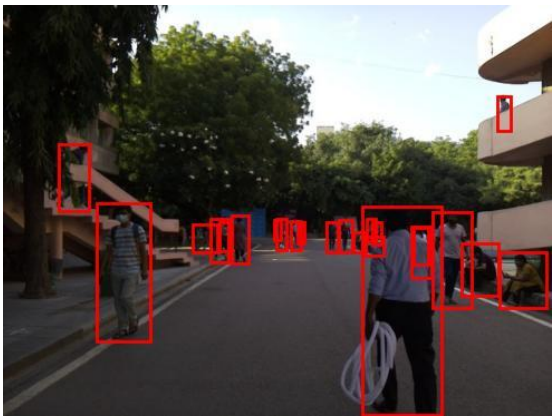
People Detection using DINO

Github link - https://github.com/surajrao2003/DINO_Object_Detection

I followed all the instructions mentioned in the assignment and setup the code using the GitHub repo provided.

Steps followed -

1) **Understanding the data -**
   Visualized the images in the dataset provided (200 images).
   This was the script I used for checking the ground truth labels using the json file provided.
   https://colab.research.google.com/drive/1zLcipOgcSFb9klYM2-i8bLEQi3DOY0Nc?usp=drive_link



2) **Repository setup -**
   Followed all the instruction given in the readme of the github repo, cloned it and installed all the requirements on google colab.

3) Since I do not have access to GPU on my laptop, I used **google colab**. It had limitations on GPU usage but I used multiple accounts to get access to GPU.

4) **Pretrained model -**
   Downloaded the pretrained model from the link provided (DINO-4scale model with the ResNet-50 (R50) backbone)
   https://drive.google.com/file/d/1eeAHgu-fzp28PGdIjeLe-pzGPMG2r2G_/view?usp=drive_link

5) Run evaluation script on the validation set -

Running evaluation script using pretrained model (given in repo)

+ Code + Text

```
[13] !bash scripts/DINO_eval.sh /content/drive/MyDrive/CV_assignment_iitd/COCODIR /content/checkpoint0011_4scale.pth
```

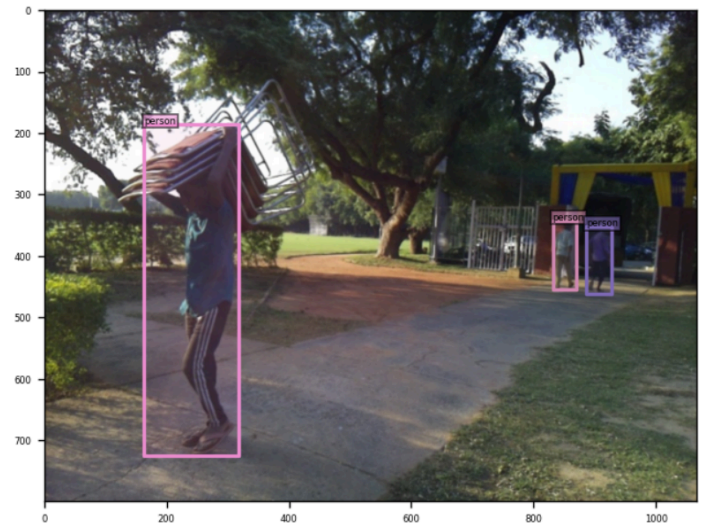
AP values -

```
  return _VF.meshgrid(tensors, **kwargs)  # type: ignore[attr-defined]
Test:  [ 0/40]  eta: 0:02:20  class_error: 0.00  loss: 10.7839 (10.7839)  loss_bbox_dn:
Test:  [10/40]  eta: 0:00:15  class_error: 0.00  loss: 10.6963 (9.7527)  loss_bbox_dn:
Test:  [20/40]  eta: 0:00:07  class_error: 0.00  loss: 10.9374 (10.9136)  loss_bbox_dn:
Test:  [30/40]  eta: 0:00:03  class_error: 16.67  loss: 12.4234 (11.8966)  loss_bbox_dr
Test:  [39/40]  eta: 0:00:00  class_error: 9.09  loss: 12.5348 (12.1007)  loss_bbox_dn:
Test: Total time: 0:00:13 (0.3279 s / it)
Averaged stats: class_error: 9.09  loss: 12.5348 (12.1007)  loss_bbox_dn: 0.0000 (0.000
Accumulating evaluation results...
DONE (t=0.04s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.479
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.805
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.501
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.355
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.563
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.629
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.123
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.522
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.589
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.508
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.647
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.705
```

6) Visualizing the predictions of pretrained model

For this, I modified the script provided in the github repo so I can compare ground truth images and prediction images.

