

# Машинное обучение (Machine Learning)

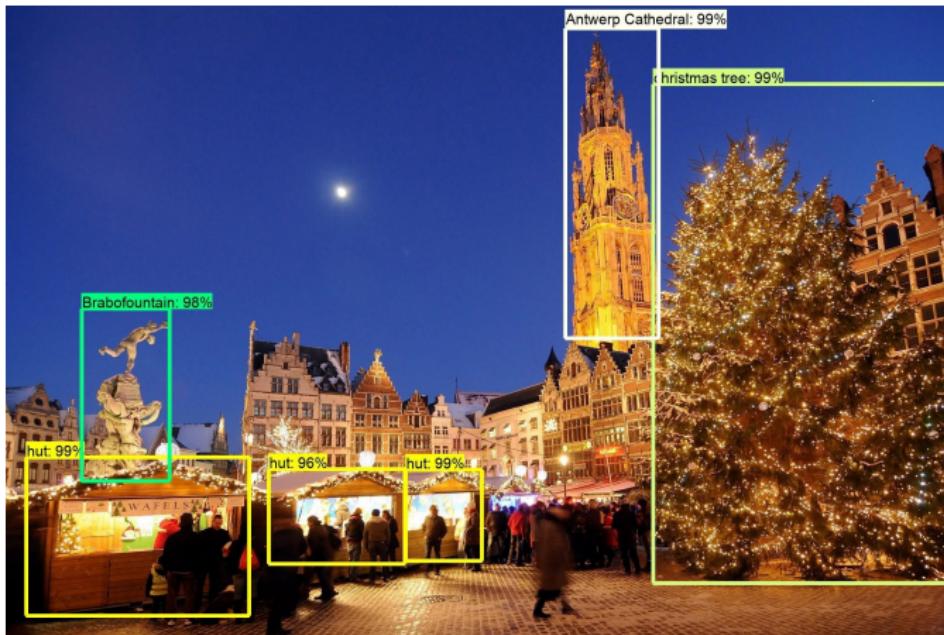
## Детекция изображений

Уткин Л.В.

Санкт-Петербургский политехнический университет Петра Великого



# Detection обнаружение (1)







# Detection обнаружение (4)

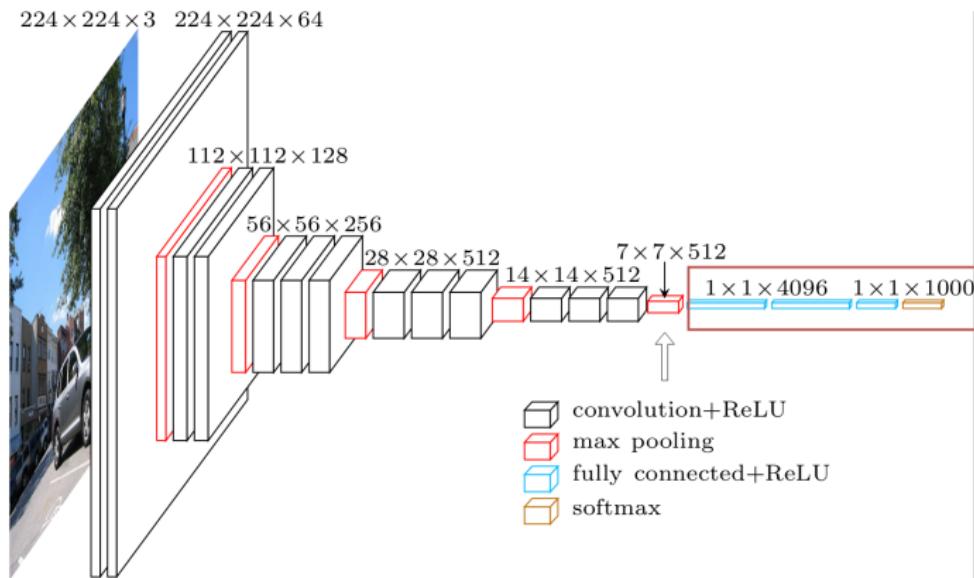
## Минусы алгоритма:

- Скользящее окно имеет фиксированную прямоугольную форму - неточное обнаружение bounding box, если ни одно из скользящих окон не соответствует реальному объекту
- Постоянное вырезание изображений и подача их в ConvNet требует больших вычислительных ресурсов

# One-stage Object Detection

- One-shot detectors обходят оба минуса, используя фиксированное множество детектироворов на сетке (grid detectors)
- Каждый bounding box детектор находится в определенной позиции на изображении
- Подаем изображение в обычную сверточную сеть, которая называется основой детектора объектов.
- Эта сеть - есть не что иное, как классификатор изображений и ее можно предварительно обучать на огромных наборах данных (ImageNet)

