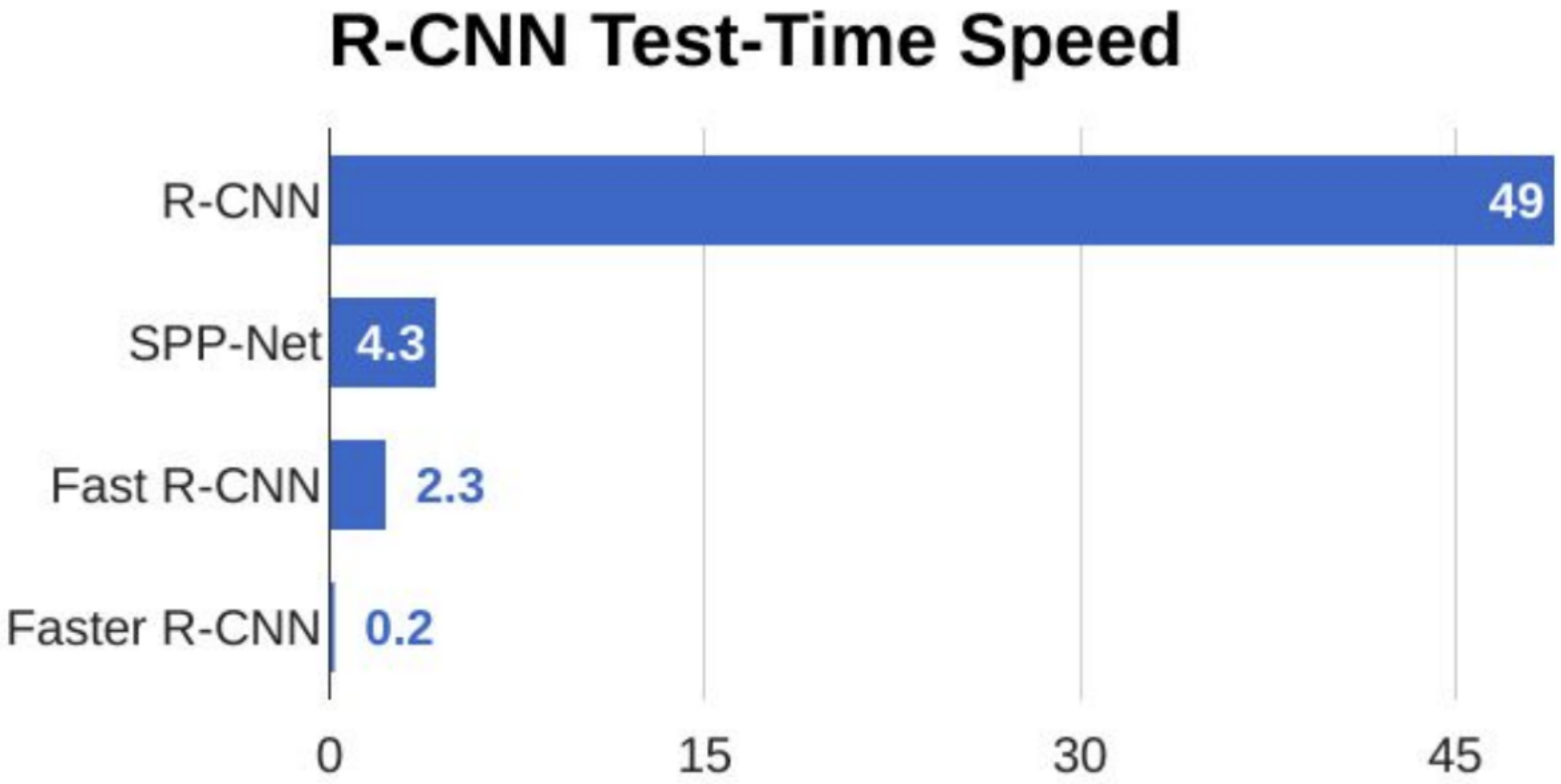
https://arxiv.org/abs/1506.01497

Faster R-CNN

	R-CNN	**Fast R-CNN**	**Faster R-CNN**
Test time per image (with proposals)	50 seconds	2 seconds	**0.2 seconds**
(Speedup)	1x	25x	**250x**
mAP (VOC 2007)	66.0	**66.9**	**66.9**

