|  |  |
| --- | --- |
| **СОГЛАСОВАНО**  Доцент кафедры  ИАНИ ННГУ, к.ф.-м.н.  \_\_\_\_\_\_\_\_\_\_\_\_\_ Д.А. Яшунин  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019 г. | **УТВЕРЖДАЮ**  Профессор кафедры  ИАНИ ННГУ, д.т.н.  \_\_\_\_\_\_\_\_\_\_\_\_\_ Н.В. Старостин  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019 г. |

**Пояснительная записка № 2**

**«ПЗ по методам детекции лиц (ПЗ2)»**

**Этап 1. Подготовка обзоров на существующие подходы к решению   
задачи и построение технологического стека**

**НИР «Разработка и реализация программного обеспечения   
для распознавания лиц на групповых фотографиях»**

**(Шифр ПО «AFR»)**

Ответственный исполнитель

\_\_\_\_\_\_\_\_\_\_\_\_ М.М. Годовицын

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019 г.

**Н. Новгород 2019**

Оглавление

[1 Введение 3](#_Toc4616921)

[2 Regional Convolution Neural Network (RCNN) 3](#_Toc4616922)

[3 Fast Regional Convolution Neural Network (Fast RCNN) 5](#_Toc4616923)

[4 Faster Regional Convolution Neural Network (Faster RCNN) 6](#_Toc4616924)

[5 You Only Look Once (YOLO) 7](#_Toc4616925)

[6 Single-Shot Detector (SSD) 8](#_Toc4616926)

[7 Выводы 9](#_Toc4616927)

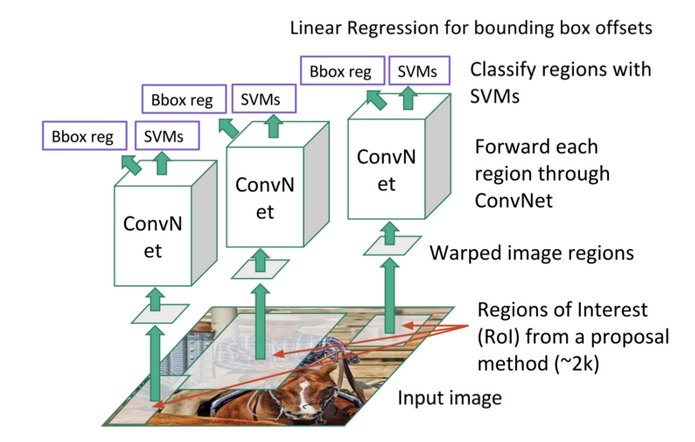
[Список используемых источников 11](#_Toc4616928)

# 

# Введение

Задача детекции лица человека на изображении является первым шагом в решении стоящей перед нами задачи его распознавания. Необходимо выбрать алгоритм её решения, а для этого провести анализ существующих методов, осветив слабые и сильные их места, ограничения, производительность. Рассмотрены не все, но многие методы детекции, удовлетворяющие ограничениям разрабатываемого ПО «AFR».

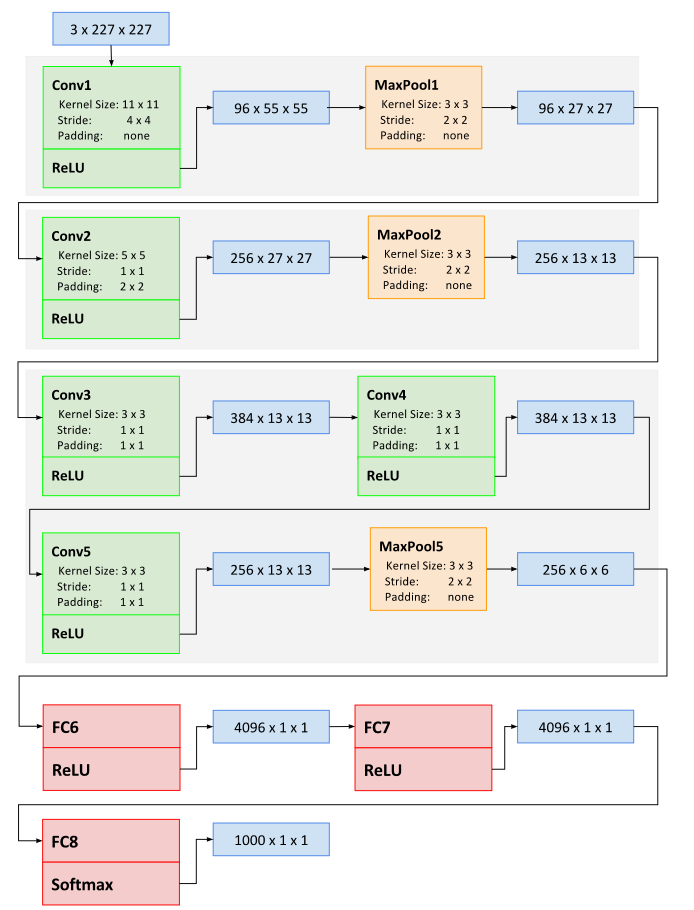
# Regional Convolution Neural Network (RCNN)



