CVPDL HW1 R12945072 鄒雨笙

1. The architecture of the object detector, detr

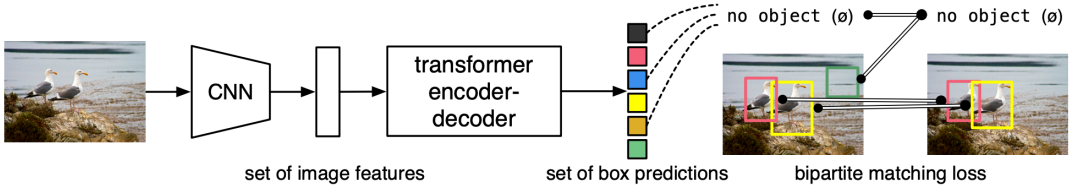

Fig. 1: DETR directly predicts (in parallel) the final set of detections by combining a common CNN with a transformer architecture. During training, bipartite matching uniquely assigns predictions with ground truth boxes. Prediction with no match should yield a "`no object`" ($\varnothing$) class prediction.

2. Implementation details

A. parameter settings: num_class = 8

B. Train the model: python -m torch.distributed.launch --nproc_per_node=1 --use_env main.py --coco_path /mnt/lab/2.course/112-1/CVPDL_hw/hw1/detr/hw1_dataset --epoch 350 --output_dir="output" --resume="./detr-r50_8.pth"

C. Use inference.py to use the data of validation and save the result in output.json

```
Accumulating evaluation results...
DONE (t=0.10s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.429
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.755
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.416
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.101
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.327
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.581
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.223
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.461
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.555
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.251
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.449
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.688
Training time 5:47:22
```

3. Table of your performance for validation set (AP, AP50, AP75)

```
100%|                                                                                | 127/127 [00:00<00:00, 198.04it/s]
{'map': tensor(0.4170), 'map_50': tensor(0.7173), 'map_75': tensor(0.4145), 'map_small': tensor(0.1334
), 'map_medium': tensor(0.3017), 'map_large': tensor(0.5628), 'mar_1': tensor(0.2184), 'mar_10': tenso
r(0.4296), 'mar_100': tensor(0.4918), 'mar_small': tensor(0.1775), 'mar_medium': tensor(0.3815), 'mar_
large': tensor(0.6224), 'map_per_class': tensor(-1.), 'mar_100_per_class': tensor(-1.), 'classes': ten
sor([1, 2, 3, 4, 5, 6, 7], dtype=torch.int32)}
```

4. Visualization