CMPT 412

Project 3

Object Detection, Semantic Segmentation, and Instance Segmentation

Instructor: Yasutaka Furukawa

Name: Kaikun Fang

Student ID: 301416542

Note: Use 3 free late-day

Note: Use 3 free late-day

Part 1: Object Detection

1. Configs and Modifications:

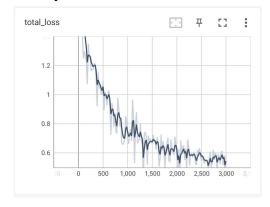
The learning rate and the number of iterations have been modified:

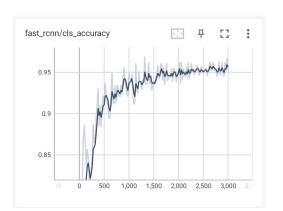
```
cfg. merge_from_file(model_zoo.get_config_file("COCO-Detection/faster_rcnn_R_101_FPN_3x.yam1"))
cfg. DATASETS. TRAIN = ("plane_train",)
cfg. DATASETS. TEST = ()
cfg. DATALOADER. NUM_WORKERS = 2
cfg. MODEL. WEIGHTS = model_zoo.get_checkpoint_url("COCO-Detection/faster_rcnn_R_101_FPN_3x.yam1")
cfg. SOLVER. IMS_PER_BATCH = 2  # batch size
cfg. SOLVER. BASE_LR = 0.0003  # LR
cfg. SOLVER. MAX_ITER = 3000
cfg. MODEL. ROI_HEADS. BATCH_SIZE_PER_IMAGE = 512
cfg. MODEL. ROI_HEADS. NUM_CLASSES = 1  # only have plane
```

2. Factors:

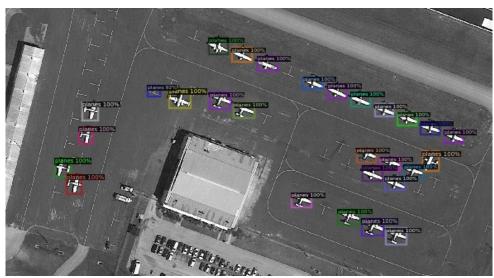
I first set the number of iterations at 500, and then set the learning rate at 0.0003 through several tests. Then I adjusted the number of iterations and found that the best accuracy could be achieved at 3000 without wasting training cost. At the end, get AP50 = 62.564

3. Final plot





4. Visualization



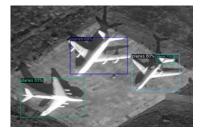




5. Ablation study:

I performed ablation study on the number of iterations. I leave all other parameters unchanged and adjust only the number of iterations.

```
MAX ITER = 500
     AP | AP50 | AP75 | APs | APm
  |:----:|:----:|:----:|
  | 25.631 | 47.240 | 25.758 | 17.370 | 33.062 | 52.314 |
MAX ITER = 2000
     AP | AP50
               | AP75 | APs | APm
  |:----:|:----:|:----:|:----:|
  | 37.804 | 58.323 | 42.507 | 26.755 | 45.778 | 71.471 |
MAX ITER = 3000
     AP | AP50 | AP75 | APs | APm
  |:----:|:----:|:----:|
  | 39.560 | 62.564 | 44.099 | 29.556 | 47.392 | 71.334 |
MAX ITER = 3500
     AP | AP50 | AP75 | APs | APm
  |:----:|:----:|:----:|
  | 40.053 | 62.700 | 44.476 | 29.529 | 48.045 | 72.047 |
```







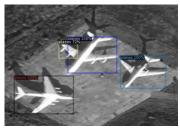


Figure 1 Iter 500

Figure 2 Iter 2000

Figure 3 Iter 3000

Figure 4 Iter 3500

The data and the visualization results show that the increase in the number of iterations has a positive effect on the training effect. Each of Figures 1, 2, and 4 has a part of the body of the plane that is also recognized as a separate plane. In contrast, Figure 3, which used a number of 3000 iterations, identified exactly the best results.