https://arxiv.org/abs/1506.01497

Faster R-CNN



https://arxiv.org/abs/1506.01497

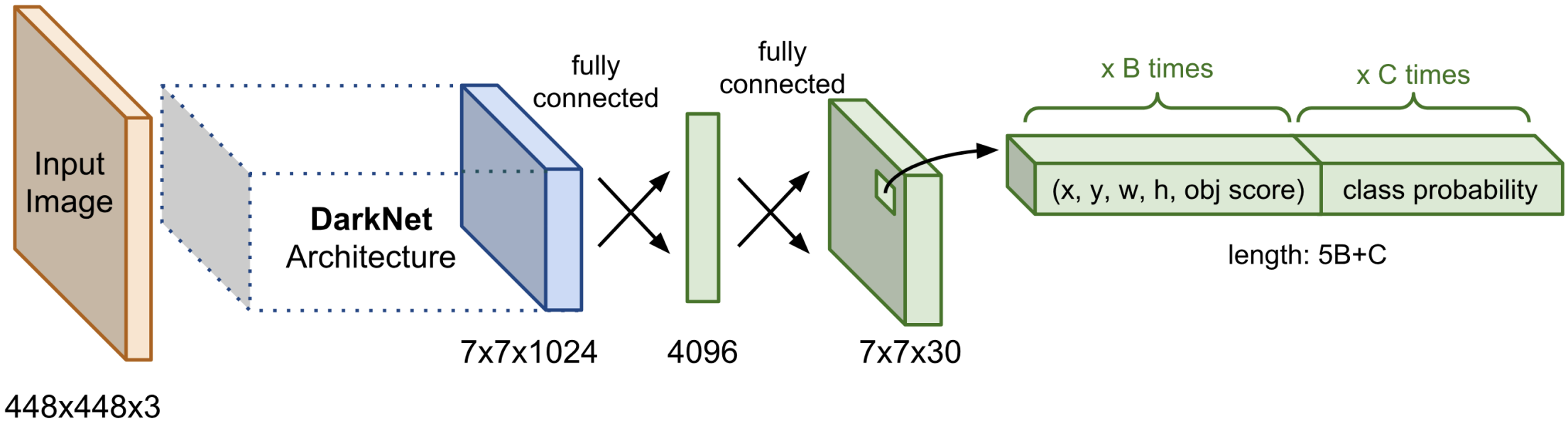
Что ещё?

- Mask R-CNN — instance segmantation

One-shot detection

- Двухэтапные детекторы работают хорошо, но не очень быстро
- Попытаемся одновременно и искать кандидатов, и определять их классы

YOLO



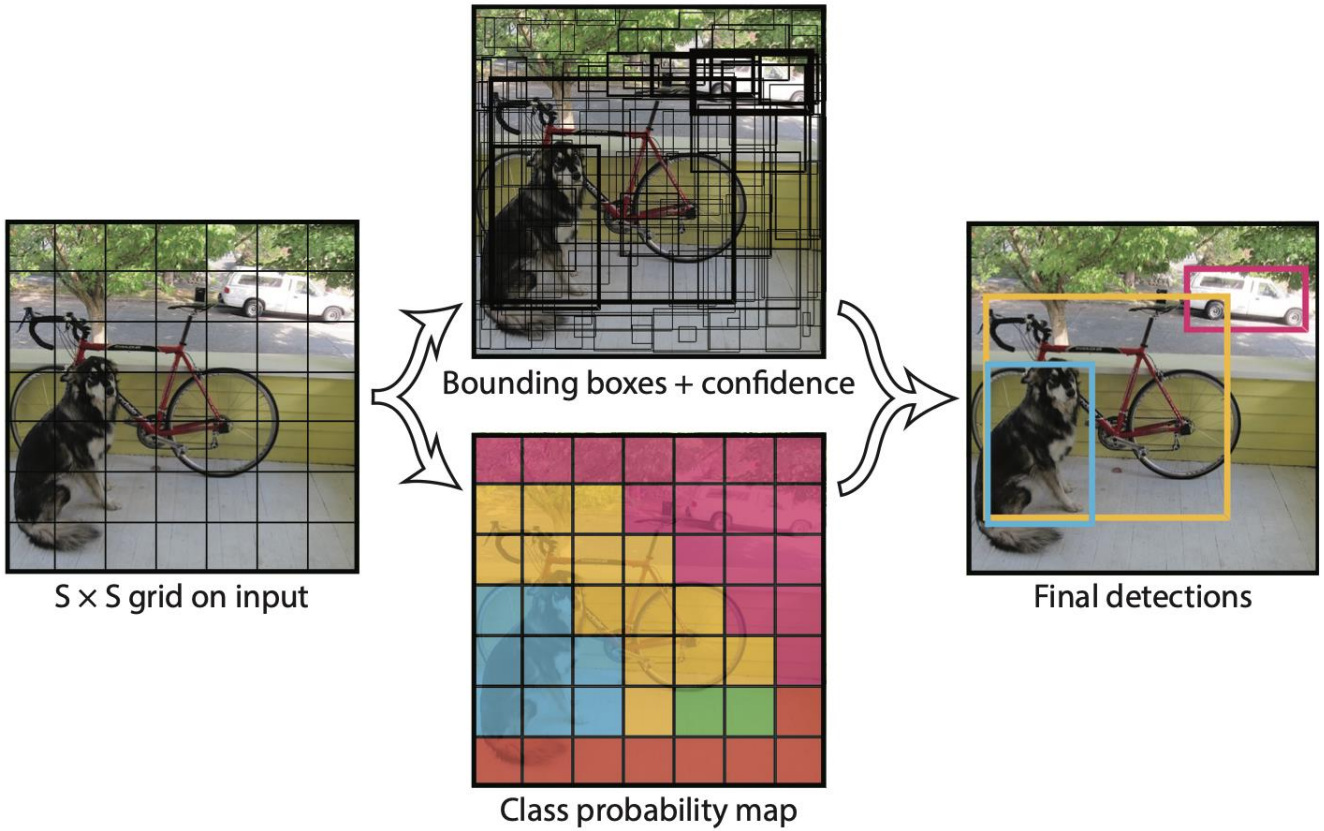
https://lilianweng.github.io/lil-log/2018/12/27/object-detection-part-4.html

