

Основы глубинного обучения

Лекция 8

Задачи компьютерного зрения

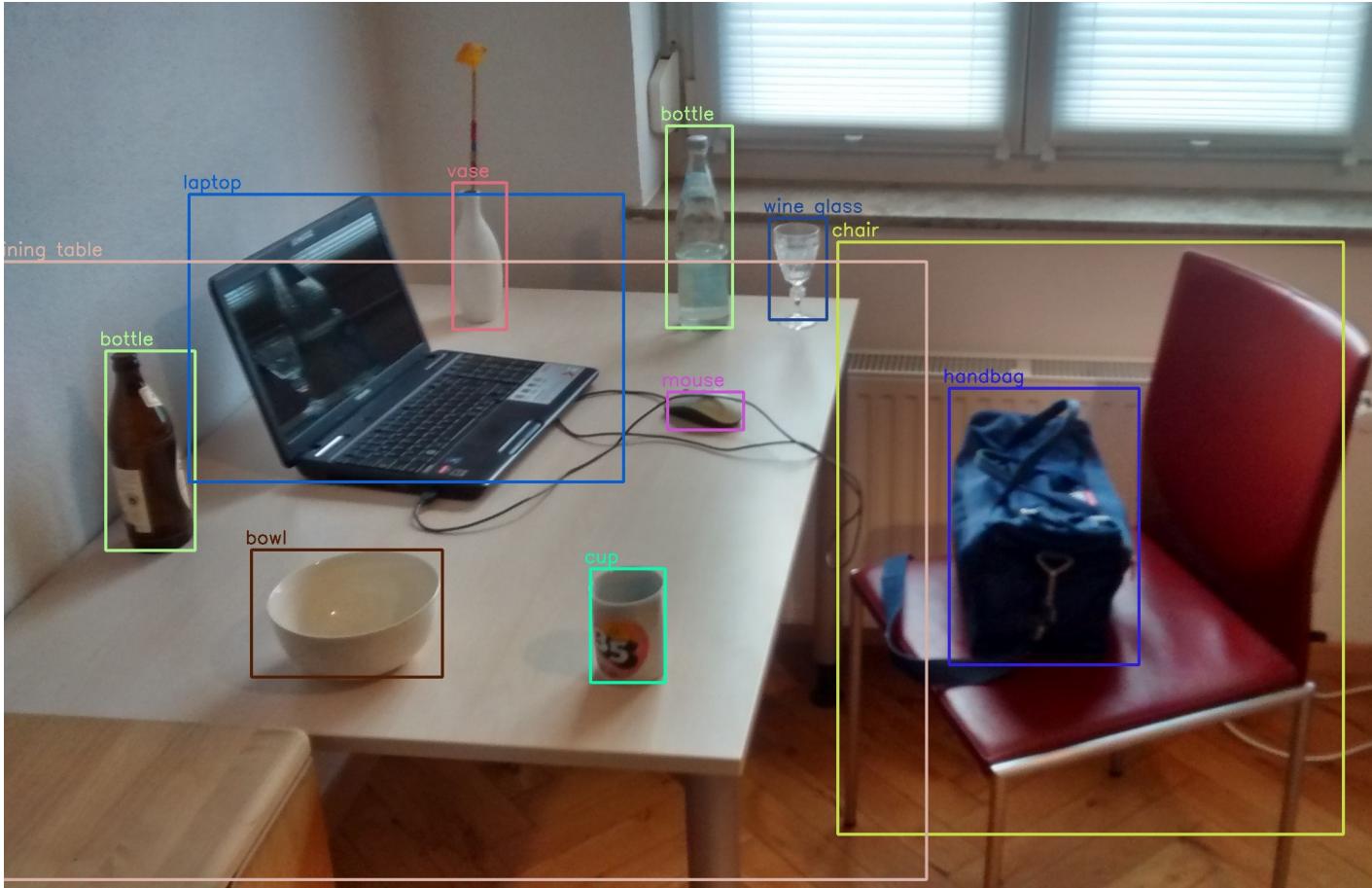
Евгений Соколов

esokolov@hse.ru

НИУ ВШЭ, 2021

Детекция объектов

Задача детекции



Задача детекции

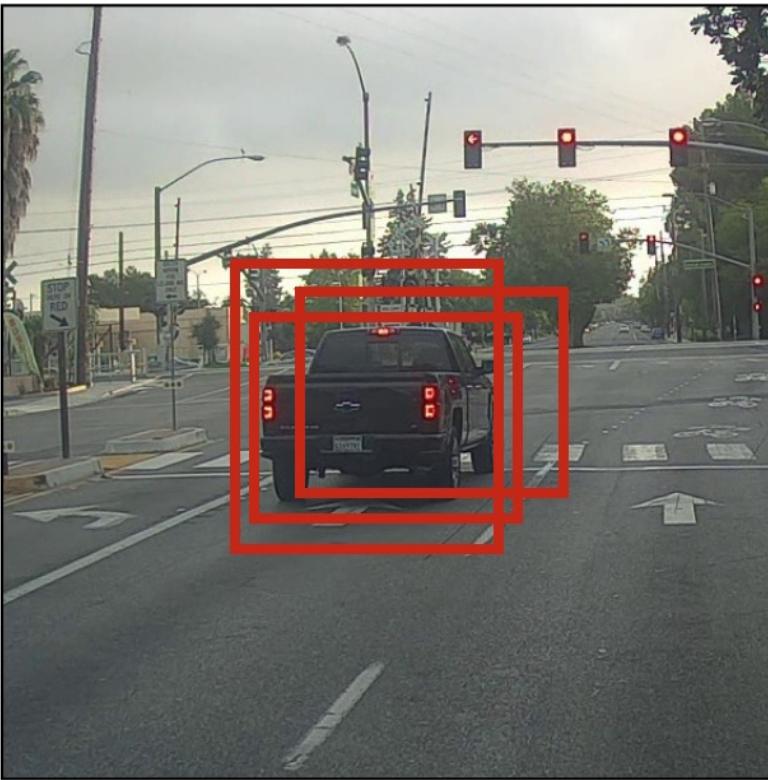
- Перебирать все прямоугольники слишком долго
- Не очень понятно, что такое хороший прямоугольник
- Прямоугольники разного размера, усложняется классификация

Non-maximum suppression

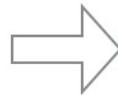
- Модель выдаёт для класса k список прямоугольников с уверенностями
- Проходим в порядке уменьшения уверенности
- Для каждого прямоугольника удаляем все последующие, с которыми Intersection over Union (IoU) > 0.5