Ground truth image



savename: /content/visualizations/gt_image_15.png/85-2024-09-24-11:55:39.921572.png

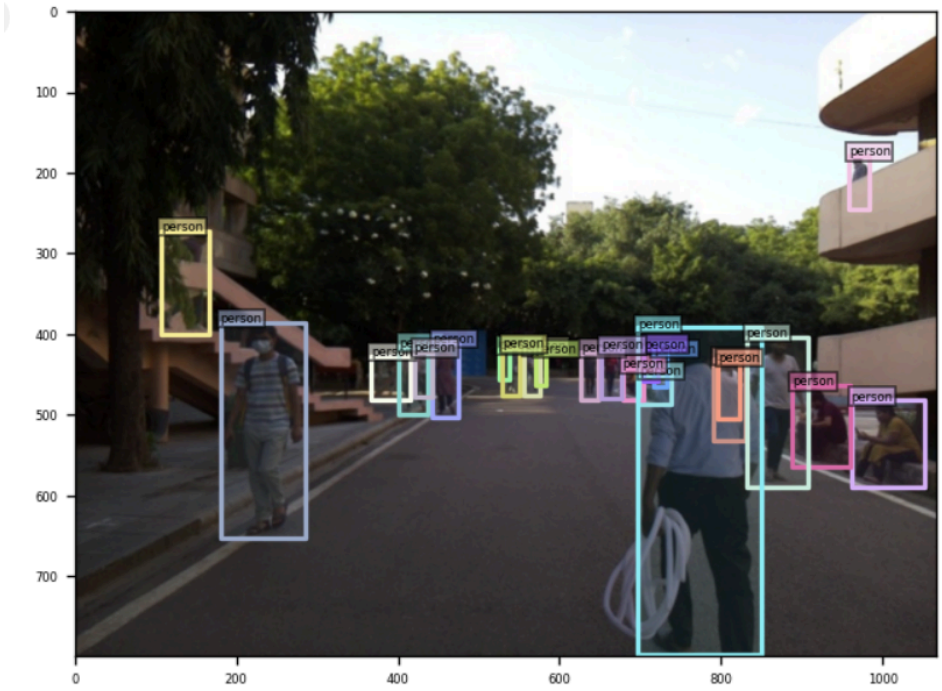
Prediction image

savename: /content/visualizations/pred_image_15.png/85-2024-09-24-11:55:41.074109.png

7) Errors cases (where model fails to detect some)