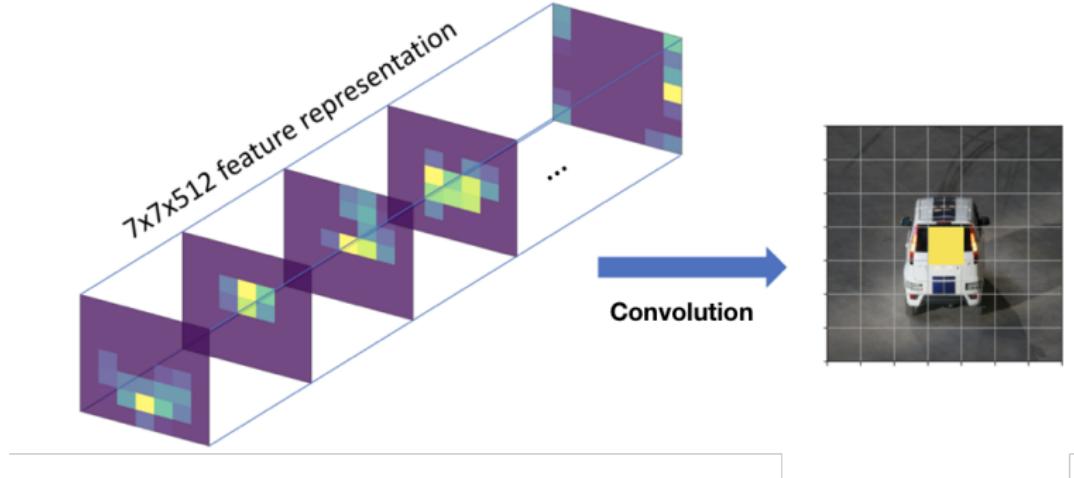
## Detection обнаружение - основная сеть



# Detection обнаружение (5)

- Удаляем последние слои (красный прямоугольник), так как не надо классифицировать входное изображение
- Выход основной сети - изображение  $7 \times 7 \times 512$ , которое имеет низкое пространственное разрешение и высокое разрешение признаков
- Располагая эту карту признаков на исходном изображении, можно сопоставить ячейки сетки и входное изображение
- При таком отображении ячейка сетки, которая содержит центр bounding box, может быть аппроксимирована.

# Detection обнаружение (6)



# Detection обнаружение (7)

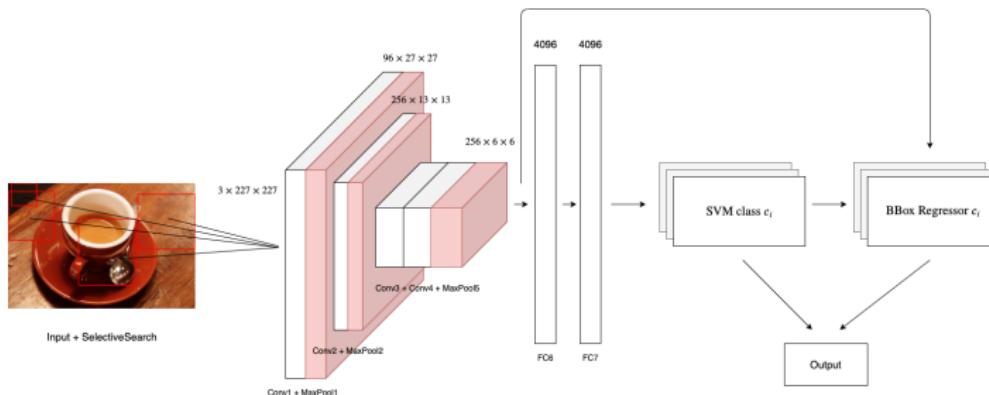
- Сетка объединит все 512 признаков, используя сверточные слои, чтобы обнаружить объект с помощью *bounding box*.
- Атрибуты, необходимые для описания обнаруженного объекта:
  - координаты x-y и ширина/высота *bounding boxes* (4)
  - вероятность ячейки сетки, содержащей объект (1)
  - класс из  $N$  классов, к которым принадлежит обнаруженный объект
- Каждая ячейка сетки выполняет  $4+1+N$  сверток, чтобы обнаружить и нарисовать *bounding box* для объекта.

# R-CNN (1)

Материал заимствован из статьи “Сергей Михайлин, Object Detection. Распознавай и властвуй.

[https://habr.com/ru/company/jetinfosystems/blog/498294/”](https://habr.com/ru/company/jetinfosystems/blog/498294/)

R-CNN - Region Convolution Neural Network



## R-CNN (2)

- “R” в названии R-CNN означает Region (область), т.е. сеть использует предварительную информацию о предполагаемом местоположении обнаруживаемых объектов.
- Возможные области обнаружения объектов получают посредством алгоритма селективного поиска.
- Для селективного поиска перспективных областей обычно используют обычные методы обработки изображений.