Non-maximum suppression

Before non-max suppression



**Non-Max
Suppression**



After non-max suppression

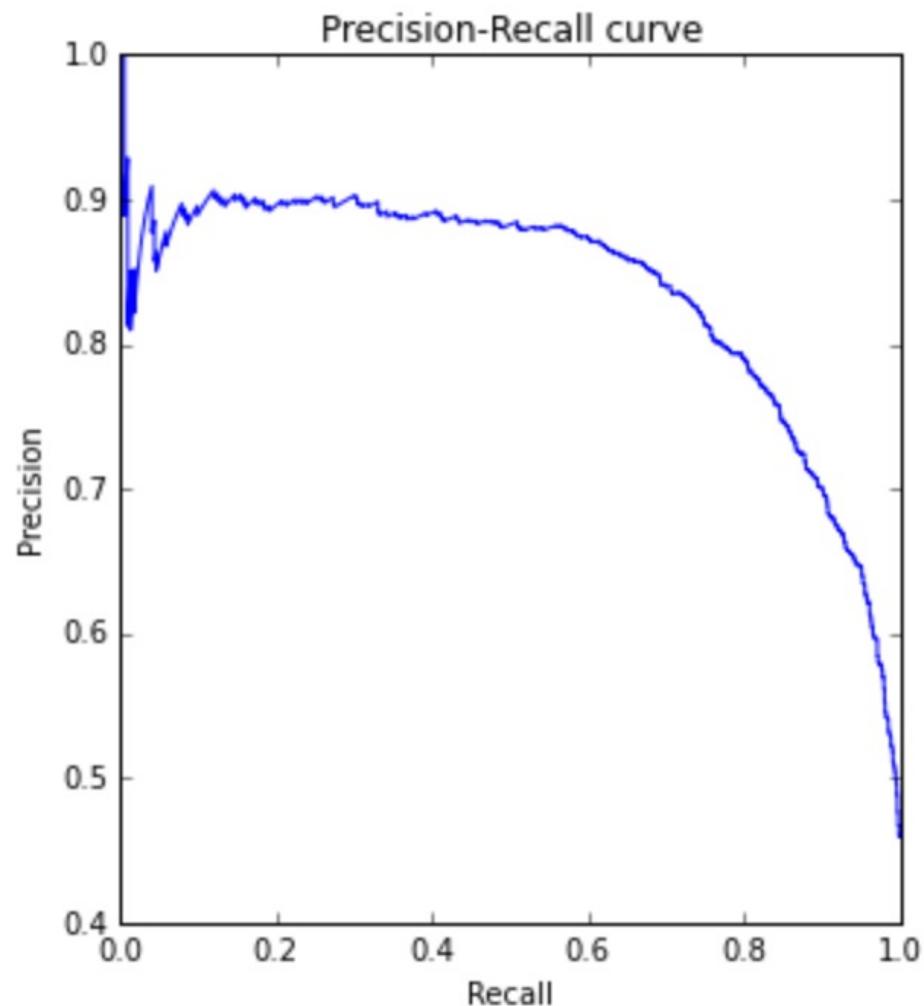


Метрики качества

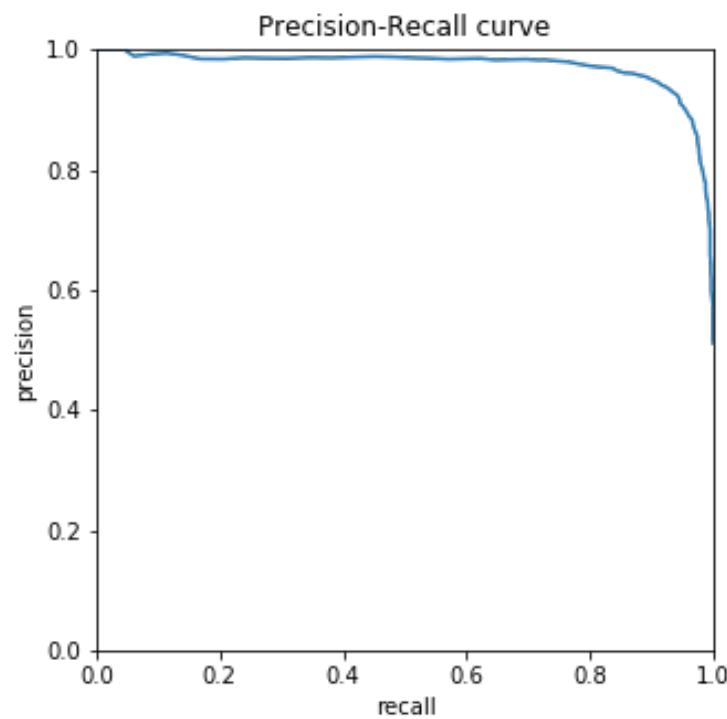
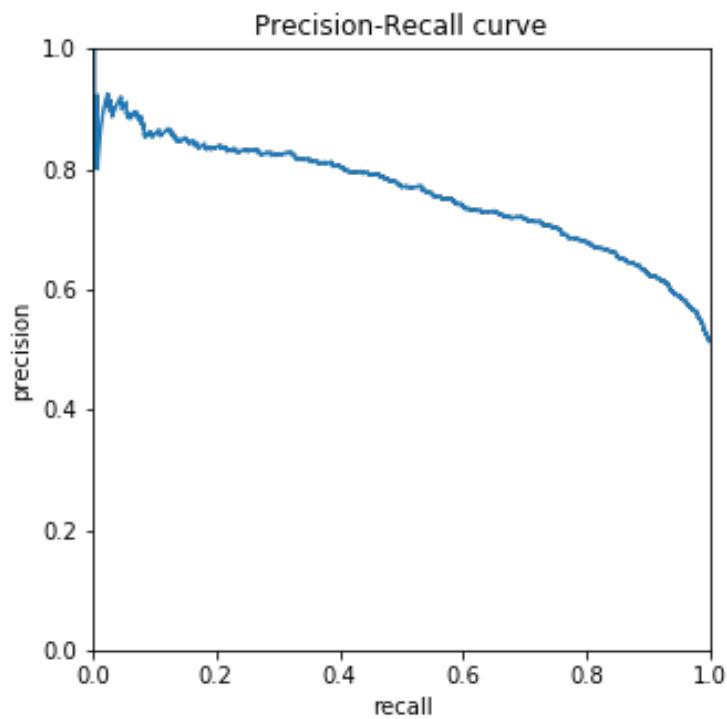
- Модель выдаёт для класса k список прямоугольников с уверенностями
- Считаем прямоугольник корректным, если
$$\text{IoU}(y, z) = \text{Jaccard}(y, z) > t$$
- Строим PR-кривую, считаем под ней площадь, получаем Average Precision
- Усредняем по всем классам, получаем mAP (mean AP)
- Раньше $t = 0.5$,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



Two-shot detection

Будем решать задачу в два этапа:

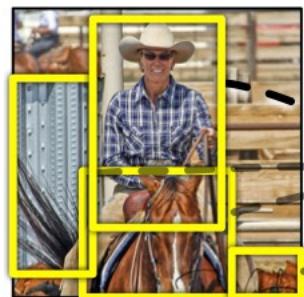
1. Находим «кандидатов» — прямоугольники, где скорее всего что-то есть
2. Классифицируем эти прямоугольники

R-CNN

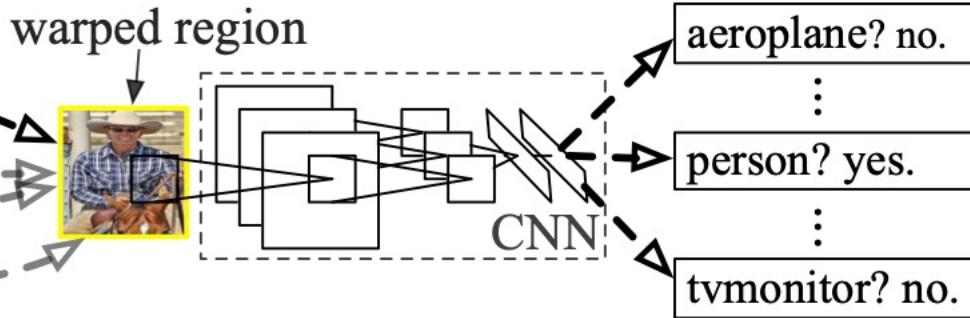
R-CNN: *Regions with CNN features*



1. Input
image



2. Extract region
proposals (~2k)



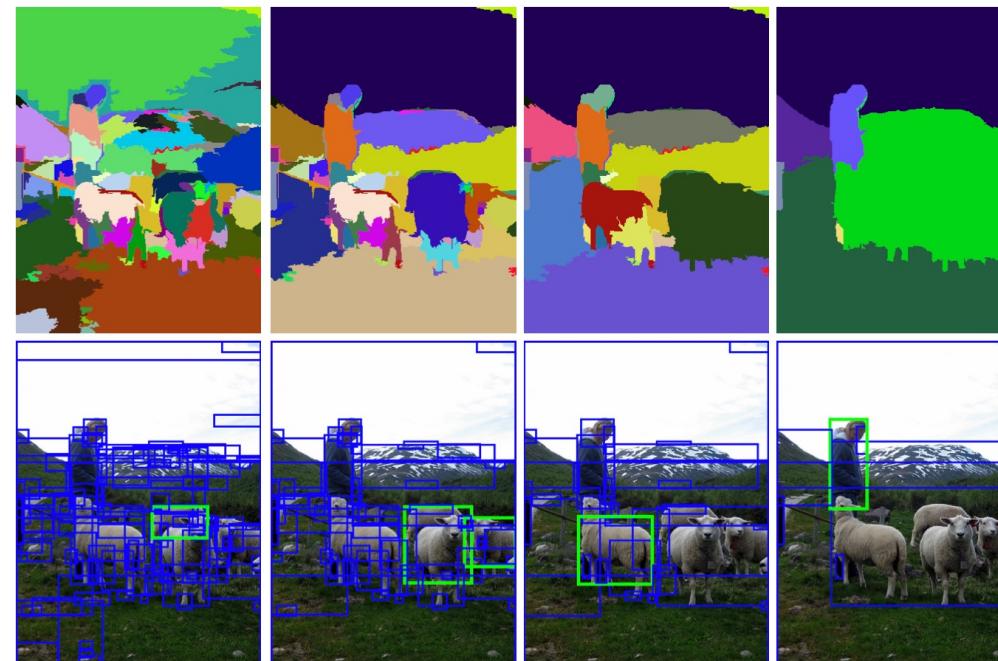
3. Compute
CNN features

4. Classify
regions

R-CNN

Генерация кандидатов:

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



R-CNN

Извлечение признаков:

- Выходы предпоследнего слоя AlexNet (4096 чисел)

R-CNN

Классификация прямоугольников:

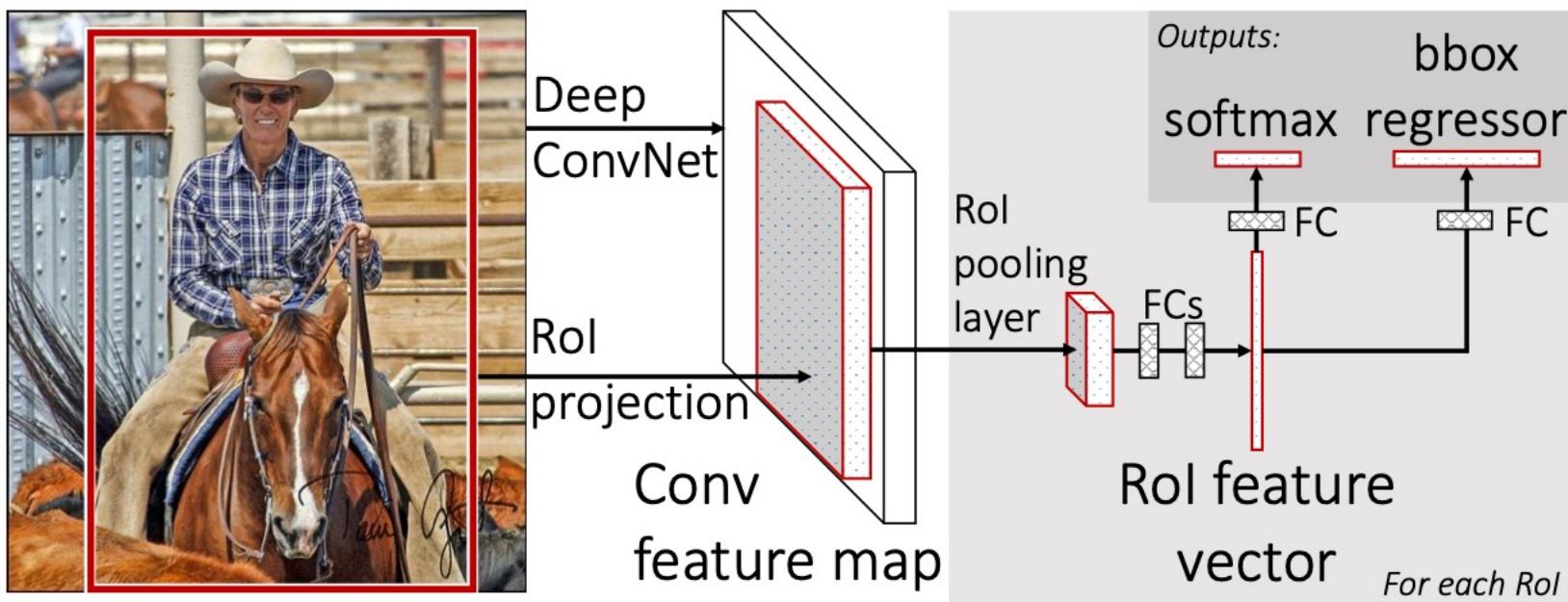
- SVM
- One-vs-all

R-CNN

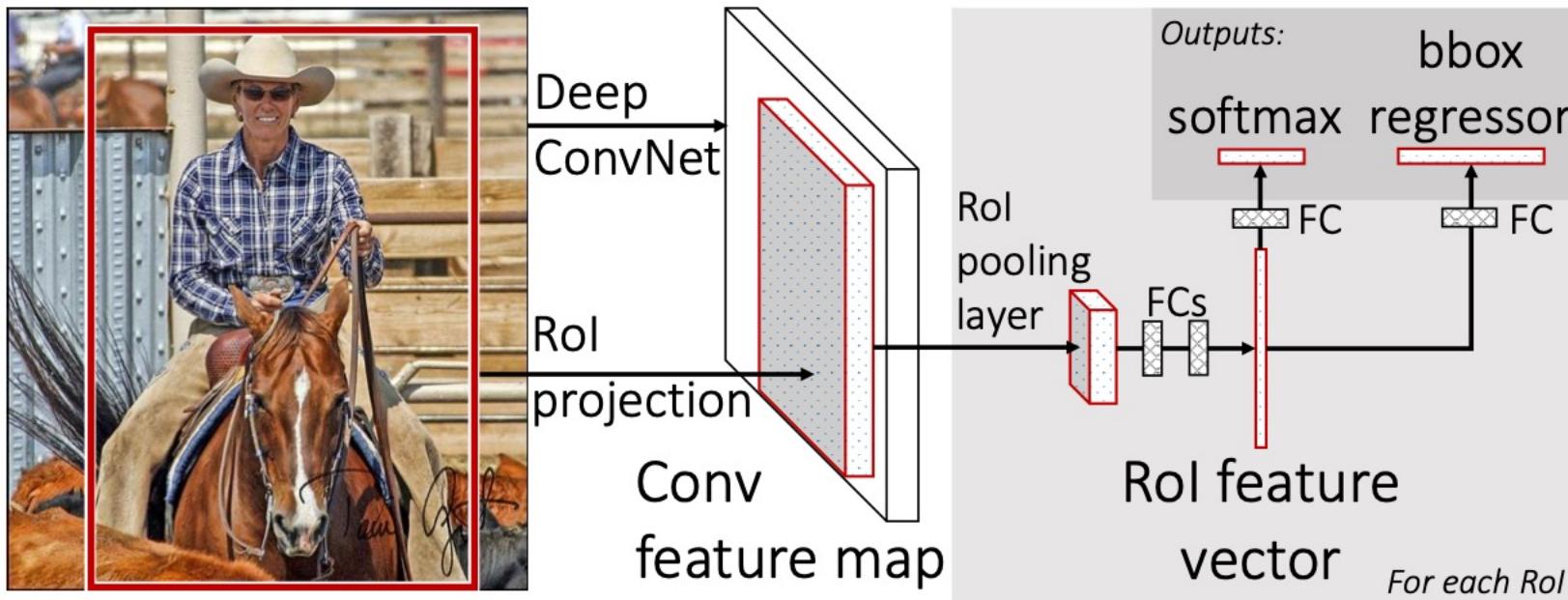
Проблемы:

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

Fast R-CNN

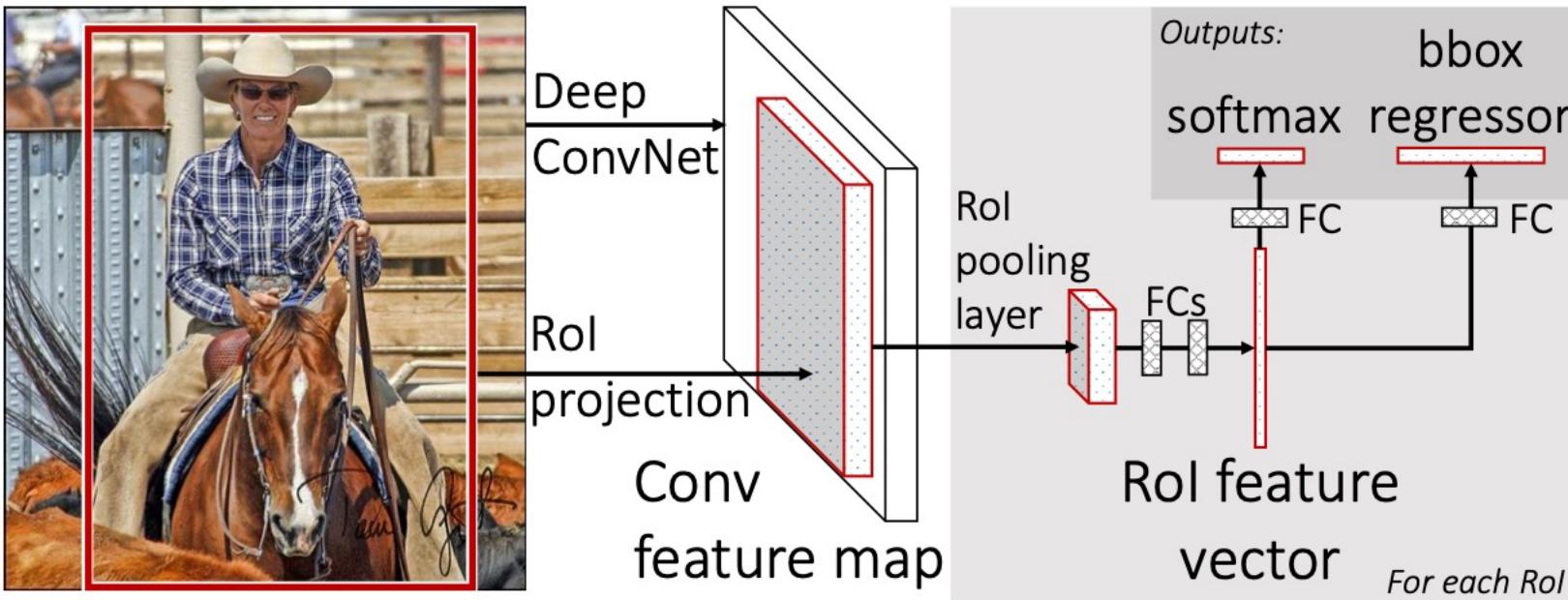


Fast R-CNN



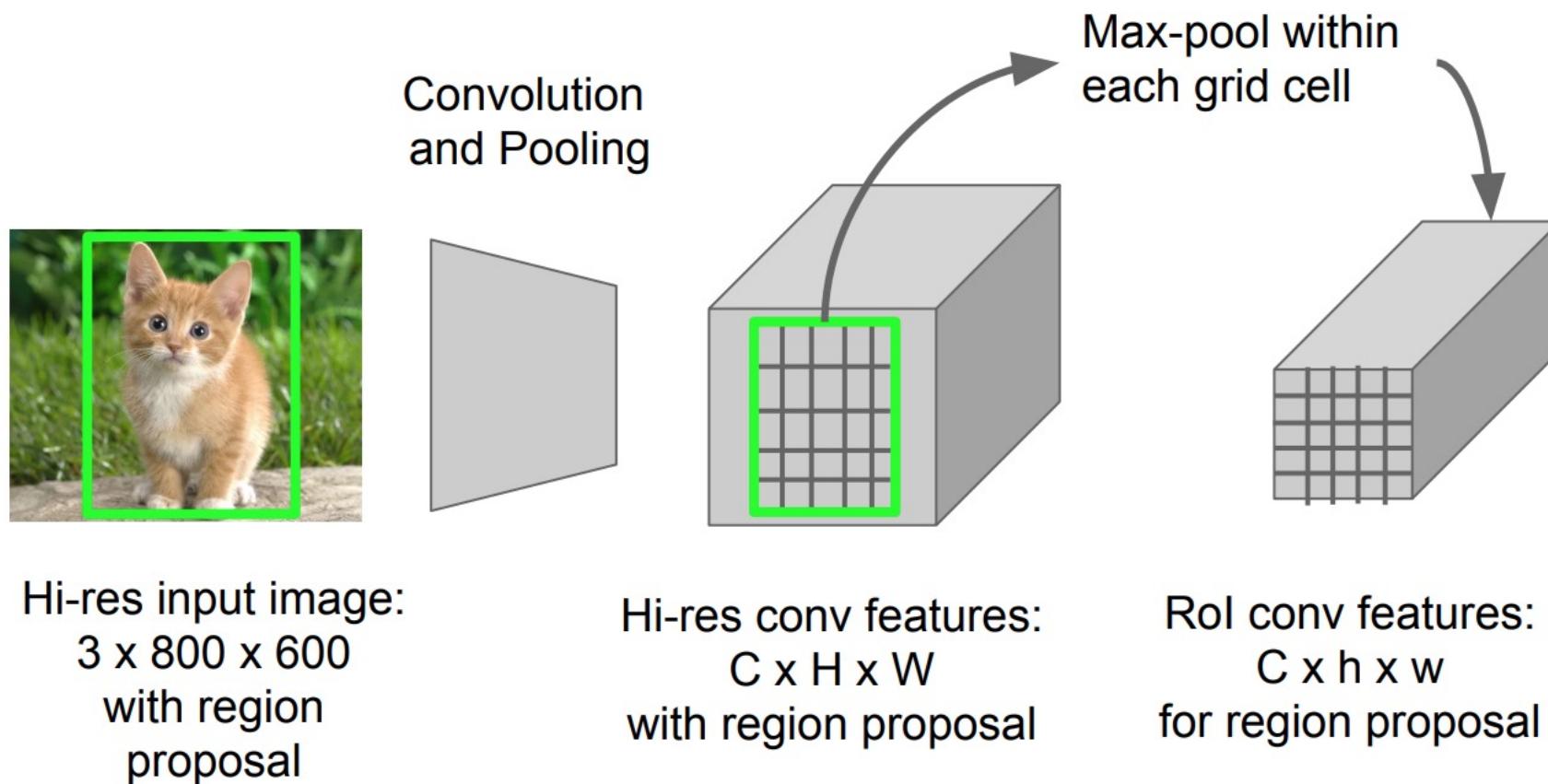
- Кандидаты всё ещё генерируются внешним методом

Fast R-CNN

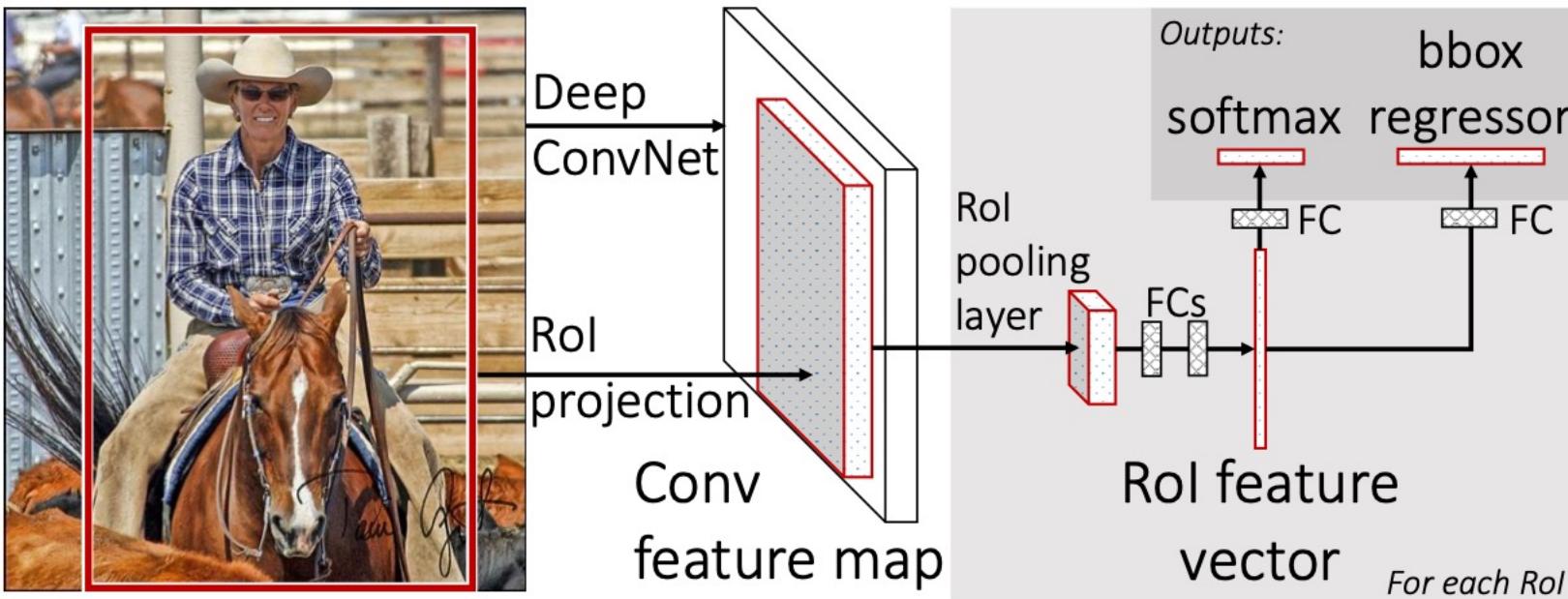


- Для конкретного кандидата извлекаются признаки: вырезаются участки из последнего тензора, режутся на блоки, из каждого блока берётся максимум

