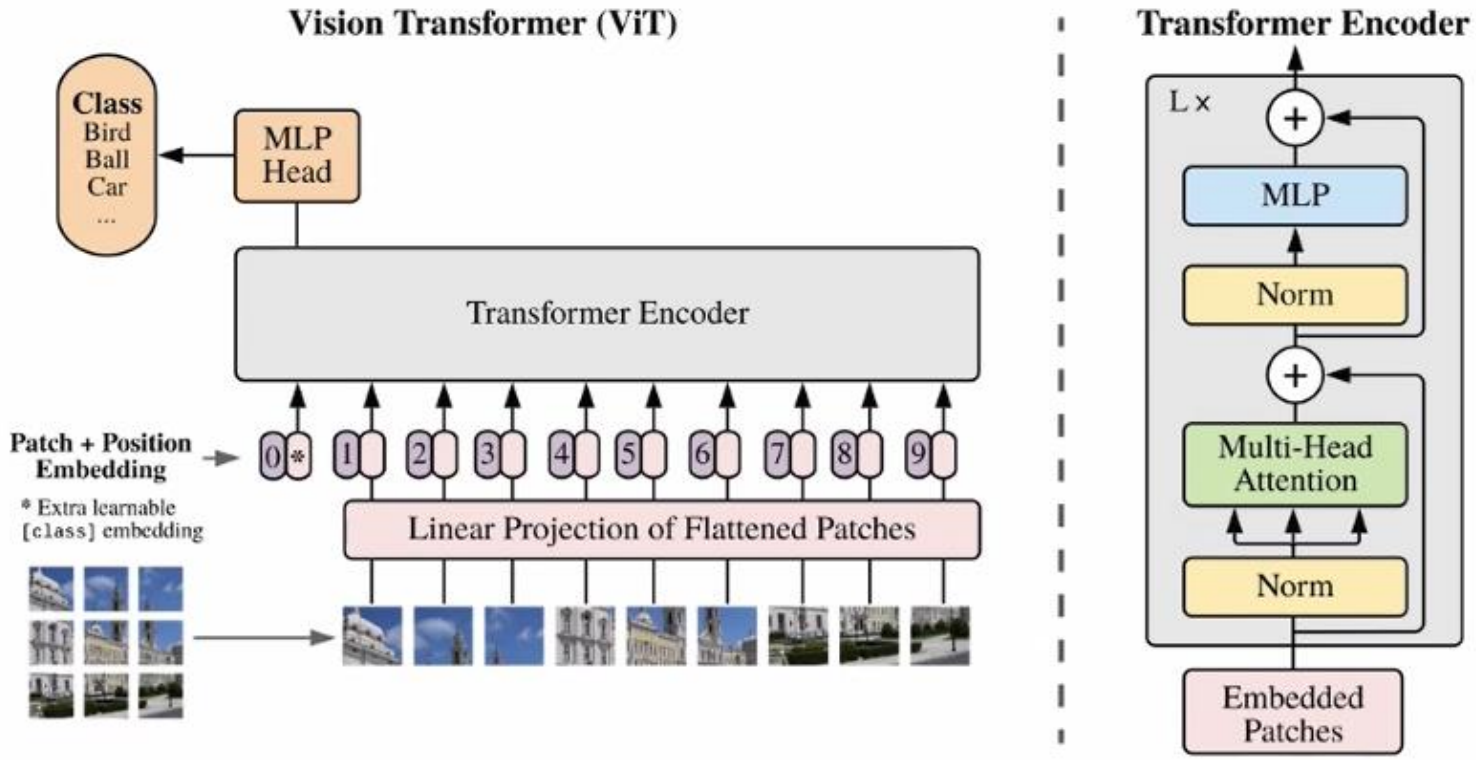
DETR: End-to-End Object Detection with Transformers

Facebook AI France, 2020: Nicolas Carion, Francisco Massa et al.

Мария Тимонина БПМИ192

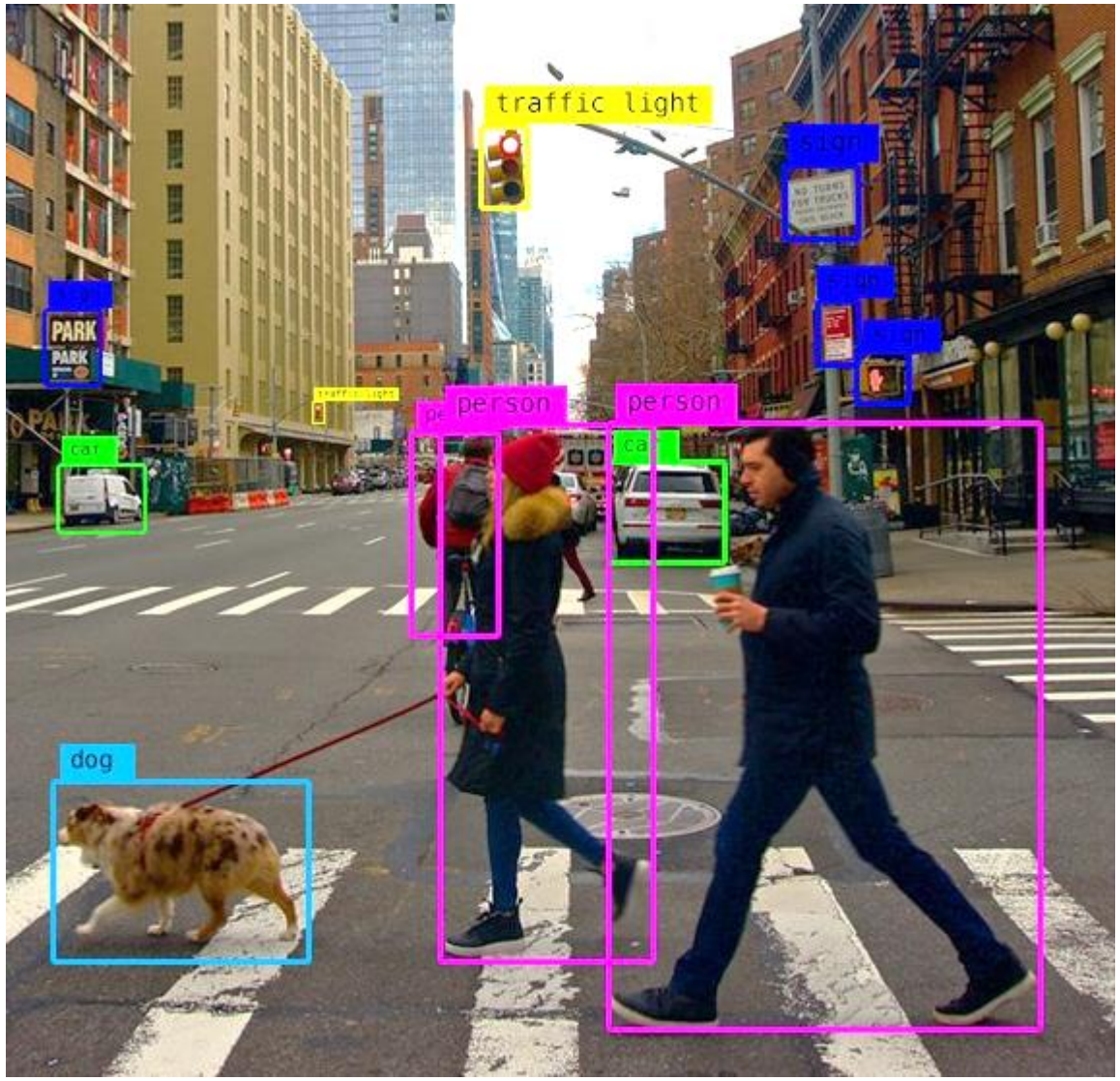
Vision Transformer (ViT) [2020, Dosovitskiy]

