

# Глубинное обучение

Лекция 9  
Детектирование объектов

Михаил Гущин

[mhushchyn@hse.ru](mailto:mhushchyn@hse.ru)

НИУ ВШЭ, 2024



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
УНИВЕРСИТЕТ

В предыдущей серии

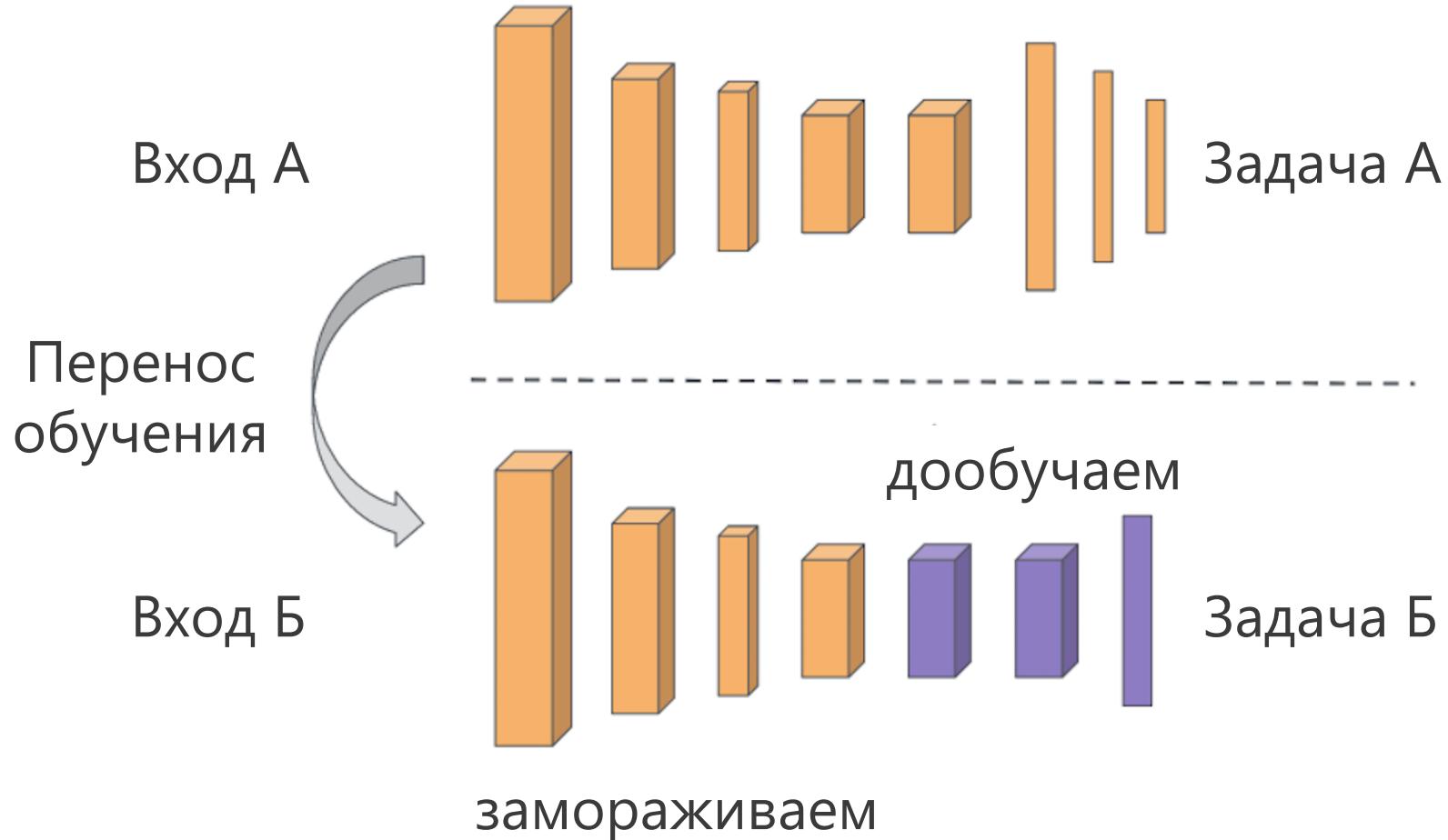
# ImageNet



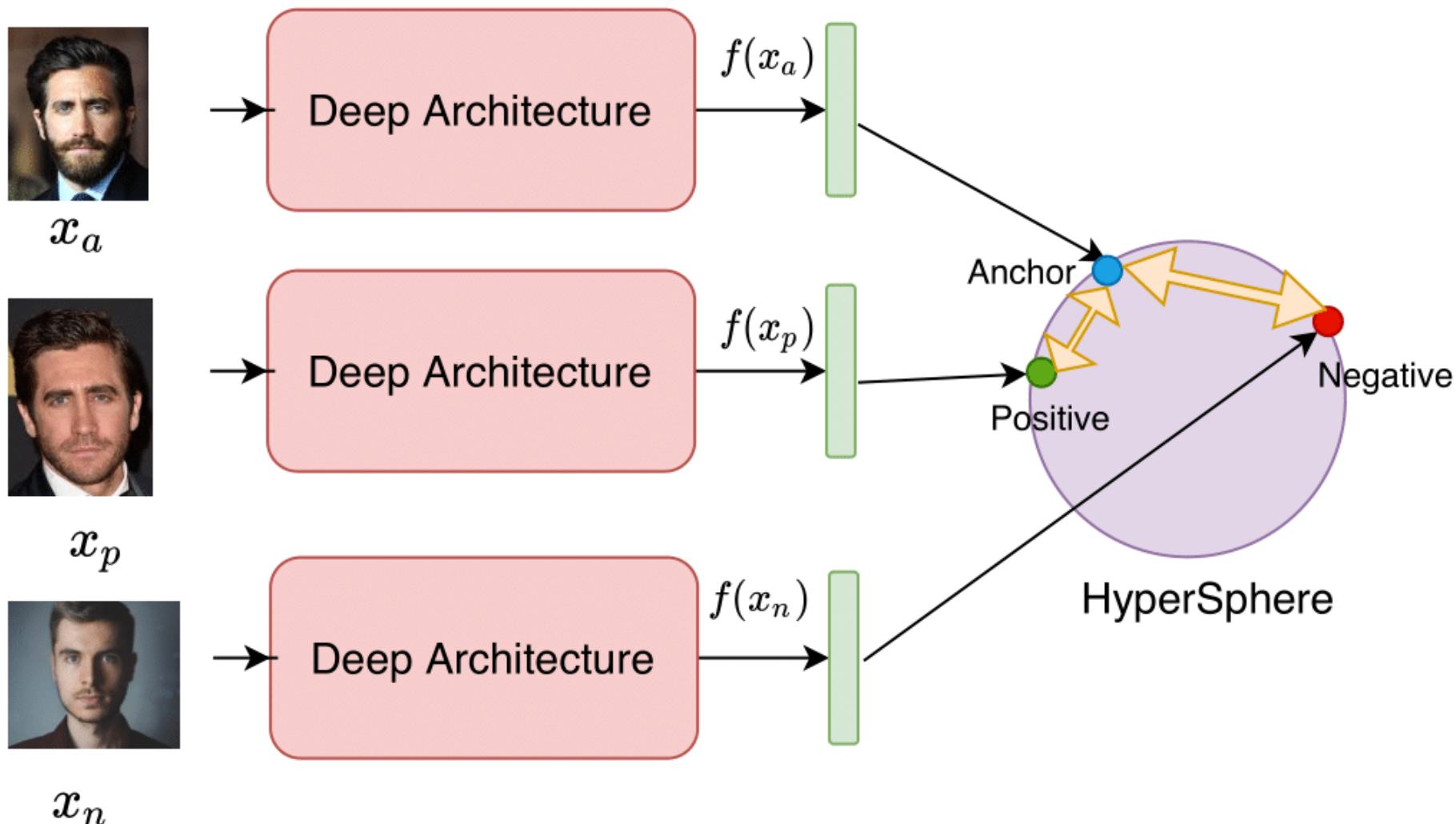
Подробнее: <https://image-net.org/challenges/LSVRC>

- ▶ ImageNet Large Scale Visual Recognition Challenge (ILSVRC)
- ▶ 1000 классов изображений
- ▶ Более 1 000 000 изображений

# Перенос обучения (fine-tuning)



# Триплет лосс



Источник: <https://pyimagesearch.com/2023/03/06/triplet-loss-with-keras-and-tensorflow/>

# Задачи компьютерного зрения

# Классификация изображений



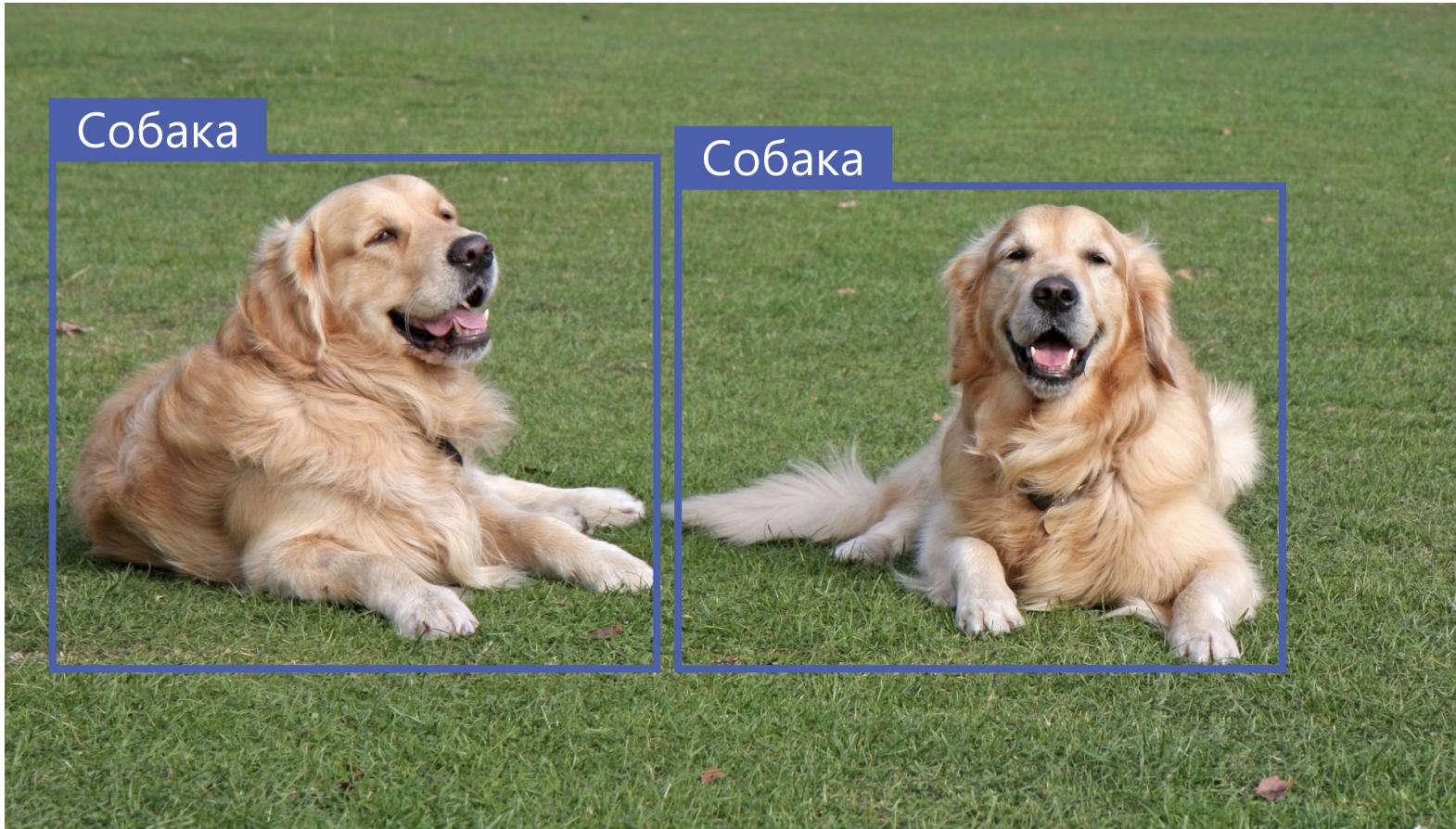
«Кот»