RoI Pooling



Fast R-CNN



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

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

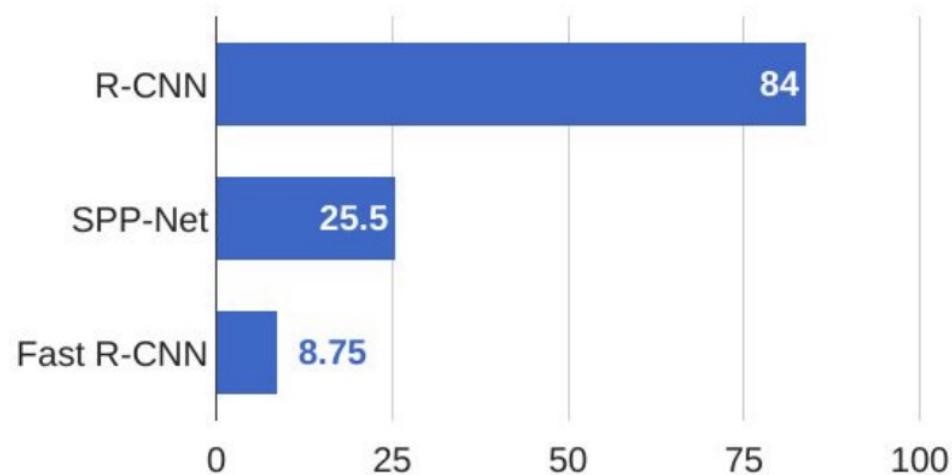
$$L_{\text{loc}}(t^u, v) = \sum_{i \in \{\text{x}, \text{y}, \text{w}, \text{h}\}} \text{smooth}_{L_1}(t_i^u - v_i),$$

in which

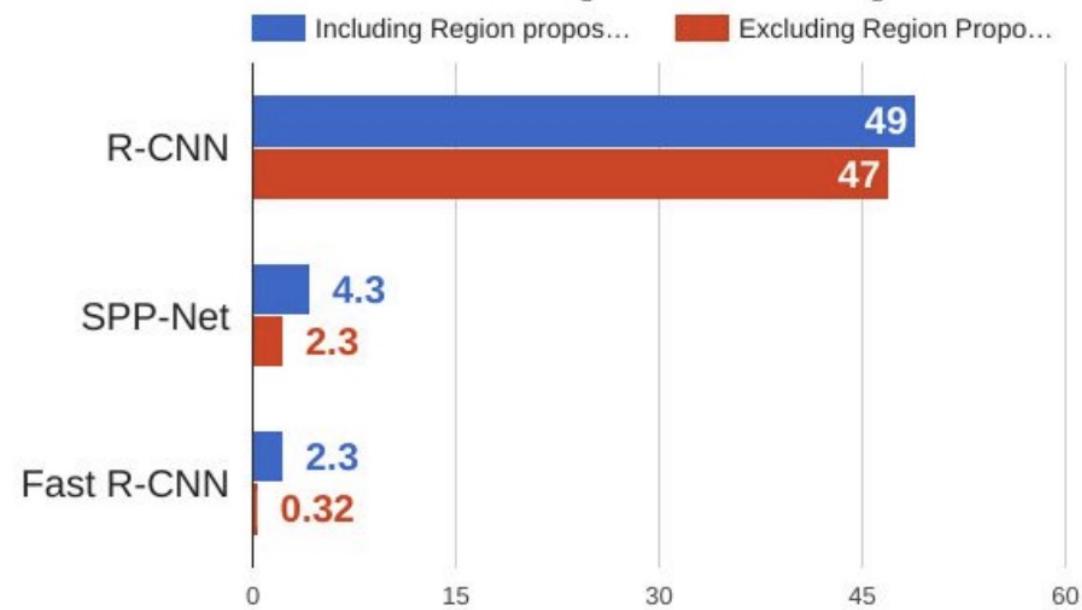
$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}$$

Fast R-CNN

Training time (Hours)



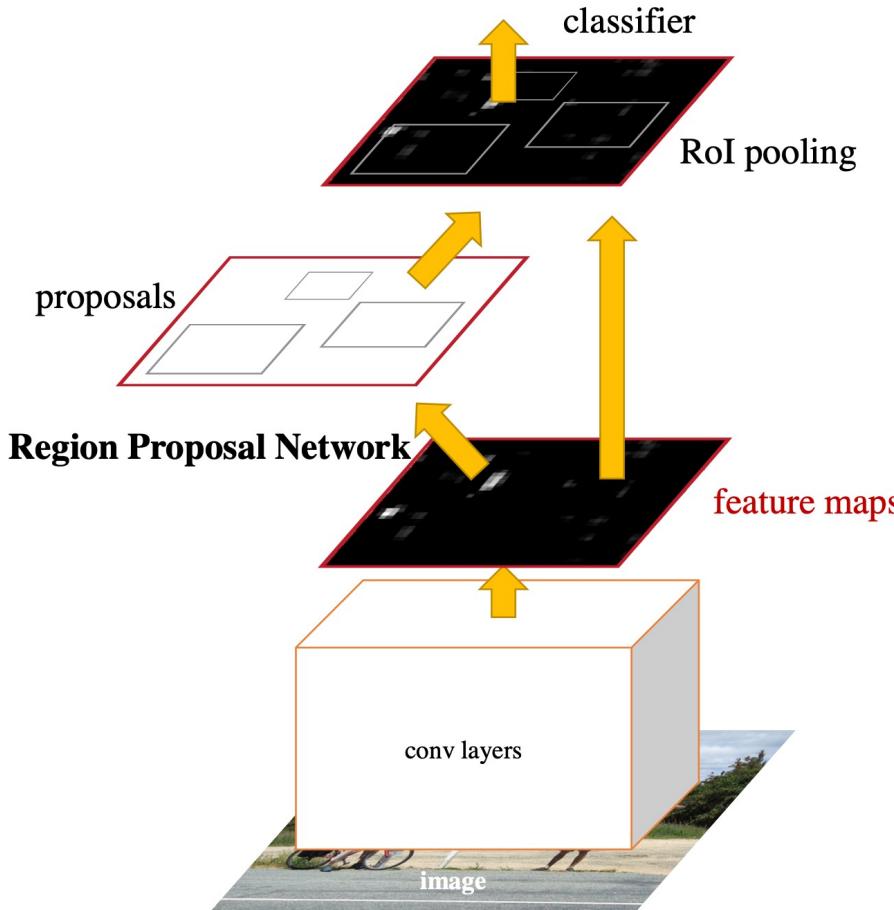
Test time (seconds)