Part 2: Semantic Segmentation

1. Hyperparameter settings:

After a series of adjustments, my parameters were set as follows:

```
batch_size = 4
learning_rate = 0.003
num epochs = 150
```

2. Final architecture:

I have made changes to the default "down" class and "up" class:

For "down" class: To increase the degree of training, I added an extra layer of

```
"conv".
```

For "up" class: To increase efficiency, I added an extra layer of

"nn.BatchNorm2d".

MyModel code:

```
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.down6(y)
    y = self.up1(y)
    y = self.up2(y)
    y = self.up3(y)
    y = self.up5(y)
    y = self.up6(y)
    output = self.output conv(y)
```

Torchsummary:

Layer (type)				İ		
BatchNorm2d-2 [-1, 4, 128, 128] 8 BatchNorm2d-56 [-1, 256, 4, 4] 60 conv-4 (-1, 4, 128, 128] 0 conv-4 (-1, 4, 128, 128] 0 conv-58 [-1, 256, 4, 4] 0 conv-58 [-1, 256, 4, 4] 0 conv-69 [-1, 18, 128, 128] 16 BatchNorm2d-6 (-1, 8, 128, 128] 16 BatchNorm2d-60 [-1, 256, 4, 4] 590,088 BatchNorm2d-6 (-1, 8, 128, 128] 16 BatchNorm2d-60 [-1, 256, 4, 4] 590,088 BatchNorm2d-10 [-1, 8, 128, 128] 16 Conv-60 [-1, 256, 4, 4] 10 BatchNorm2d-10 [-1, 8, 128, 128] 16 Conv-60 [-1, 256, 4, 4] 10 BatchNorm2d-10 [-1, 8, 128, 128] 16 Conv-12 [-1, 18, 44, 4] 10 BatchNorm2d-15 [-1, 16, 64, 64] 1, 168 BatchNorm2d-15 [-1, 16, 64, 64] 1, 168 BatchNorm2d-16 [-1, 156, 64, 64] 1, 168 BatchNorm2d-16 [-1, 156, 64, 64] 10 Conv-18 [-1, 156, 64, 64] 10 BatchNorm2d-20 [-1, 156, 64, 64] 10 Conv-18 [-1, 156, 64, 64] 10 BatchNorm2d-20 [-1, 156, 64, 64] 10 Conv-18 [-1, 156, 64, 64] 10 BatchNorm2d-20 [-1, 156,						
BatchNorm2d-2 [-1, 4, 128, 128] 8 BatchNorm2d-56 [-1, 256, 4, 4] 60 conv-4 (-1, 4, 128, 128] 0 conv-4 (-1, 4, 128, 128] 0 conv-58 (-1, 256, 4, 4] 0 conv-58 (-1, 3, 128, 128) 0 conv-69 (-1, 256, 4, 4] 0 conv-58 (-1, 3, 128, 128) 0 conv-12 (-1, 8, 128, 128) 0 conv-69 (-1, 128, 4, 4] 0 conv-12 (-1, 8, 128, 128) 0 conv-12 (-1, 8, 128, 128) 0 conv-14 (-1, 8, 128, 128) 0 conv-14 (-1, 18, 128, 128) 0 conv-15 (-1, 15, 64, 64] 0 conv-15 (-1, 15, 64, 64] 0 