# Классификация и локализация

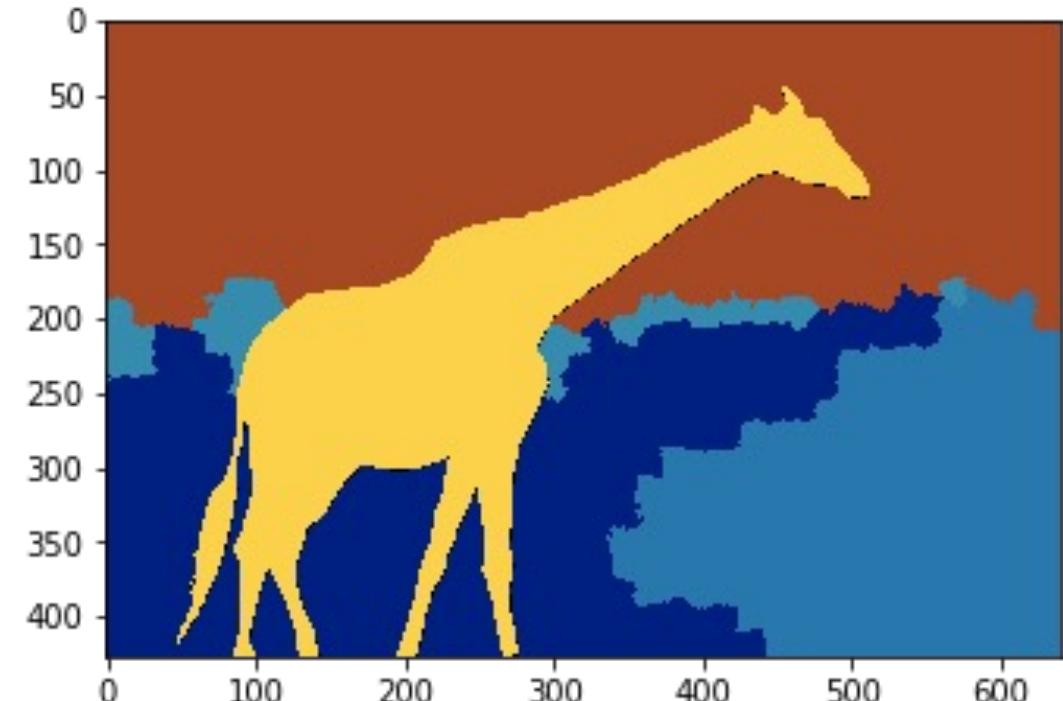
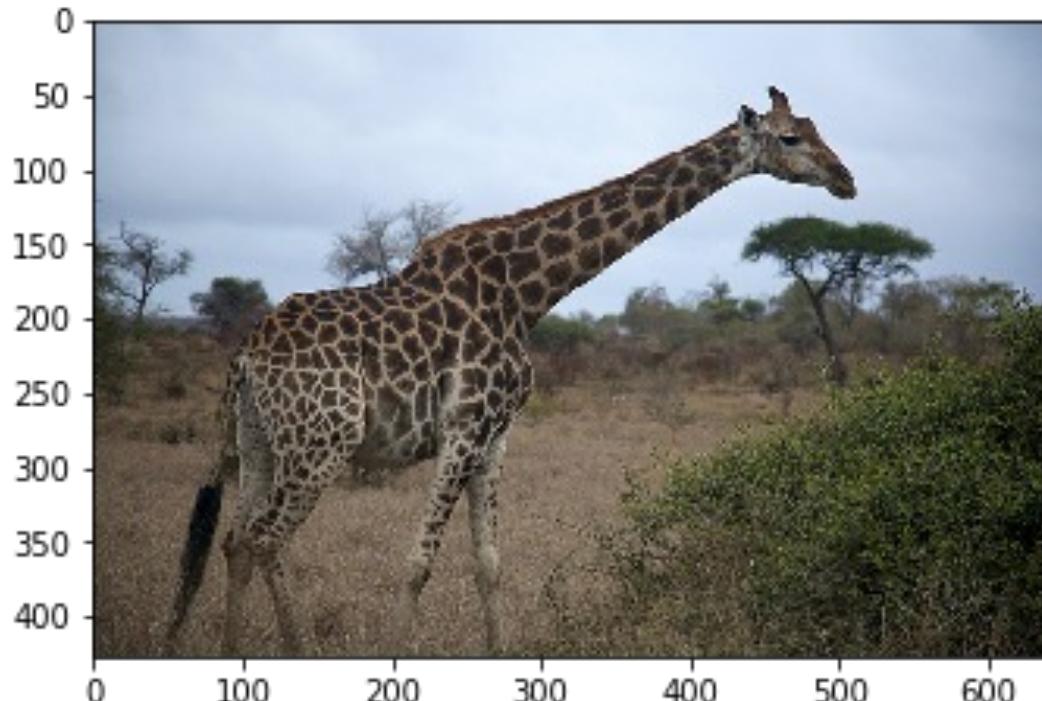


«Кот»

# Детектирование объектов



# Семантическая сегментация



[mxnet.apache.org](http://mxnet.apache.org)

# Локализация объекта

# Классификация

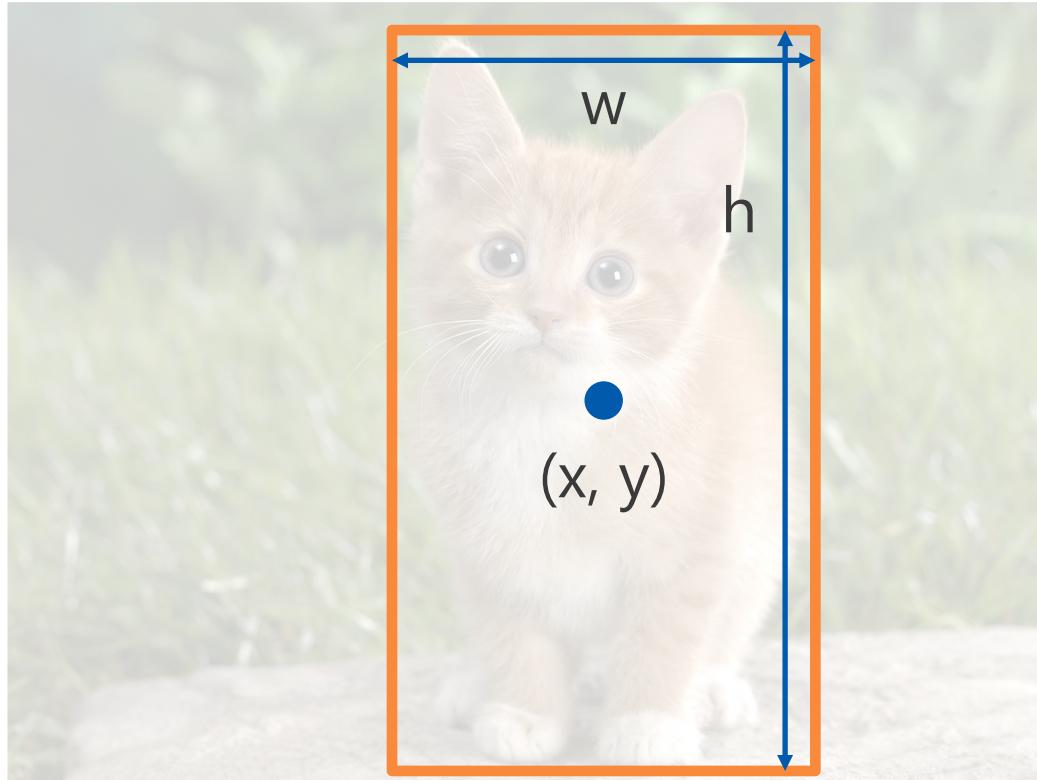


# Классификация и локализация



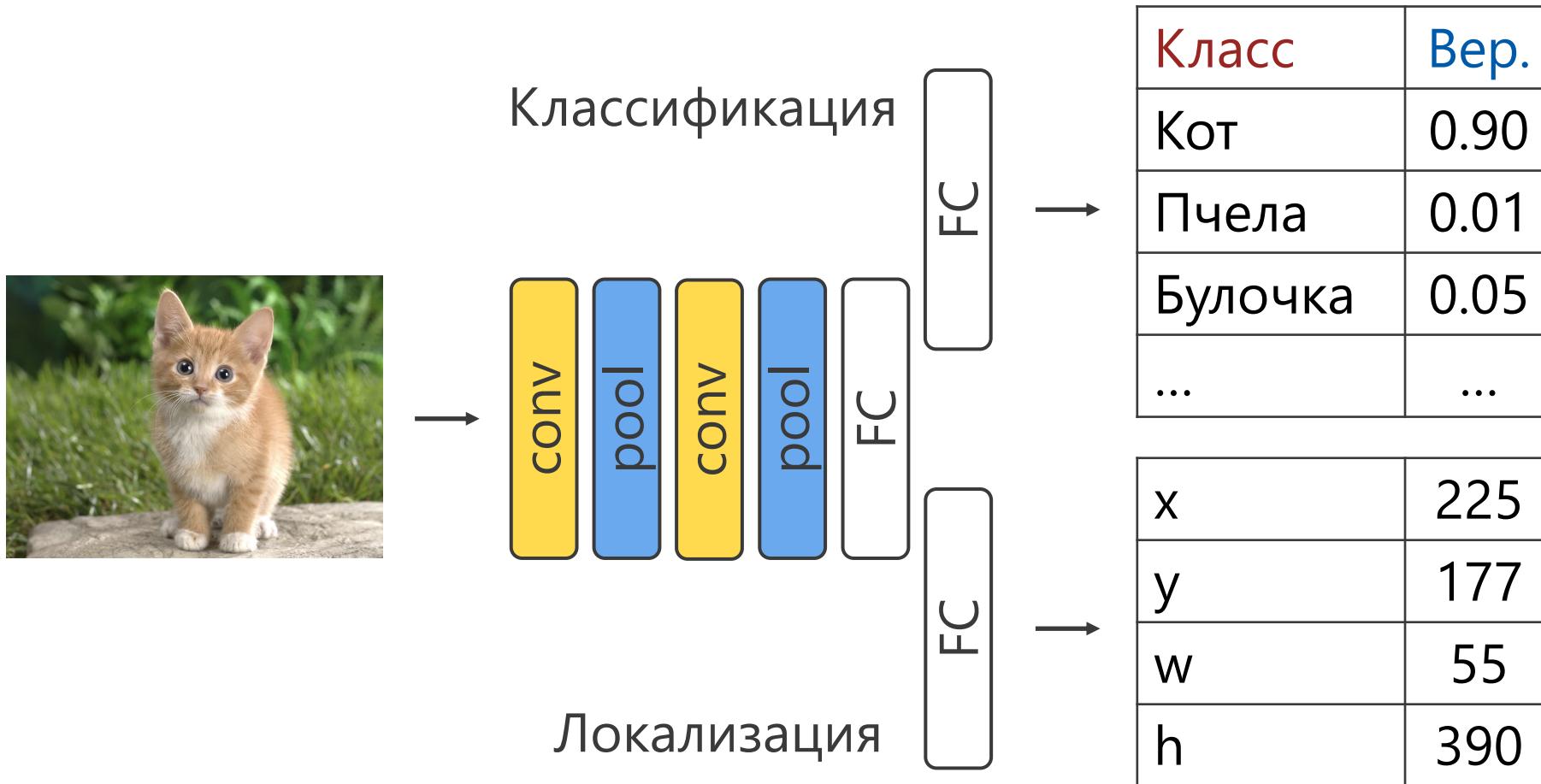
«Кот»

# Классификация и локализация



Класс: «Кот»  
Прямоугольник:  $(x, y, w, h)$

# Классификация и локализация



# Перенос обучения

- ▶ За основу берем сеть, обученную на ImageNet (VGG-16)
- ▶ Добавляем голову для локализации
- ▶ Дообучаем последние слои сети

# Функция потерь

Многозадачная функция потерь  $L$ :

$$L = \textcolor{brown}{L}_{\text{классификация}} + \alpha \textcolor{blue}{L}_{\text{локализация}} \rightarrow \min$$

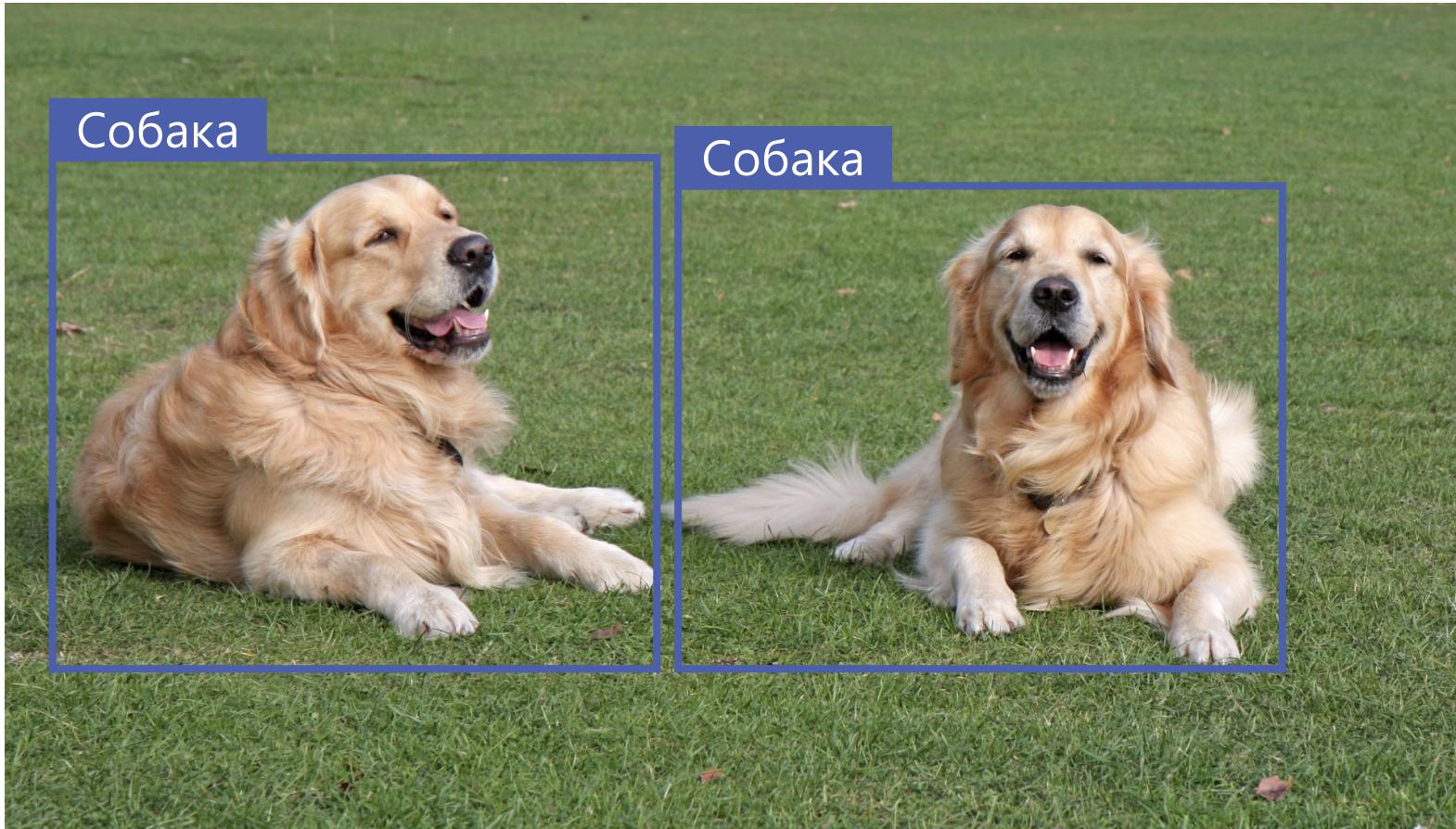
- ▶  $\textcolor{brown}{L}_{\text{классификация}}$  — функция кросс-энтропии для классификации изображения
- ▶  $\textcolor{blue}{L}_{\text{локализация}}$  — MSE функция потерь для параметров прямоугольника