# R-CNN: Шаги алгоритма

- ① Определение набора гипотез (регионов изображения), т.е. генерируется много областей изображения, каждая из которых может принадлежать только к одному классу. Меньшие области рекурсивно комбинируются в более крупные посредством жадного алгоритма. На каждом этапе сливаются две области, обладающие наибольшим сходством. Этот процесс повторяется до тех пор, пока не останется только одна область. В ходе этого процесса создается иерархия все более крупных областей, что позволяет алгоритму предлагать широкий набор вероятных областей изображения, где могут быть обнаружены объекты. Сгенерированные таким способом области используются в качестве рекомендованных предложений.

# R-CNN: Шаги алгоритма (продолжение)

2. Извлечение из предполагаемых регионов признаков с помощью сверточной нейронной сети и их кодирование в вектор. Области, представляющие интерес, передаются классифицирующей и локализующей сети для предсказания классов объектов и ассоциированных с ними ограничительных прямоугольников.
3. Классификация объекта внутри гипотезы на основе вектора из шага 2. Классифицирующая сеть представляет собой сверточную нейронную сеть, после которой применяется SVM для окончательной классификации.
4. Улучшение (корректировка) координат гипотезы.
5. Все повторяется, начиная с шага 2, пока не будут обработаны все гипотезы с шага 1.

# R-CNN: набор гипотез

## 1 Определение набора гипотез (регионов изображения).

- составляется набор гипотез и на основе сегментации определяются границы объектов по интенсивности пикселей, перепаду цветов, контраста и т.д.
- регионы частично перекрывают друг друга.
- гипотезы дополнительно расширяются на  $m$  пикселей во всех 4 направлениях (добавляется контекст) для уточнения

# R-CNN: кодирование признаков регионов

- 2 Извлечение из предполагаемых регионов признаков с помощью сверточной нейронной сети и их кодирование в вектор.
  - Каждая гипотеза из предыдущего шага по отдельности друг от друга поступает на вход сверточной сети (AlexNet) без последнего softmax-слоя
  - Результат - кодирование изображения в **векторное представление**, которое извлекается из последнего полносвязного FC7 слоя

# R-CNN: классификация

3 Классификация объекта внутри гипотезы на основе вектора из шага 2.

- SVM (One vs. Rest – один против всех)
- Выход - вектор длины  $+1$  по количеству классов плюс нулевой класс - фон

# R-CNN: корректировка координат

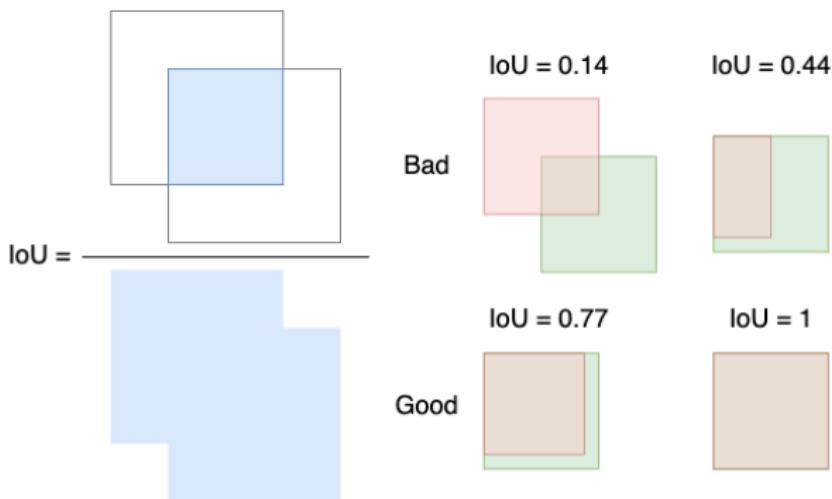
## 4 Улучшение (корректировка) координат гипотезы.

- Гипотезы, содержащие какой-либо объект, дополнительно обрабатываются линейной регрессией (кроме гипотез с классом «фон», там нет объекта)
- Для каждого класса используется свой регрессор.
- На входе - не вектор из слоя FC7, а карты признаков, извлеченные из последнего MaxPooling слоя, так как вектор сохраняет информацию о наличии объекта с какими-то характеристическими подробностями, а карта признаков наилучшим образом сохраняет информацию о местоположении объектов.