**Входные данные.**

Область изображения, представленная вектором размерности 3x227x227, найденная селективным поиском.

**Выходные данные.**

Вектор 1000x1x1.

**Производительность.**

50 секунд на изображение (GPU)

**Алгоритм работы.**

С помощью селективного поиска на входном изображении выделяются области (ограничительные рамки), предположительно содержащие искомый объект (гипотезы). Похожие гипотезы объединяются вместе жадным алгоритмом. Далее трансформируются в квадрат (3x227x227) и подаются на вход свёрточной нейронной сети, которая генерирует карту признаков, представленную вектором (4096x1x1). Далее производится классификация либо с использованием метода опорных веторов, либо с помощью softmax слоя (в таком случае получим ускорение, но точность может снизиться в среднем на 4%). Потом с помощью лиейного регрессора уточняется местоположение ограничительной рамки.

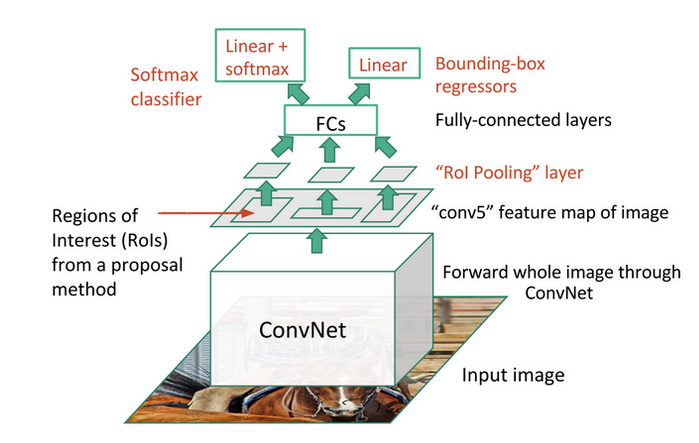
**Комментарий.**

Занимает много времени: алгоритм поиска областей является фиксированным (нет элемента обучения).

Figure . Селективный поиск.



# Fast Regional Convolution Neural Network (Fast RCNN)

****

**Входные данные.**

Целое изображение (сеть самостоятельно формирует карту признаков).

**Выходные данные.**

Вектор 1000x1x1

**Производительность.**

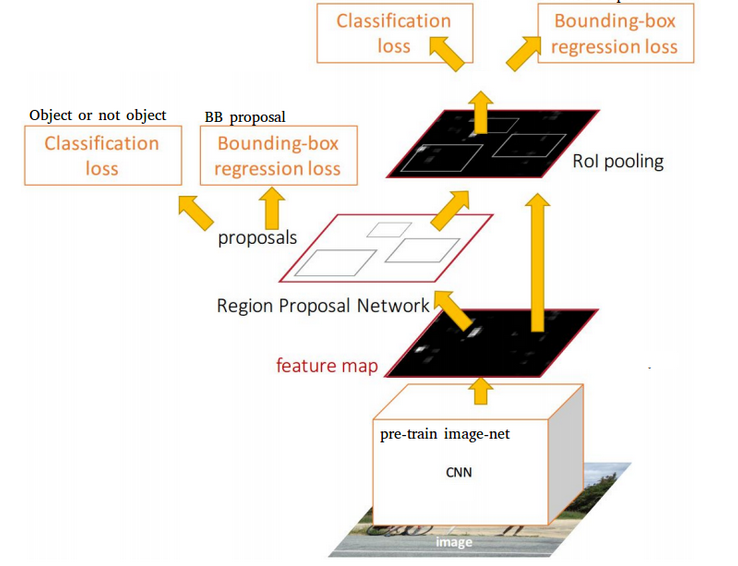
2 секунды на изображение (GPU)

Быстрее, потому что операция свёртки выполняется только один раз для каждого изображения, а не каждый раз для каждой области.