R-CNN



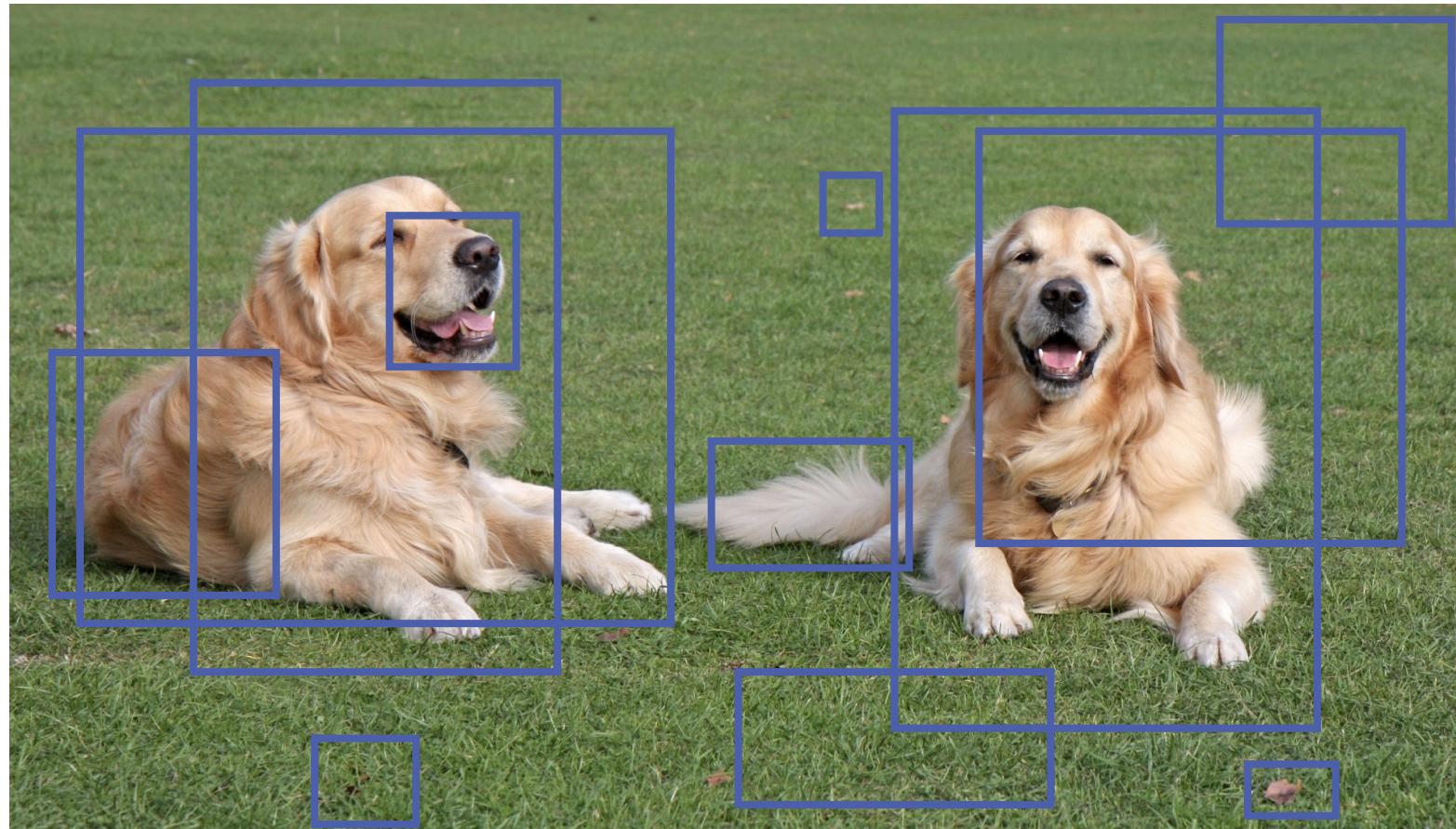
# Детектирование объектов



# Идея R-CNN алгоритма

- ▶ Генерация областей изображения (region proposals)
- ▶ Классификация с локализацией объектов для каждой области
- ▶ Отбор лучших прямоугольников ( $x, y, w, h$ ) (non-maximum suppression)

# Region proposal

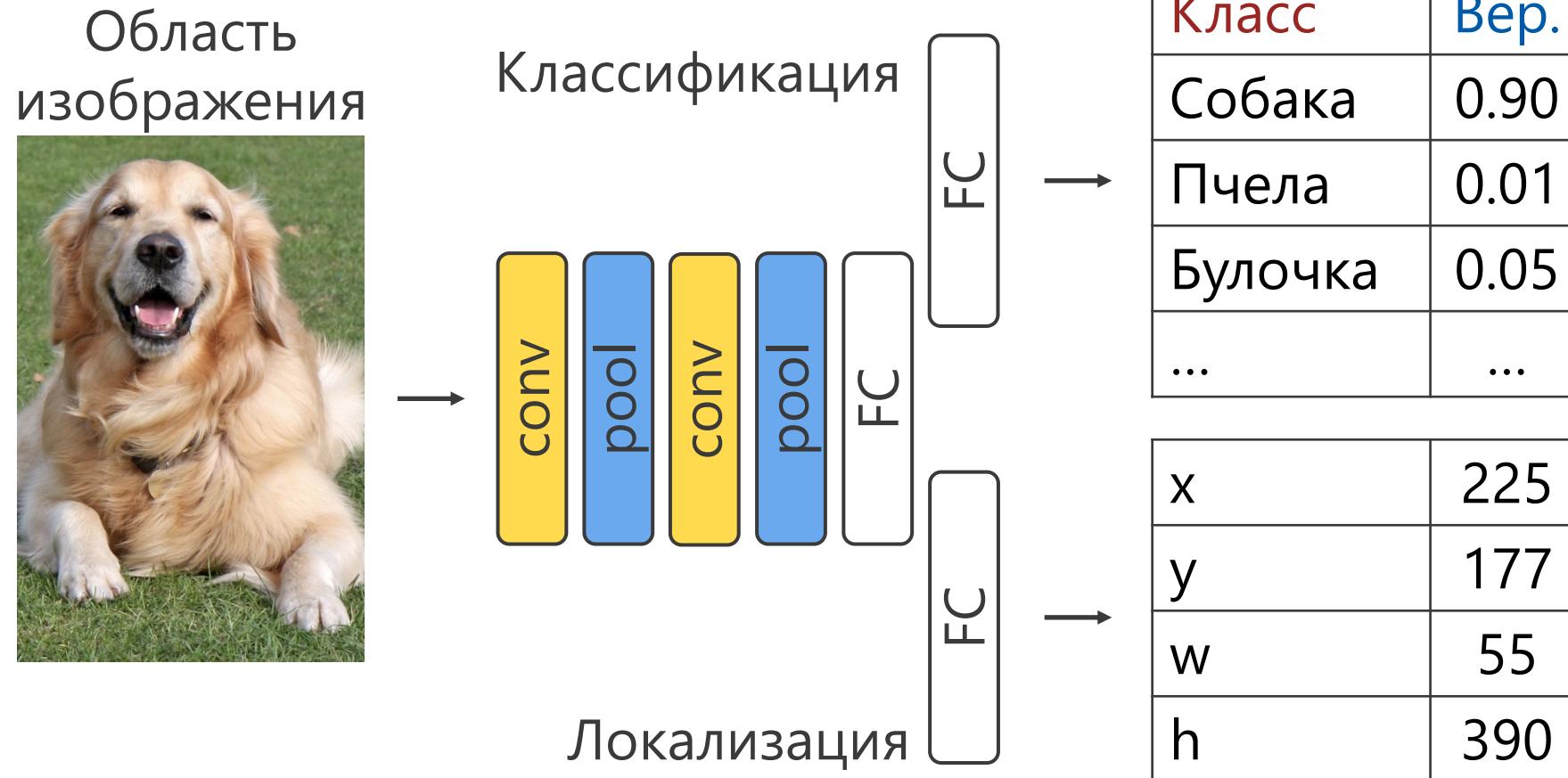


Выбираем около 2000 областей

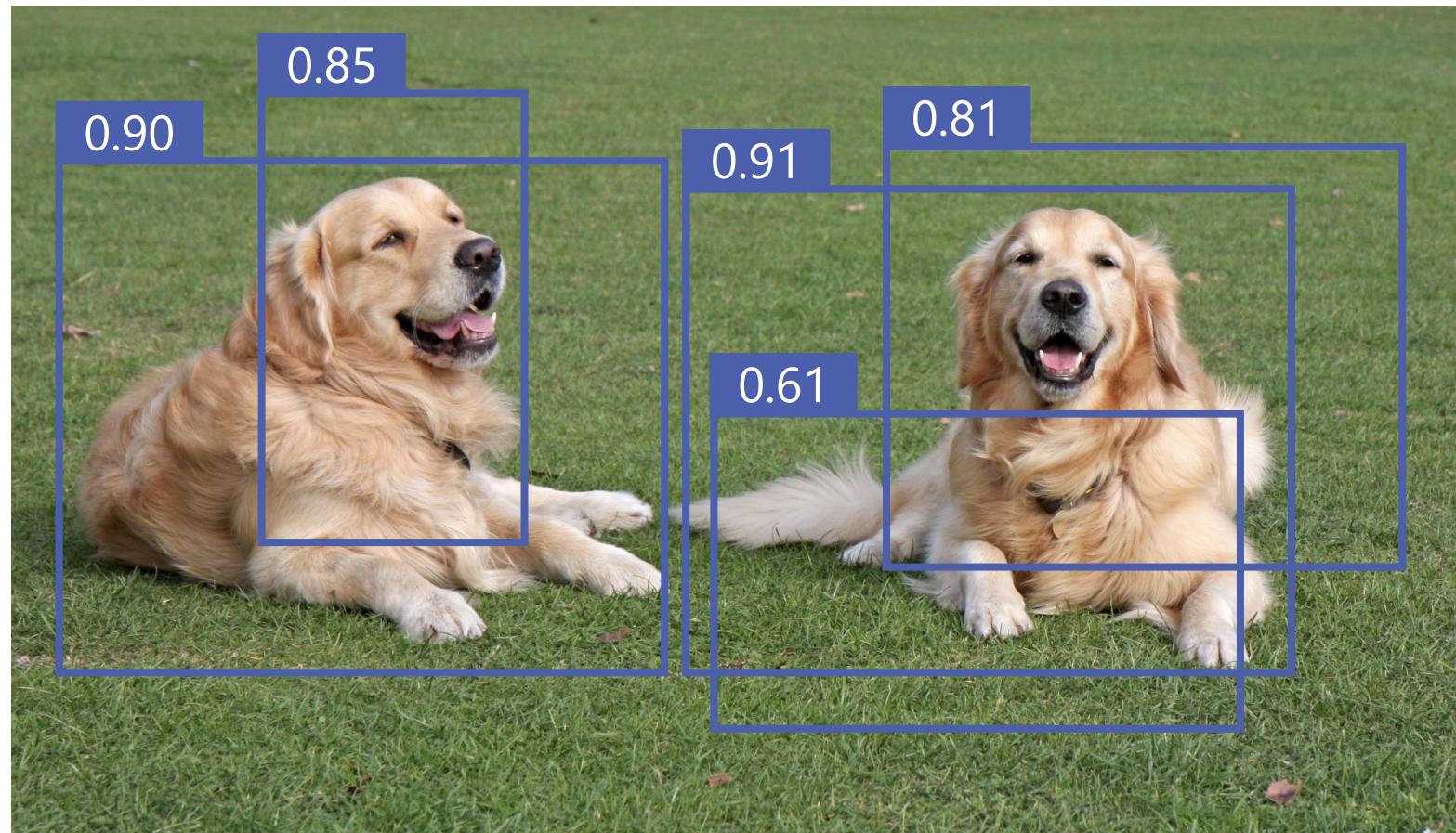
# Region proposal

- ▶ **Sliding window**
  - последовательный перебор всех областей разных размеров;
  - простой, но неэффективный.
- ▶ **Selective search**
  - иерархическое объединение областей по цвету, текстуре, размеру и форме;
  - популярный и эффективный алгоритм.