conv-15 (-1, 15, 64, 64] 0 conv-18 (-1, 15, 64, 64] 0 conv-24 (-1, 15, 12, 23) 0 conv-24 (-1, 15, 23, 23) 0 conv-24 (-1, 128, 23, 23) 0 conv-24 (-1, 128, 23, 23) 0 conv-24 (-1, 128, 23, 23) 0 conv-24 (-1, 23, 23, 23) 0 conv-24 (-1, 23, 23, 23) 0 conv-24 (-1, 23, 23, 23) 0 conv-24 (-1, 24, 16, 16) 0 conv-24 (-1, 24, 16, 16) 0 conv-24 (-1, 44, 18, 8) 0 conv-24 (-1, 44, 16, 16) 0 conv-24 (-1, 44, 18, 8) 0 conv-24 (-1, 44, 16, 16) 0 conv-24 (-1, 44, 18, 8) 0 conv-24 (-1, 44, 16, 16) 0 conv-24 (-1, 44, 18, 8) 0 conv-24 (-1, 44, 18, 8) 0 conv-24 (-1, 44, 18, 8				Conv2d-55	[-1, 256, 4, 4]	295,168
Conv24-5 [-1, 4, 128, 128] 9	BatchNorm2d-2	[-1, 4, 128, 128]	8	BatchNorm2d-56	[-1, 256, 4, 4]	
Conv2d-5	ReLU-3	[-1, 4, 128, 128]	0			
BatchNorm2d-6 [-1, 8, 128, 128] 16 ReLU-7 [-1, 8, 128, 128] 0 Conv-8 [-1, 8, 128, 128] 0 Conv-8 [-1, 8, 128, 128] 0 ReLU-10 [-1, 8, 128, 128] 0 Rethorm2d-10 [-1, 8, 128, 128] 544 Rexponded to the second of			0			
ReLU-7 [-1, 8, 128, 128] 0 ReLU-61 [-1, 256, 4, 4] 0 Conv2d-9 [-1, 8, 128, 128] 0 Conv2d-9 [-1, 8, 128, 128] 584 Conv2d-9 [-1, 8, 128, 128] 166 ReLU-11 [-1, 8, 128, 128] 16 Conv2d-13 [-1, 8, 64, 64] 0 RelU-13 [-1, 8, 64, 64] 0 RelU-16 [-1, 126, 64, 4] 256, 400 RelU-17 [-1, 16, 64, 64] 1, 168 RelU-17 [-1, 16, 64, 64] 0 RelU-18 RelU-17 [-1, 16, 64, 64] 0 RelU-18 RelU-17 [-1, 16, 64, 64] 0 RelU-18 RelU-11 [-1, 16, 64, 64] 0 RelU-11 [-1, 16, 64, 64] 0 RelU-12 [-1, 12, 23, 32] 0 RelU-12 [-1, 12, 32, 32] 0 RelU-12 [-1, 32, 32, 32] 0 RelU-12 [-1, 12, 44, 4] 0 RelU-12 [-1, 44, 16, 16] 0 RelU-12 [-1,			296			
Conv2d-9 Conv2d-10 Conv2d-	BatchNorm2d-6	[-1, 8, 128, 128]	16			
Conv2d-9						
BatchNorm2d-10 [-1, 8, 128, 128] 16						
ReLU-11 [-1, 8, 128, 128] 0 ConVTranspose2d-65 [-1, 125, 4, 4] 252, 400 Conv2d-15 [-1, 8, 64, 64] 0 Conv2d-15 [-1, 16, 64, 64] 1, 168 BatchNorm2d-16 [-1, 128, 4, 4] 256 ReLU-17 [-1, 16, 64, 64] 0 Conv2d-15 [-1, 16, 64, 64] 0 Conv2d-15 [-1, 16, 64, 64] 0 Conv2d-16 [-1, 128, 4, 4] 0 Conv2d-17 [-1, 128, 4, 4] 0 Conv2d-18 [-1, 16, 64, 64] 0 Conv2d-19 [-1, 64, 8, 8] 73,792 BatchNorm2d-20 [-1, 16, 64, 64] 0 Conv2d-19 [-1, 64, 8, 8] 0 Conv2d-19 [-1, 64, 64] 0 Conv2d-19 [-1, 64, 8, 8] 0 Conv2d-19 [-1, 64, 16, 16] 0 Conv2d-						