Fast R-CNN

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

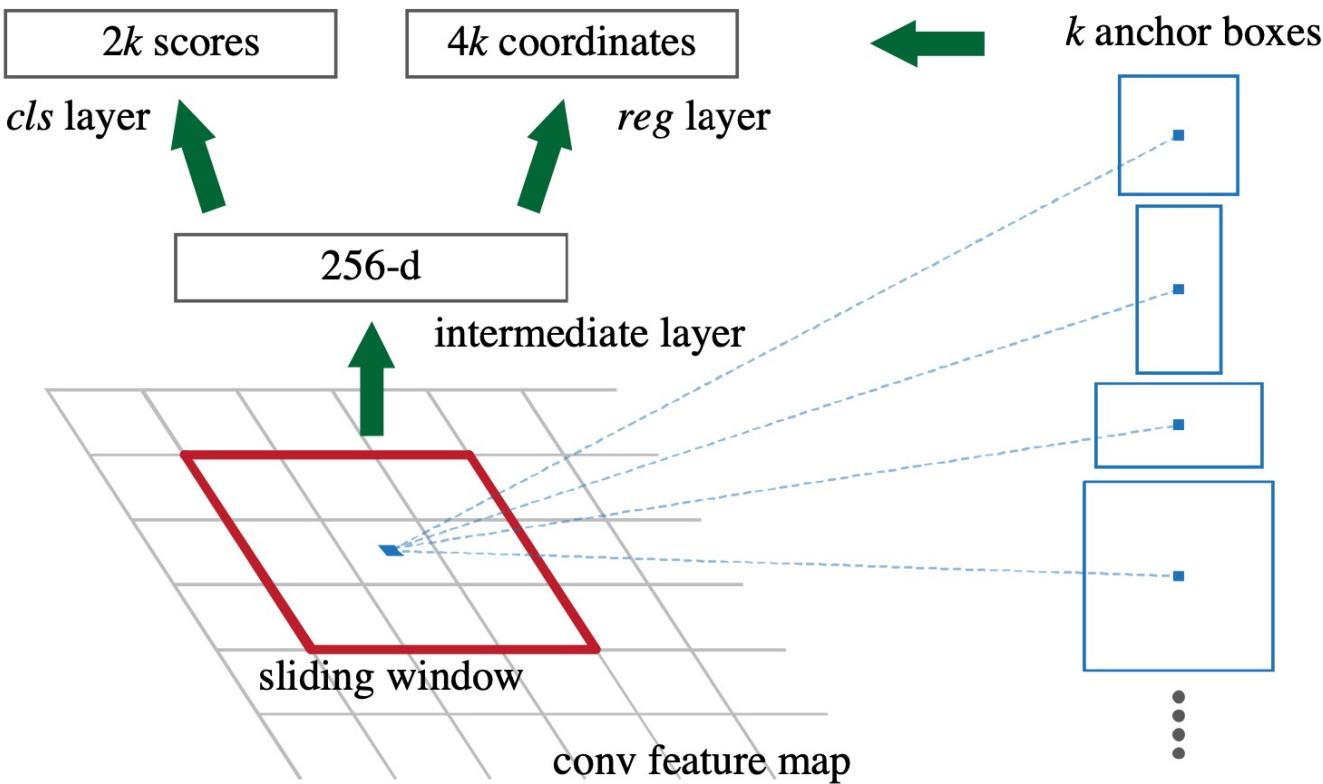
Faster R-CNN



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



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

Faster R-CNN

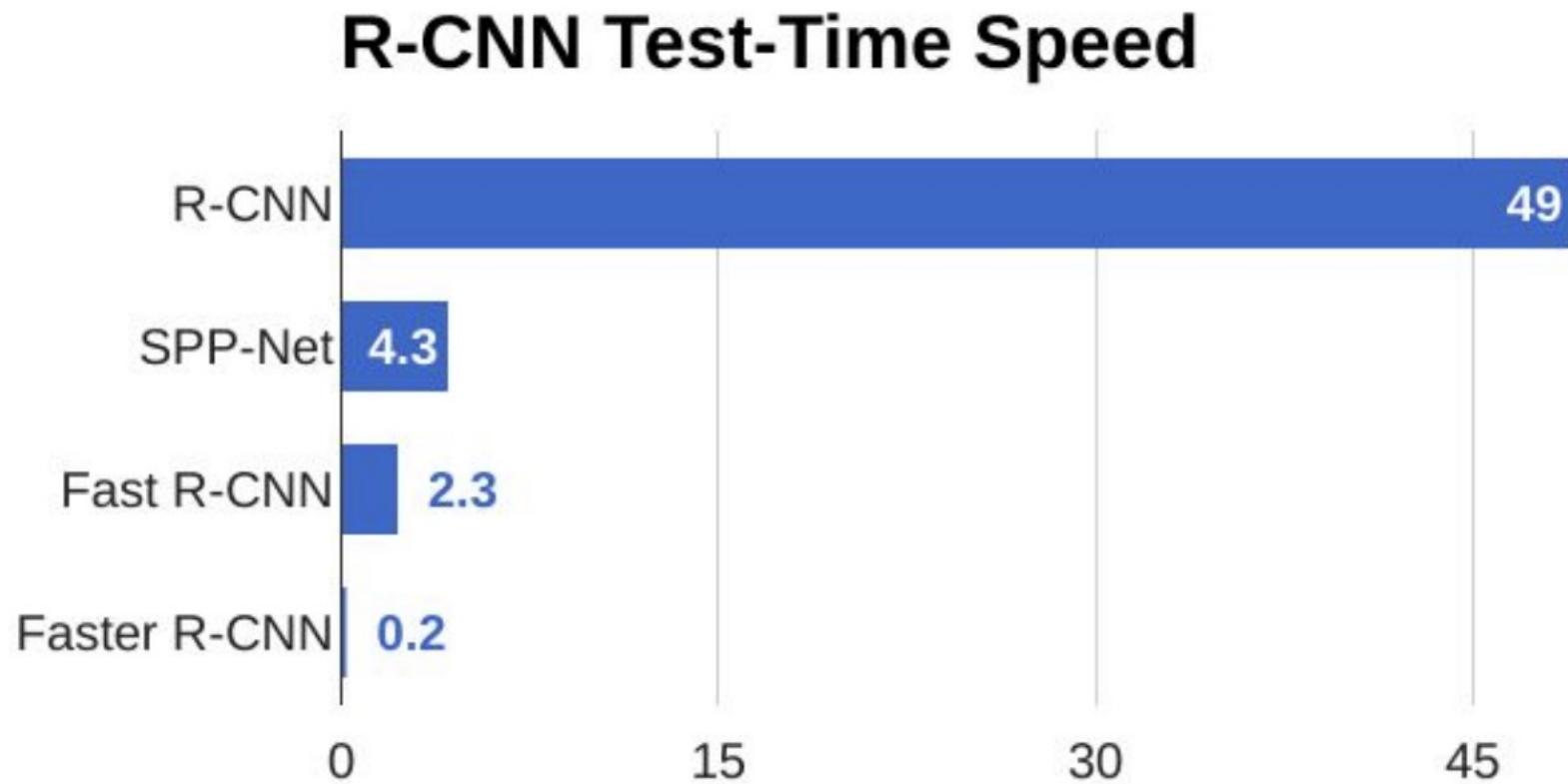
На этапе применения:

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

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

Faster R-CNN



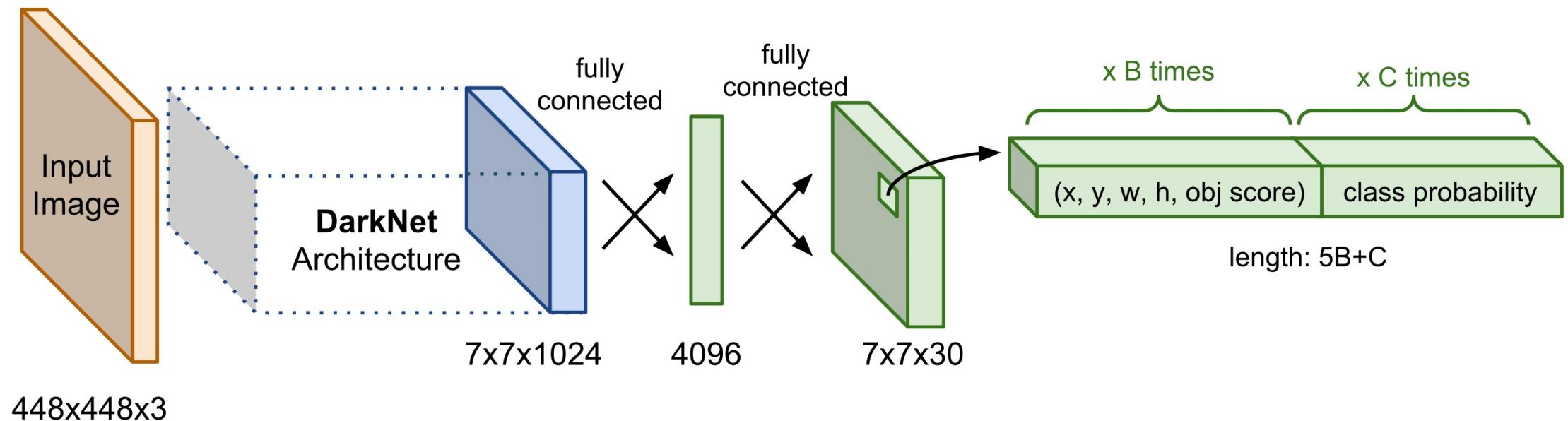
Что ещё?