# Классификация + локализация для регионов

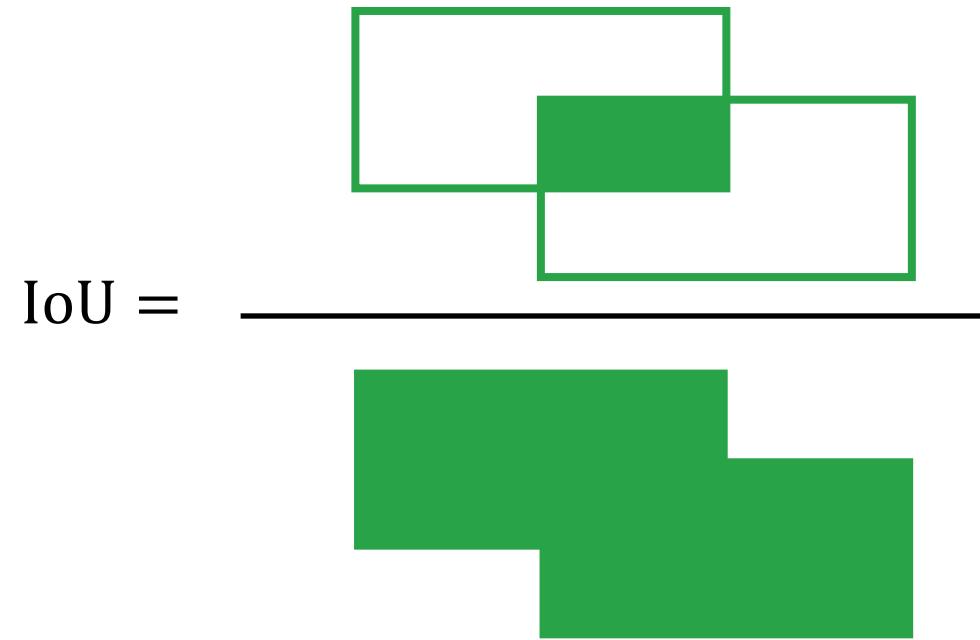


# Классификация + локализация для регионов



Как выбрать нужные прямоугольники?

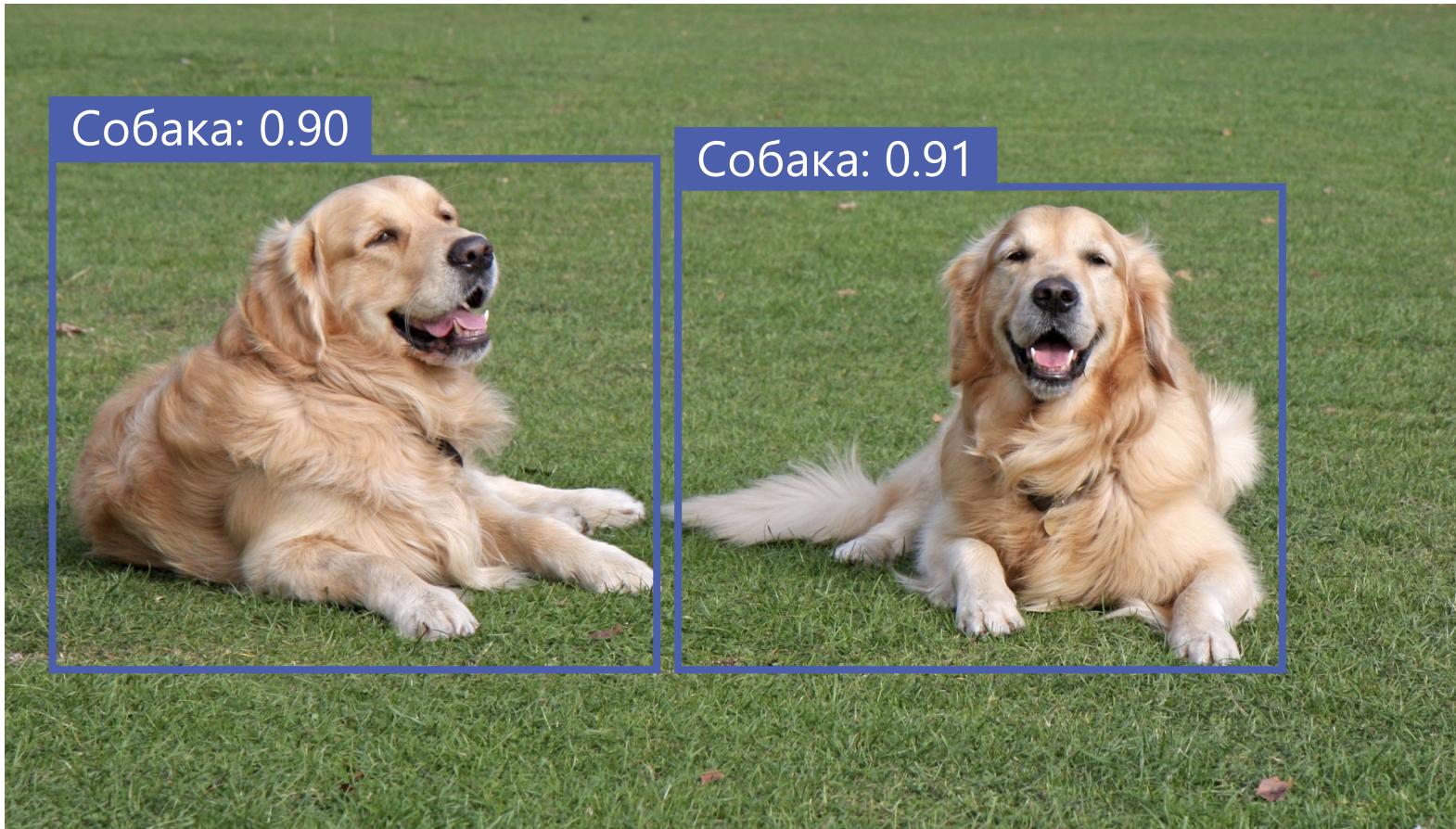
# Intersection over Union (IoU)



# Non-maximum suppression

- ▶ Для **каждого класса** берем прямоугольник ( $x, y, w, h$ ) с наибольшей вероятностью;
- ▶ Удаляем прямоугольники с меньшими вероятностями, для которых  $\text{IoU} > k$ ;
- ▶ Повторяем шаги, пока не переберем все прямоугольники.

# Пример



# R-CNN алгоритм

- ▶ Генерация областей (region proposals)
- ▶ Классификация с локализацией объекта для каждого прямоугольника
- ▶ Отбор лучших прямоугольников ( $x, y, w, h$ ) (non-maximum suppression)

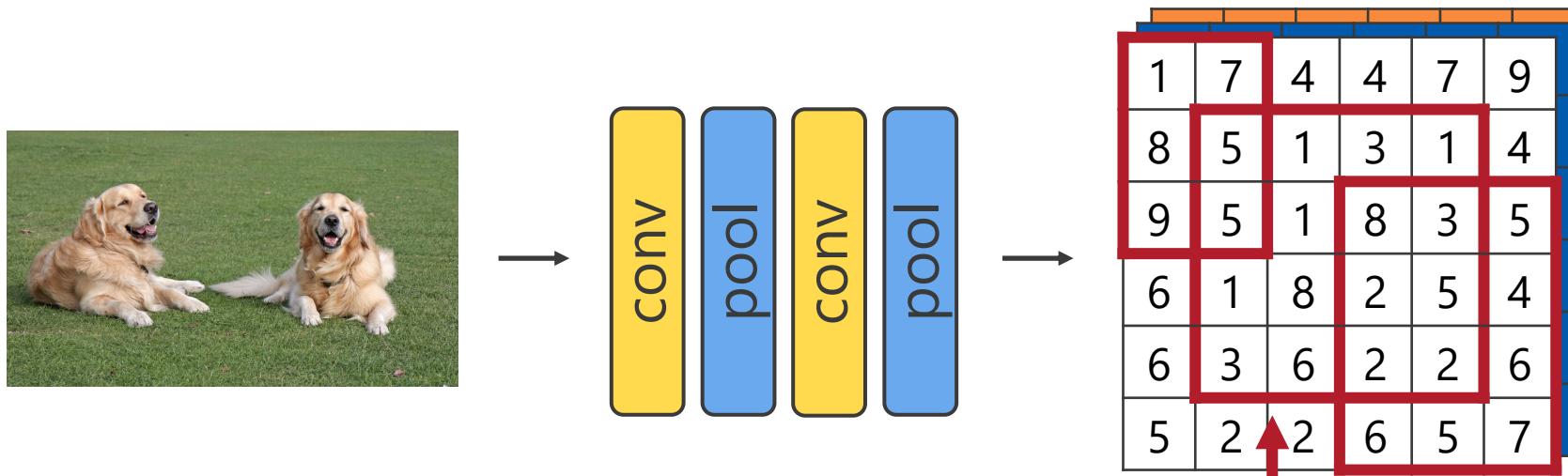


Fast R-CNN

# R-CNN

- ▶ Простой и эффективный
- ▶ Около 2000 областей для классификации
- ▶ Долго обрабатывает одно изображение
- ▶ Как можно ускорить?

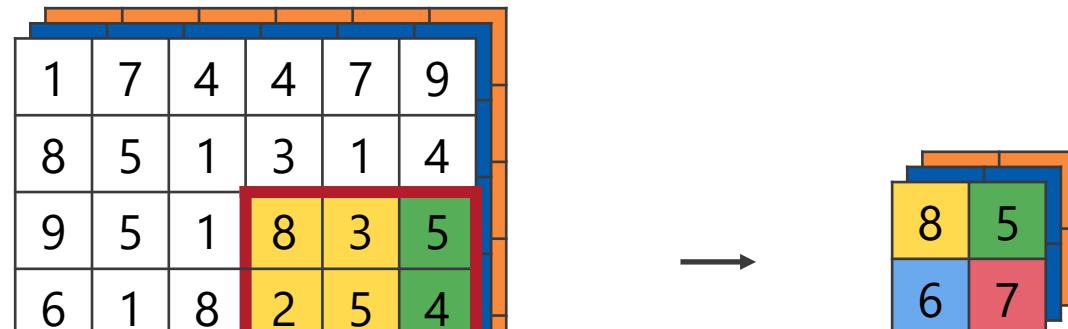
# Region proposals



Сгенерированные области, ROIs  
(Regions of Interest)

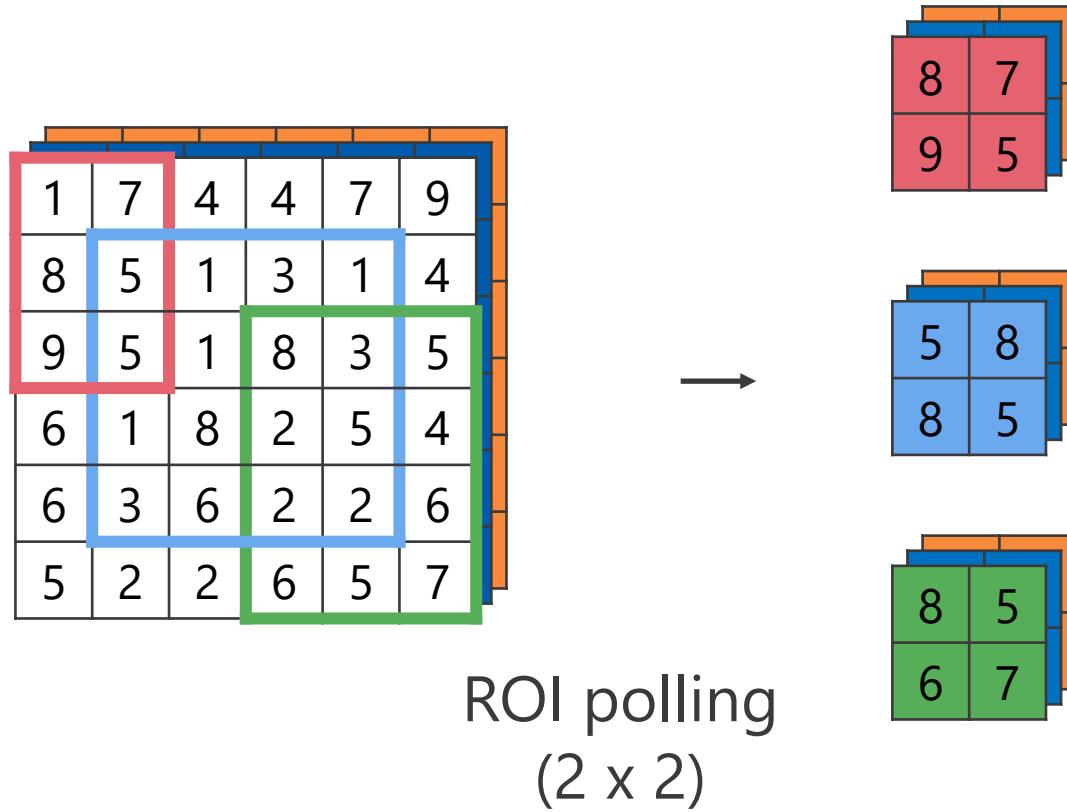
# ROI pooling

- ▶ Каждый ROI делим сеткой ( $2 \times 2$ )
- ▶ В каждой ячейке сетки находим максимум