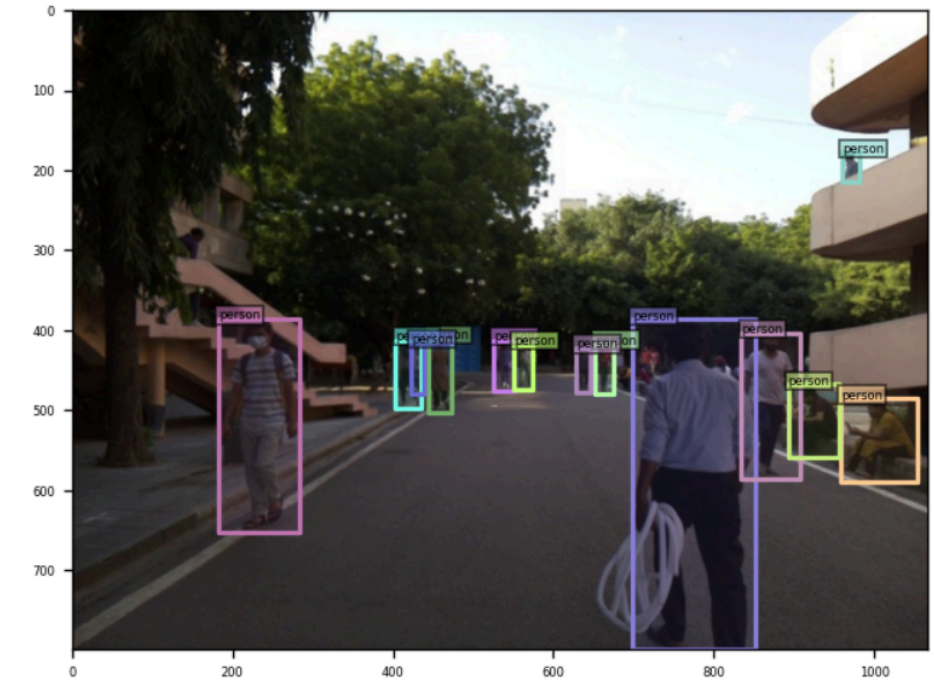
1st case -

Ground truth image -



savename: /content/visualizations/gt image 2.png/11-2024-09-23-16:39:47.239817.png

Predicted image -

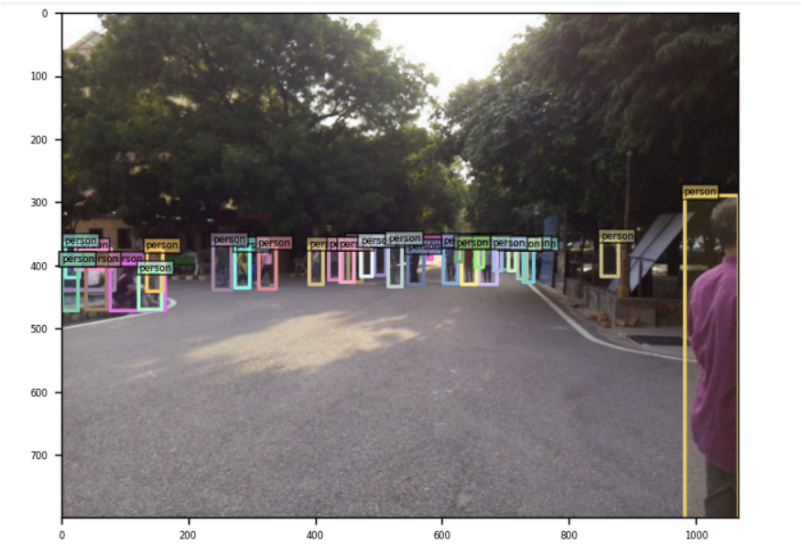


savename: /content/visualizations/pred_image_2.png/11-2024-09-23-16:39:47.924703.png

When the person is very small and distant from the camera, the model often fails to identify correctly. The model confuses a person as an object and vice versa in the case of distant small objects/persons.

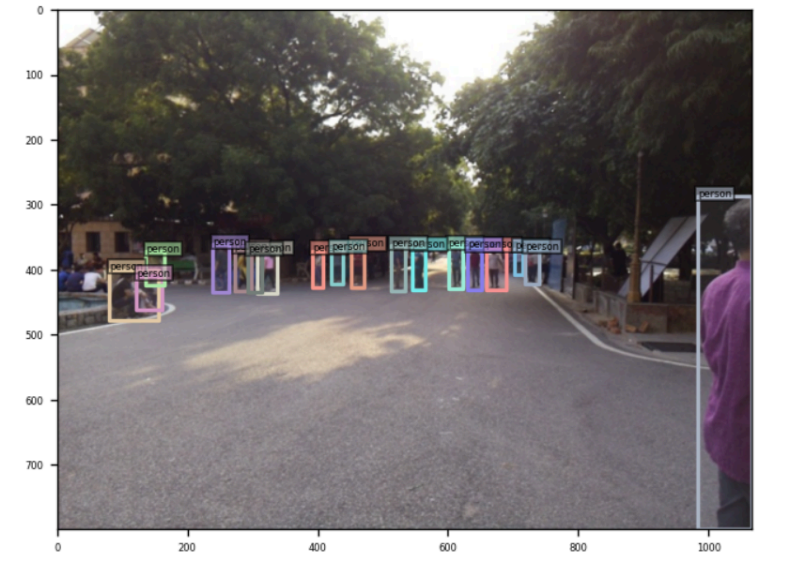
2nd case -

Ground truth image -



savename: /content/visualizations/gt_image_12.png/61-2024-09-23-16:40:02.052886.png

Predicted image-



savename: /content/visualizations/pred_image_12.png/61-2024-09-23-16:40:02.807381.png

In this case, the model fails to detect the people sitting at the left corner (turned backwards). In my opinion, the model has difficulties in detecting people from behind when they are at far distances.