YOLO

- Разбиваем изображение на S*S блоков
- Для каждого блока предсказываем B прямоугольников
- Для каждого прямоугольника:
  - Координаты
  - Вероятность наличия объекта
  - Вероятность каждого класса **при условии** наличия объекта

https://arxiv.org/abs/1506.02640

YOLO



https://arxiv.org/abs/1506.02640

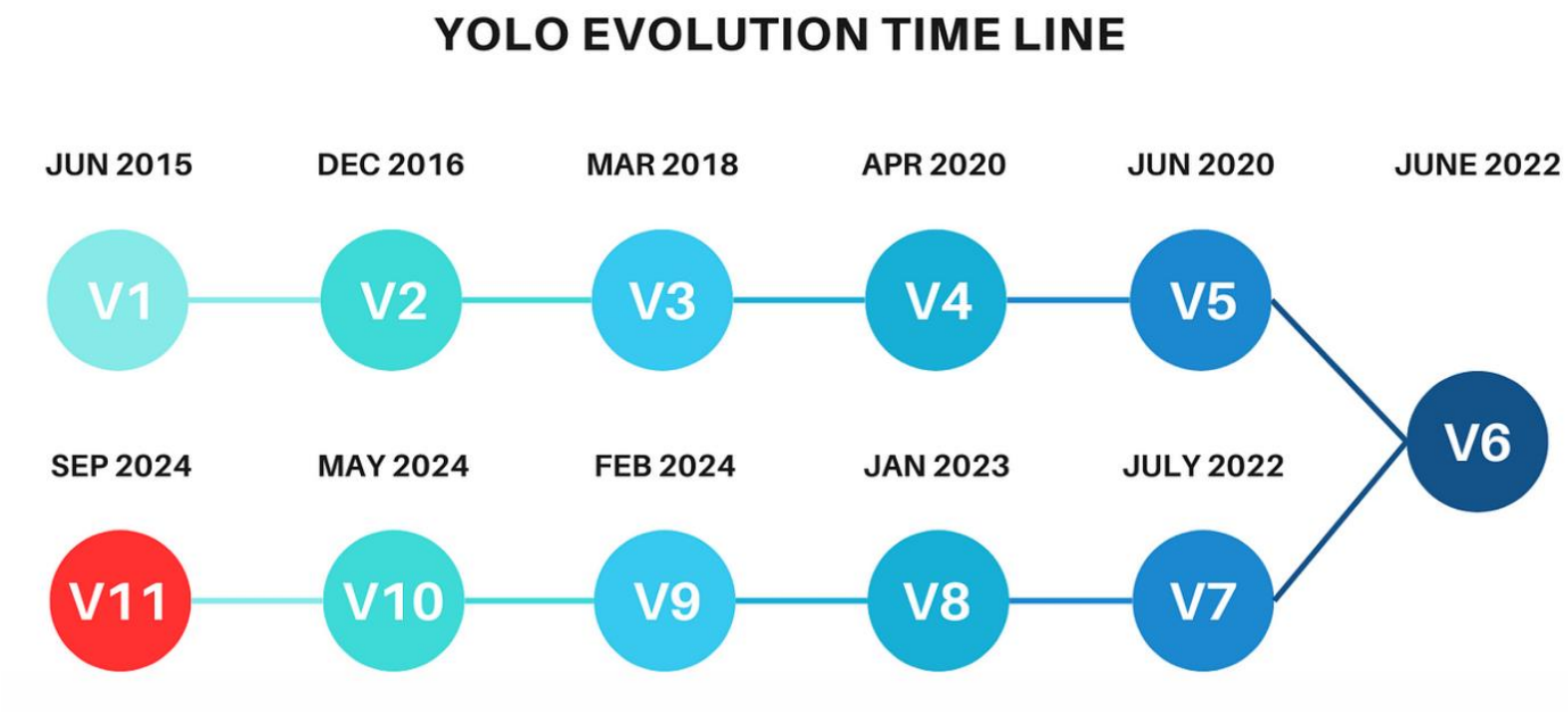
YOLO

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

https://arxiv.org/abs/1506.02640

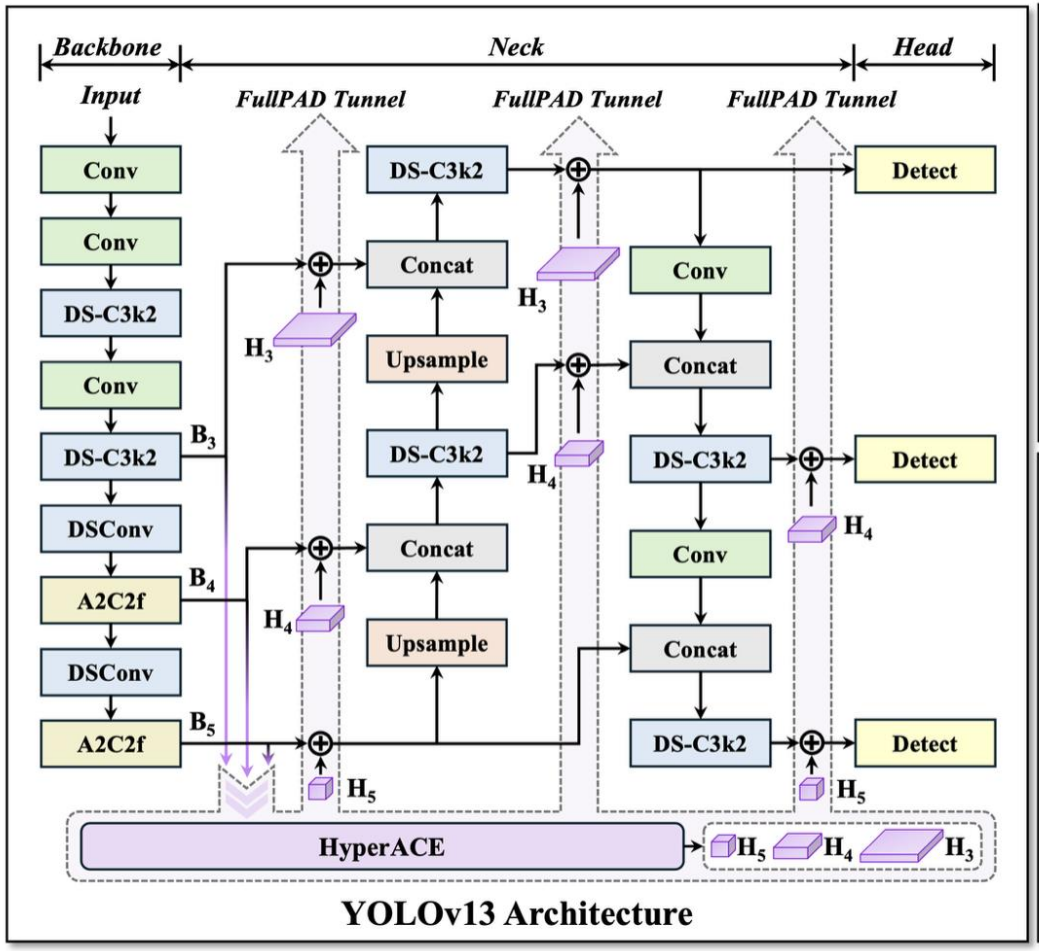
Что ещё?