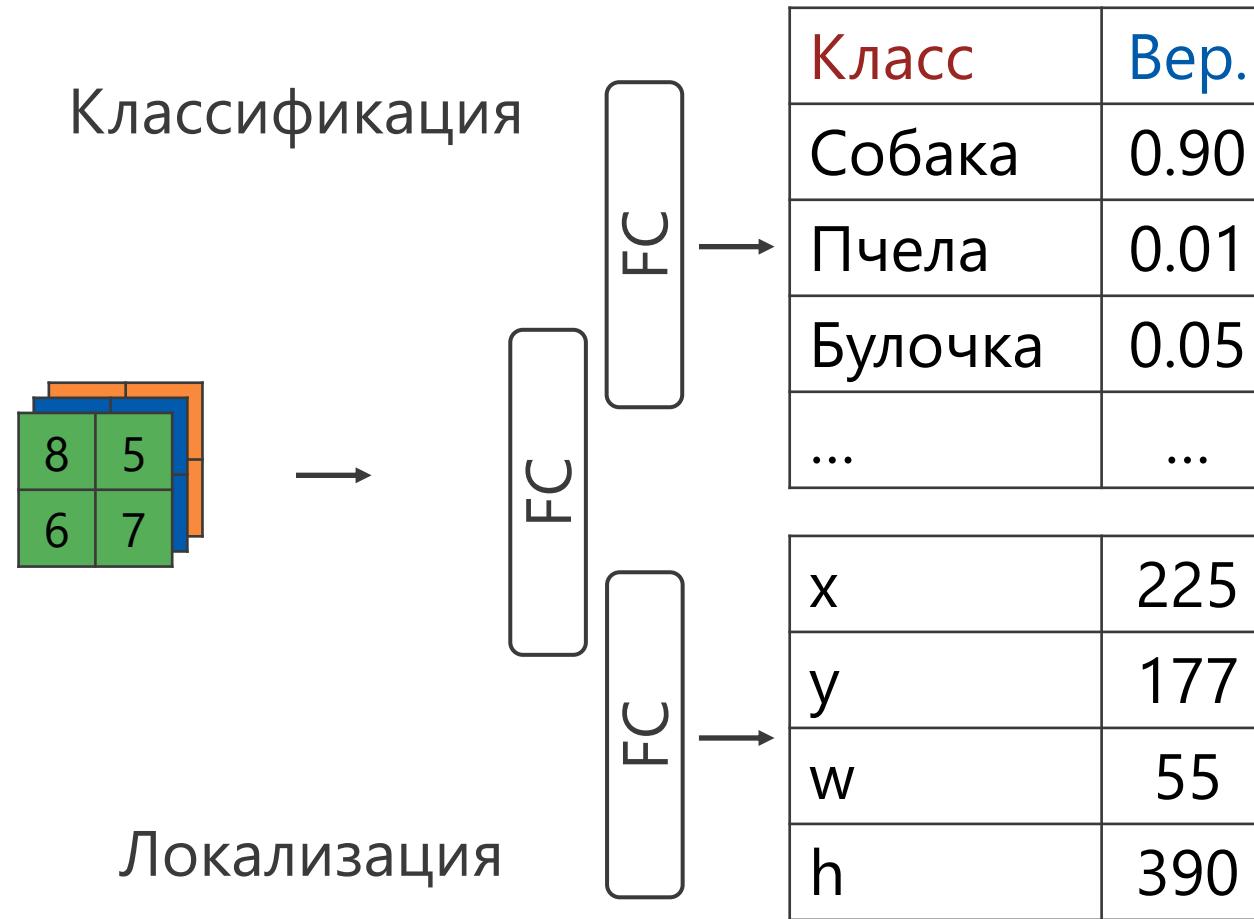


ROI pooling  
( $2 \times 2$ )