# R-CNN - позитив. и негат. гипотезы (1)

- Проблема - несбалансированность гипотез: на картинке с единственным объектом лишь несколько гипотез из множества содержат этот самый объект  $c > 0$  (**позитивные гипотезы**), а все остальные являются фоном  $c = 0$  (**негативные гипотезы**)
- Метрика пересечения регионов - Intersection over Union (**IoU**) - площадь пересечения двух областей делится на общую площадь регионов

## R-CNN - позитив. и негат. гипотезы (2)



# R-CNN - позитив. и негат. гипотезы (3)

- Позитивные гипотезы – если класс определен неверно, то штраф
- Негативные? Их много. Фон (easy negative) просто классифицировать, чем другие объекты (hard negative)
  - Easy negative и hard negative определяются по пересечению ограничивающей рамки (IoU) с правильным положением объекта на изображении.
  - Если пересечения нет или оно крайне мало – это easy negative ( $c = 0$ ), если большое – это hard negative или positive.
  - Подход Hard Negative Mining: для обучения только hard negative, поскольку, научившись распознавать их, мы автоматический добиваемся наилучшей работы с easy negative

# Non-maximum suppression (1)

- Проблема - модель выделяет несколько гипотез с большой уверенностью указывающие на один и тот же объект
- Non-maximum suppression (NMS) позволяет оставить только **одну**, наилучшую, ограничивающую рамку.
- И на изображении может быть два разных объекта одного класса

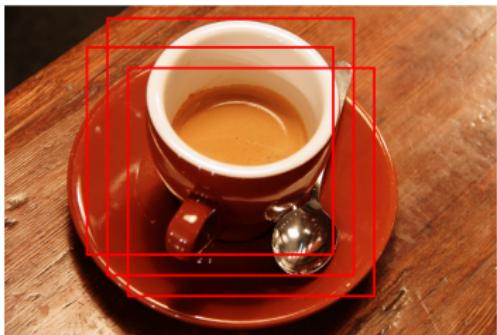
Detection  
oooooooooooo

R-CNN  
oooooooooooo●oooooooooooooooooooo

Region Proposal Network  
oooooooooooooooo

# Non-maximum suppression - пример

Before non-maximum suppression



After non-maximum suppression



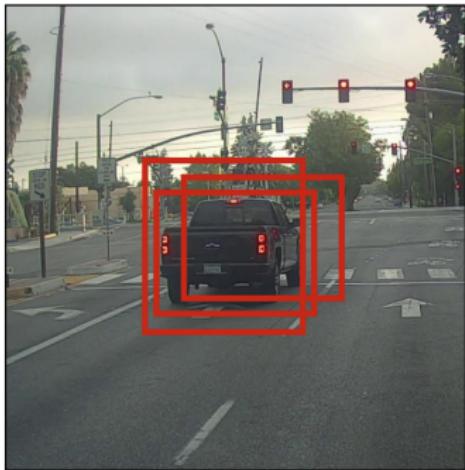
Detection  
oooooooooo

R-CNN  
oooooooooooooooooooo

Region Proposal Network  
oooooooooooooooooooo

# Non-maximum suppression - еще пример

Before non-max suppression



Non-Max  
Suppression



After non-max suppression



# Non-maximum suppression - алгоритм

Алгоритм на одном классе:

- ① На вход функция принимает набор гипотез для одного класса и порог, задающий величину максимального IoU между гипотезами.
- ② Гипотезы сортируются по их «уверенности» (IoU).
- ③ В цикле выбирается первая гипотеза (имеет наибольшую величину IoU) и добавляется в результирующий набор.
- ④ В цикле выбирается вторая гипотеза (среди оставшихся после шага 3).
- ⑤ Если между выбранными гипотезами IoU больше, чем выбранный порог, то вторая гипотеза отбрасывается и далее не присутствует в результирующем наборе.
- ⑥ Все повторяется, начиная с шага 3, до полного перебора гипотез.

# Обучение R-CNN (векторного представления)

- Предобученная на ImageNet сеть, например AlexNet, дообучается, чтобы работать с нужными классами
- Для этого изменяют размерность выходного слоя на  $C$  и обучают модифицированный вариант
- Первые слои можно заблокировать, т.к. они извлекают первичные признаки, а последующие во время обучения адаптируются под признаки нужных классов
- Когда сеть обучилась классифицировать объекты, последний слой с SoftMax активацией отбрасывается и выходом становится слой FC7, выход которого - векторное представление гипотезы
- Позитивные гипотезы -  $\text{IoU} > 0.5$ , остальные негативные. Мини-батч размерности 128 состоит из 32 позитивных и 96 негативных гипотез

# Обучение R-CNN (классификаторы и регрессоры)

- SVMы получают на вход векторное представление гипотезы и обучаются как обычные SVM модели, но негативы берутся с  $\text{IoU} > 0.3$
- Регрессии:
  - $G = (g_x, g_y, g_w, g_h)$  – правильные координаты объекта;
  - $\hat{G} = (\hat{g}_x, \hat{g}_y, \hat{g}_w, \hat{g}_h)$  – исправленное положение координат гипотез (должно совпадать с  $G$ );
  - $T = (t_x, t_y, t_w, t_y)$  – правильные поправки к координатам;
  - $P = (p_x, p_y, p_w, p_h)$  – координаты гипотезы;
- Необходимо построить преобразование  $P$  в  $G$