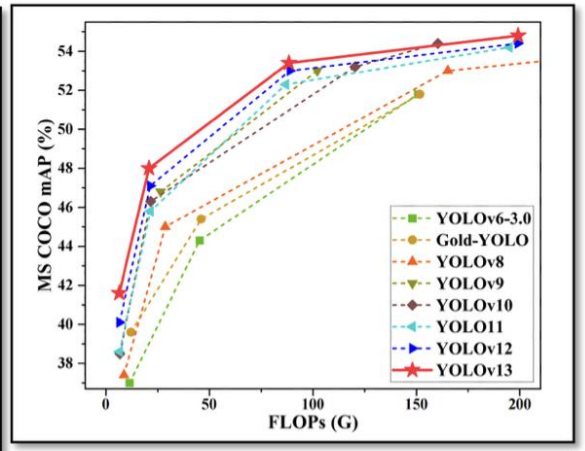
- SSD (Single Shot Detector)
- RetinaNet
- YOLOv{number}
- Детекторы на трансформере

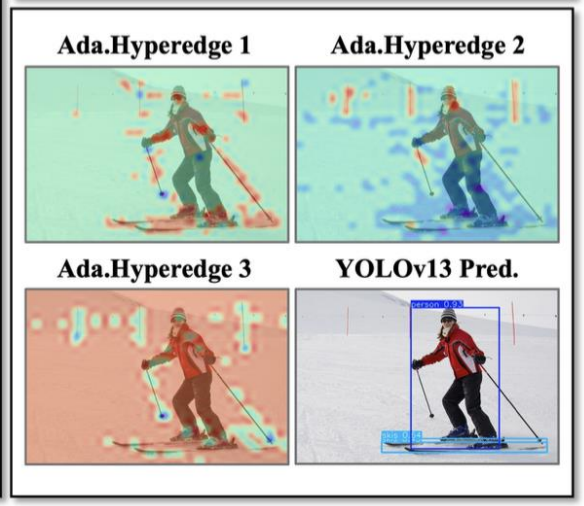


https://medium.com/@nikhil-rao-20/yolov11-explained-next-level-object-detection-with-enhanced-speed-and-accuracy-2dbe2d376f71

YOLOv13 внутри







https://github.com/iMoonLab/yolov13

Идентификация объектов

Labeled Faces in the Wild

- Около 13 тысяч фотографий
- Около 6 тысяч человек

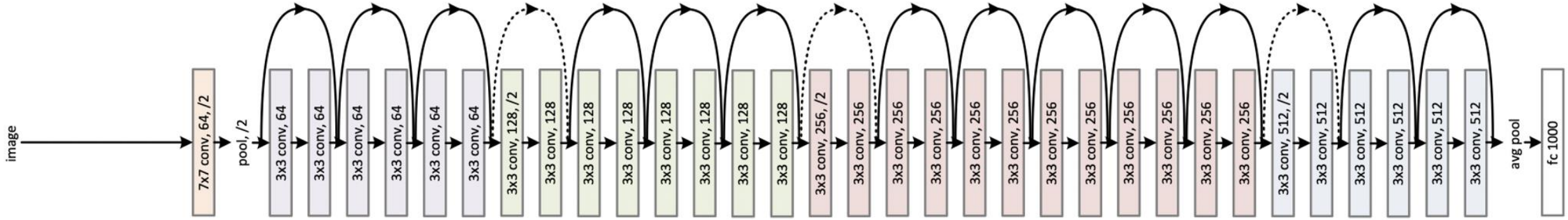
http://vis-www.cs.umass.edu/lfw/#information