Conv-12						
MaxPool2d-13						
Conv2d-15						
Conv2d-15						
BatchNorm2d-16			-			
ReLU-17						
Conv2d-19 [-1, 16, 64, 64] 2,320 Conv7ansposeZd-72 (-1, 128, 8, 8] 65, 664 Conv2d-19 [-1, 16, 64, 64] 2,320 BatchNorm2d-20 [-1, 16, 64, 64] 0 Conv2d-19 [-1, 16, 64, 64] 0 BatchNorm2d-20 [-1, 16, 64, 64] 0 Conv2d-23 [-1, 16, 32, 32] 0 BatchNorm2d-77 [-1, 64, 8, 8] 0 Conv2d-25 [-1, 32, 32, 32] 0 Conv2d-25 [-1, 32, 32, 32] 0 Conv2d-25 [-1, 32, 32, 32] 0 Conv2d-26 [-1, 32, 32, 32] 0 Conv2d-26 [-1, 32, 32, 32] 0 Conv2d-27 [-1, 64, 16, 16] 18,464 Conv-28 [-1, 32, 32, 32] 0 Conv2d-29 [-1, 32, 32, 32] 0 Conv2d-80 [-1, 32, 16, 16] 0 Conv2d-35 [-1, 32, 16, 16] 0 Conv2d-35 [-1, 64, 16, 16] 0 Conv2d-37 [-1, 64, 16, 16] 0 Conv2d-38 [-1, 64, 16, 16] 0 Conv2d-39 [-1, 64, 16, 16] 0 Conv2d-36 [-1, 64, 16, 16] 0 Conv2d-37 [-1, 64, 16, 16] 0 Conv2d-38 [-1, 64, 16, 16] 0 Conv2d-39 [-1, 64, 16, 16] 0 Conv2d-39 [-1, 64, 16, 16] 0 Conv2d-36 [-1, 64, 16, 16] 0 Conv2d-37 [-1, 64, 16, 16] 0 Conv2d-38 [-1, 64, 16, 16] 0 Conv2d-39 [-1, 64, 16, 16] 0 Conv2d-49 [-1, 64, 16, 16] 0 Conv2d-49 [-1, 128, 8, 8] 0						
Conv2d-19						
BatchNorm2d-20						
ReLU-21				BatchNorm2d-74	[-1, 64, 8, 8]	
Conv-22				ReLU-75	[-1, 64, 8, 8]	
MaxPool2d-23				conv-76	[-1, 64, 8, 8]	
down-24				BatchNorm2d-77	[-1, 64, 8, 8]	
Conv2d-25						
BatchNorm2d-26 ReLU-27 ReLU-27 ReLU-27 ReLU-28 ReLU-28 ReLU-31 ReLU-31 ReLU-31 ReLU-33 ReLU-33 ReLU-33 ReLU-33 ReLU-33 ReLU-33 Relu-33 Relu-33 Relu-33 Relu-34 Relu-37 Relu-38 Relu-39 Relu-38 Relu-39 Relu-30						
ReLU-27						
Conv-28			0			
Conv2d-39	conv-28		0			
BatchNorm2d-36	Conv2d-29	[-1, 32, 32, 32]	9,248			
RELU-31	BatchNorm2d-30	[-1, 32, 32, 32]	64			