- Патчи с помощью обучаемой свертки вытягиваются в векторы
- Изображение как фраза из 256 слов
- Нулевой эмбеддинг через backprop выучит класс всей картинки



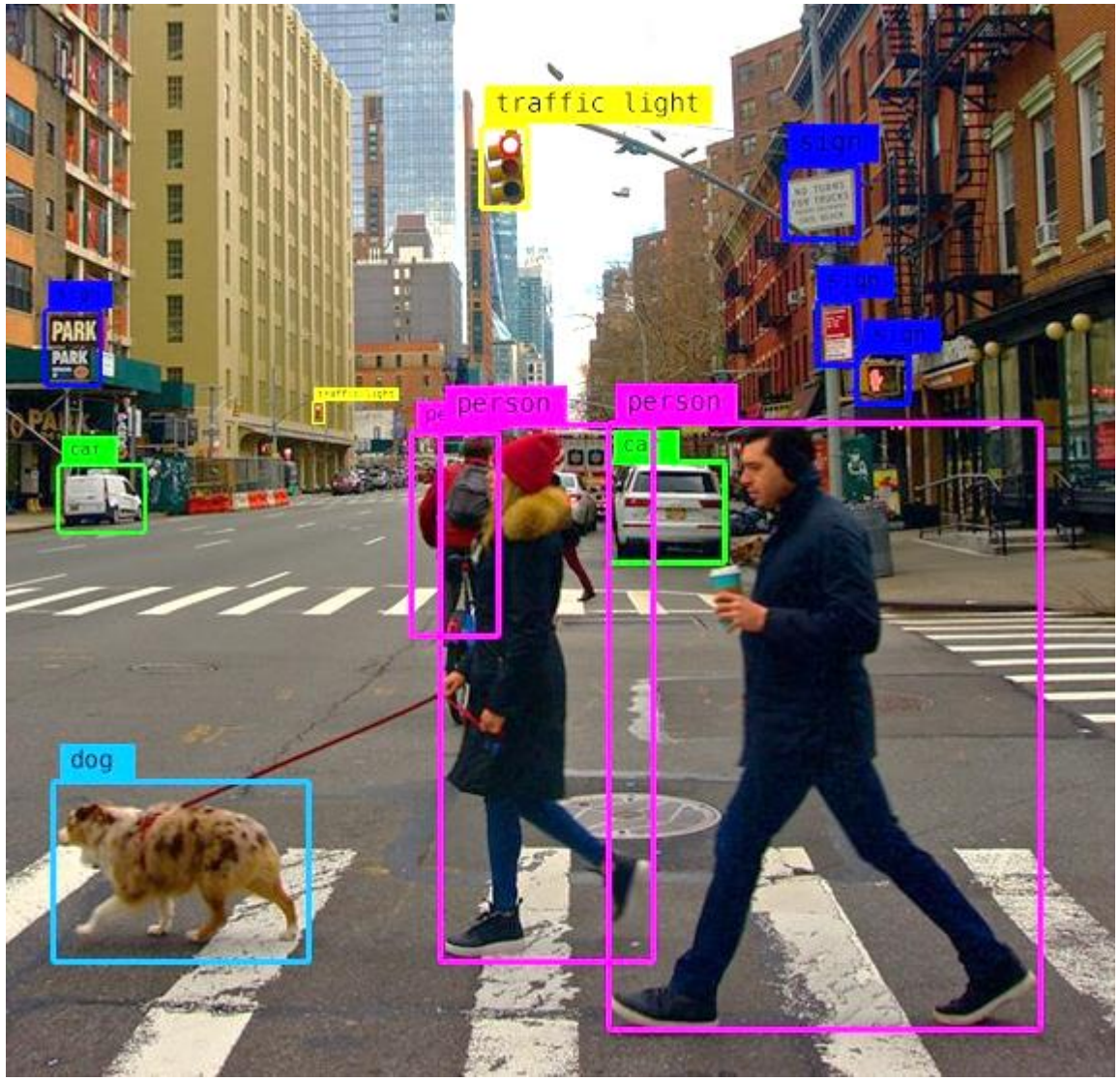
Object Detection

- 2D-оболочка
- Хотим ограничить в bounding box все объекты заранее известного списка классов и разметить их
- В отличие от сегментации, обнаруживаем только объекты (без фона, stuff)