MegaFace

- 4.7 миллионов фотографий
- Около 700 тысяч человек
- В среднем 7 фото на человека

http://megaface.cs.washington.edu

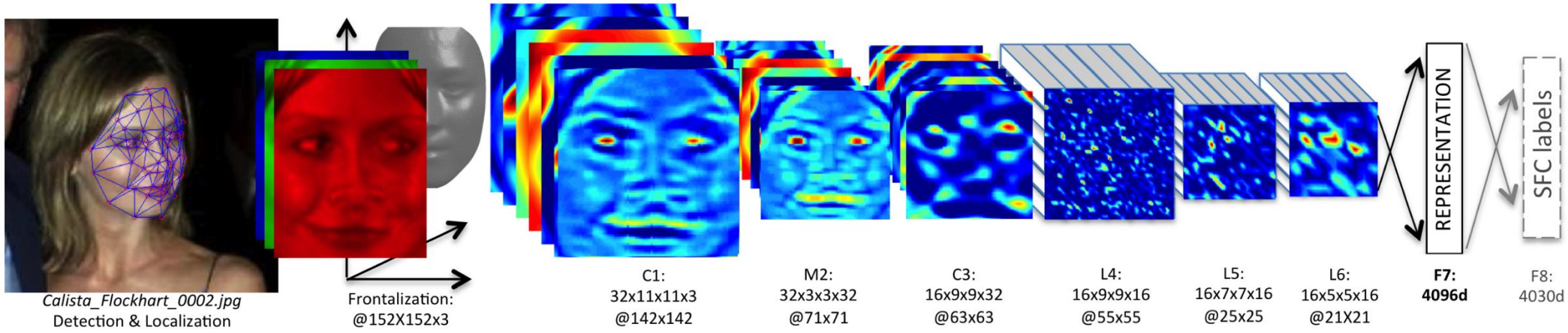
Дообучение



Если данных совсем мало:

- Берём модель из другой задачи
- Заменяем последний слой на слой с нужным числом выходов
- Обучаем только его
- По сути, это обучение линейной модели

DeepFace



https://www.cs.toronto.edu/~ranzato/publications/taigman_cvpr14.pdf

DeepFace

- Обучаем некоторую архитектуру для классификации (число классов = число людей в данных)
- Используем выходы предпоследнего слоя как признаковое описание изображения
- Признаки нормализуются (чтобы норма была единичной)
- Считаем близость векторов по какой-нибудь метрике

