9/12

Recall:

Faster R-CNN tried to "merge" RPN as part of one network (sliding window based)

Each sliding window: *k* anchor boxes

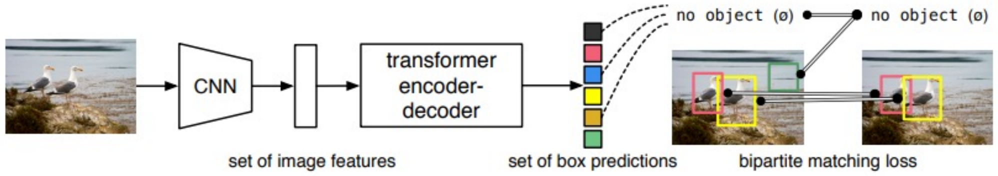
Predict bbox regression and classification

DETR main principles:

One-shot detection – do away with surrogate predictions, postprocessing (NMS)

Transformer (attention, attention, attention)

Direct set prediction (set of bboxes)



Key Ingredients

- Set prediction loss that enforce unique matches
  - Recall in Faster R-CNN, there often are multiple overlapping bboxes
- An architecture that predicts (in a single pass) a set of objects and models their relation.

y the ground truth set of objects, and $\hat{y} = \{\hat{y}_i\}_{i=1}^{N}$

N is a large number such that an image will not contain more than *N* objects

consider y also as a set of size N padded with $\varnothing$ (no object)

Step 1 – Hungarian Matching:

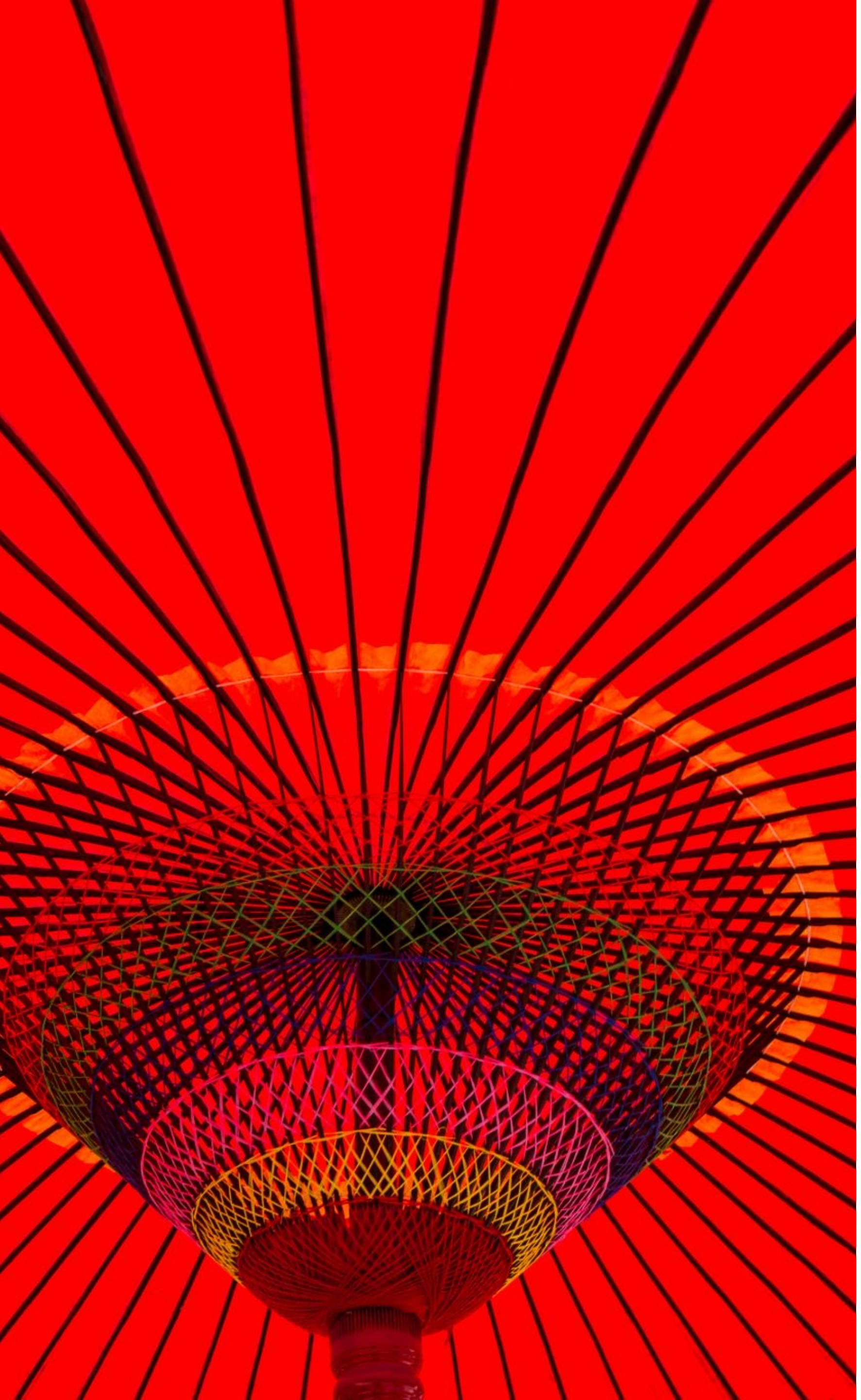
$$\hat{\sigma} = \underset{\sigma \in \mathfrak{S}_N}{\arg\min} \sum_{i}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)})$$