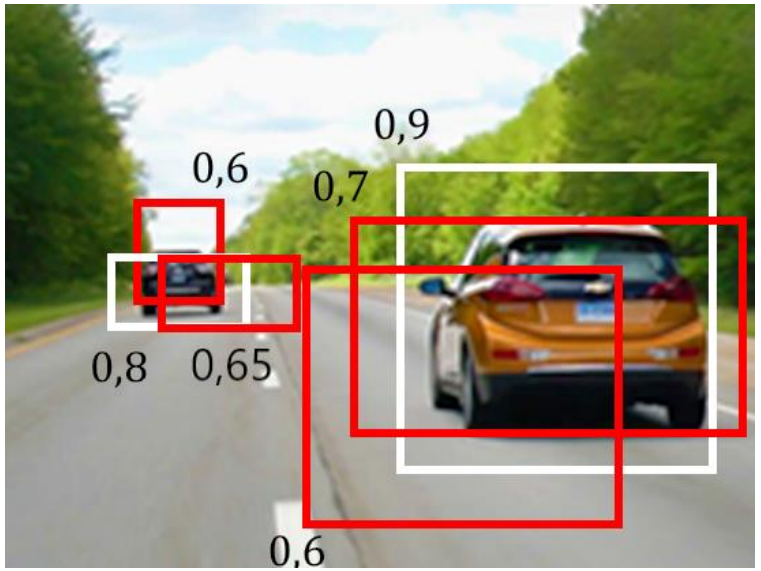
Правильно отделить объекты не просто

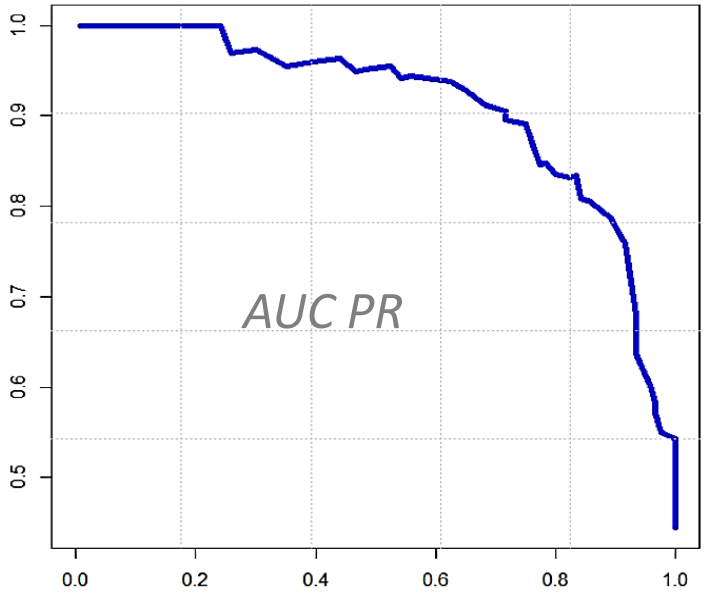
- Вариантов рамки объекта – квадрат от числа пикселей
- Правильный контейнер может иметь разное отношение сторон
- Объекты на фото могут перекрываться другими

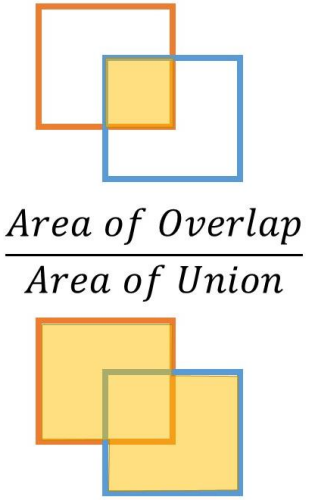


Как оценить качество ответа детектора?

- Уверенность модели в ответе
  Sure > threshold (0.5-0.7)
- [Recall] прямоугольник-кандидат считаем правильным, если:
  IoU > threshold
- [Precision] учитываем объект **один раз**, остальные – false positives

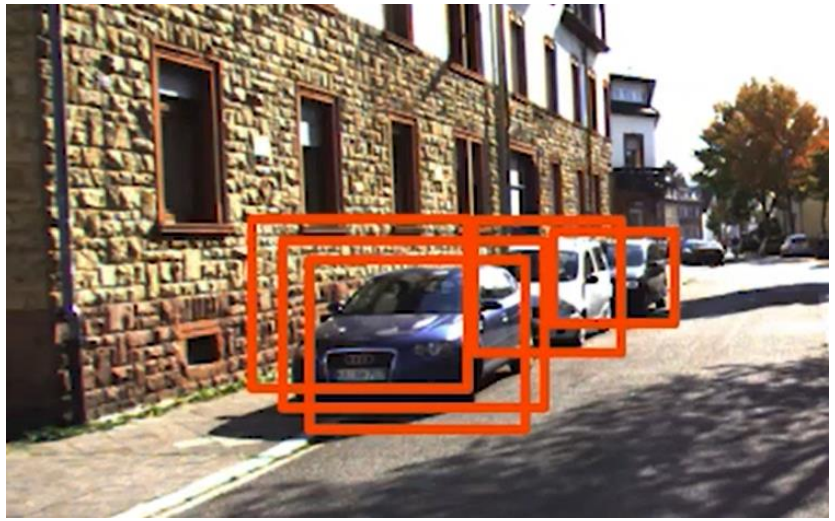






$$\frac{Area\ of\ Overlap}{Area\ of\ Union}$$

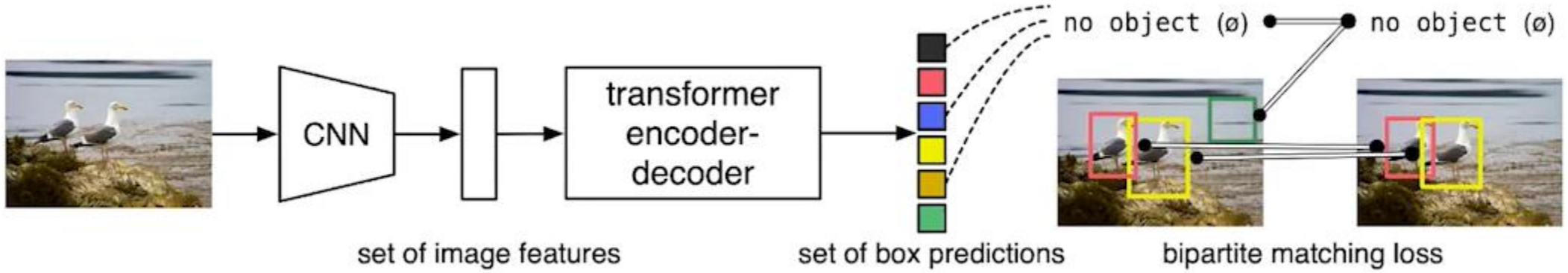
Non-Maximum Suppression



- Дано множество пар {(Coords_i, score_i)}
- Одному истинному объекту должно соответствовать не более одной рамки