The model also fails to detect a person (right side near tree) who is submerged into the background due to shade. When the person cannot be differentiated from the background, the model is facing difficulties in detecting.

3rd case -

Ground truth image-

savename: /content/visualizations/gt_image_25.png/131-2024-09-23-16:40:24.459056.png

Predicted image -

savename: /content/visualizations/pred_image_25.png/131-2024-09-23-16:40:25.754794.png

In this image, the model confuses the current pole as a person. Again proving that the model is having difficulties in accurately prediction person at far distances.

4th case -

Ground truth image

savename: /content/visualizations/gt_image_30.png/154-2024-09-23-16:40:33.533386.png

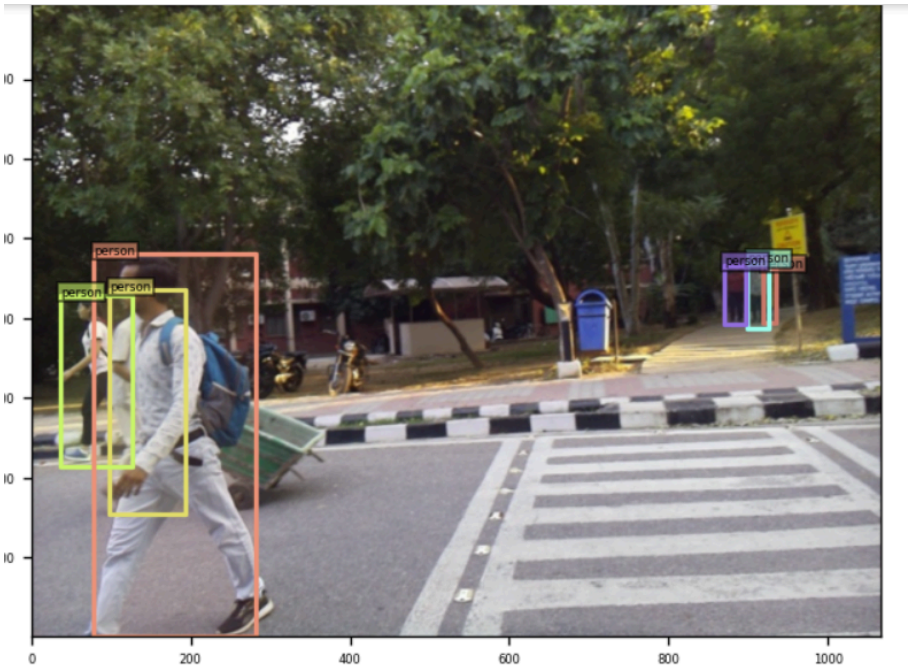
Predicted image -

savename: /content/visualizations/pred_image_30.png/154-2024-09-23-16:40:34.383581.png

Although there are only 2 persons, it gives 5 bounding boxes.

5th case -

Ground truth image-



ename: /content/visualizations/gt_image_35.png/169-2024-09-23-16:40:41.476889.png

Predicted image-

savename: /content/visualizations/pred_image_35.png/169-2024-09-23-16:40:42.448281.png

Here, the model wrongly detects the bike and scooty as persons.
In some cases, the model also has difficulties with occlusions. However, in most cases, the model was successful in tackling occlusions.

8) Finetuning the pretrained model on our custom dataset

I followed the github repo instructions and made the necessary changes to finetune.

Finetuning the pretrained model on custom dataset (script in repo)

```
[16] !bash /content/DINO/scripts/DINO_train.sh /content/drive/MyDrive/CV_assignment_iitd/COCODIR \
        --output_dir logs/DINO/R50-MS4 \
        --config_file /content/DINO/config/DINO/DINO_4scale.py \
        --pretrain_model_path /content/checkpoint0011_4scale.pth \
        --finetune_ignore label_enc.weight class_embed
```

Number of epochs - 12

In each epoch, the model trains over training set (160 images) and performs evaluation on the validation set (40 images).