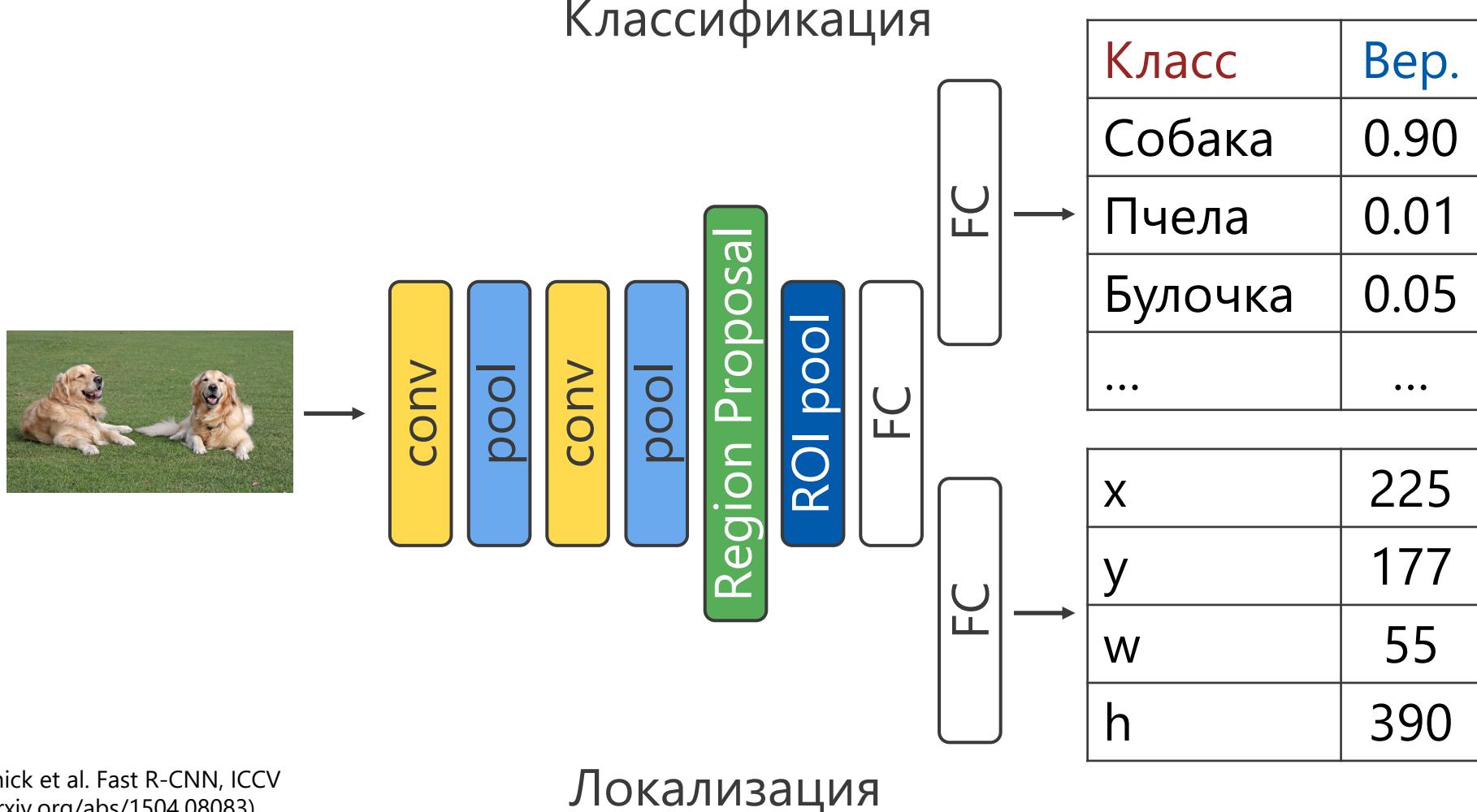
# ROI pooling



# Классификация + локализация

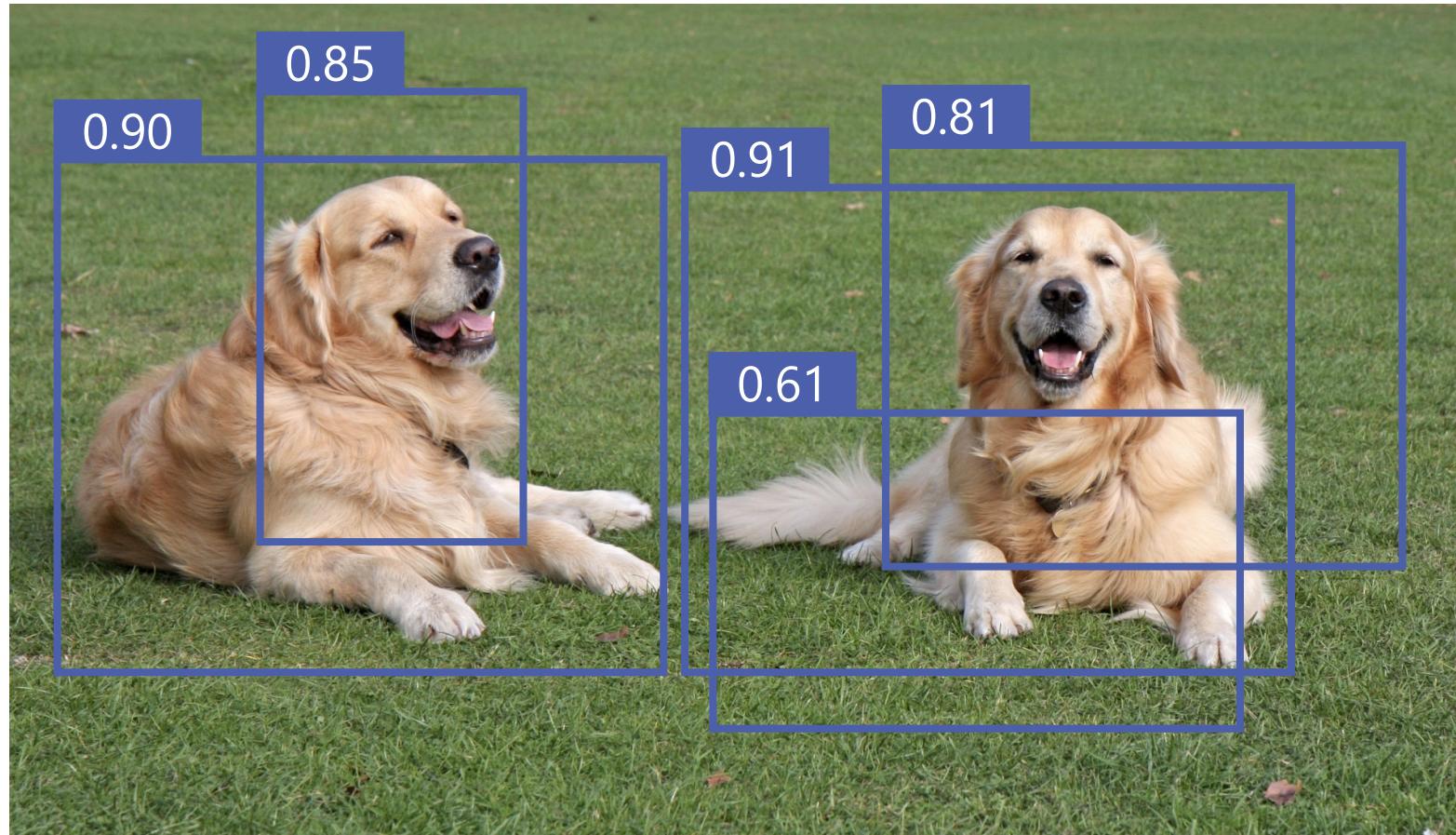


# Fast R-CNN



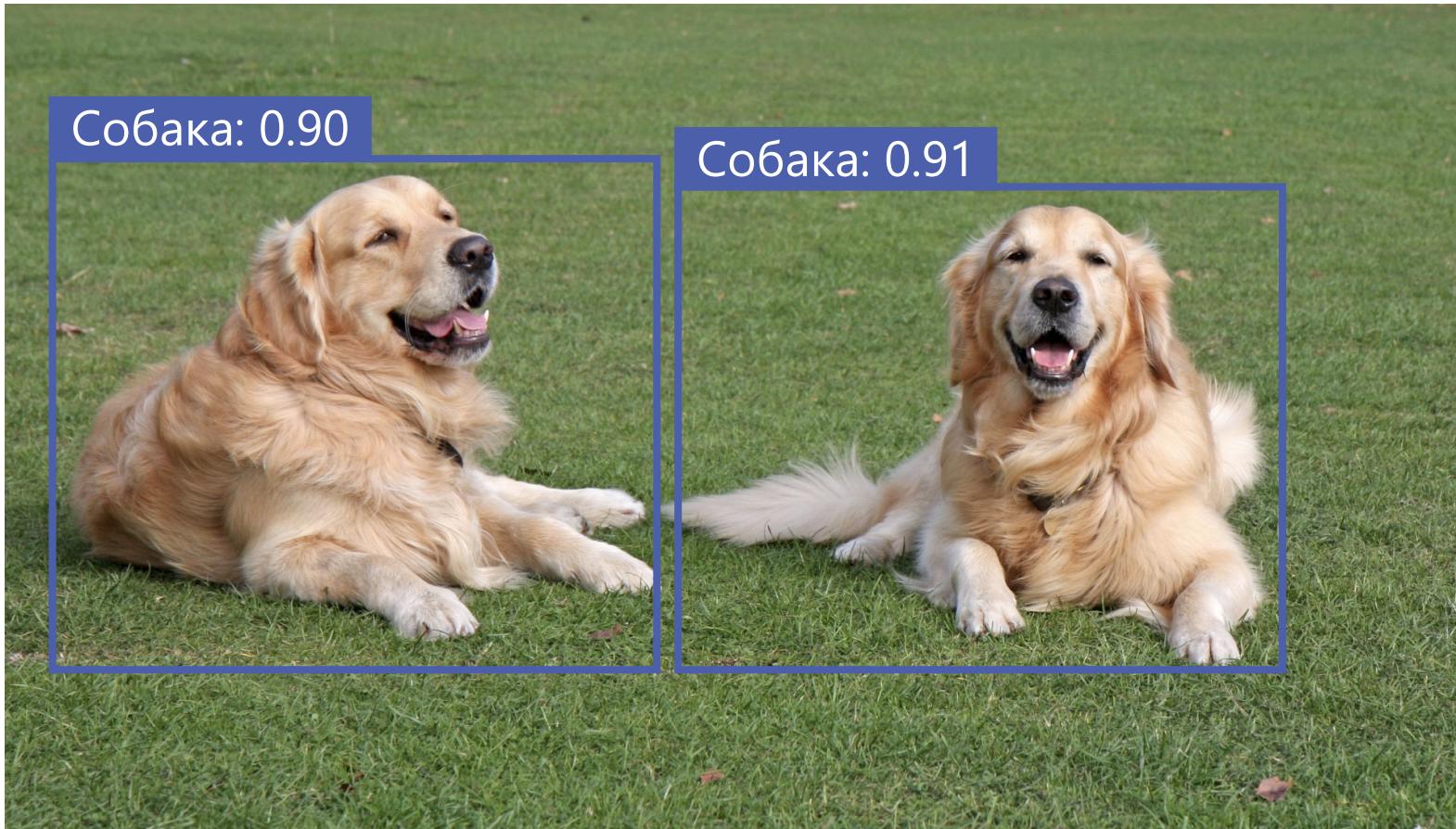
Подробнее: R. Girshick et al. Fast R-CNN, ICCV  
2015 (URL: <https://arxiv.org/abs/1504.08083>)

# Классификация + локализация для регионов



Как выбрать нужные прямоугольники?

# Пример

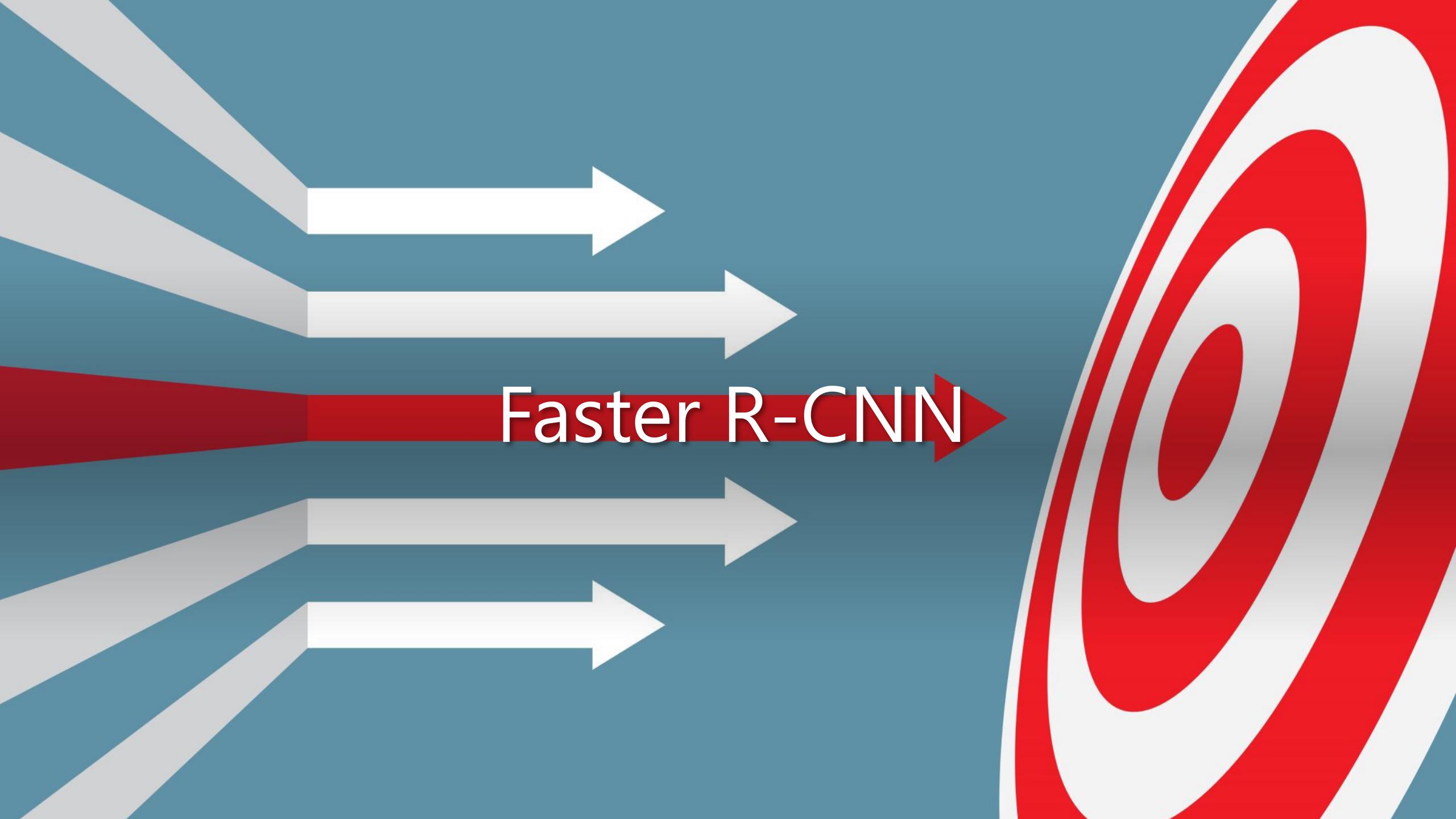


# Fast R-CNN

- ▶ Пропускаем изображение через сверточные слои
- ▶ Генерация области (ROI) на свернутом изображении
- ▶ ROI pooling
- ▶ Классификация с локализацией объекта для каждого ROI
- ▶ Отбор лучших прямоугольников ( $x, y, w, h$ ) (non-maximum suppression)

# Выводы

- ▶ Изображение целиком подается на вход модели
- ▶ Изображение проходит через сверточные слои сети только один раз
- ▶ Ускорение в 25 раз по сравнению с R-CNN

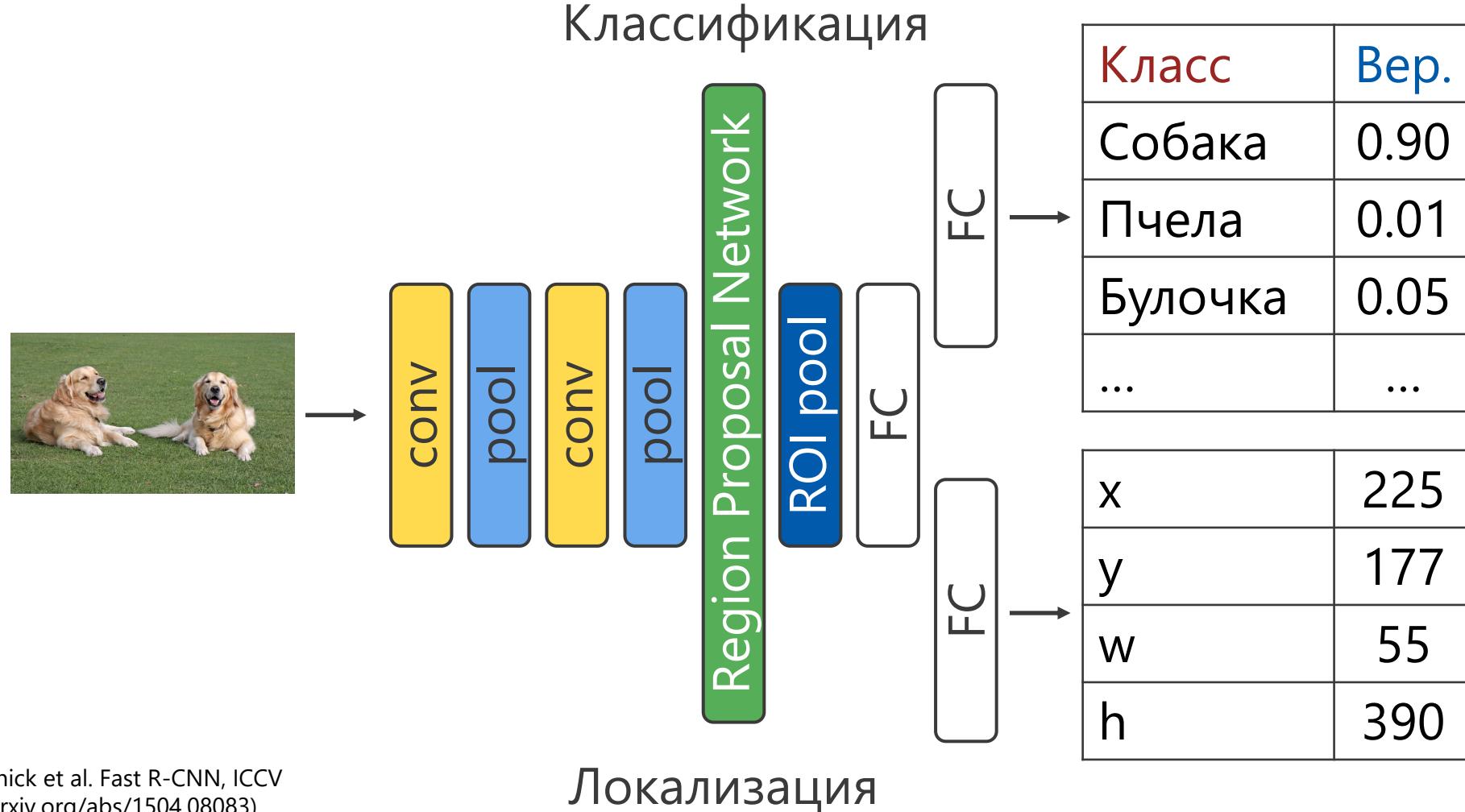


Faster R-CNN

# Fast R-CNN

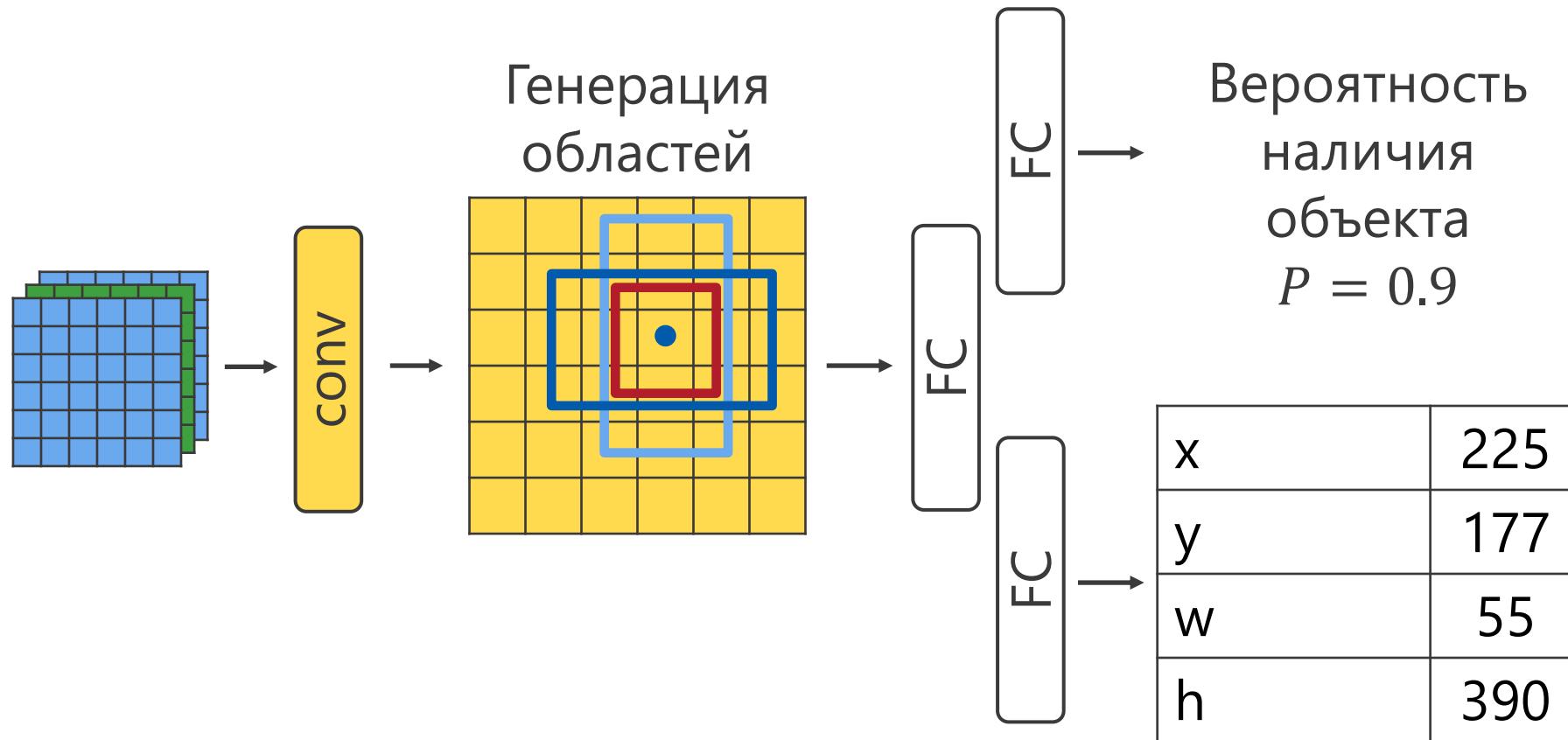
- ▶ Быстрее R-CNN в 25 раз
- ▶ Region proposal занимает больше всего времени
- ▶ Как можно ускорить?