1. Сортируем пары по убыванию score_i
2. Перебираем прямоугольники
    1. Выбираем очередную рамку в ответ
    2. Все рамки из оставшихся в очереди, у которых IoU с выбранной больше threshold, удаляем из очереди

DETR: Detection using Transformers



N=100 выходов вида
(x_center, y_center, width, height, class, score)

Как считать ошибку? Нужно сопоставить результат и разметку

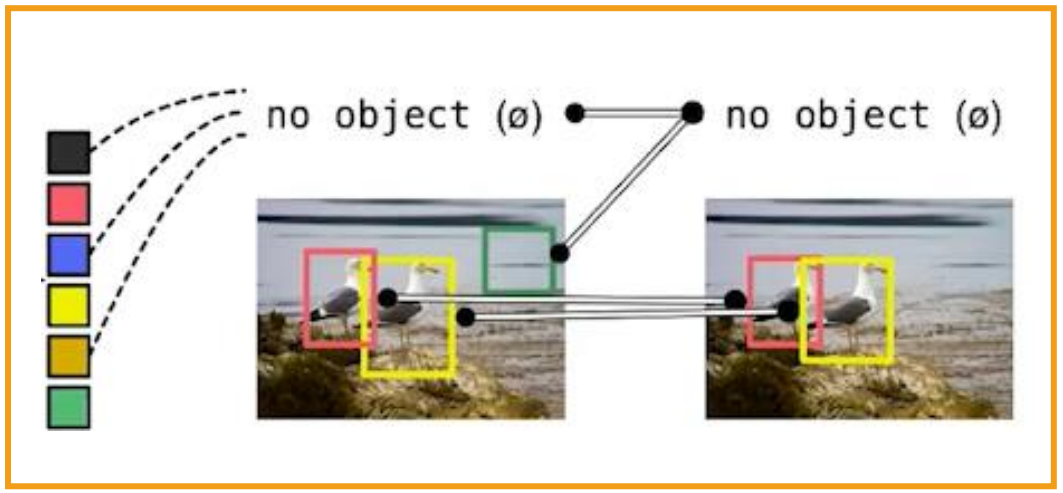
https://arxiv.org/pdf/2005.12872.pdf

Bipartite Matching

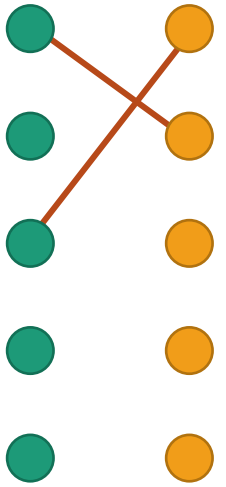
Венгерский алгоритм

$$\hat{\sigma} = \arg\min_{\sigma \in \mathfrak{S}_N} \sum_{i}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)})$$

$$-\mathbb{1}_{\{c_i \neq \varnothing\}} \hat{p}_{\sigma(i)}(c_i) + \mathbb{1}_{\{c_i \neq \varnothing\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)})$$



- Лучшая перестановка выходов трансформера
- Максимизирует уверенности в ответе
- И минимизирует потери от искажения рамки

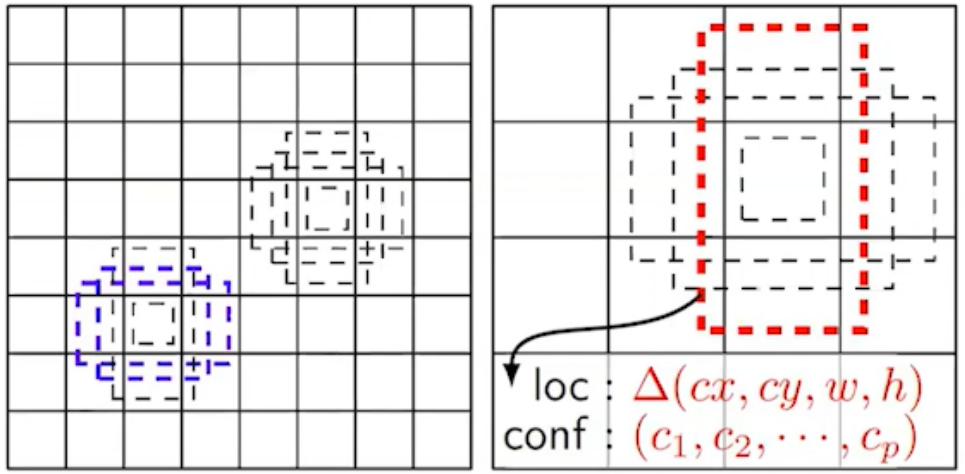


$$b_i \in [0, 1]^4$$

https://github.com/facebookresearch/detr

Bounding Box Loss

- Маленькие и большие рамки объектов, в отличие от других (якорных) архитектур обучаются вместе
- Решение от авторов: обобщенная IoU



SSD: Single-Shot Detector (якорная)

$$GIoU = IoU - \frac{|C \backslash (A \cup B)|}{|C|}$$

$$b_i \in [0,1]^4$$

$$\mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)}) = \lambda_{\text{iou}} \mathcal{L}_{\text{iou}}(b_i, \hat{b}_{\sigma(i)}) + \lambda_{\text{L1}} ||b_i - \hat{b}_{\sigma(i)}||_1$$

