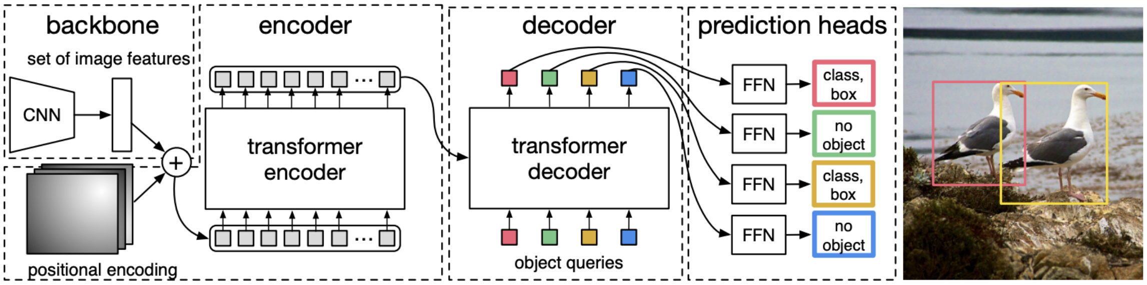
hw1_R12944044

Architecture of object detector



Implement details

1. 先將作業給的Dataset放入detr資料夾，並載入 https://github.com/facebookresearch/detr.git 所提供的 pre-trained model. （需自行下載），並按造此repo將Dataset更改符合coco format

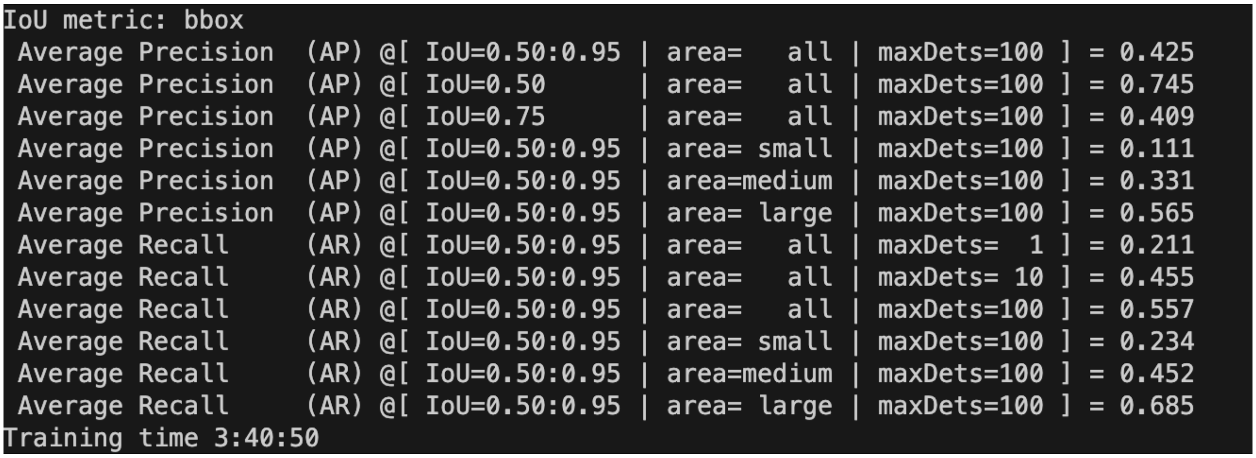
	name	backbone	schedule	inf_time	box AP	url	size
0	DETR	R50	500	0.036	42.0	model | logs	159Mb

2. 將此model透過detr的 `change_class_num.py` 將model class做更改以符合作業dataset只有8個class。
3. 更改 models/detr.py 裡面的num_classes = 8
4. train model

```
python -m torch.distributed.launch --nproc_per_node=1 --use_env main.py --coco_path /local/tomlord1122/1121-CVPDL/detr/hw1_dataset --epoch
```

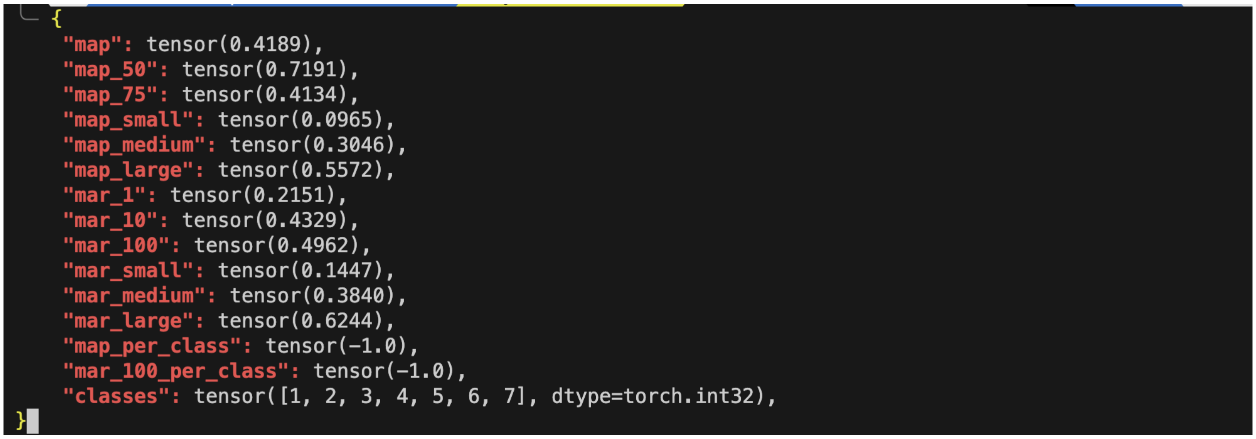
5. `inference.py` 將訓練好的model用validation image data測試並儲存為output.json

Training result



```
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.425
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.745
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.409
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.111
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.331
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.565
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.211
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.455
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.557
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.234
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.452
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.685
Training time 3:40:50
```

Performance for validation set (AP, AP50, A75)



```
{
  "map": tensor(0.4189),
  "map_50": tensor(0.7191),
  "map_75": tensor(0.4134),
  "map_small": tensor(0.0965),
  "map_medium": tensor(0.3046),
  "map_large": tensor(0.5572),
  "mar_1": tensor(0.2151),
  "mar_10": tensor(0.4329),
  "mar_100": tensor(0.4962),
  "mar_small": tensor(0.1447),
  "mar_medium": tensor(0.3840),
  "mar_large": tensor(0.6244),
  "map_per_class": tensor(-1.0),
  "mar_100_per_class": tensor(-1.0),
  "classes": tensor([1, 2, 3, 4, 5, 6, 7], dtype=torch.int32),
}
```

Visualization