# Faster R-CNN



Подробнее: R. Girshick et al. Fast R-CNN, ICCV  
2015 (URL: <https://arxiv.org/abs/1504.08083>)

# Region proposal network



\* Обучаем эту сеть отдельно

# Region proposal network

- ▶ Простая генерация областей
- ▶ Оставляем только области, где предсказан объект
- ▶ Применяем non-maximum suppression для отбора лучших областей
- ▶ Выводим предсказанные локализации ( $x, y, w, h$ ) в лучших областях
- ▶ Локализации ( $x, y, w, h$ ) используем как ROI в Faster R-CNN

	<b>R-CNN</b>	<b>Fast R-CNN</b>	<b>Faster R-CNN</b>
Время на изображение	50 сек	2 сек	0.2 сек
Ускорение	1	25	250
mAP (VOC 2007)	66	66.9	66.9

\* mAP — mean Average Precision

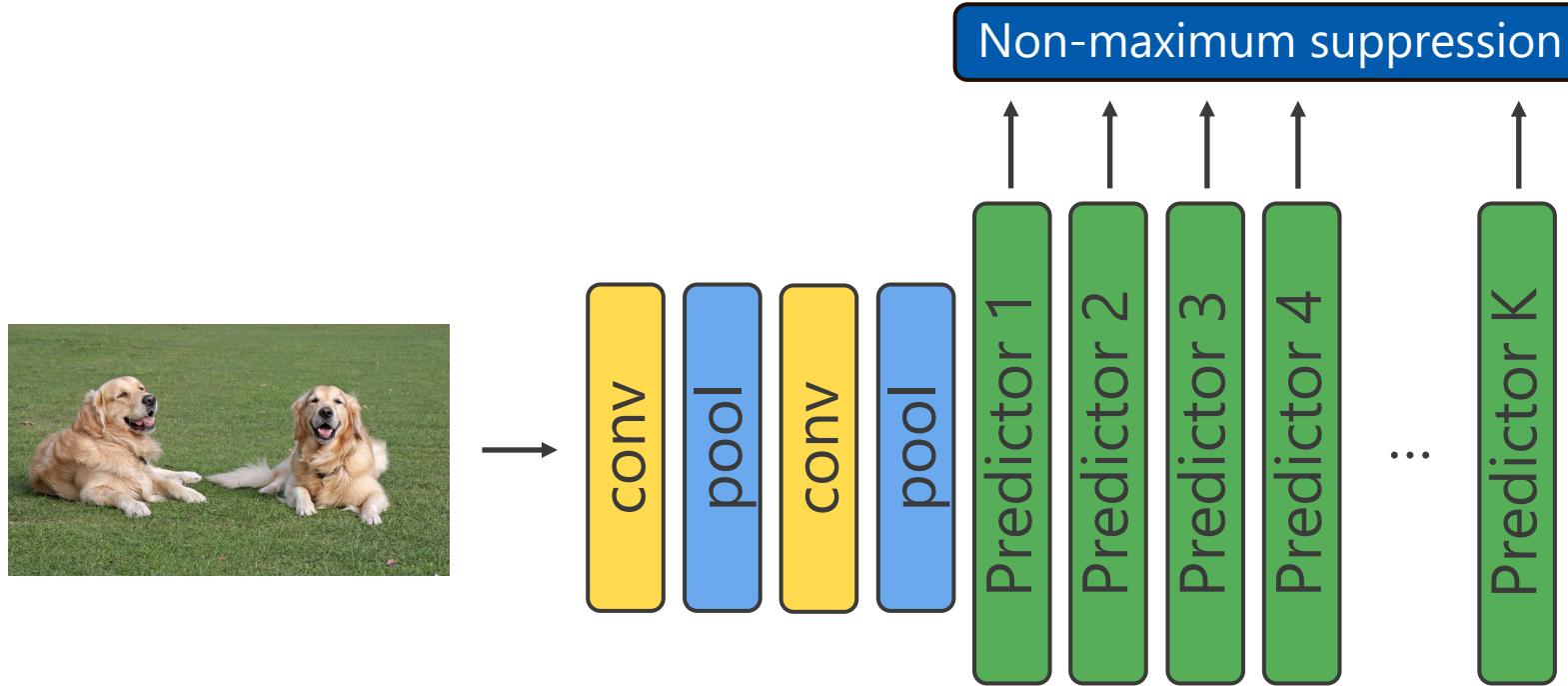


# SSD: Single Shot MultiBox Detector

# R-CNN

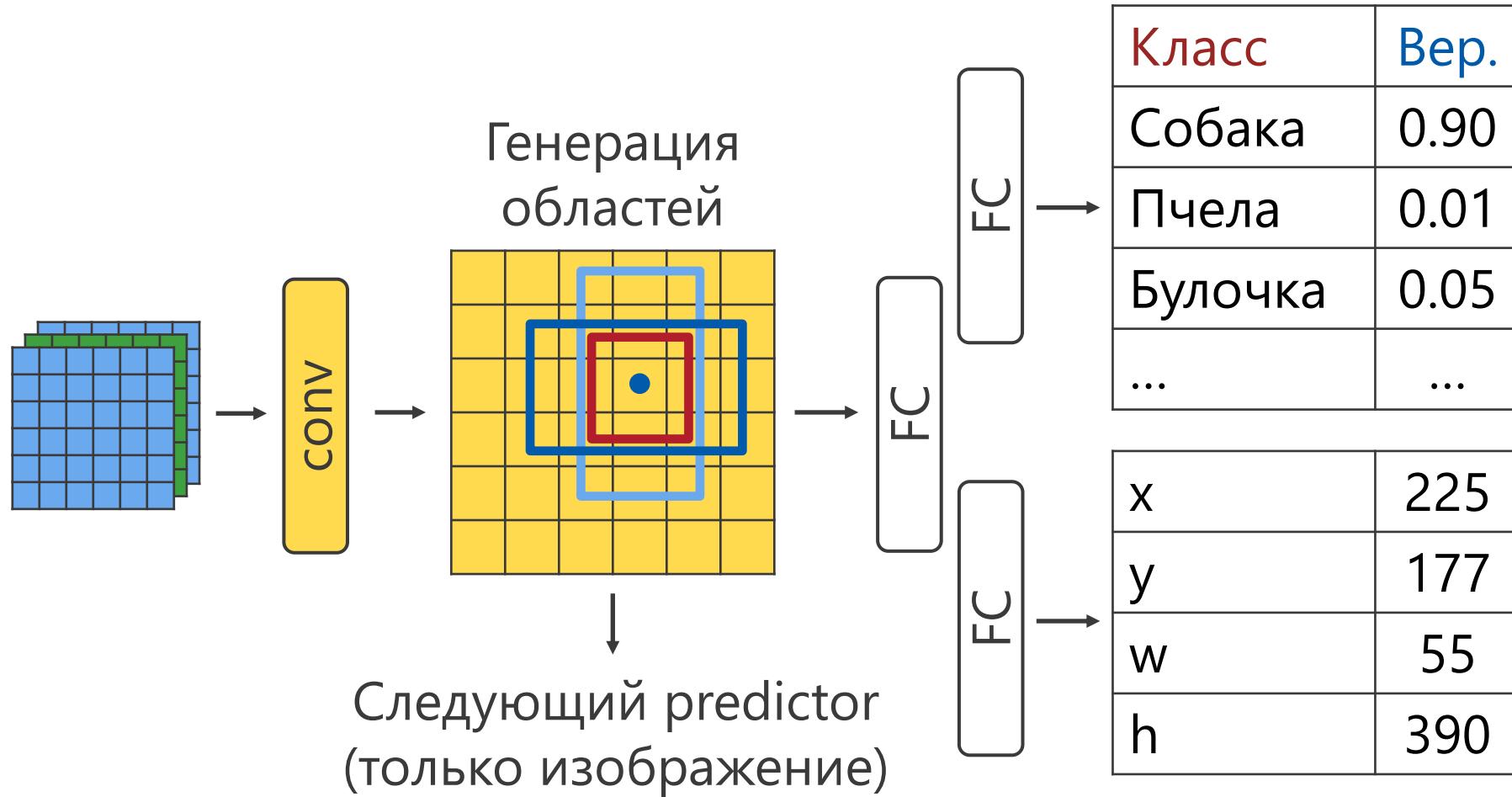
- ▶ Два этапа:
  - Сложная генерация областей
  - Классификация и локализация внутри каждой области
- ▶ Высокое качество
- ▶ Медленные
- ▶ Можно ли обойтись одним этапом?

# SSD: Single Shot MultiBox Detector



Подробнее: W. Liu et al. SSD: Single Shot MultiBox Detector, ECCV  
2016 (URL: <https://arxiv.org/abs/1512.02325>)