Финальная функция потерь

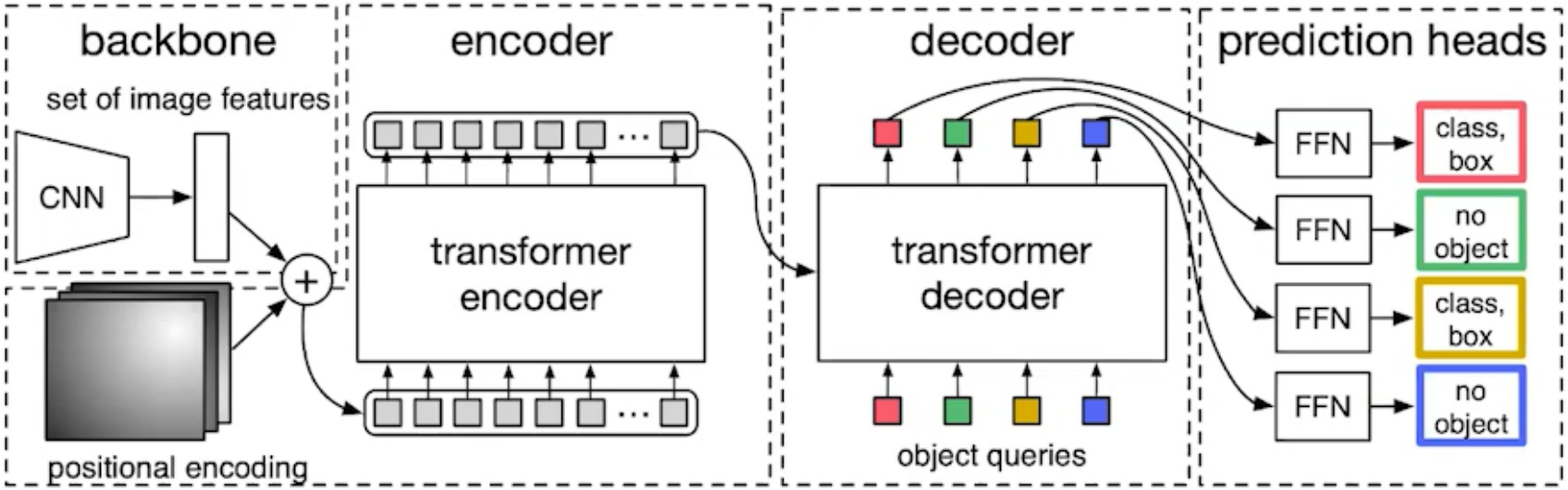
$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^{N} \left[-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{\{c_i \neq \varnothing\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}}(i)) \right]$$

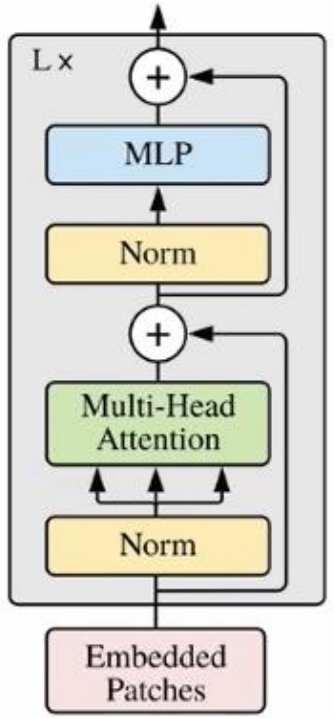
$y_i = (c_i, b_i)$ • Правильный прямоугольник задается как

$b_i \in [0, 1]^4$ • Координаты центра, высота и ширина

c_i • Метка класса

DETR: Encoder & Decoder





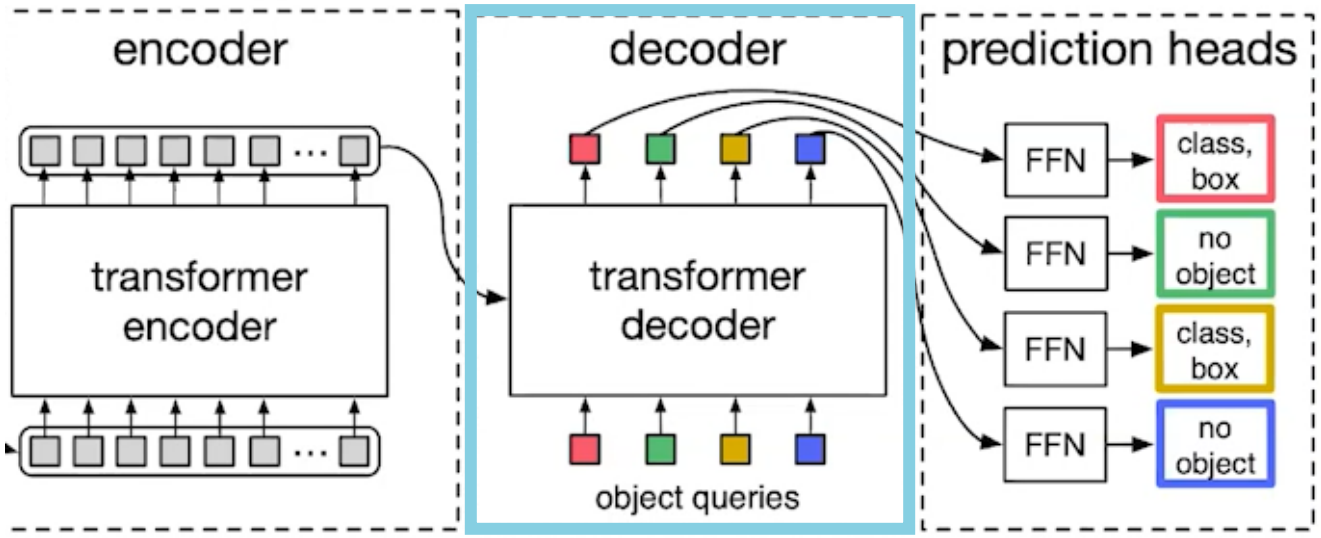
- Вход: векторы признаков
- Стандартный трансформер:
  - multi-head self-attention
  - Полносвязный слой

- Вход: N токенов, N = 100
- Для каждого токена предсказать координаты рамки и класс