- Mask R-CNN — instance segmantation

One-shot detection

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

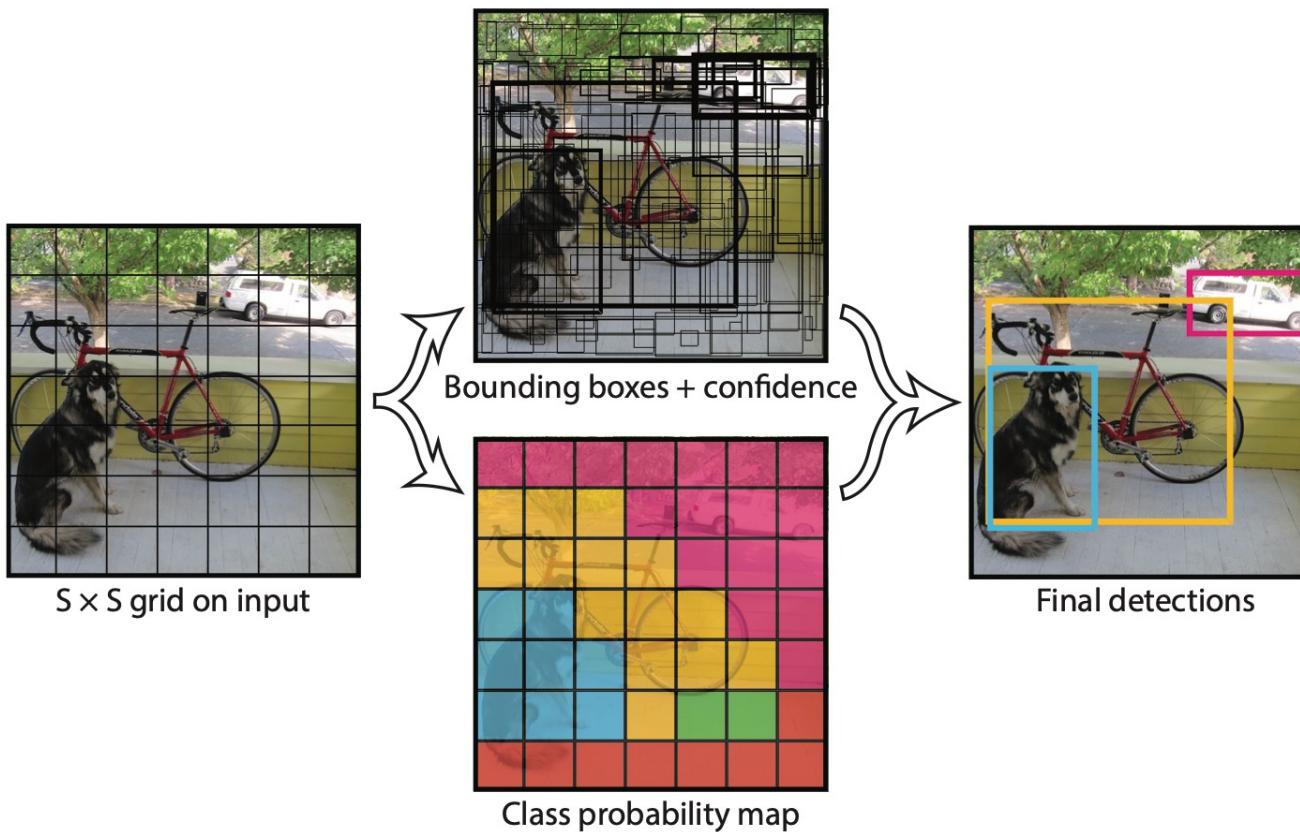
YOLO



YOLO

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

YOLO



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

Что ещё?