1st epoch -

```
Epoch: [0]  [20/79]  eta: 0:01:05  lr: 0.000100  class_error: 0.00  loss: 42.5291 (47.3038)  loss_ce_dn: 0.3673 (0.5409)
Epoch: [0]  [30/79]  eta: 0:00:51  lr: 0.000100  class_error: 0.00  loss: 51.1387 (45.8317)  loss_ce_dn: 0.3207 (0.4768)
Epoch: [0]  [40/79]  eta: 0:00:39  lr: 0.000100  class_error: 0.00  loss: 31.9202 (41.7803)  loss_ce_dn: 0.3503 (0.4669)
Epoch: [0]  [50/79]  eta: 0:00:28  lr: 0.000100  class_error: 0.00  loss: 27.0994 (39.0720)  loss_ce_dn: 0.3749 (0.4491)
Epoch: [0]  [60/79]  eta: 0:00:18  lr: 0.000100  class_error: 0.00  loss: 26.3492 (36.9602)  loss_ce_dn: 0.3688 (0.4344)
Epoch: [0]  [70/79]  eta: 0:00:08  lr: 0.000100  class_error: 0.00  loss: 24.6979 (35.1398)  loss_ce_dn: 0.3253 (0.4106)
Epoch: [0]  [78/79]  eta: 0:00:00  lr: 0.000100  class_error: 0.00  loss: 23.9956 (33.9909)  loss_ce_dn: 0.2538 (0.3927)
Epoch: [0] Total time: 0:01:15 (0.9596 s / it)
Averaged stats: lr: 0.000100  class_error: 0.00  loss: 23.9956 (33.9909)  loss_ce_dn: 0.2538 (0.3927)  loss_bbox_dn: 0.3
/content/DINO/engine.py:164: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autc
  with torch.cuda.amp.autocast(enabled=args.amp):
Test:  [ 0/40]  eta: 0:01:03  class_error: 0.00  loss: 9.0090 (9.0090)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0.0
Test:  [10/40]  eta: 0:00:10  class_error: 0.00  loss: 12.3496 (11.9691)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0
Test:  [20/40]  eta: 0:00:05  class_error: 0.00  loss: 12.0425 (12.0478)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0
Test:  [30/40]  eta: 0:00:02  class_error: 0.00  loss: 11.9907 (12.6386)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0
Test:  [39/40]  eta: 0:00:00  class_error: 0.00  loss: 11.6645 (12.5052)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0
Test: Total time: 0:00:11 (0.2791 s / it)
Averaged stats: class_error: 0.00  loss: 11.6645 (12.5052)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0.0000 (0.0000)
Accumulating evaluation results...
DONE (t=0.06s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.013
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.050
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.001
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.009
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.026
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.012
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.067
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.169
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.153
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.216
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.000
```