  + степень уверенности в ответе
- No object class

https://arxiv.org/pdf/2005.12872.pdf

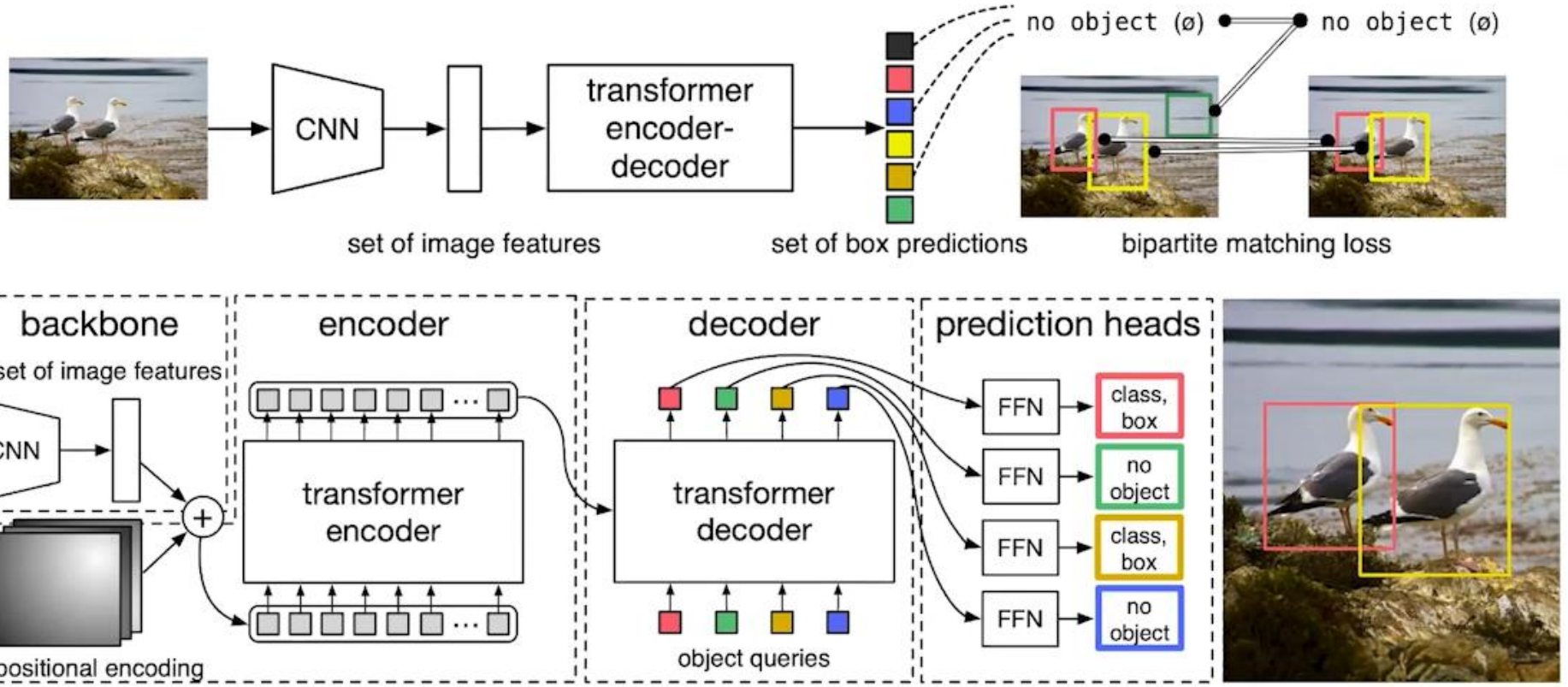
DETR Decoder



$$\text{Attention}(Q, K, V) = \\ = \text{Softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

- Берет на вход N=100 различных обучаемых токенов
- Есть как Self-Attention блоки, так и
- Encoder-Decoder блоки (K, V из энкодера и Q из декодера)

Полная архитектура DETR



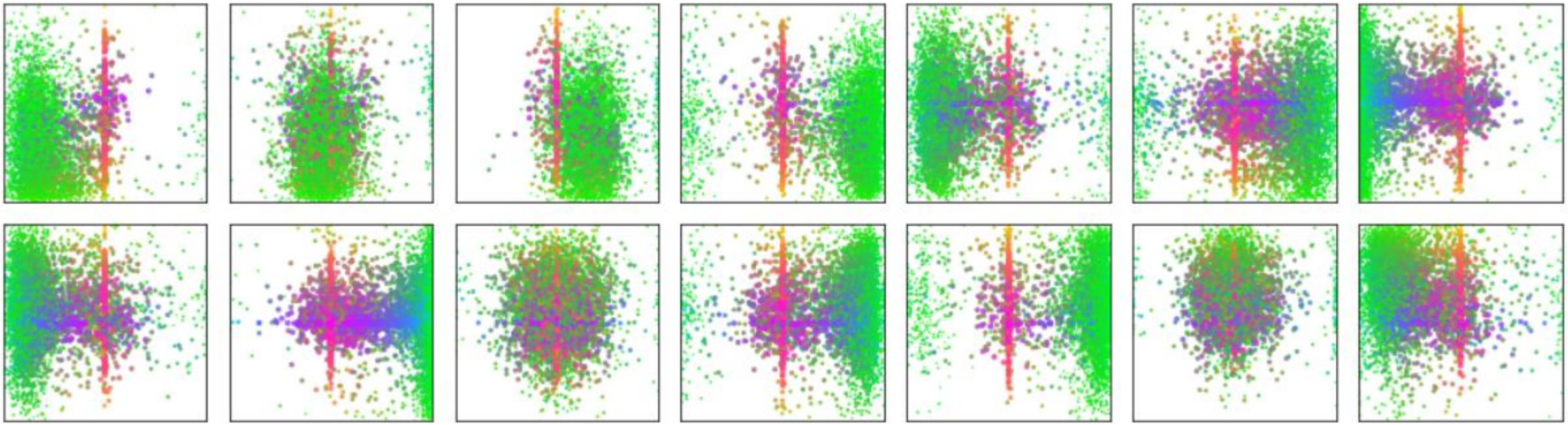
- CNN – ResNet50 или ResNet101
- FFN для предсказания – 3-слойный MLP + ReLU для координат центра, softmax для уверенности в классе