$$y_i = (c_i, b_i)$$

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

$\lambda_{\text{iou}} \mathcal{L}_{\text{iou}}(b_i, \hat{b}_{\sigma(i)}) + \lambda_{\text{L1}} ||b_i - \hat{b}_{\sigma(i)}||_1$ where $\lambda_{\text{iou}}, \lambda_{\text{L1}} \in \mathbb{R}$ are hyperparameters

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



Details

- Step 2 – Hungarian Loss:

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

where $\hat{\sigma}$ is the optimal assignment computed in the first step (1)

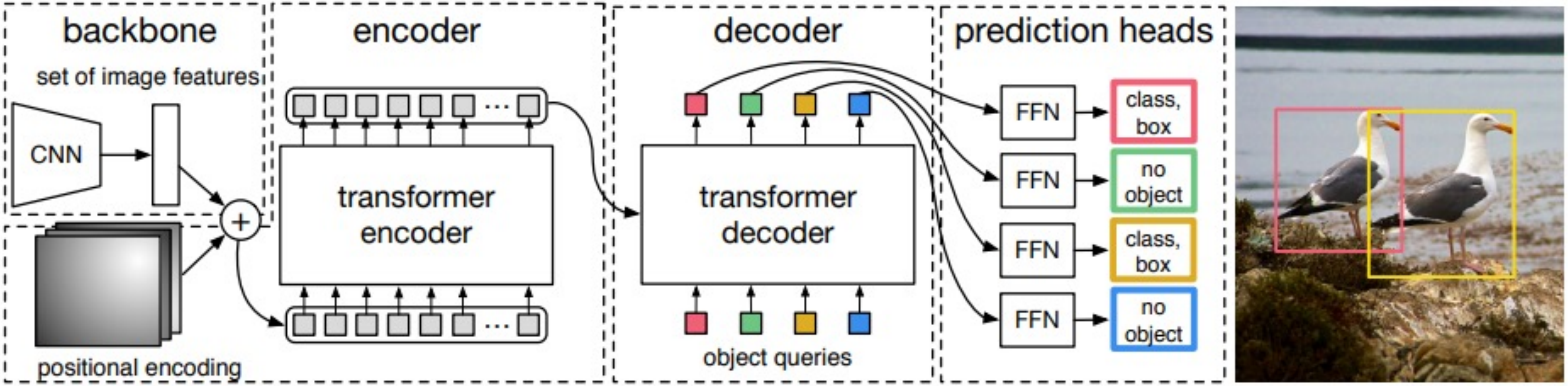
down-weight the log-probability term when $c_i = \varnothing$ by a factor 10