6th epoch -

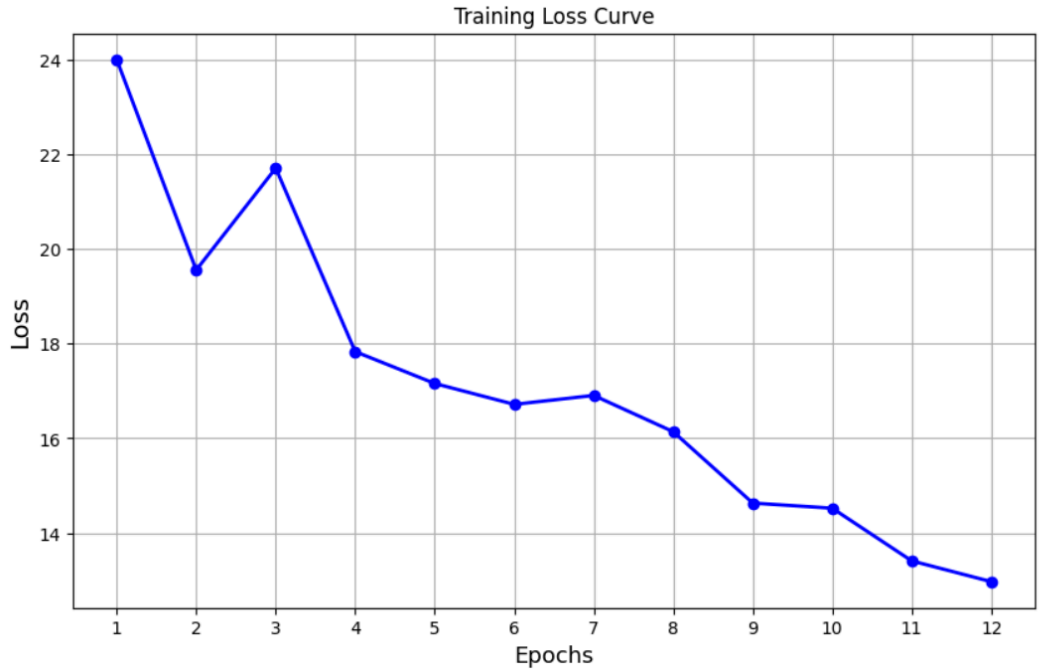
```
Epoch: [5]  [ 0/79]  eta: 0:03:27  lr: 0.000100  class_error: 0.00  loss: 19.0353 (19.0353)  loss_ce_dn: 0.0706 (0.0706)
Epoch: [5]  [10/79]  eta: 0:01:18  lr: 0.000100  class_error: 0.00  loss: 17.6524 (17.8585)  loss_ce_dn: 0.0513 (0.0536)
Epoch: [5]  [20/79]  eta: 0:01:00  lr: 0.000100  class_error: 0.00  loss: 16.8767 (17.2115)  loss_ce_dn: 0.0513 (0.0542)
Epoch: [5]  [30/79]  eta: 0:00:48  lr: 0.000100  class_error: 0.00  loss: 16.4441 (17.3522)  loss_ce_dn: 0.0561 (0.0568)
Epoch: [5]  [40/79]  eta: 0:00:38  lr: 0.000100  class_error: 0.00  loss: 17.7173 (17.7202)  loss_ce_dn: 0.0592 (0.0609)
Epoch: [5]  [50/79]  eta: 0:00:28  lr: 0.000100  class_error: 0.00  loss: 17.9908 (17.6426)  loss_ce_dn: 0.0660 (0.0608)
Epoch: [5]  [60/79]  eta: 0:00:18  lr: 0.000100  class_error: 0.00  loss: 16.4829 (17.3540)  loss_ce_dn: 0.0553 (0.0594)
Epoch: [5]  [70/79]  eta: 0:00:08  lr: 0.000100  class_error: 0.00  loss: 15.9051 (17.2211)  loss_ce_dn: 0.0558 (0.0597)
Epoch: [5]  [78/79]  eta: 0:00:00  lr: 0.000100  class_error: 0.00  loss: 16.7171 (17.1763)  loss_ce_dn: 0.0497 (0.0585)
Epoch: [5] Total time: 0:01:17 (0.9752 s / it)
Averaged stats: lr: 0.000100  class_error: 0.00  loss: 16.7171 (17.1763)  loss_ce_dn: 0.0497 (0.0585)  loss_bbox_dn: 0.1
Test:  [ 0/40]  eta: 0:01:33  class_error: 0.00  loss: 8.5804 (8.5804)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0.0
Test:  [10/40]  eta: 0:00:14  class_error: 0.00  loss: 9.1235 (9.5228)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0.0
Test:  [20/40]  eta: 0:00:07  class_error: 0.00  loss: 8.5025 (8.9763)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0.0
Test:  [30/40]  eta: 0:00:03  class_error: 0.00  loss: 8.7567 (9.4001)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0.0
Test:  [39/40]  eta: 0:00:00  class_error: 0.00  loss: 8.7990 (9.4108)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0.0
Test: Total time: 0:00:12 (0.3235 s / it)
Averaged stats: class_error: 0.00  loss: 8.7990 (9.4108)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0.0000 (0.0000)
Accumulating evaluation results...
DONE (t=0.04s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.055
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.142
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.037
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.037
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.085
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.084
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.061
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.186
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.395
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.255
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.520
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.465
```