Сравнение с другими моделями

Model	GFLOPS/FPS	#params	AP	AP_S	AP_M	AP_L
Faster RCNN-DC5	320/16	166M	39.0	21.4	43.5	52.5
Faster RCNN-FPN	180/26	42M	40.2	24.2	43.5	52.0
Faster RCNN-R101-FPN	246/20	60M	42.0	25.2	45.6	54.6
Faster RCNN-DC5+	320/16	166M	41.1	22.9	45.9	55.0
Faster RCNN-FPN+	180/26	42M	42.0	26.6	45.4	53.4
Faster RCNN-R101-FPN+	246/20	60M	44.0	**27.2**	48.1	56.0
DETR	86/28	41M	42.0	20.5	45.8	61.1
DETR-DC5	187/12	41M	43.3	22.5	47.3	61.1
DETR-R101	152/20	60M	43.5	21.9	48.0	61.8
DETR-DC5-R101	253/10	60M	**44.9**	23.7	**49.5**	**62.3**

- Сверточная сеть лучше работает на маленьких объектах, но глобальное внимание показывает лучшее качество на средних и больших
- AP – Average Precision

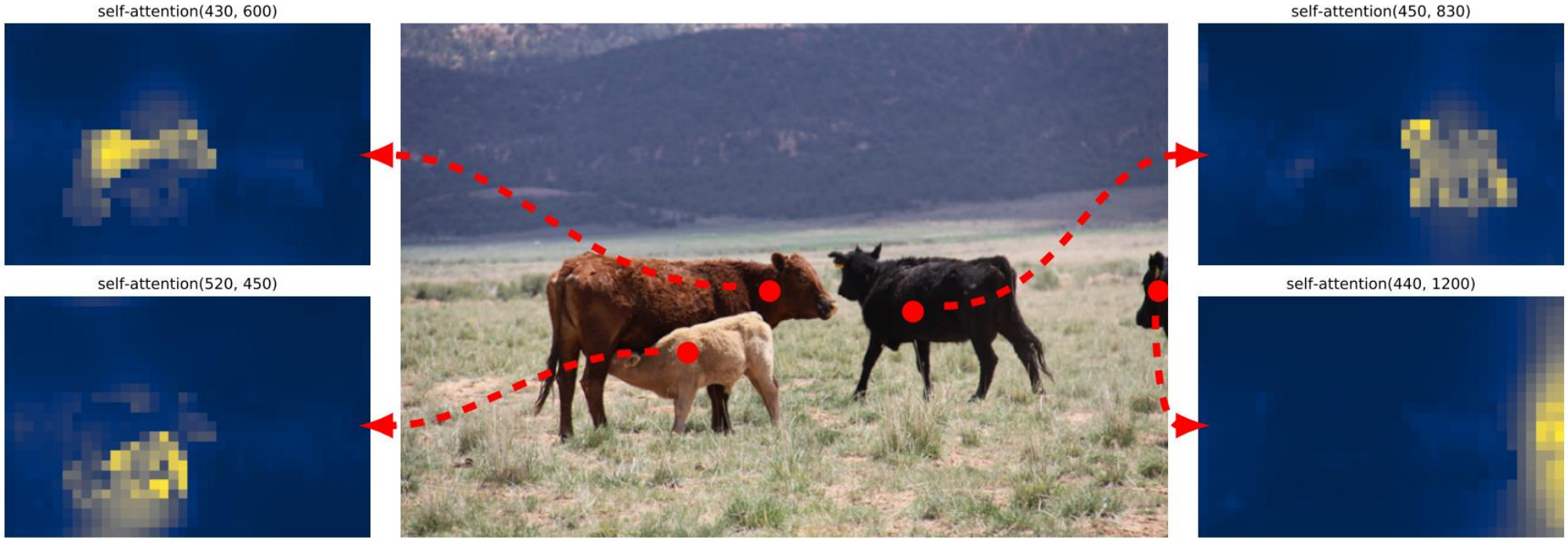
Токены декодера:
как выучивают позицию рамки?



Для 14 выходов из N=100

Визуализация центров всех предсказанных ими прямоугольников
для набора данных валидации из COCO 2017