DETR Architecture

The input images are batched together, applying 0-padding adequately to ensure they all have the same dimensions (H_0, W_0) as the largest image of the batch.

multi-head self-attention module and a feed forward network (FFN)

decodes the N objects in parallel at each decoder layer



$f \in \mathbb{R}^{C \times H \times W}$ $C = 2048$ and $H, W = \frac{H_0}{32}, \frac{W_0}{32}$

box coordinates and class labels

1x1 convolution

$z_0 \in \mathbb{R}^{d \times H \times W}$

$d \times HW$

Training Details

Technical details. We train DETR with AdamW [26] setting the initial transformer's learning rate to 10^{-4}, the backbone's to 10^{-5}, and weight decay to 10^{-4}. All transformer weights are initialized with Xavier init [11], and the backbone is with ImageNet-pretrained ResNet model [15] from TORCHVISION with frozen batchnorm layers. We report results with two different backbones: a ResNet-50 and a ResNet-101. The corresponding models are called respectively DETR and DETR-R101. Following [21], we also increase the feature resolution by adding a dilation to the last stage of the backbone and removing a stride from the first convolution of this stage. The corresponding models are called respectively DETR-DC5 and DETR-DC5-R101 (dilated C5 stage). This modification increases the resolution by a factor of two, thus improving performance for small objects, at the cost of a 16x higher cost in the self-attentions of the encoder, leading to an overall 2x increase in computational cost. A full comparison of FLOPs of these models and Faster R-CNN is given in Table 1.

We use scale augmentation, resizing the input images such that the shortest side is at least 480 and at most 800 pixels while the longest at most 1333 [50]. To help learning global relationships through the self-attention of the encoder, we also apply random crop augmentations during training, improving the performance by approximately 1 AP. Specifically, a train image is cropped with probability 0.5 to a random rectangular patch which is then resized again to 800-1333. The transformer is trained with default dropout of 0.1. At inference time, some slots predict empty class. To optimize for AP, we override the prediction of these slots with the second highest scoring class, using the corresponding confidence. This improves AP by 2 points compared to filtering out empty slots. Other training hyperparameters can be found in section A.4. For our ablation experiments we use training schedule of 300 epochs with a learning rate drop by a factor of 10 after 200 epochs, where a single epoch is a pass over all training images once. Training the baseline model for 300 epochs on 16 V100 GPUs takes 3 days, with 4 images per GPU (hence a total batch size of 64). For the longer schedule used to compare with Faster R-CNN we train for 500 epochs with learning rate drop after 400 epochs. This schedule adds 1.5 AP compared to the shorter schedule.

Assignment Due 9/19

Model	GFLOPS/FPS	#params	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster RCNN-DC5	320/16	166M	39.0	60.5	42.3	21.4	43.5	52.5
Faster RCNN-FPN	180/26	42M	40.2	61.0	43.8	24.2	43.5	52.0
Faster RCNN-R101-FPN	246/20	60M	42.0	62.5	45.9	25.2	45.6	54.6
Faster RCNN-DC5+	320/16	166M	41.1	61.4	44.3	22.9	45.9	55.0
Faster RCNN-FPN+	180/26	42M	42.0	62.1	45.5	26.6	45.4	53.4
Faster RCNN-R101-FPN+	246/20	60M	44.0	63.9	**47.8**	**27.2**	48.1	56.0
DETR	86/28	41M	42.0	62.4	44.2	20.5	45.8	61.1
DETR-DC5	187/12	41M	43.3	63.1	45.9	22.5	47.3	61.1
DETR-R101	152/20	60M	43.5	63.8	46.4	21.9	48.0	61.8
DETR-DC5-R101	253/10	60M	**44.9**	**64.7**	47.7	23.7	**49.5**	**62.3**