12th epoch -

```
Epoch: [11]  [ 0/79]  eta: 0:05:48  lr: 0.000010  class_error: 0.00  loss: 17.7067 (17.7067)  loss_ce_dn: 0.04
Epoch: [11]  [10/79]  eta: 0:01:29  lr: 0.000010  class_error: 0.00  loss: 14.5140 (15.4412)  loss_ce_dn: 0.03
Epoch: [11]  [20/79]  eta: 0:01:08  lr: 0.000010  class_error: 0.00  loss: 13.6230 (14.6490)  loss_ce_dn: 0.02
Epoch: [11]  [30/79]  eta: 0:00:52  lr: 0.000010  class_error: 0.00  loss: 13.1826 (14.5948)  loss_ce_dn: 0.02
Epoch: [11]  [40/79]  eta: 0:00:40  lr: 0.000010  class_error: 0.00  loss: 13.8651 (14.4926)  loss_ce_dn: 0.03
Epoch: [11]  [50/79]  eta: 0:00:29  lr: 0.000010  class_error: 0.00  loss: 13.6192 (14.3666)  loss_ce_dn: 0.02
Epoch: [11]  [60/79]  eta: 0:00:19  lr: 0.000010  class_error: 0.00  loss: 13.2667 (14.1464)  loss_ce_dn: 0.02
Epoch: [11]  [70/79]  eta: 0:00:09  lr: 0.000010  class_error: 0.00  loss: 12.8710 (13.9829)  loss_ce_dn: 0.02
Epoch: [11]  [78/79]  eta: 0:00:01  lr: 0.000010  class_error: 0.00  loss: 12.9757 (13.9386)  loss_ce_dn: 0.02
Epoch: [11] Total time: 0:01:19 (1.0100 s / it)
Averaged stats: lr: 0.000010  class_error: 0.00  loss: 12.9757 (13.9386)  loss_ce_dn: 0.0244 (0.0339)  loss_bb
Test:  [ 0/40]  eta: 0:01:23  class_error: 0.00  loss: 6.2190 (6.2190)  loss_bbox_dn: 0.0000 (0.0000)  loss_gi
Test:  [10/40]  eta: 0:00:14  class_error: 0.00  loss: 7.0130 (7.4874)  loss_bbox_dn: 0.0000 (0.0000)  loss_gi
Test:  [20/40]  eta: 0:00:07  class_error: 0.00  loss: 6.5867 (7.1173)  loss_bbox_dn: 0.0000 (0.0000)  loss_gi
Test:  [30/40]  eta: 0:00:03  class_error: 0.00  loss: 6.7777 (7.6653)  loss_bbox_dn: 0.0000 (0.0000)  loss_gi
Test:  [39/40]  eta: 0:00:00  class_error: 0.00  loss: 7.2787 (7.7447)  loss_bbox_dn: 0.0000 (0.0000)  loss_gi
Test: Total time: 0:00:12 (0.3209 s / it)
Averaged stats: class_error: 0.00  loss: 7.2787 (7.7447)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn: 0.0000
Accumulating evaluation results...
DONE (t=0.04s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.054
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.099
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.055
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.038
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.085
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.107
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.080
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.236
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.458
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.397
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.508
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.515
Training time 0:19:41
```