Conv2d-87 [-1, 16, 32, 32] 4,624 MaxPool2d-33 [-1, 32, 16, 16] 0 Conv2d-35 [-1, 64, 16, 16] 0 BatchNorm2d-36 [-1, 64, 16, 16] 128 BatchNorm2d-36 [-1, 64, 16, 16] 0 Conv2d-39 [-1, 64, 16, 16] 0 BatchNorm2d-39 [-1, 64, 16, 16] 0 Conv2d-39 [-1, 64, 16, 16] 0 BatchNorm2d-40 [-1, 64, 16, 16] 128 BatchNorm2d-91 [-1, 16, 32, 32] 32 Conv3d-39 [-1, 64, 16, 16] 0 BatchNorm2d-39 [-1, 64, 16, 16] 128 BatchNorm2d-91 [-1, 16, 32, 32] 32 Conv7ranspose2d-93 [-1, 16, 32, 32] 0 Conv2d-39 [-1, 16, 46, 16, 16] 36,928 BatchNorm2d-40 [-1, 64, 16, 16] 0 ReLU-41 [-1, 64, 16, 16] 0 Conv-42 [-1, 64, 16, 16] 0 MaxPool2d-43 [-1, 64, 16, 16] 0 Conv3d-45 [-1, 18, 8, 8] 0 Conv3d-45 [-1, 128, 8, 8] 0 BatchNorm2d-98 [-1, 8, 64, 64] 16 BatchNorm2d-46 [-1, 128, 8, 8] 0 Conv3d-45 [-1, 128, 8, 8] 0 ReLU-47 [-1, 128, 8, 8] 256 Conv3d-49 [-1, 128, 8, 8] 0 ReLU-47 [-1, 128, 8, 8] 147,584 Conv2d-49 [-1, 128, 8, 8] 147,584 BatchNorm2d-98 [-1, 4, 128, 128] 0 ReLU-51 [-1, 128, 8, 8] 0 Conv-52 [-1, 128, 8, 8] 0 Conv-52 [-1, 128, 8, 8] 0 Conv-64 [-1, 128, 8, 8] 0 ReLU-51 [-1, 128, 8, 8] 0 Conv-52 [-1, 128, 8, 8] 0 Conv-188 [-1, 1, 128, 128] 0 Conv-188 [-1, 1, 128, 128] 0 Conv-196 [-1, 1, 128, 128] 0 Conv-198 [-1, 1, 128, 128	ReLU-31	[-1, 32, 32, 32]	0			
MaxPool2d-33 down-34 [-1, 32, 16, 16] 0 down-34 [-1, 16, 32, 32] 0 down-34 [-1, 16, 16, 16] 0 down-34 [-1, 16, 32, 32] 0 down-34 [-1, 16, 16] 0 down-34 [-1, 12, 12, 12, 12] 0 down-34 [-1, 12, 12, 12] 0 down-34 [-1, 12, 12, 12] 0 down-34 [-1, 12, 12, 12] 0 down-34 [-	conv-32	[-1, 32, 32, 32]	0			
down-34 [-1, 32, 16, 16] 0 ReLU-89 [-1, 16, 32, 32] 0 Conv2d-35 [-1, 64, 16, 16] 18,496 conv-90 [-1, 16, 32, 32] 0 BatchNorm2d-36 [-1, 64, 16, 16] 128 BatchNorm2d-91 [-1, 16, 32, 32] 32 conv-38 [-1, 64, 16, 16] 0 up-92 [-1, 16, 32, 32] 0 Conv2d-39 [-1, 64, 16, 16] 36,928 Conv72d-94 [-1, 8, 64, 64] 1,940 ReLU-41 [-1, 64, 16, 16] 0 ReLU-96 [-1, 8, 64, 64] 16 conv-42 [-1, 64, 8, 8] 0 ReLU-96 [-1, 8, 64, 64] 16 down-44 [-1, 64, 8, 8] 0 BatchNorm2d-98 [-1, 8, 64, 64] 16 down-45 [-1, 128, 8, 8] 73,856 ConvTanspose2d-100 [-1, 8, 64, 64] 16 down-44 [-1, 64, 8, 8] 0 BatchNorm2d-98 [-1, 8, 64, 64] 16 BatchNorm2d-46 [-1, 128, 8, 8] 0 Conv2d-101 [-1, 4, 128, 128] 264 BatchNorm2d-49	MaxPool2d-33	[-1, 32, 16, 16]				
Conv2d-35			0			
BatchNorm2d-36						