# Обучение R-CNN (регрессоры)

- Регрессоры - это четыре функции:
  - $d_x(P), d_y(P)$  – определяют поправки к координатам центра  $(x, y)$
  - $d_w(P), d_h(P)$  – определяют поправки к ширине и высоте в логарифмическом пространстве
- $\varphi_5(P)$  - карта признаков, полученная из  $MaxPool_5$  слоя сети, при подаче в сеть гипотезы, ограниченной координатами  $P$ .
- Преобразование  $P$  в  $G$  как:

$$\begin{aligned}\hat{g}_x &= p_w d_x(P) + p_x, \quad \hat{g}_y = p_h d_y(P) + p_y \\ \hat{g}_w &= p_w e^{d_w(P)}, \quad \hat{g}_h = p_h e^{d_h(P)}\end{aligned}$$

# Обучение R-CNN (регрессоры)

- $d_*(P) = w_*^T \varphi_5(P)$  (здесь  $*$   $\in (x, y, w, h)$ ) - линейная функция, а вектор  $w$  ищется с помощью задачи оптимизации (гребневая регрессия):

$$w_* = \arg \max_{\hat{w}_*} \sum_i^N (T_{i*} - d_*(P))^2 + \lambda \|\hat{w}_x\|^2$$

- Поправки к координатам - пары между правильным положением гипотез  $G$  и их текущем состоянием  $P$ :

$$T_x = \frac{g_x - p_x}{p_w}, \quad T_y = \frac{g_y - p_y}{p_h}$$

$$T_w = \log \frac{g_w}{p_w}, \quad T_h = \log \frac{g_h}{p_h}$$

# R-CNN - недостатки

- Огромное число предлагаемых областей-кандидатов, что замедляет работу сети, т.к. каждая область должна независимо пропускаться через сверточные нейронные сети.
- Гипотезы на шаге 1 могут частично дублировать друг друга – разные гипотезы могут состоять из одинаковых частей и отдельно обрабатываться сетью
- Алгоритм выделения гипотез никак не обучается, а поэтому дальнейшее улучшение качества почти невозможно (есть плохие гипотезы).

# Fast R-CNN - ускоренный вариант R-CNN (1)

В сети Fast R-CNN вычислительные трудности удается частично преодолеть за счет использования общего пути свертки для всего изображения вплоть до определенного количества слоев, после прохождения которых рекомендованные области изображения проецируются на выходные карты признаков, и соответствующие области извлекаются для дальнейшей обработки посредством полносвязных слоев с их последующей окончательной классификацией.

## Fast R-CNN - ускоренный вариант R-CNN (2)

Извлечение соответствующих рекомендованных областей из карт признаков на выходе свертки и изменение их размера для приведения его в соответствие с размером, используемым полно связанным слоем, выполняется посредством операции субдискретизации, известной как RoI-пулинга (Region of Interest).

# Fast R-CNN - ускоренный вариант R-CNN (3)

Алгоритм:

- ① Извлечение карты признаков изображения (не для каждой гипотезы по отдельности, а для всего изображения целиком)
- ② Поиск гипотез (аналогично R-CNN на основе Selective Search)
- ③ Сопоставление каждой гипотезы с местом на карте признаков
- ④ Классификация каждой гипотезы и исправление координат ограничивающей рамки

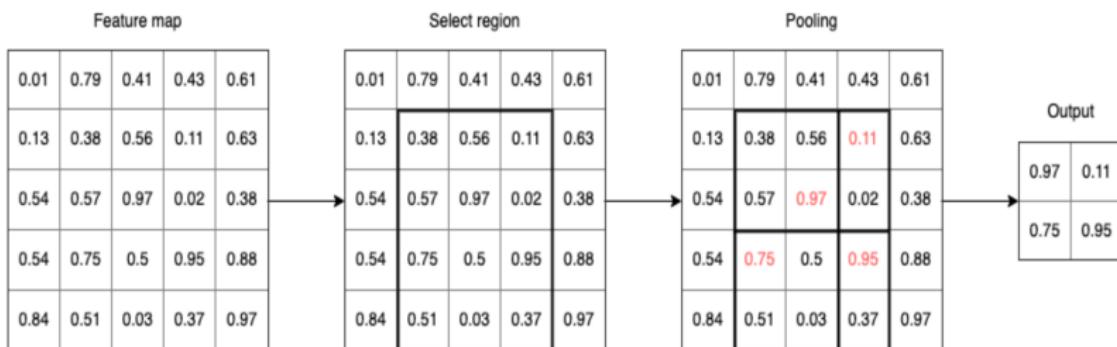
# Region of Interest (RoI) слой

- RoI - позволяет один раз обрабатывать изобр. целиком сетью, получая на выходе карту признаков, которая используется для обработки каждой гипотезы
- Основная задача RoI - сопоставление координат ограничивающих рамок с соответствующими координатами карты признаков
- RoI слой подает его на вход полносвязному слою «срез» карты признаков для определения класса и поправок к координатам