https://www.cs.toronto.edu/~ranzato/publications/taigman_cvpr14.pdf

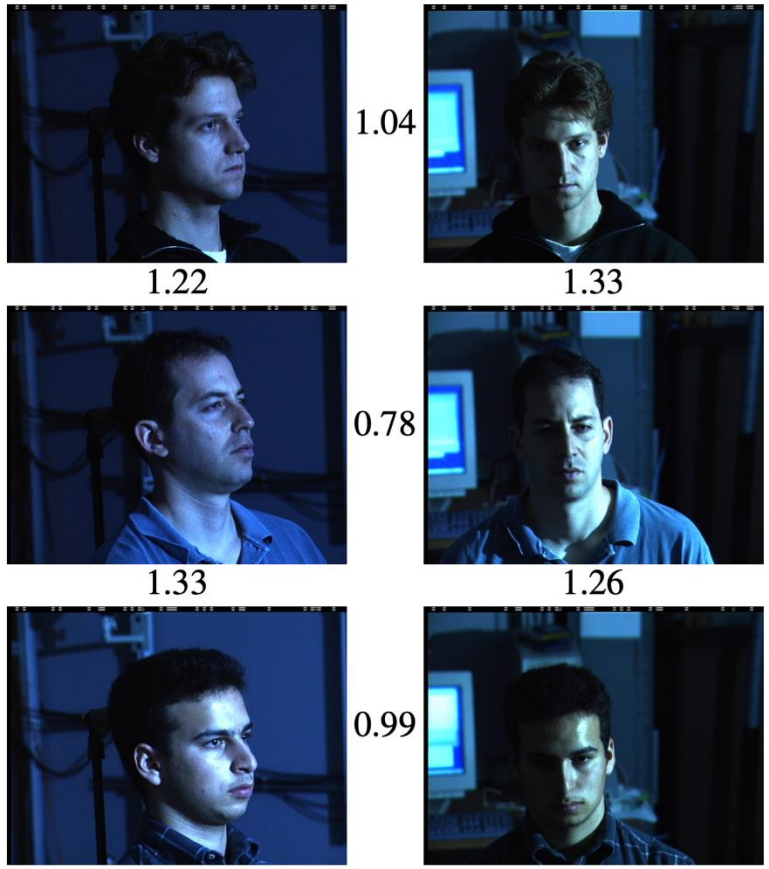
DeepFace

- Можно сравнить расстояние с порогом, чтобы идентифицировать человека

https://www.cs.toronto.edu/~ranzato/publications/taigman_cvpr14.pdf

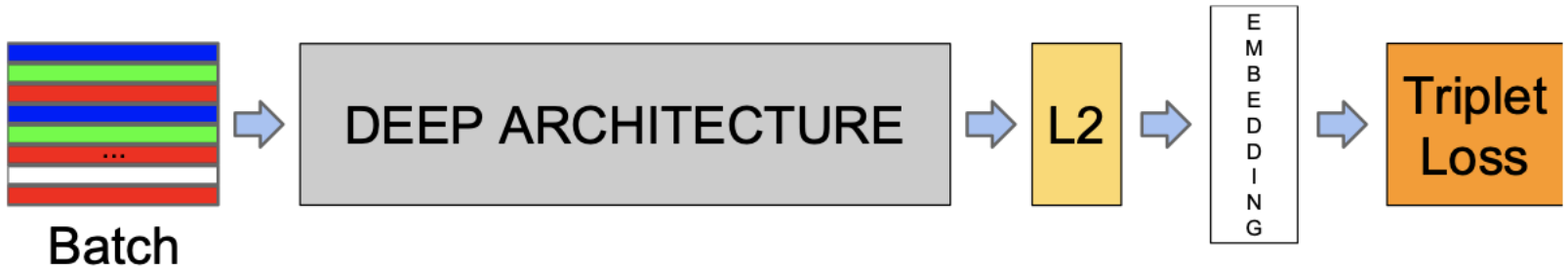
FaceNet

- Почему бы в явном виде не обучать представления изображений так, чтобы фотографии одного человека имели близкие представления?



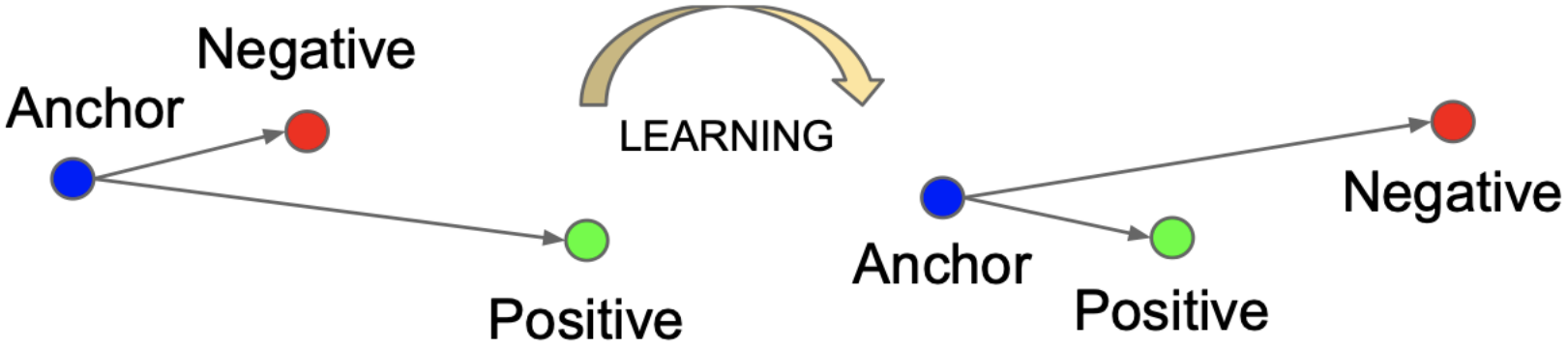
https://arxiv.org/pdf/1503.03832.pdf

FaceNet



https://arxiv.org/pdf/1503.03832.pdf

FaceNet



$$\sum_{i}^{N} \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+$$

https://arxiv.org/pdf/1503.03832.pdf

FaceNet

- Важно правильно выбирать триплеты
- Обычно: выбираем positive и ищем semi-hard negatives

$$\|f(x_i^a) - f(x_i^p)\|_2^2 < \|f(x_i^a) - f(x_i^n)\|_2^2$$

https://arxiv.org/pdf/1503.03832.pdf

Триплетная и попарная ошибки

- Попарная ошибка:

$$\sum_{(i,j)\in R} \left[a(x_i) - a\left(x_j\right) < 0\right] \to \min$$

- Не совсем про обучение расстояния

FaceNet

- Точность на LFW: 99.63%

https://arxiv.org/pdf/1503.03832.pdf