ReLU-37 [-1, 64, 16, 16] 0 Up-92 [-1, 16, 32, 32] 0 Conv2d-38 [-1, 64, 16, 16] 0 Conv2d-39 [-1, 64, 16, 16] 36,928 Conv2d-94 [-1, 8, 64, 64] 1,160 BatchNorm2d-40 [-1, 64, 16, 16] 128 BatchNorm2d-95 [-1, 8, 64, 64] 1,160 Conv-42 [-1, 64, 16, 16] 0 Conv-42 [-1, 64, 8, 8] 0 BatchNorm2d-98 [-1, 8, 64, 64] 16 Conv-44 [-1, 64, 8, 8] 0 BatchNorm2d-98 [-1, 8, 64, 64] 16 Conv-44 [-1, 128, 8, 8] 0 Conv-45 [-1, 128, 8, 8] 0 Conv-46 [-1, 128, 8, 8] 0 Conv-47 [-1, 128, 128] 264 BatchNorm2d-46 [-1, 128, 8, 8] 0 BatchNorm2d-100 [-1, 4, 128, 128] 292 ReLU-47 [-1, 128, 8, 8] 0 BatchNorm2d-102 [-1, 4, 128, 128] 292 Conv-48 [-1, 128, 8, 8] 0 BatchNorm2d-102 [-1, 4, 128, 128] 0 Conv-48 [-1, 128, 8, 8] 147,584 Conv-104 [-1, 4, 128, 128] 0 ReLU-51 [-1, 128, 8, 8] 0 Conv-104 [-1, 4, 128, 128] 0 Conv-105 [-1, 128, 8, 8] 0 Conv-106 [-1, 4, 128, 128] 0 Conv-106 [-1, 4, 4, 4] 0 Conv-108 [-1, 4, 4, 4] 0 C				BatchNorm2d-91		
Conv2d-39						
BatchNorm2d-40				ConvTranspose2d-93		1,040
ReLU-41 [-1, 64, 16, 16] 0 ReLU-96 [-1, 8, 64, 64] 0 conv-97 [-1, 8, 64, 64] 0 ReLU-96 [-1, 8, 64, 64] 0 ReLU-96 [-1, 8, 64, 64] 0 ReLU-96 [-1, 8, 64, 64] 10 ReLU-96 [-1, 8, 64, 64] 10 ReLU-96 [-1, 64, 8, 8] 0 ReLU-96 [-1, 8, 64, 64] 16 ReLU-96 [-1, 128, 8, 8] 0 ReLU-97 [-1, 128, 64, 64] 16 ReLU-97 [-1, 128, 8, 8] 256 ReLU-47 [-1, 128, 8, 8] 256 ReLU-47 [-1, 128, 8, 8] 0 ReLU-97 [-1, 4, 128, 128] 292 ReLU-47 [-1, 128, 8, 8] 0 ReLU-98 [-1, 4, 128, 128] 8 ReLU-98 [-1, 128, 8, 8] 147,584 Conv-104 [-1, 4, 128, 128] 8 ReLU-98 [-1, 128, 8, 8] 256 ReLU-99 [-1, 4, 128, 128] 0 ReLU-103 [-1, 4, 128, 128] 0 ReLU-104 [-1, 4, 128, 128] 0 ReLU-105 [-1, 128, 8, 8] 0 ReLU-106 [-1, 4, 128, 128] 0 ReLU-107 [-1, 128, 128				Conv2d-94	[-1, 8, 64, 64]	1,160
conv-42				BatchNorm2d-95	[-1, 8, 64, 64]	
MaxPool2d-43						
down-44						
Conv2d-45						