# RoI слой

- Как подать на вход полносвязному слою гипотезы разного размера и соотношения сторон?
- RoI слой преобразует изображение с размерами  $I_h \times I_w$  в размеры  $O_h \times O_w$
- Для этого исходное изображение разделяют на сетку размером  $O_h \times O_w$  (размер ячейки примерно  $\frac{I_h}{O_h} \times \frac{I_w}{O_w}$ ) и из каждой ячейки выбирают максимальное число

## RoI слой - пример



## ROI слой - пример

- Пусть имеются карта признаков размером 5x5 и нужная гипотеза на этой карте имеет координаты (1,1,4,5).
- Полносвязный слой ожидает размерность 4x1 (вытянутую 2x2 матрицу).
- Поделим гипотезу на неодинаковые блоки разной размерности (Pooling) и возьмем в каждом максимальное число (Pooling и Output).
- Таким образом становится возможным обработать целиком изображение, а потом работать с каждой гипотезой на основе карты признаков.

# Полносвязный слой и его выходы

- В предыдущей версии R-CNN использовались SVM-классификаторы, в Fast R-CNN они заменены одним SoftMax выходом размерности + 1
- Выход регрессоров обрабатывается с помощью NMS (Non-Maximum Suppression)
- Результат - вероятности принадлежности гипотезы к классам и поправки к координатам ограничивающей рамки

# Multi-task loss

Для задач регрессии ограничивающих рамок и классификации применяется специальная loss function:

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

$u$  - правильный класс

$L_{cls}$  - функция ошибки для классификации

$$L_{cls}(P, u) = -\log P_u$$

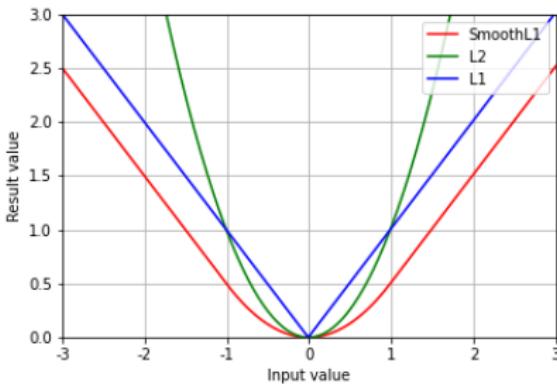
$L_{loc}$  - является SmoothL1-функцией и измеряет разницу между  $v = (v_x, v_y, v_w, v_h)$  и  $t^u = (t_x^u, t_y^u, t_w^u, t_h^u)$  значениями:

$$\text{SmoothL1} = \begin{cases} \frac{1}{2}x^2, & |x| < 1 \\ |x| - \frac{1}{2}, & \text{otherwise} \end{cases}$$

Здесь  $x$  обозначает разность целевого значения и предсказания  $t_i^u - v_i$ .

# SmoothL1

Функция SmoothL1 сочетает в себе преимущества L1 и L2 функции, является устойчивой при больших значениях и не сильно штрафует при малых значениях.



# Fast R-CNN - обучение

Формирования батча:

- ① Выбирается количество гипотез в батче  $R$
- ② Выбирается случайно изображений  $N$
- ③ Для каждого из  $N$  изображений берется  $R/N$  гипотез (равномерно на каждое изображение)
- ④ В  $R$  включаются как позитивные (25 % всего батча), так и негативные (75 % всего батча) гипотезы.  
Позитивные -  $\text{IoU} > 0.5$

# Faster R-CNN

- Улучшение - устранение зависимости от алгоритма Selective Search (в R-CNN)
- Система представляется как композиция двух модулей – определение гипотез и их обработка
- Первый модуль - Region Proposal Network (RPN)
- Второй модуль аналогично Fast R-CNN (начиная с ROI слоя)