**Алгоритм работы.**

В свёрточную сеть подаются не гипотезы, а входное изображение для создания свёрточной карты признаков. Далее гипотезы проецируется на эту карту признаков и перемасштабируются (3x227x227). Далее следуют полносвязные слои, которые для каждой гипотезы вычисляют класс и смещение.

# Faster Regional Convolution Neural Network (Faster RCNN)



**Входные данные.**

Целое изображение.

**Выходные данные.**

Вектор 1000x1x1.

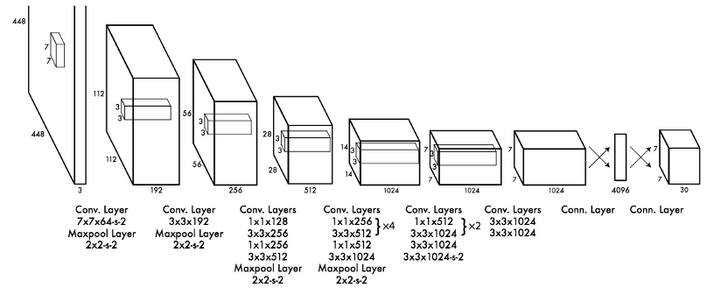
**Производительность.**

0.2 секунды на изображение.

**Алгоритм.**

В свёрточную сеть подаётся изображение, затем создаётся свёрточная карта признаков(Fast RCNN). Затем для этой карты используется ещё одна сеть для генерации гипотез.

# You Only Look Once (YOLO)



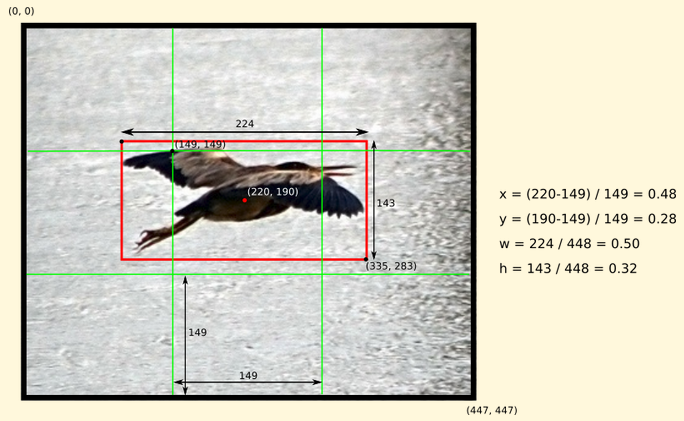
**Входные данные.**

Целое изображение.

**Выходные данные.**

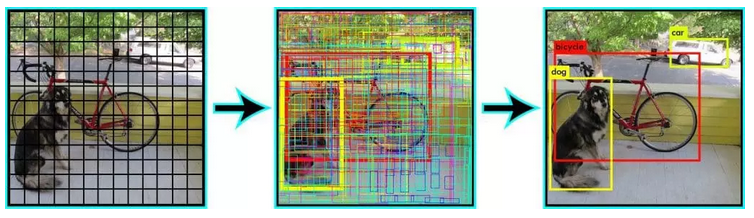
Тензор N\*N(A+B\*5), определяющий прогноз для каждой ячейки сетки, где A — оценка вероятности соответствия каждому классу, B — число рамок в ячейке ,каждой рамке соответствуют 5 значений: координаты центра, ширина и высота (нормализованы в интервале [0,1]), а также её точность (=Pr(Object) \* IOU(pred, truth); если в рамке нет объектов, точность будет равна нулю). Подробнее об этом можно узнать, прочитав раздел «алгоритм».

Figure Пример вычисления координат рамки для изображения 448x448 пикселей, N=3



**Производительность.**

55 кадров/с.



**Алгоритм.**

Делит изображене на сетку NxN, каждая ячейка сетки формирует M ограничительных рамок и значение её точности, т.е. насколько вероятно содержание в ней объекта (независимо от класса). Из сформированных NxNxM рамок часть может быть убрана из рассмотрения, если их точность не соответствует установленному порогу.

**Комментарий.**

Сталкивается с трудностями при детектировании объектов малого размера.

Сеть имеет 24 свёрточных слоя и два полносвязных. Существует облегчённая её версия, Fast YOLO, с 9 свёрточными слоями и меньшим количеством фильтров. Работает она ещё быстрее (150 кадров/с), однако точность её значительно ниже (примерно на 8%).