BatchNorm2d-46 [-1, 128, 8, 8] 256 Conv2d-101 [-1, 4, 128, 128] 292 ReLU-47 [-1, 128, 8, 8] 0 BatchNorm2d-102 [-1, 4, 128, 128] 8 conv-48 [-1, 128, 8, 8] 0 ReLU-103 [-1, 4, 128, 128] 0 Conv2d-49 [-1, 128, 8, 8] 147,584 conv-104 [-1, 4, 128, 128] 0 BatchNorm2d-50 [-1, 128, 8, 8] 256 BatchNorm2d-105 [-1, 4, 128, 128] 8 ReLU-51 [-1, 128, 8, 8] 0 conv-104 [-1, 4, 128, 128] 0 conv-52 [-1, 128, 8, 8] 0 conv-204 [-1, 1, 128, 128] 0 Conv-52 [-1, 128, 4, 4] 0 conv-108 [-1, 1, 128, 128] 0 MaxPool2d-53 [-1, 128, 4, 4] 0 conv-108 [-1, 1, 128, 128] 0						
ReLU-47 [-1, 128, 8, 8] 0 BatchNorm2d-102 [-1, 4, 128, 128] 8 conv-48 [-1, 128, 8, 8] 0 ReLU-103 [-1, 4, 128, 128] 0 Conv-49 [-1, 128, 8, 8] 147,584 conv-104 [-1, 4, 128, 128] 0 BatchNorm2d-50 [-1, 128, 8, 8] 256 BatchNorm2d-105 [-1, 4, 128, 128] 8 ReLU-51 [-1, 128, 8, 8] 0 up-106 [-1, 4, 128, 128] 8 Conv-52 [-1, 128, 8, 8] 0 Conv-2d-107 [-1, 1, 128, 128] 37 MaxPool2d-53 [-1, 128, 4, 4] 0 conv-108 [-1, 1, 128, 128] 0						
conv-48 [-1, 128, 8, 8] 0 RelU-103 [-1, 4, 128, 128] 0 Conv2d-49 [-1, 128, 8, 8] 147,584 conv-104 [-1, 4, 128, 128] 0 BatchNorm2d-50 [-1, 128, 8, 8] 256 BatchNorm2d-105 [-1, 4, 128, 128] 8 RelU-51 [-1, 128, 8, 8] 0 up-106 [-1, 4, 128, 128] 0 conv-52 [-1, 128, 8, 8] 0 Conv2d-107 [-1, 1, 128, 128] 37 MaxPool2d-53 [-1, 128, 4, 4] 0 conv-108 [-1, 1, 128, 128] 0						
Conv2d-49 [-1, 128, 8, 8] 147,584 conv-104 [-1, 4, 128, 128] 0 BatchNorm2d-50 [-1, 128, 8, 8] 256 BatchNorm2d-105 [-1, 4, 128, 128] 8 ReLU-51 [-1, 128, 8, 8] 0 up-106 [-1, 4, 128, 128] 0 conv-52 [-1, 128, 8, 8] 0 Conv2d-107 [-1, 1, 128, 128] 37 MaxPool2d-53 [-1, 128, 4, 4] 0 conv-108 [-1, 1, 128, 128] 0						8
BatchNorm2d-50 [-1, 128, 8, 8] 256 BatchNorm2d-105 [-1, 4, 128, 128] 8 ReLU-51 [-1, 128, 8, 8] 0 up-106 [-1, 4, 128, 128] 0 conv-52 [-1, 128, 8, 8] 0 Conv2d-107 [-1, 1, 128, 128] 37 MaxPool2d-53 [-1, 128, 4, 4] 0 conv-108 [-1, 1, 128, 128] 0						
ReLU-51 [-1, 128, 8, 8] 0 up-106 [-1, 4, 128, 128] 0 conv-52 [-1, 128, 8, 8] 0 Conv2d-107 [-1, 1, 128, 128] 37 MaxPool2d-53 [-1, 128, 4, 4] 0 conv-108 [-1, 1, 128, 128] 0						
conv-52 [-1, 128, 8, 8] 0 Conv2d-107 [-1, 1, 128, 128] 37 MaxPool2d-53 [-1, 128, 4, 4] 0 conv-108 [-1, 1, 128, 128] 0						
MaxPool2d-53 [-1, 128, 4, 4] 0 conv-108 [-1, 1, 128, 128] 0						
i i i i i i i i i i i i i i i i i i i						
	down-54		0			

