Assignment3
Gourp member: Mike Wang, Ying Zhou, Hanjie(Huntley) Liu
Kaggle marks account for part3: Mike Wang

Hanjie Liu
Student ID: 301404949
Kaggle Name: Huntley Liu
Late day: 1

Part1
List of configs

```
'''
# Set the configs for the detection part in here.
# TODO: approx 15 lines
'''
cfg = get_cfg()
cfg.OUTPUT_DIR = "{}/output/".format(BASE_DIR)

cfg.merge_from_file(model_zoo.get_config_file("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml"))
#cfg.merge_from_file(model_zoo.get_config_file("COCO-Detection/faster_rcnn_X_101_32x8d_FPN_3x.yaml"))
cfg.DATASETS.TRAIN = ("plane_train",)
cfg.DATASETS.TEST = ("plane_test",)

cfg.DATALOADER.NUM_WORKERS = 4
cfg.MODEL.WEIGHTS = model_zoo.get_checkpoint_url("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml")
#cfg.MODEL.WEIGHTS = model_zoo.get_checkpoint_url("COCO-Detection/faster_rcnn_X_101_32x8d_FPN_3x.yaml")
cfg.SOLVER.IMS_PER_BATCH = 2
cfg.SOLVER.BASE_LR = 0.00025
cfg.SOLVER.MAX_ITER = 500
cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 128
cfg.MODEL.ROI_HEADS.NUM_CLASSES = 1
```

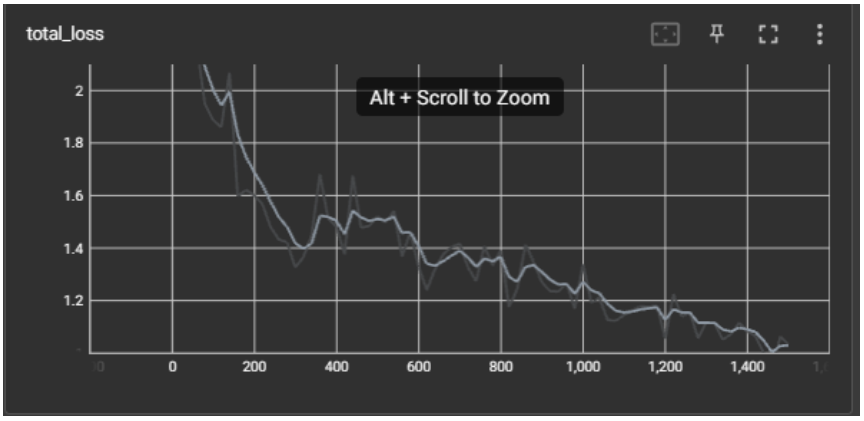
List of modifications and Improving factors:

1. The Base_Lr has been modified from 0.001 to 0.00025. When I using 0.001 as my Base_Lr, the loss goes down very fast at beginning but it is hard to converge in last 100 iterations. So I change it to 0.00025, the accuracy steadily improved. I found that the higher Base_Lr will have a lower accuracy and lower Base_Lr will take longer time to converge.
2. I used “faster_rcnn_X_101_32x8d_FPN_3x.yaml” from Model_ZOO website as my pre-training model, it causes more time than tutorial’s but has higher accuracy.
3. I change the iterations from 500 to 1500(5000 for part3), because 500 iterations still has fast speed of decreasing loss, but 1500 iterations become stable, and results are crediable.

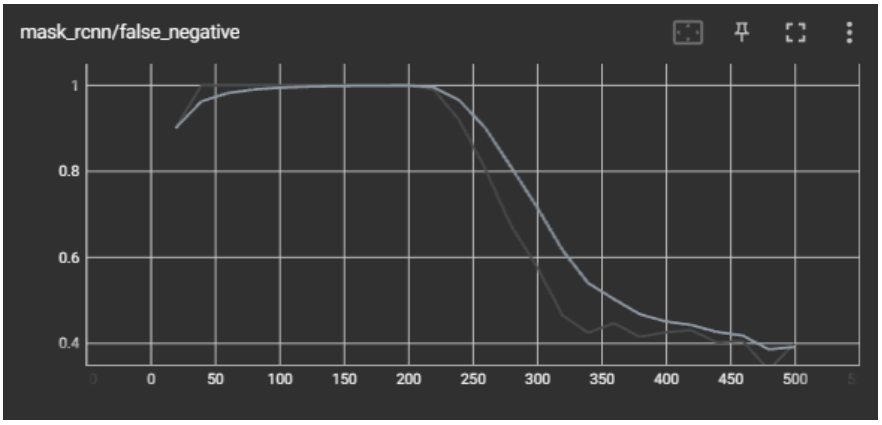
Training loss on the last part of iterations:

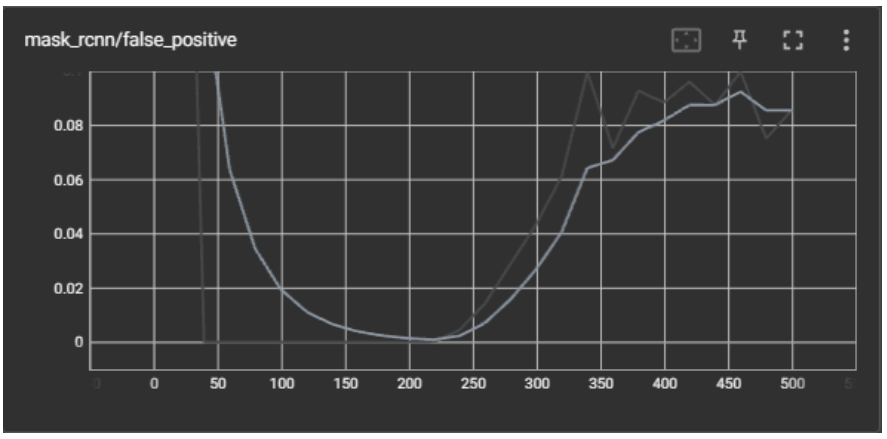
```
eta: 0:07:05  iter: 1319  total_loss: 1.114  loss_cls: 0.2389  loss_box_reg: 0.5352  loss_rpn_cls: 0.1155
eta: 0:06:18  iter: 1339  total_loss: 1.051  loss_cls: 0.2438  loss_box_reg: 0.5174  loss_rpn_cls: 0.09838
eta: 0:05:30  iter: 1359  total_loss: 1.07  loss_cls: 0.2429  loss_box_reg: 0.5092  loss_rpn_cls: 0.09847
eta: 0:04:43  iter: 1379  total_loss: 1.115  loss_cls: 0.2425  loss_box_reg: 0.5291  loss_rpn_cls: 0.08864
eta: 0:03:56  iter: 1399  total_loss: 1.081  loss_cls: 0.2768  loss_box_reg: 0.4954  loss_rpn_cls: 0.09134
eta: 0:03:09  iter: 1419  total_loss: 1.064  loss_cls: 0.2567  loss_box_reg: 0.4988  loss_rpn_cls: 0.08788
eta: 0:02:22  iter: 1439  total_loss: 0.9969  loss_cls: 0.2459  loss_box_reg: 0.4948  loss_rpn_cls: 0.0919
eta: 0:01:34  iter: 1459  total_loss: 0.9374  loss_cls: 0.2385  loss_box_reg: 0.5003  loss_rpn_cls: 0.0853
eta: 0:00:47  iter: 1479  total_loss: 1.062  loss_cls: 0.2201  loss_box_reg: 0.5333  loss_rpn_cls: 0.1102
eta: 0:00:00  iter: 1499  total_loss: 1.032  loss_cls: 0.2396  loss_box_reg: 0.4709  loss_rpn_cls: 0.1092
```

Visualization of test samples
Total loss:



Accuracy:



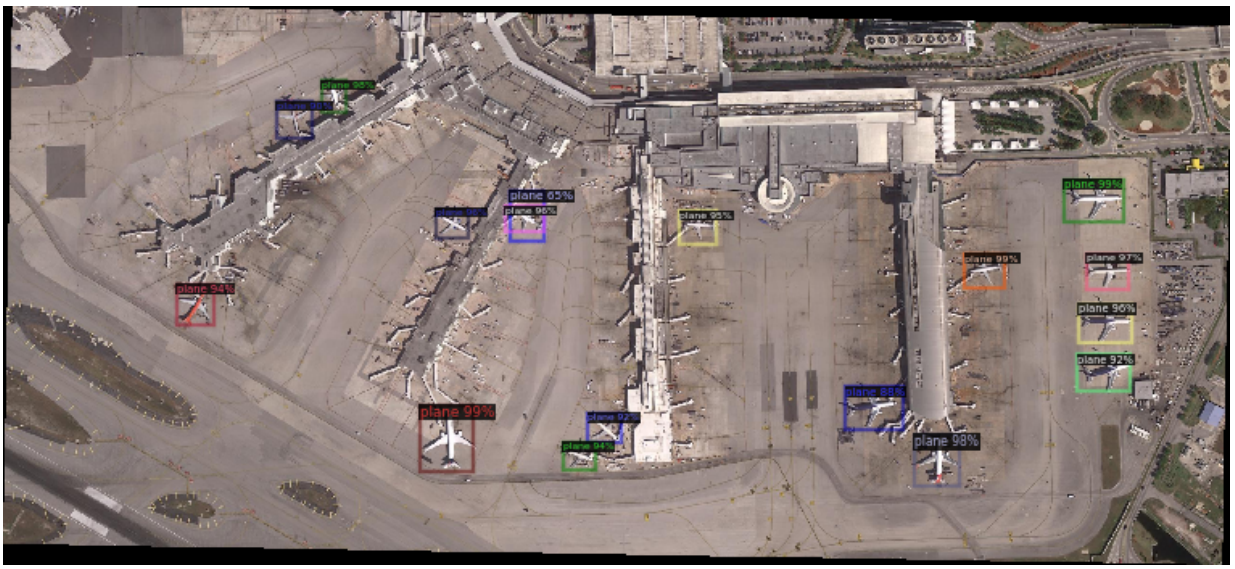


Test sample 1 for iter 500:

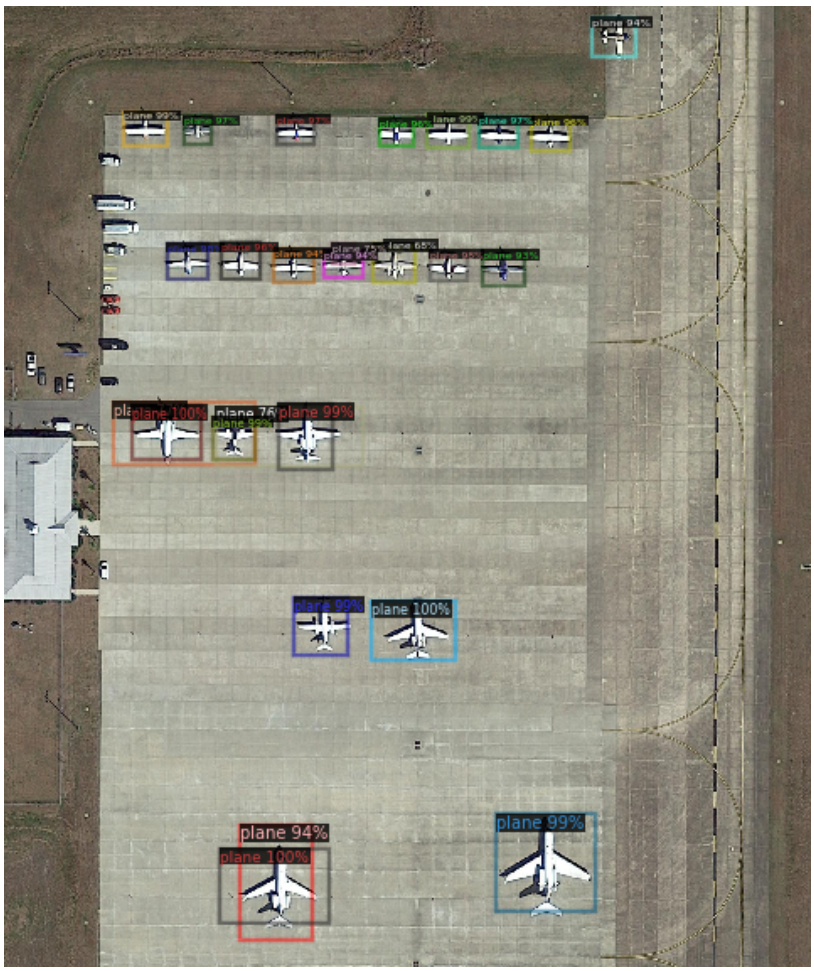
Test sample 2 for 500 iter:

Test sample 3 for 500 iter:

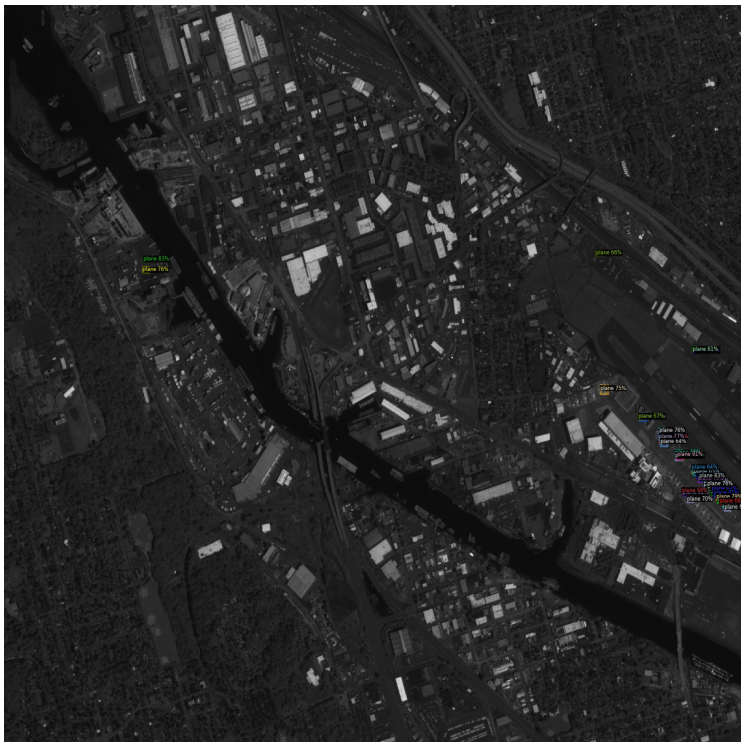
Test sample 1 for 1500 iter:



Test sample 2 for 1500 iter:



Test sample 3 for 1500 iter:



Final accuracy for the AP50 on 500 iter:

```
[02/28 21:00:44 d2.evaluation.fast_eval_api]: Evaluate annotation type *bbox*
[02/28 21:00:44 d2.evaluation.fast_eval_api]: COCOeval_opt.evaluate() finished in 0.26 seconds.
[02/28 21:00:44 d2.evaluation.fast_eval_api]: Accumulating evaluation results...
[02/28 21:00:44 d2.evaluation.fast_eval_api]: COCOeval_opt.accumulate() finished in 0.02 seconds.
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.317
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.532
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.351
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.220
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.404
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.618
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.015
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.130
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.357
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.222
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.447
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.738
[02/28 21:00:44 d2.evaluation.coco_evaluation]: Evaluation results for bbox:
|   AP   |  AP50  |  AP75  |  APs   |  APm   |  APl   |
|:------:|:------:|:------:|:------:|:------:|:------:|
| 31.708 | 53.240 | 35.065 | 22.005 | 40.373 | 61.828 |
Loading and preparing results...
DONE (t=0.15s)
creating index...
index created!
[02/28 21:00:45 d2.evaluation.fast_eval_api]: Evaluate annotation type *segm*
[02/28 21:00:46 d2.evaluation.fast_eval_api]: COCOeval_opt.evaluate() finished in 0.94 seconds.
[02/28 21:00:46 d2.evaluation.fast_eval_api]: Accumulating evaluation results...
[02/28 21:00:46 d2.evaluation.fast_eval_api]: COCOeval_opt.accumulate() finished in 0.02 seconds.
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.098
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.328
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.022
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.053
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.114
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.336
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.008
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.060
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.129
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.070
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.157
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.396
[02/28 21:00:46 d2.evaluation.coco_evaluation]: Evaluation results for segm:
|  AP   |  AP50  |  AP75  |  APs  |  APm   |  APl   |
|:-----:|:------:|:------:|:-----:|:------:|:------:|
| 9.835 | 32.806 | 2.193  | 5.293 | 11.432 | 33.554 |
OrderedDict([('bbox', {'AP': 31.70845464725845, 'AP50': 53.239532202057106, 'AP75': 35.06519663659673,
```

Final accuracy for the AP50 on 1500 iter:

```
[03/02 06:26:01 d2.evaluation.fast_eval_api]: Evaluate annotation type *bbox*
[03/02 06:26:02 d2.evaluation.fast_eval_api]: COCOeval_opt.evaluate() finished in 0.29 seconds.
[03/02 06:26:02 d2.evaluation.fast_eval_api]: Accumulating evaluation results...
[03/02 06:26:02 d2.evaluation.fast_eval_api]: COCOeval_opt.accumulate() finished in 0.03 seconds.
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.287
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.585
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.247
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.231
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.349
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.409
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.014
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.115
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.338
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.244
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.407
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.558
[03/02 06:26:02 d2.evaluation.coco_evaluation]: Evaluation results for bbox:
|   AP   |  AP50  |  AP75  |  APs   |  APm   |  APl   |
|:------:|:------:|:------:|:------:|:------:|:------:|
| 28.694 | 58.545 | 24.709 | 23.135 | 34.925 | 40.933 |
OrderedDict([('bbox', {'AP': 28.694165524522607, 'AP50': 58.545235702691336, 'AP75': 24.708872603354685,
```

Ablation study:

Effect of increasing maximum iterations from 500 to 1500: We increase our iteration round from 500 to 1500, in this situation, the accuracy of object are impressivly increase.Some of them has been improved from 70% to 99%, and the program can detect more planed as it predictor probability exceed 0.6. The AP50 detection accurate also gose up from 53 to 58. So we know that more iteration before overfit will increase our accuracy

Part 2

Hyperparameter setting:

```
# Set the hyperparameters
num_epochs = 50
batch_size = 4
learning_rate = 0.001
weight_decay = 1e-5
```

The optimizer: "torch.optim.SGD"

Net architecture:

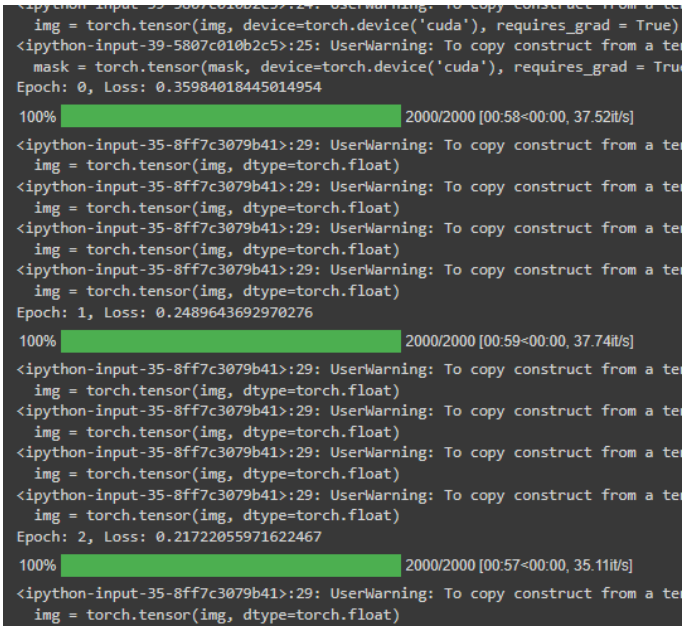
I add four more down and up sampling layers in network. The input layer was given (3, 16) and each down sampling layer increase the output to the twice of input, until we got 512. The up sampling will decrease the output to half of the input until we got 4. We increase the feature channels for helping us detecting feature more credible. The loss is going stable around 0.068.

```
def __init__(self):
    super(MyModel, self).__init__()

    self.input_conv = conv(3, 16)
    self.down1 = down(16, 32)
    self.down2 = down(32, 64)
    self.down3 = down(64, 128)
    self.down4 = down(128, 256)
    self.down5 = down(256, 512)


    self.up1 = up(512, 256)
    self.up2 = up(256, 128)
    self.up3 = up(128, 64)
    self.up4 = up(64, 32)
    self.up5 = up(32, 4)
    self.output_conv = conv(4, 1, False) #

def forward(self, input):
  y = self.input_conv(input)
  y = self.down1(y)
  y = self.down2(y)
  y = self.down3(y)
  y = self.down4(y)
  y = self.down5(y)
  y = self.up1(y)
  y = self.up2(y)
  y = self.up3(y)
  y = self.up4(y)
  y = self.up5(y)
  output = self.output_conv(y)
  return output
```