# Faster R-CNN - алгоритм

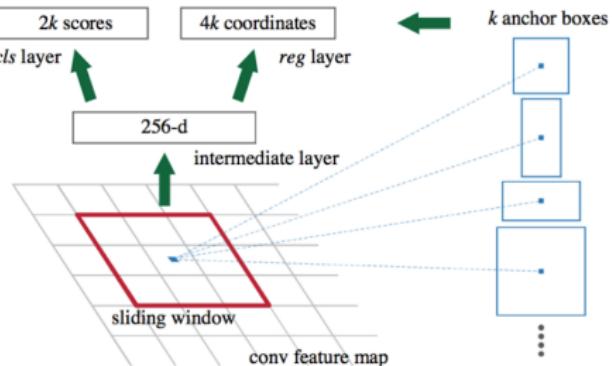
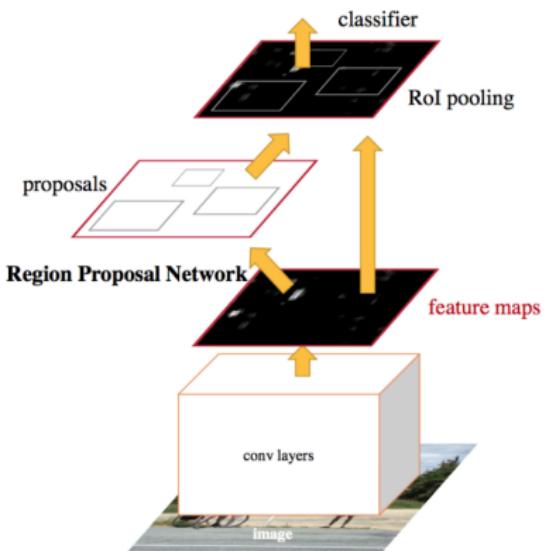
- ① Извлечение карты признаков изображения с помощью нейронной сети
- ② Генерация на основе полученной карты признаков гипотез – определение приблизительных координат и наличие объекта любого класса
- ③ Сопоставление координат гипотез с помощью RoI с картой признаков, полученной на первом шаге
- ④ Классификация гипотез (уже на определение конкретного класса) и дополнительное уточнение координат

Detection  
oooooooooo

R-CNN  
oooooooooooooooooooo

Region Proposal Network  
●oooooooooooo

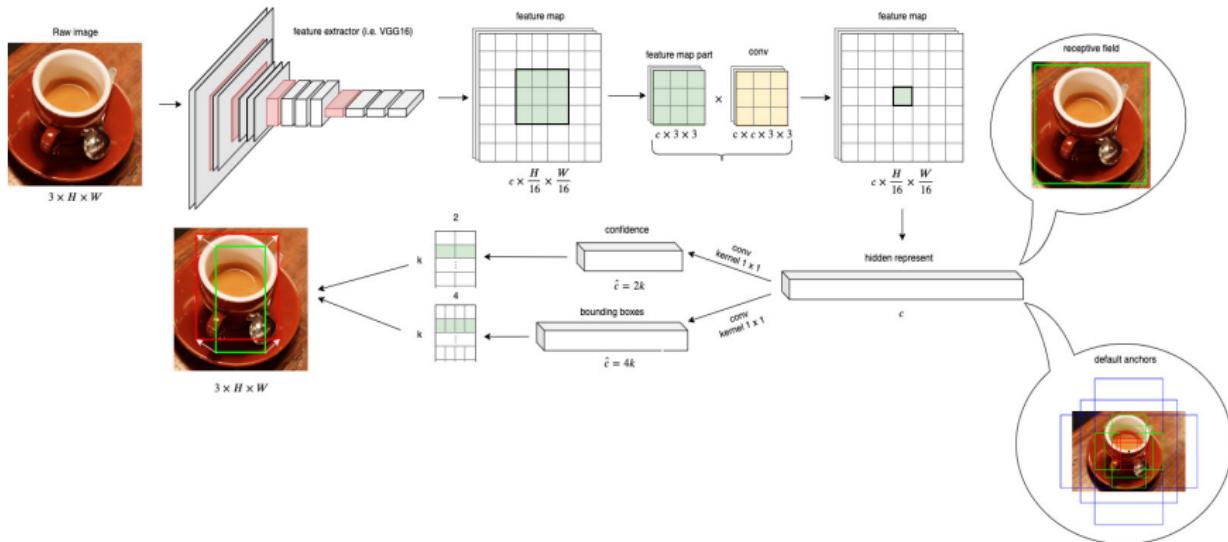
# Region Proposal Network (1)



# Region Proposal Network (2)

- Признаки извлекаются из сеть VGG16 с выходом - последний сверточный слой – conv5\_3.
- Характеристики рецептивного поля:
  - Эффективное сжатие (effective strides, ): 16
  - Размер рецептивного поля (receptive field size, ): 196
- Карта признаков в 16 раз меньше изначального размера изображения, количество каналов - 512

# Region Proposal Network (3)



# Region Proposal Network (4)

- ➊ Получим карту признаков  $c \times \frac{H}{16} \times \frac{W}{16}$  с пред. шага
- ➋ Применим сверт. слой 3x3 (отступ 1 – итоговая матрица не меняется в размерах) для доп. наращивания рецептивного поля ( $P_0 = 106, r_0 = 228$ ). Ячейке  $(i, j)$  карты признаков соответствует вектор разм.  $c$  (512).
- ➌ К каждому вектору применимы два сверт. слоя с ядром 1x1 и кол-вом вых. каналов  $\hat{c}$  (ядро отображает размерность  $c$  в  $\hat{c}$ ):
  - ➊ Первой слой (cls): параметр  $\hat{c} = 2k$  – для определения вер-ти наличия (отсутствия) объекта внутри гипотезы (бинарная классификация)
  - ➋ Второй слой (reg): параметр  $\hat{c} = 4k$  – для определения координат гипотез.