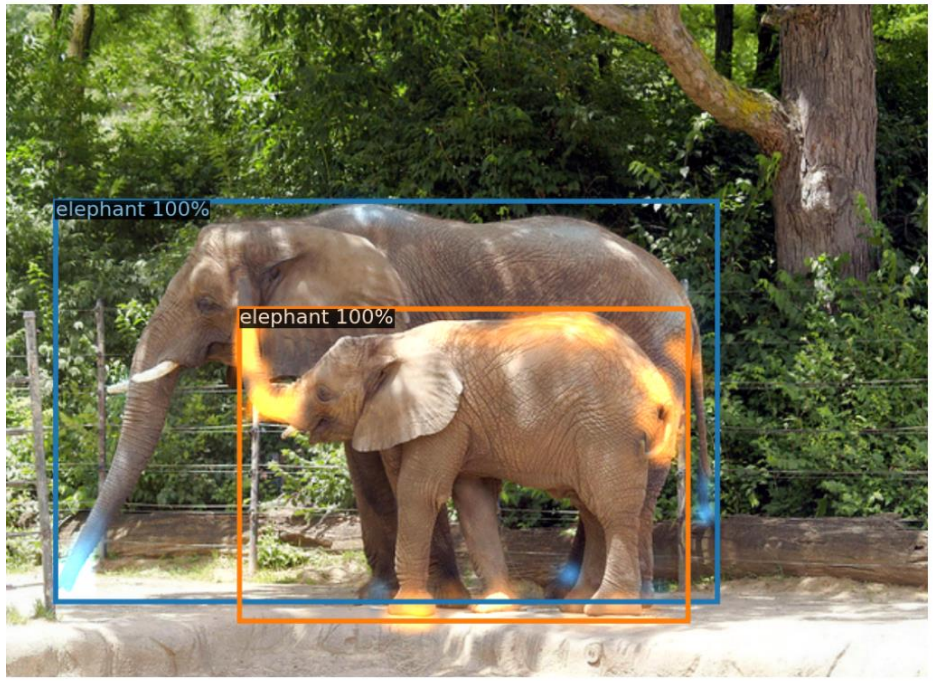
Encoder отделяет объекты от фона

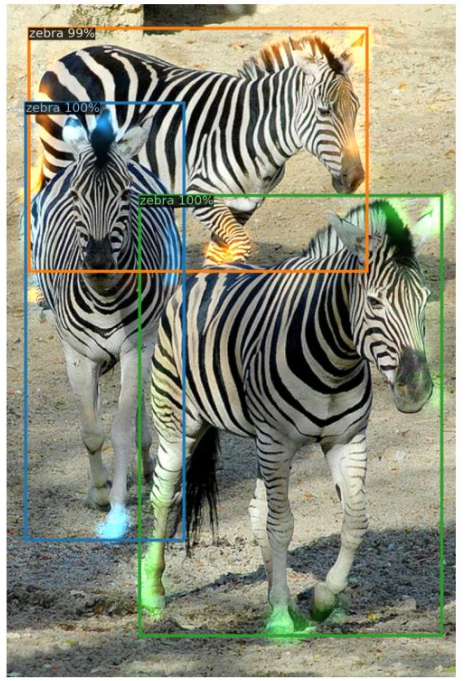


- Карты распространения внимания для выходов первого блока

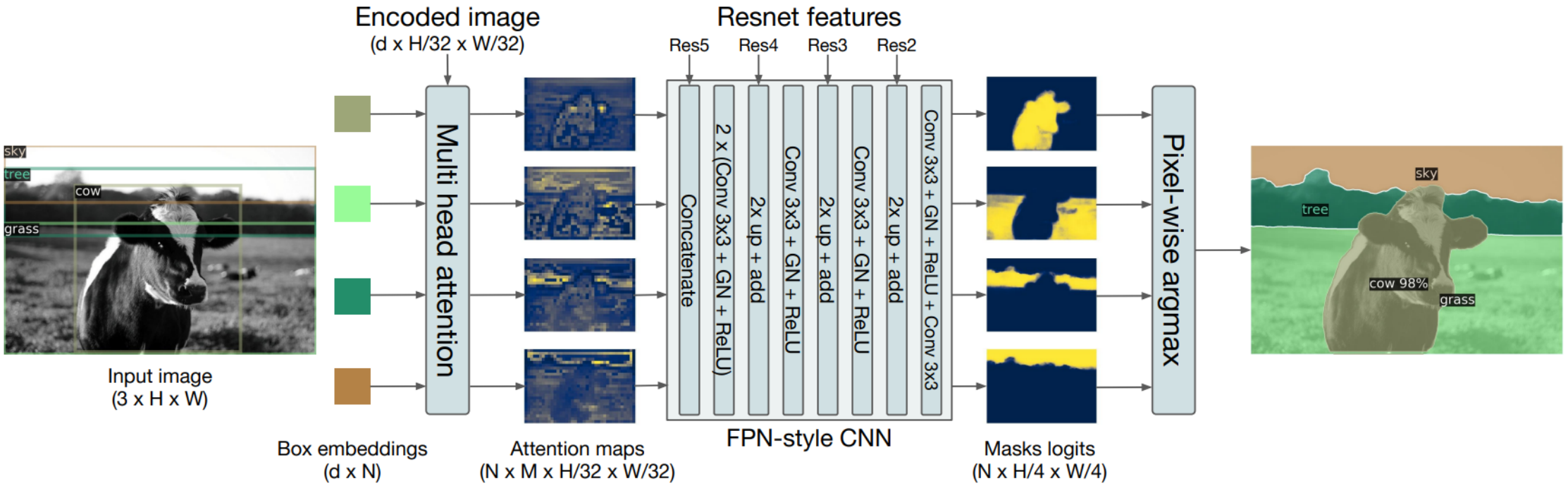
Decoder уже смотрит на выступы – яркие черты класса

- Визуализация ключей декодера (выходов энкодера), наиболее релевантных для того токена, из которого получилась рамка объекта
- В фокусе оказываются края объекта – ноги, холка





Задача сегментации с DETR



- Выделяются классы не только объектов, но и фона
- Карты распространения внимания (выходы декодера) подаются в сверточную голову

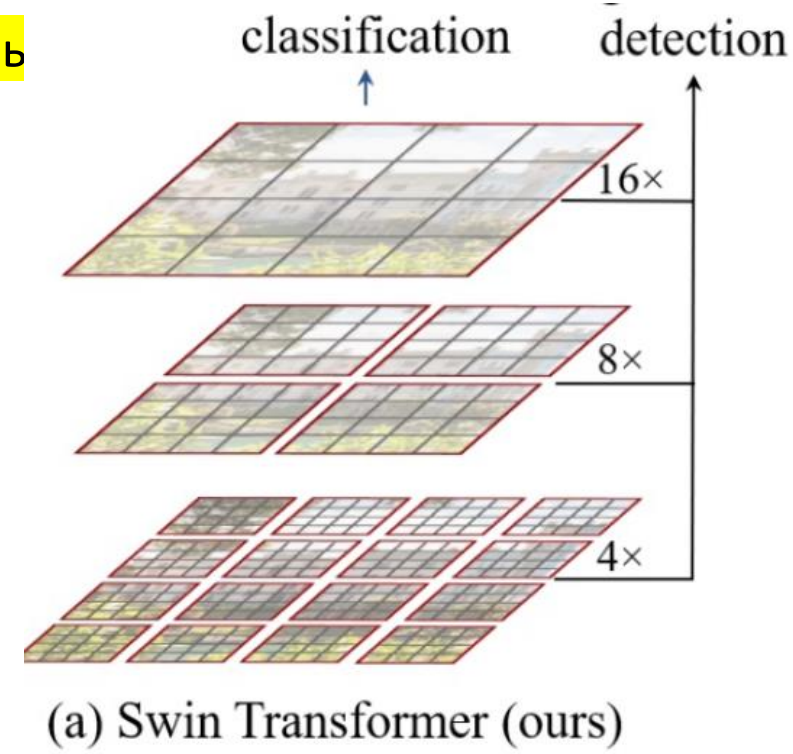
Как идея развивалась дальше?

- 2017 – Attention is All You Need (машинной перевод)
- 2020 – ViT (глобальное внимание для классификации картинок)
- 2020 – DETR (детекция объектов, тоже глобально) – мы здесь
- 2021 – SWIN (Shifted WINdows, **локальное внимание**

  best paper ICCV21)
- Апрель 2021 – DINO (DETR with Improved

  DeNoising Anchor Boxes)
- Object Detection SOTA сейчас **DINO** (Swin-L, multi-scale)

box AP	AP50	AP75	APS	APM	APL
63.3	80.8	69.9	46.7	66.0	76.5
			23.7	49.5	62.3



(a) Swin Transformer (ours)