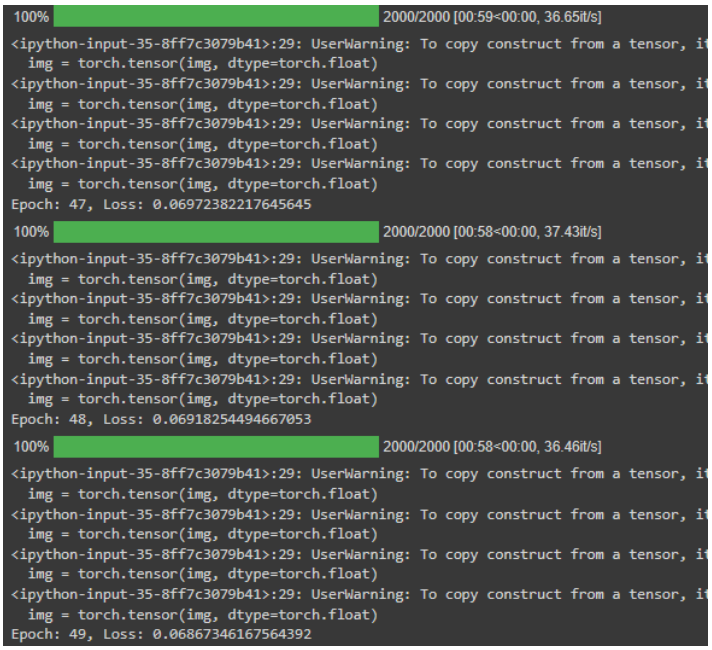
Loss function:

nn.BCEWithLogitsLoss()

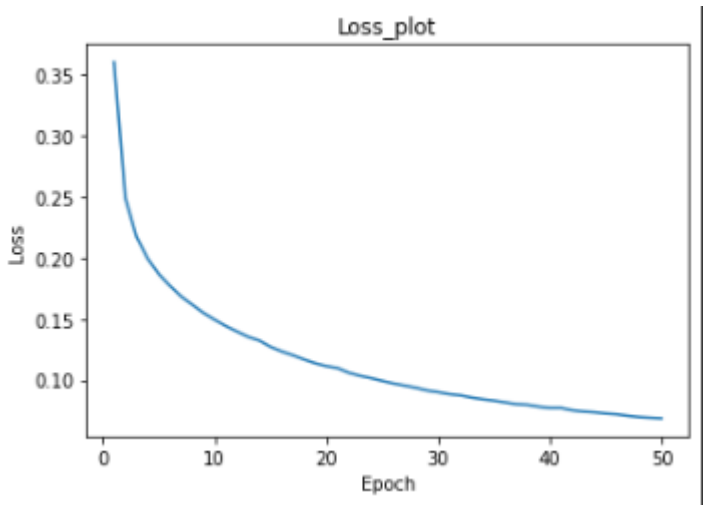
Traning loss of first 3 epochs:



Traning loss of last 3 epochs:

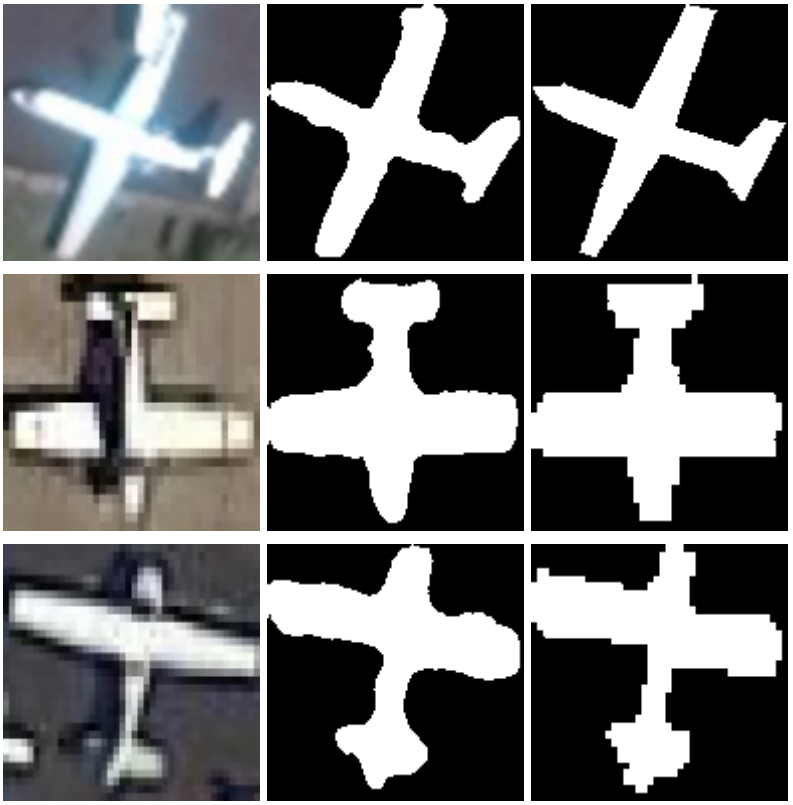


Plot of loss:



Final mean IoU: `IoU: 0.9006885895082279`

Test sample: Original; predicted; test

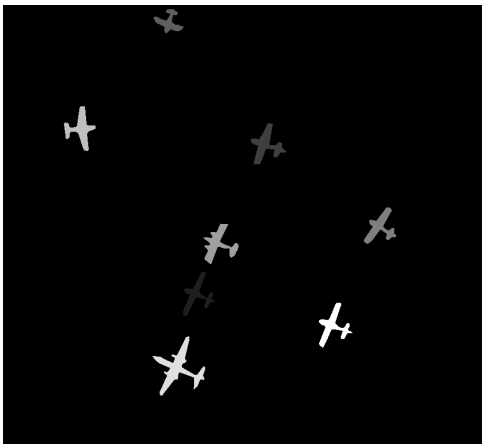


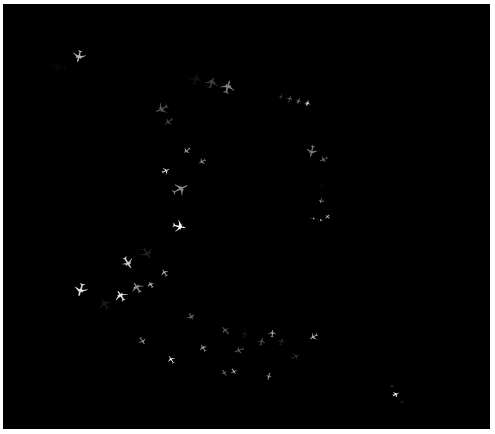
Part 3:

Kaggle ID: Mike Wang (uploader)

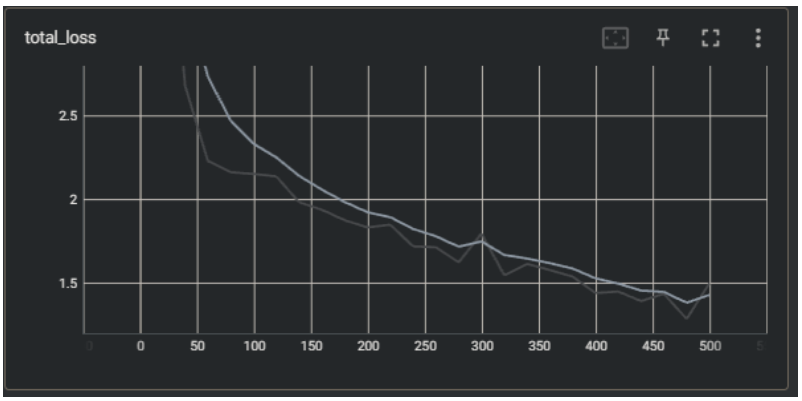
Best score: 0.46629

Test and Train sample:

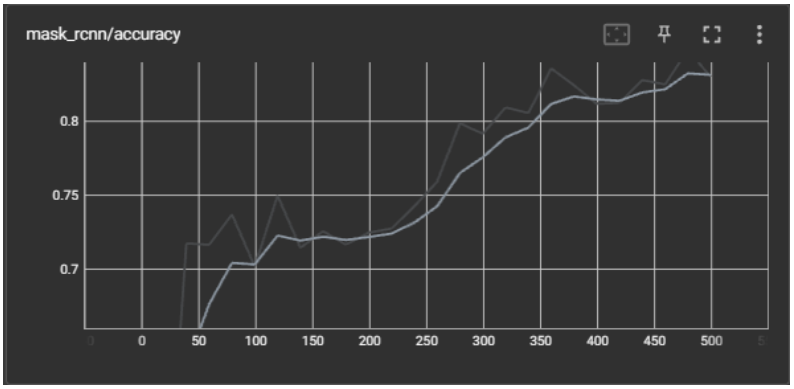



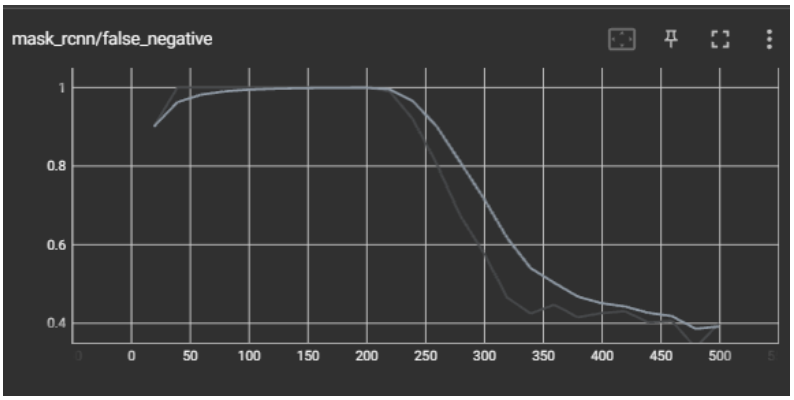


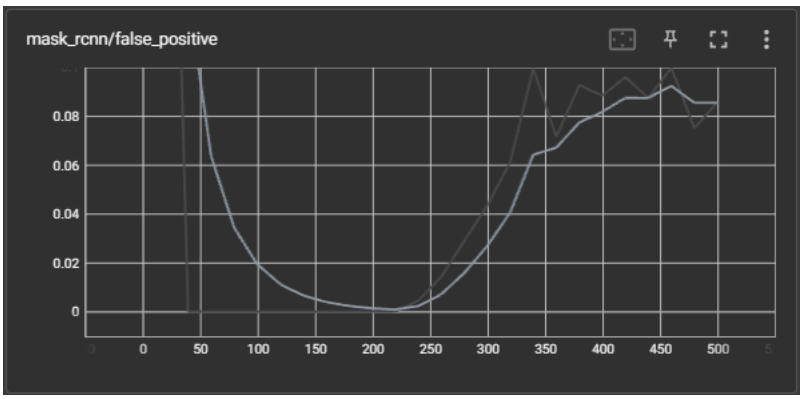
Part 4:
Our model is "COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml"
We run 100 iterations and 500 iterations:
Total loss:



Accuracy:



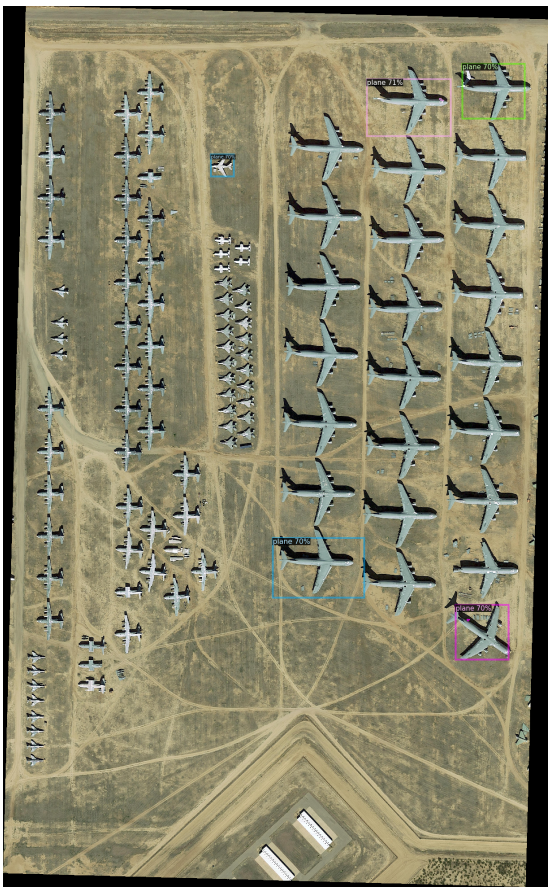




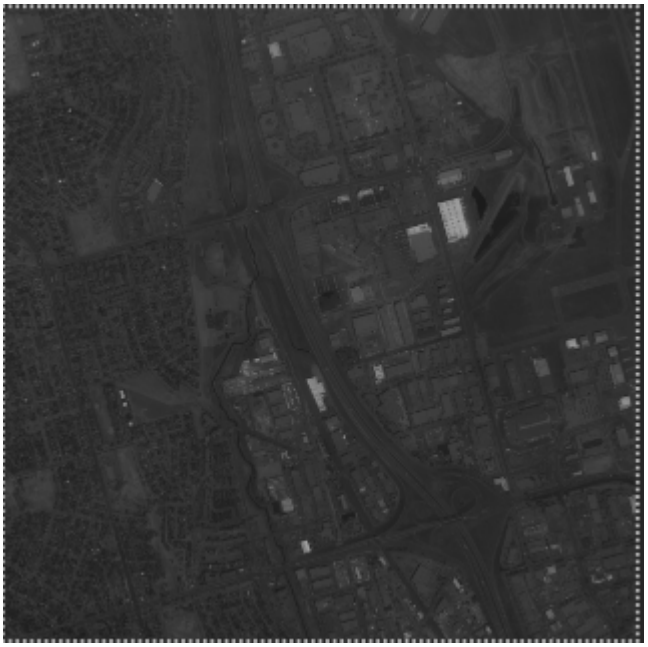
Accuracy of AP50:

```
[02/23 07:25:09 d2.evaluation.fast_eval_api]: Evaluate annotation type *bbox*
[02/23 07:25:09 d2.evaluation.fast_eval_api]: COCOeval_opt.evaluate() finished in 0.28 seconds.
[02/23 07:25:09 d2.evaluation.fast_eval_api]: Accumulating evaluation results...
[02/23 07:25:09 d2.evaluation.fast_eval_api]: COCOeval_opt.accumulate() finished in 0.03 seconds.
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.249
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.467
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.246
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.166
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.330
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.506
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.013
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.113
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.296
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.173
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.378
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.650
[02/23 07:25:09 d2.evaluation.coco_evaluation]: Evaluation results for bbox:
|   AP   |  AP50  |  AP75  |  APs   |  APm   |  APl   |
|:------:|:------:|:------:|:------:|:------:|:------:|
| 24.939 | 46.740 | 24.626 | 16.622 | 32.966 | 50.553 |
Loading and preparing results...
DONE (t=0.16s)
creating index...
index created!
[02/23 07:25:10 d2.evaluation.fast_eval_api]: Evaluate annotation type *segm*
[02/23 07:25:11 d2.evaluation.fast_eval_api]: COCOeval_opt.evaluate() finished in 0.97 seconds.
[02/23 07:25:11 d2.evaluation.fast_eval_api]: Accumulating evaluation results...
[02/23 07:25:11 d2.evaluation.fast_eval_api]: COCOeval_opt.accumulate() finished in 0.03 seconds.
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.054
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.214
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.007
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.030
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.058
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.218
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.006
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.041
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.080
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.041
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.094
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.274
[02/23 07:25:11 d2.evaluation.coco_evaluation]: Evaluation results for segm:
|   AP   |  AP50  |  AP75  |  APs   |  APm   |  APl   |
|:------:|:------:|:------:|:------:|:------:|:------:|
| 5.406  | 21.382 | 0.716  | 2.980  | 5.849  | 21.831 |
OrderedDict([('bbox', {'AP': 24.93927395330302, 'AP50': 46.73957178167791, 'AP75': 24.62580693029191, 'APs': 16.621
```

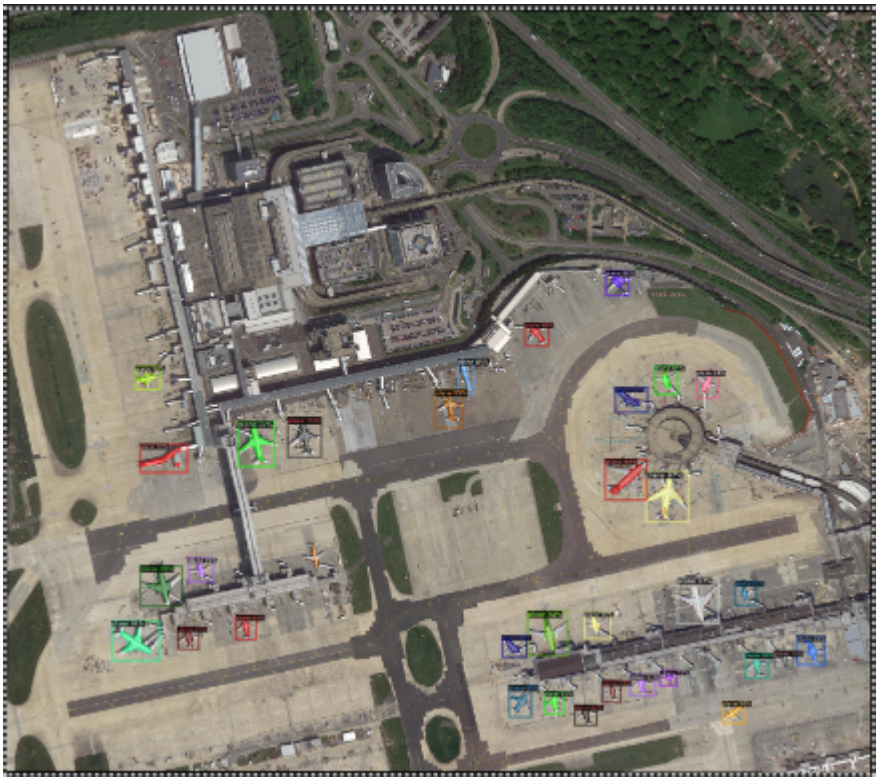
Test for 100 iter:



Test for iter 500:




planeplane 95%
plane 92%
plane 98%
plane 99%
plane 99%
plane 99%
plane 99%
plane 95%
plane 99%
plane 99%
plane 99%
plane 99%
plane 99%
plane 99%
plane 99%
plane 99%
plane 99%
plane 99%
plane 99%
plane 99%
plane 99%



Mean IOU:

```
IoU: 0.8160094420176907
```

Difference:

Compared both results, this model can be run less than previous one for same bbox accuracy, such as 500. The part 1 model does not work well in 500 iterations, but this one looks fine. And Its accuracy are increasing impressivly when iteration goes from 100 to 500. So we can say this model can be used faster than previous one.

From the evaluation output, the AP50 of this model for the bounding box is less than previous model, eventhough while they have same iterations. So due to AP50 of this one is lower than previous one, we can say this model is less accurate than previous one.