# Region Proposal Network (5)

- Полученные вектора можно переформировать в матрицы  $k \times 2$  и  $k \times 4$ , где строке  $i$  соответствуют значения для конкретной гипотезы
- Для правильного определения координат необходимо использовать якоря (anchors) и поправки к их координатам
- Якорем называют четырехугольник разного соотношения сторон (1:1, 2:1, 1:2) и размеров (128x128, 256x256, 512x512).
- Центром якоря считается центр ячейки ( $i.j$ ) карты признаков

# Функция потерь

- Для обучения Region Proposal Network используется следующее обозначение классов:
  - Позитивными являются все якоря, имеющие  $IoU > 0.7$  или имеющие наибольшее пересечение среди всех якорей (в случае, если нет  $IoU > 0.7$ )
  - Негативными являются все якоря, имеющие  $IoU < 0.3$
  - Все остальные якоря не участвуют в обучении
- Таким образом класс  $p_i^*$  якоря присуждается по следующему правилу:

$$p_i^* = \begin{cases} 1, & IoU > 0.7 \\ 0, & IoU < 0.3 \\ \text{nothing}, & \text{otherwise} \end{cases}$$

# Функция потерь

- Функция потерь

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{loc}} \sum_i p_i^* L_{reg}(t_i, t_i^*)$$

- $i$  – номер якоря;  $p_i$  – вероятность нахождения объекта в якоре;
- $p_i^*$  – правильный номер класса;
- $t_i$  и  $t_i^*$  – 4 предсказанные и ожидаемые (ground truth) поправки к координатам
- $L_{cls}(p_i, p_i^*)$  – бинарный log-loss;  $L_{reg}(t_i, t_i^*)$  – SmoothL1 лосс
- $\{p_i\}, \{t_i\}$  – выходы классификационной и регрессионной модели
- $N_{cls}$  и  $N_{loc}$  - размер мини-батча (256) и количество якорей

# Функция потерь

Для регрессии поправок к ограничивающим рамкам:

$$t_x = \frac{(x - x_a)}{w_a}, \quad t_x^* = \frac{(x^* - x_a)}{w^*}, \quad t_y = \frac{(y - y_a)}{h_a}, \quad t_y^* = \frac{(y^* - y_a)}{h_a}$$

$$t_w = \log \frac{w}{w_a}, \quad t_w^* = \log \frac{w^*}{w_a}, \quad t_h = \log \frac{h}{h_a}, \quad t_h^* = \log \frac{h^*}{h_a}$$

$x, y, w, h$  - центр, ширина и высота ограничивающей рамки  
 $x, x^*, x_a$  - предсказание, ground truth и значение якорей

# Общее обучение сети

- ➊ Тренировка RPN. Инициализация весами на ImageNet. Дообучаем на задаче определения регионов с каким-либо классом
- ➋ Тренировка Fast R-CNN. Инициализация весами на ImageNet. Дообучаем, используя гипотезы с помощью RPN сети. Задачи - уточнение координат и определение конкретного класса объекта
- ➌ Используя веса из ш.2, обучаем только RPN часть (слои до RPN, принадлежащие feature extractor, замораживаются и никак не изменяются)
- ➍ Используя веса из ш.3, обучаем слои для Fast R-CNN (остальные веса – идущие ранее или относящиеся к RPN — заморожены)

# Предсказание

- 1 Изображение поступает на вход нейронной сети, генерируя карту признаков
- 2 Каждая ячейка карты признаков обрабатывается RPN, результат - поправки к положению якорей и вероятность наличия объекта любого класса
- 3 Предсказанные рамки на основе карты признаков и RoI слоя поступают на обработку Fast R-CNN
- 4 На выходе - класс объектов и их точное положение на изображении

# R-CNN

- ➊ Использование Selective Search как генератор гипотез
- ➋ Использование SVM + Ridge для классификации и регрессии гипотез
- ➌ Запуск нейронной сети для обработки каждой гипотезы по отдельности.
- ➍ Низкая скорость работы.

# Fast R-CNN

- ➊ Нейронная сеть запускается только один раз на изображение – все гипотезы проверяются на основе единой карты признаков
- ➋ «Умная» обработка гипотез разного размера за счет RoI слоя
- ➌ Замена SVN на SoftMax слой
- ➍ Возможность параллельной работы классификации и регрессии.

# Faster R-CNN

- ① Генерация гипотез с помощью специального отдельно дифференцируемого модуля
- ② Изменения в процессе обработки изображения, связанные с появлением RPN модуля
- ③ Самая быстрая из этих трех моделей
- ④ Является одной из самых точных и по сей день.

Detection  
ooooooooo

R-CNN  
oooooooooooooooooooooooooooo

Region Proposal Network  
oooooooooooo●

# Вопросы

?