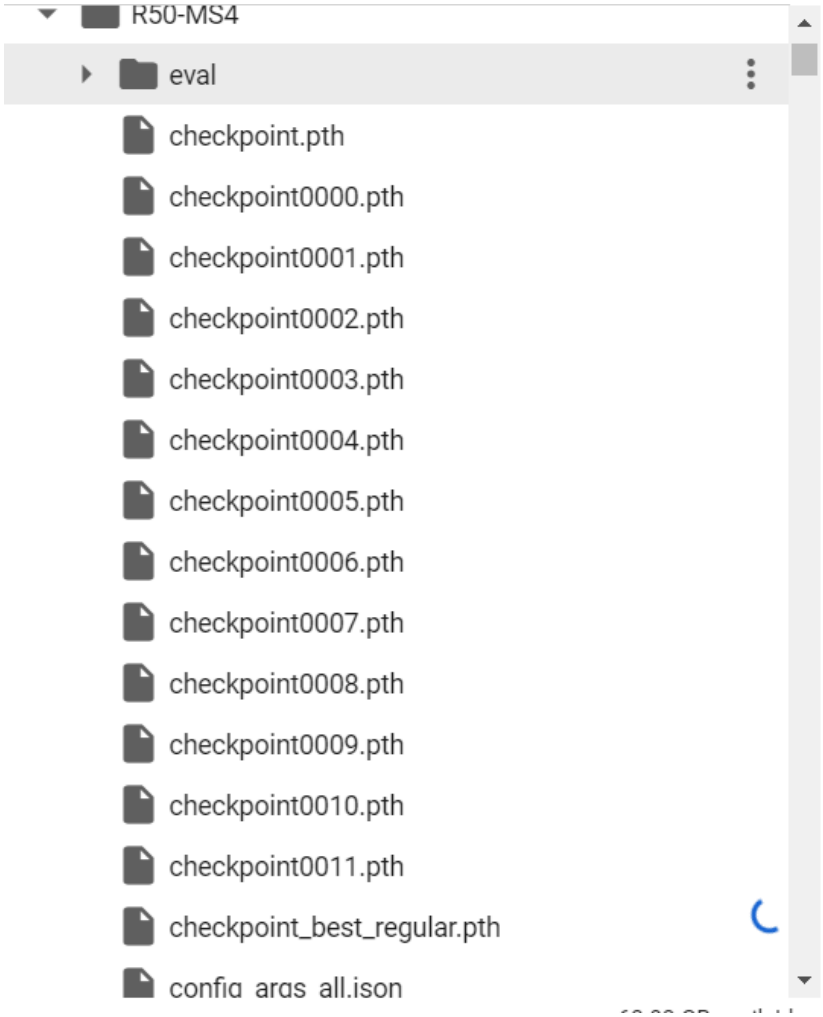
Loss Graphs (during finetuning) -

Training Loss -



Testing Loss -

The model checkpoints were saved after each epoch,



The checkpoint_best_regular.pth is our finetuned model (link below)
https://drive.google.com/file/d/1tZPCWB_5BGTknm_Phvys9pufLIDohEJ2/view?usp=sharing

9) Evaluation using the finetuned model -

Running evaluation script using finetuned model

```
# Give path to the finetuned model weights and run the evaluation script
!bash scripts/DINO_eval.sh /content/drive/MyDrive/CV_assignment_iitd/COCODIR /content/DINO/logs/DINO/R50-MS4/checkpoint_best_regular.pth
```

AP values -

```
Test:  [ 0/40]  eta: 0:01:56  class_error: 0.00  loss: 8.8260 (8.8260)  loss_bbox_dn: 0.0000 (0.0000)
Test:  [10/40]  eta: 0:00:14  class_error: 0.00  loss: 9.1632 (9.2636)  loss_bbox_dn: 0.0000 (0.0000)
Test:  [20/40]  eta: 0:00:07  class_error: 0.00  loss: 8.5402 (8.5845)  loss_bbox_dn: 0.0000 (0.0000)
Test:  [30/40]  eta: 0:00:03  class_error: 0.00  loss: 8.0905 (8.9624)  loss_bbox_dn: 0.0000 (0.0000)
Test:  [39/40]  eta: 0:00:00  class_error: 0.00  loss: 8.0905 (9.0627)  loss_bbox_dn: 0.0000 (0.0000)
Test: Total time: 0:00:13 (0.3291 s / it)
Averaged stats: class_error: 0.00  loss: 8.0905 (9.0627)  loss_bbox_dn: 0.0000 (0.0000)  loss_giou_dn:
Accumulating evaluation results...
DONE (t=0.04s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.092
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.176
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.090
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.055
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.120
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.197
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.064
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.219
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.483
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.393
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.565
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.515
```

10) Possible reasons for decrease in accuracy (my assumptions) -

- Size of dataset we used for finetuning is small (ie. 160 images). So the model may have overfitted on the date. Retraining the entire model over a small dataset may be the reason for poor performance.

  Solution-

  We can try data augmentation to replicate the dataset or we can finetune by freezing few layers.

- I thought learning rate may be an issue, so I tried changing the lr hyperparameter in the config file, but there were no much improvements.

Conclusion - I think spending more time into studying why the model performance decreased and tuning few hyperparameters might eventually give good results on a larger dataset. I couldn't experiment more due to the GPU usage constraints on google colab. I will try to work on ways to improve model's performance.