- SSD (Single Shot Detector)
- RetinaNet
- YOLOv3

Идентификация объектов

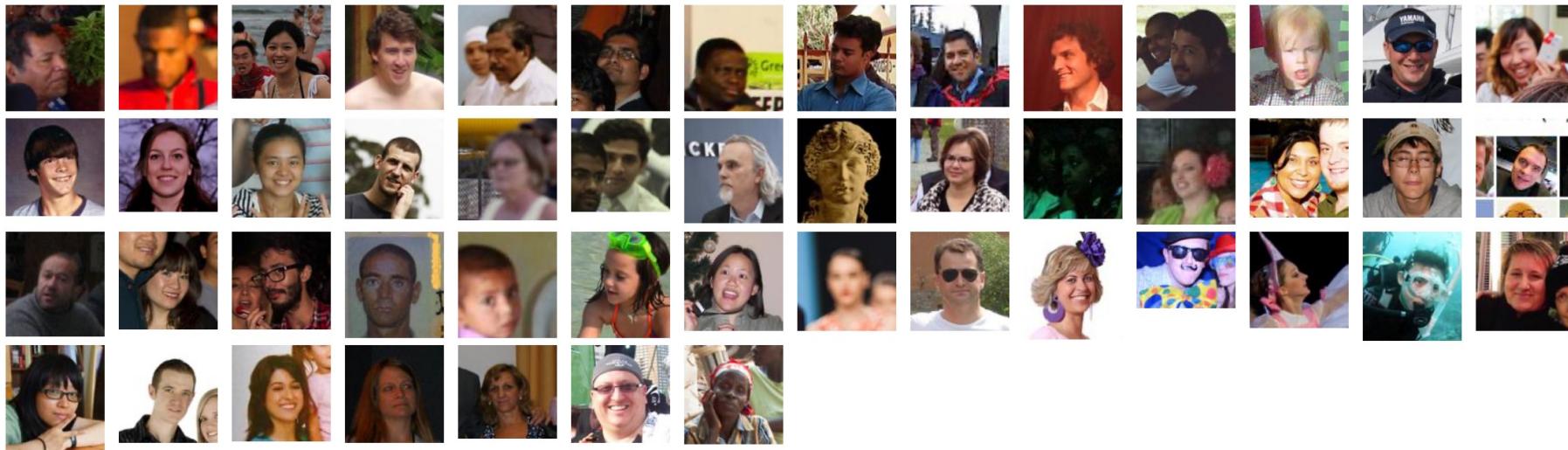
Labeled Faces in the Wild

- Около 13 тысяч фотографий
- Около 6 тысяч человек

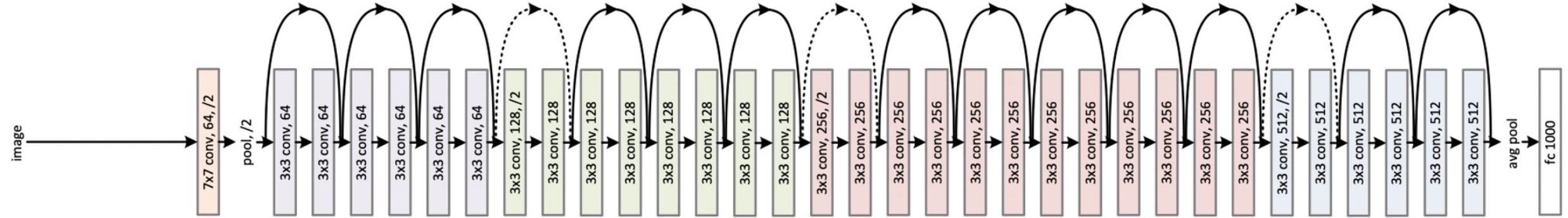


MegaFace

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



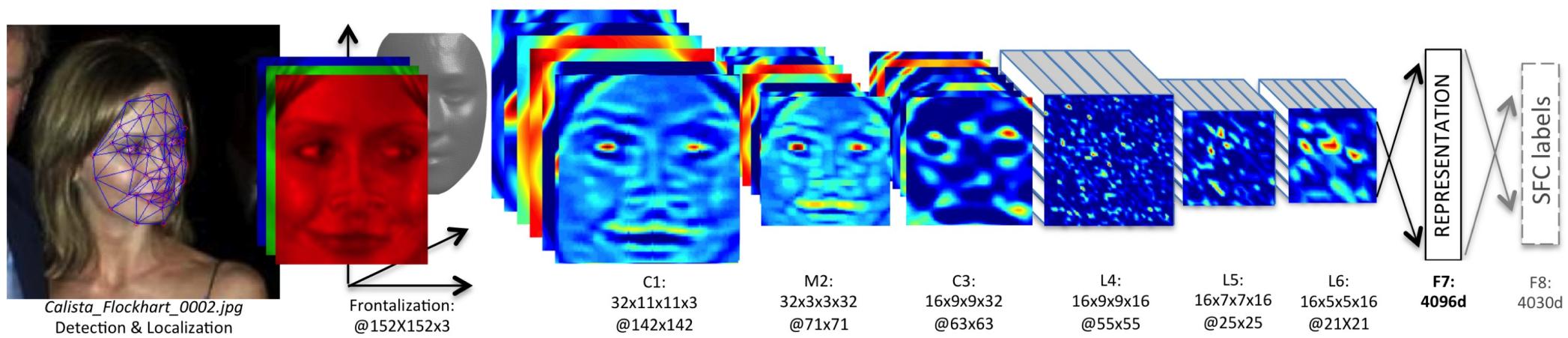
Дообучение



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



DeepFace

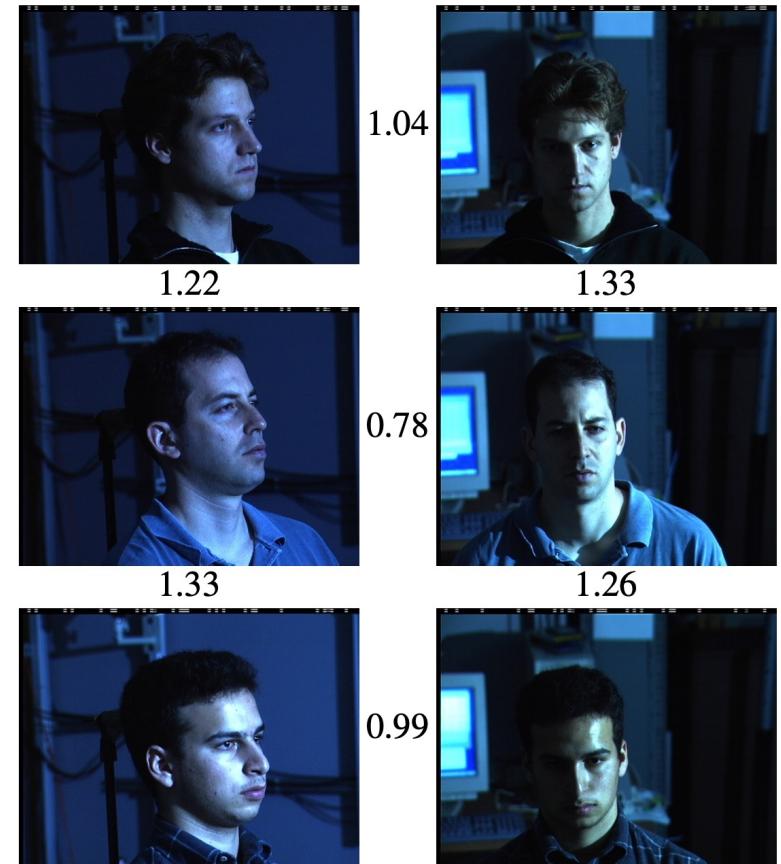
- Обучаем некоторую архитектуру для классификации (число классов = число людей в данных)
- Используем выходы предпоследнего слоя как признаковое описание изображения
- Признаки нормализуются (чтобы норма была единичной)
- Считаем близость векторов по какой-нибудь метрике

DeepFace

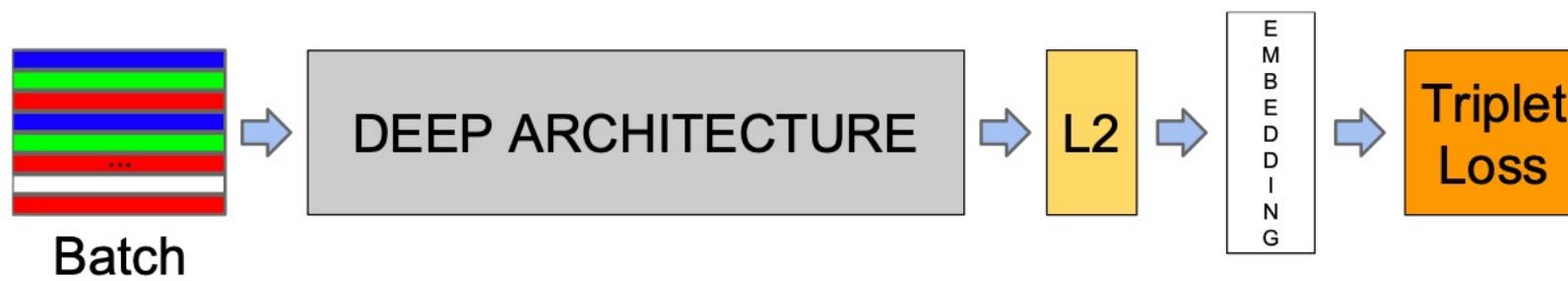
- Можно сравнить расстояние с порогом, чтобы идентифицировать человека

FaceNet

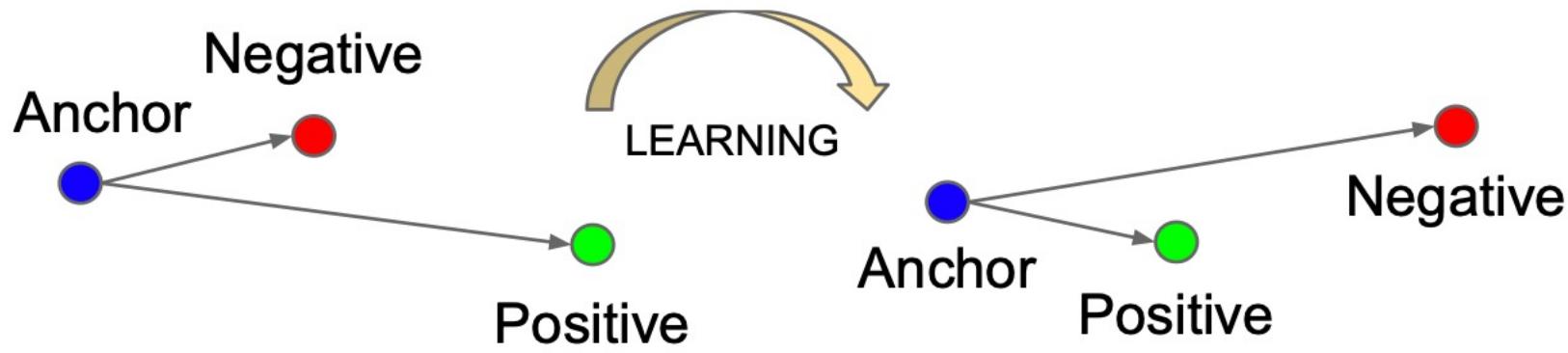
- Почему бы в явном виде не обучать представления изображений так, чтобы фотографии одного человека имели близкие представления?



FaceNet



FaceNet



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

FaceNet

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

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

Триплетная и попарная ошибки

- Попарная ошибка:

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

- Не совсем про обучение расстояния

FaceNet

- Точность на LFW: 99.63%