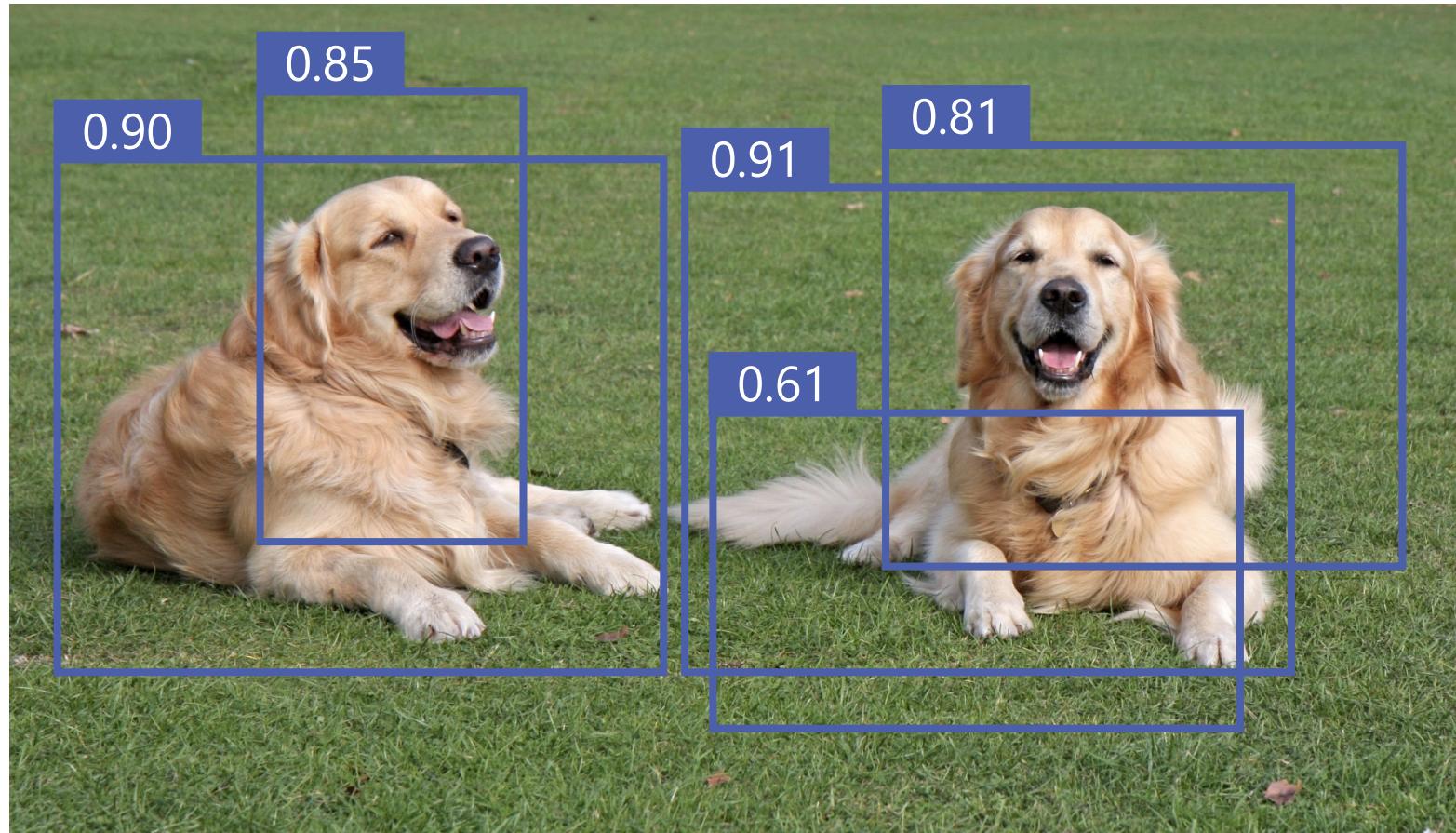
# Predictor



# SSD: Single Shot MultiBox Detector

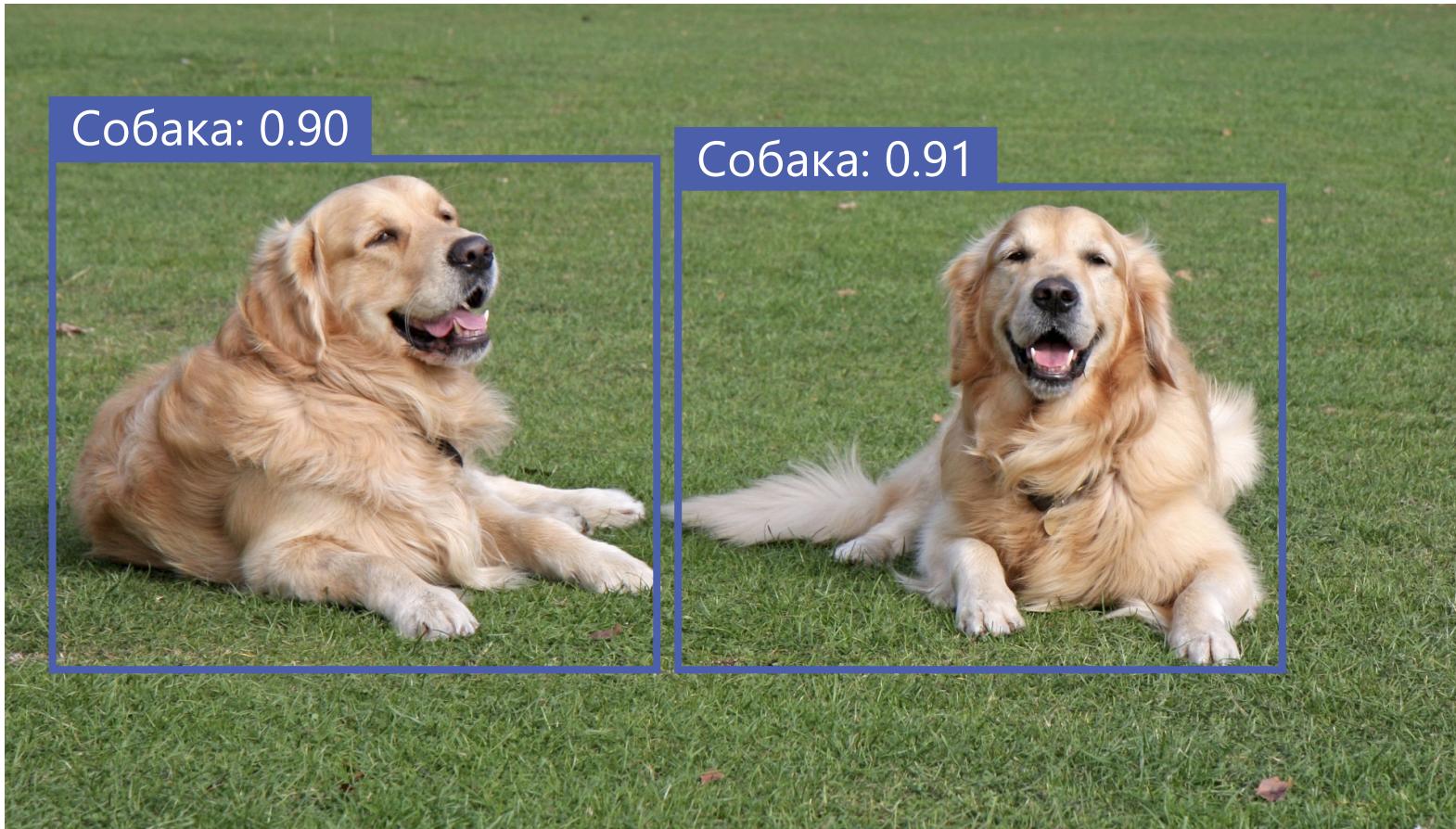
- ▶ Каждый предиктор сворачивает изображение
- ▶ Классификация и локализация в предикторе применяются к изображениям разных размеров
- ▶ Предсказания всех предикторов объединяются

# Классификация + локализация для регионов

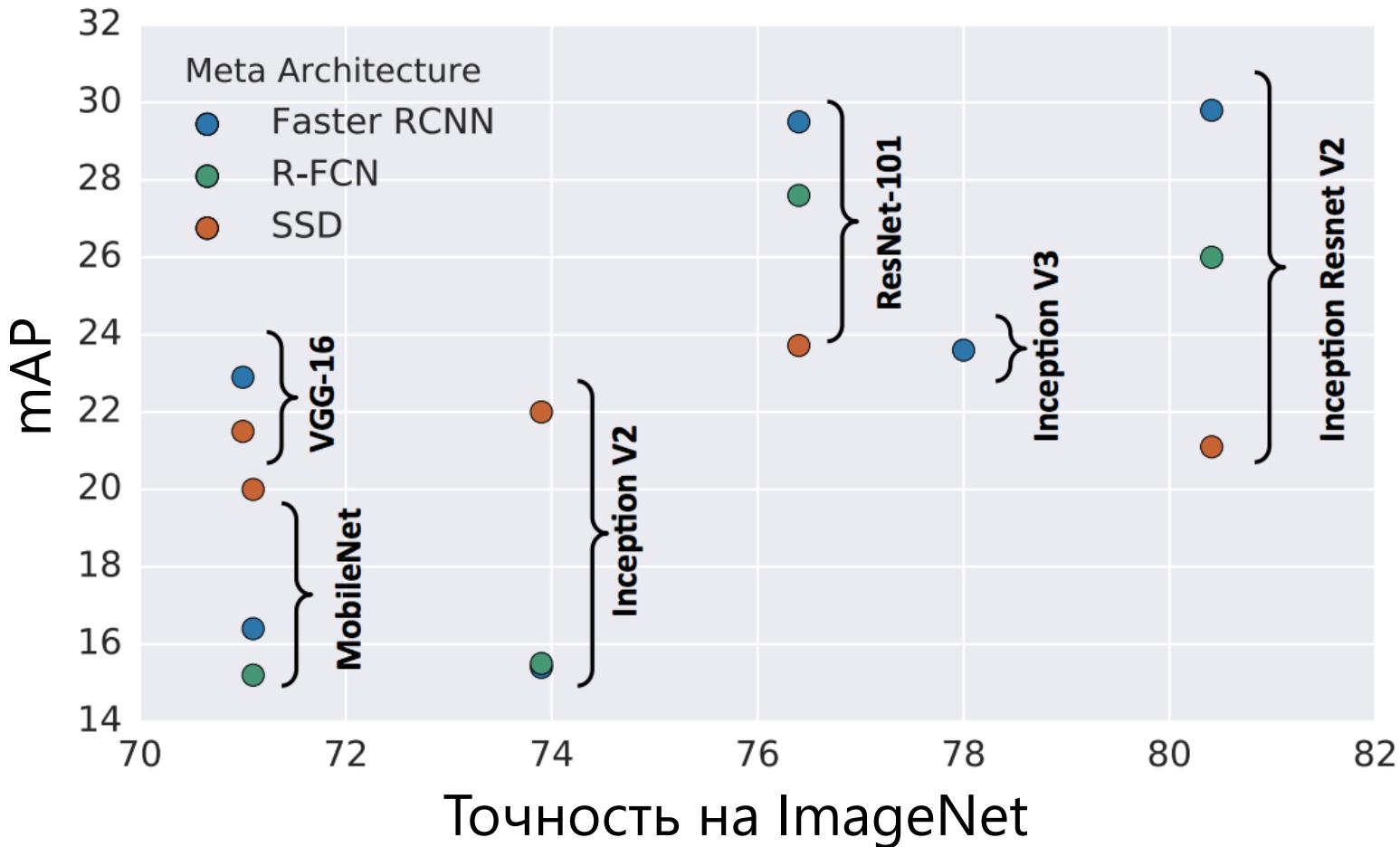


Как выбрать нужные прямоугольники?

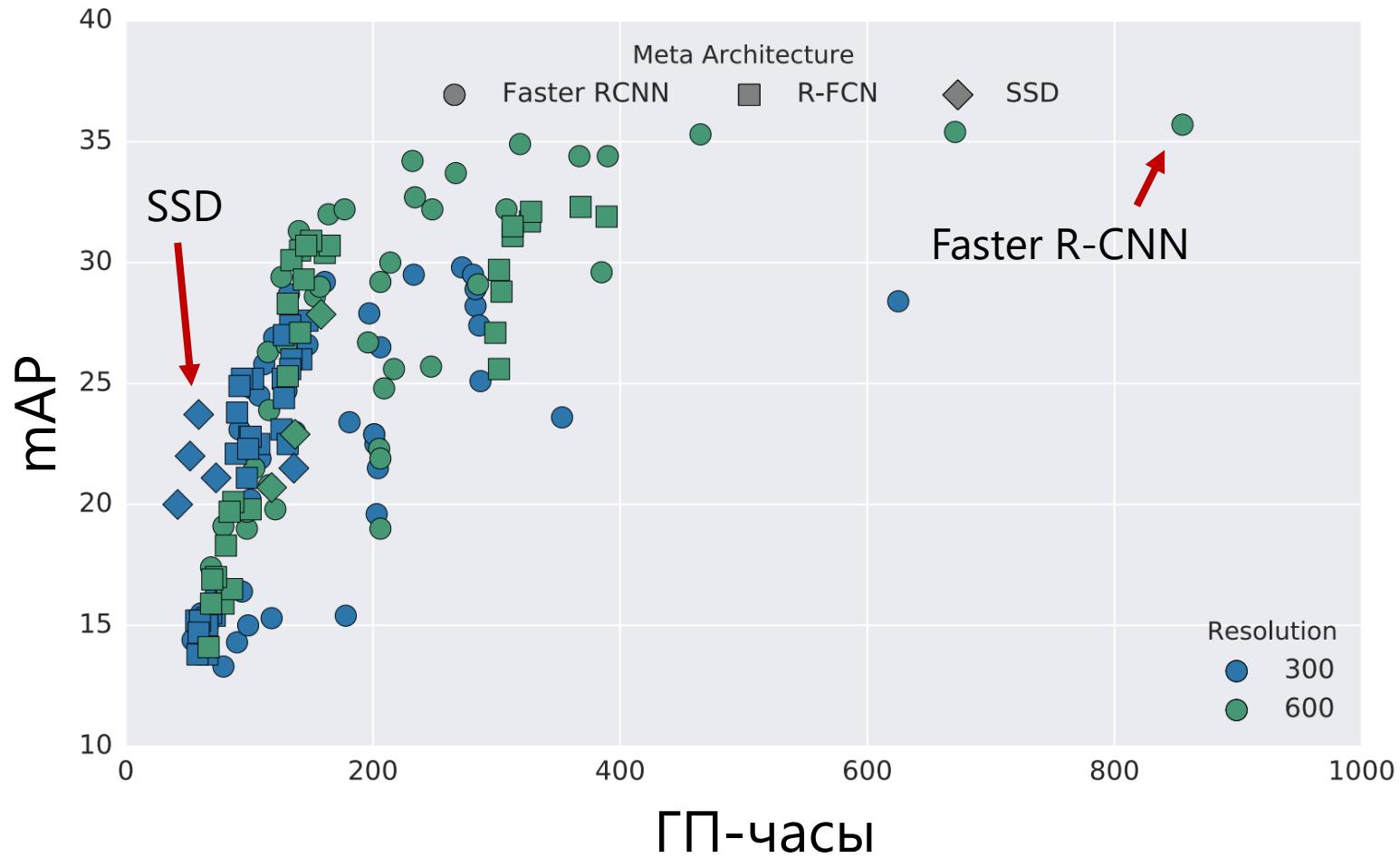
# Пример



# Сравнение алгоритмов



# Сравнение алгоритмов



# Сравнение алгоритмов

- ▶ Faster R-CNN лучше по качеству, но медленнее
- ▶ SSD быстрее, но хуже по качеству

# Где почитать еще

- ▶ SSD: [https://d2l.ai/chapter\\_computer-vision/ssd.html](https://d2l.ai/chapter_computer-vision/ssd.html)
- ▶ R-CNN: [https://d2l.ai/chapter\\_computer-vision/rcnn.html](https://d2l.ai/chapter_computer-vision/rcnn.html)

# Другие алгоритмы

- ▶ Двухэтапные:
  - R-FCN
  - Cascade R-CNN
  - Hybrid Task Cascade
- ▶ Одноэтапные
  - YOLO, YOLOv3
  - RetinaNet
  - FCOS
  - CentripetalNet