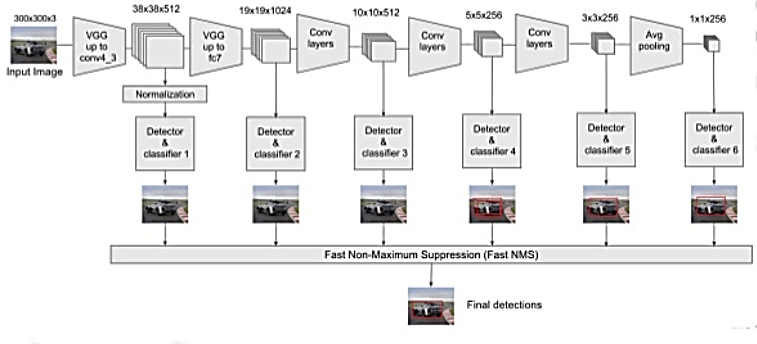
# Single-Shot Detector (SSD)

**Входные данные.**

Изображение фиксированного размера (300x300 для SSD300; максимальное разрешение — 512x512 для SSD512)

**Выходные данные.**

Карта объектов: каждое место этой карты хранит информацию об объекте (класс), который здесь содержится, и об ограничивающей рамке. Например, SSD512 выдаёт 7 карт объектов с разрешениями 64x64, 32x32, 16x16, 8x8, 4x4, 2x2, и 1x1. Для SSD300 информацию о выходных данных можно увидеть на рисунке ниже.

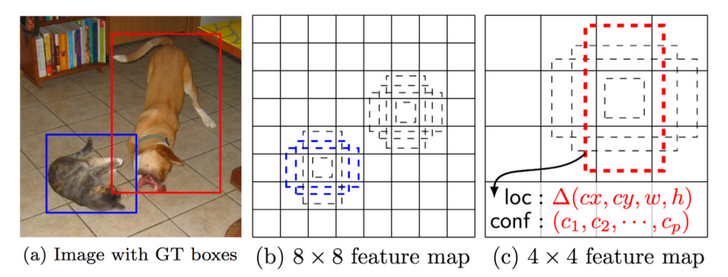


**Производительность.**

45 кадров/с.

**Алгоритм.**

Изображение обрабатывается сетью с различными фильтрами (10x10, 5x5, 3x3). Карты признаков, получаемые в разных частях сети, обрабатываются фильтрами 3x3 и формируют ограничительные рамки (подобно Fast RCNN). Каждая рамка имеет пять параметров: координаты центра, ширину, высоту, а также вектор соответствия каждому классу (подобно YOLO).

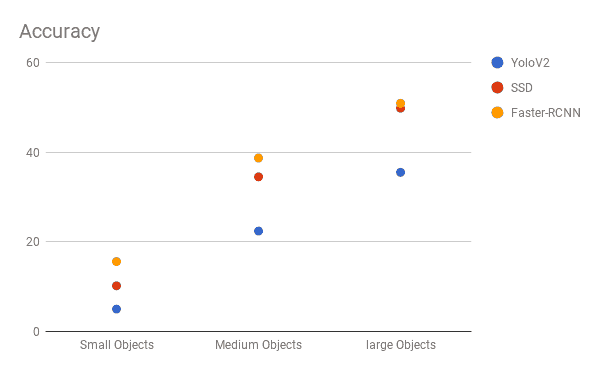


# Выводы

YOLO и SSD несомненно демонстрируют хорошее соотношение между скоростью и точностью, но содержат в себе существенный недостаток, который, возможно, не так уж и критичен для нашей задачи: трудности с обнаружением маленьких объектов. Пока не ясно, насколько далеки будут от камеры лица, которые необходимо будет обнаружить.

Faster RCNN пусть и работает медленне, да и сам по себе несколько прозаичен, не содержит в себе существенных недостатков. Время обработки изображения этим алгоритмом вполне укладывается в ограничения.





# Список используемых источников

1. <https://medium.com/@smallfishbigsea/faster-r-cnn-explained-864d4fb7e3f8>
2. <https://medium.com/@smallfishbigsea/understand-ssd-and-implement-your-own-caa3232cd6ad>
3. <https://medium.com/diaryofawannapreneur/yolo-you-only-look-once-for-object-detection-explained-6f80ea7aaa1e>
4. <https://towardsdatascience.com/fasterrcnn-explained-part-1-with-code-599c16568cff>
5. <https://courses.engr.illinois.edu/cs445/fa2015/projects/final/object-detection-video.pdf>
6. <https://medium.com/zylapp/review-of-deep-learning-algorithms-for-object-detection-c1f3d437b852>
7. <https://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/>