3. Loss functions:

The loss functions I used is the default loss function:

crit = nn.BCEWithLogitsLoss() # Define the loss function

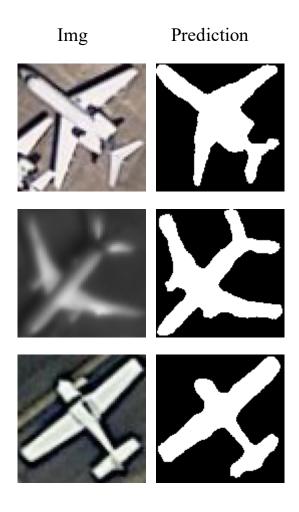
Epoch	Loss	Epoch	Loss
0	0.548725962638855	74	0.10606835782527924
1	0.35738807916641235	75	0.10531891882419586
2	0.29464346170425415	76	0.10455069690942764
3	0.2662748098373413	77	0.10415859520435333
4	0.24825114011764526	78	0.10370544344186783
69	0.10856733471155167	145	0.08330231159925461
70	0.10829269886016846	146	0.08313968032598495
71	0.10713459551334381	147	0.08315932005643845
7 <i>2</i>	0.10693884640932083	148	0.08290217071771622
73	0.10650912672281265	149	0.08267544955015182

4. IoU:

The final mean IoU of my model is:

Mean IoU: 0.8916377923312807

5. Visualize:



Part 3: Instance Segmentation

1. Kaggle Group Name: ChengH.

Members: Cheng Hu (301435966), Kaikun Fang (301416542)

2. Best accuracy:

The highest score of our group on Kaggle is 0.83098

3. Visualization:

My best result:







Part4: Mask R-CNN

1. Config:

```
cfg. merge_from_file(model_zoo.get_config_file("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yam1"))
cfg. DATASETS. TRAIN = ("plane_train",)
cfg. DATASETS. TEST = ()
cfg. DATALOADER. NUM_WORKERS = 2
cfg. MODEL. WEIGHTS = model_zoo.get_checkpoint_url("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yam1")
cfg. SOLVER. IMS_PER_BATCH = 2  # batch size
cfg. SOLVER. BASE_LR = 0.0003  # LR
cfg. SOLVER. MAX_ITER = 3000
cfg. MODEL. ROI_HEADS. BATCH_SIZE_PER_IMAGE = 512
cfg. MODEL. ROI_HEADS. NUM_CLASSES = 1  # only have plane
```

2. Visualization:



Figure 5



Figure 6

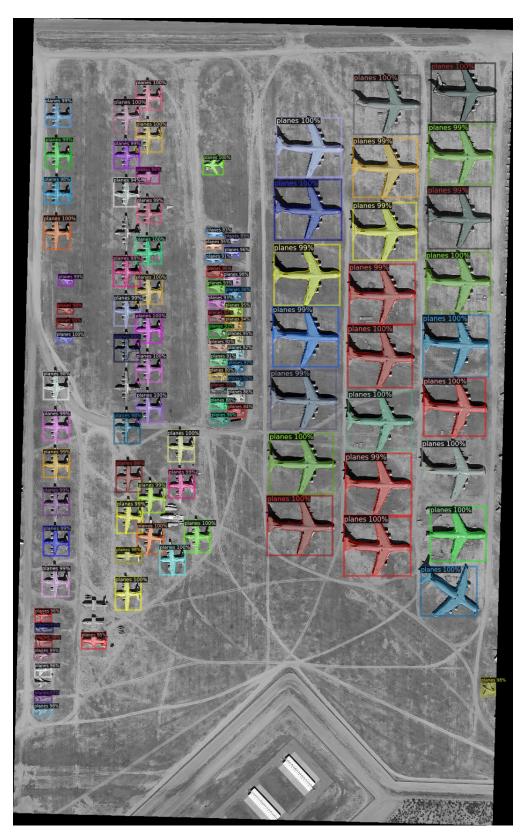


Figure 7

2. Evaluation:

Con: The segmentation of the planes fuselage is not accurate enough. In some results, other objects next to the plane are incorrectly counted as part of the plane. And In Figure 5, the number of ships incorrectly identified as planes is relatively high, this is not as good as part1's performance.

Pro: The trained model is able to find the vast majority of planes in the picture. And faster.

3. Difference between Part 3 and Part 4:

The comparison of the results revealed that. In the detection of the number of planes, part4 is higher than the number detected by part3. But in the partitioning of the airplane fuselage and background, part3 is more accurate in partitioning the fuselage. part4 will incorrectly partition some things near the fuselage as part of the fuselage as well.