HW 3

Oscar Lin 309553018
Project repository: https://github.com/clashroyaleisgood/Course_VRDL/tree/main/HW3_Instance_Segmentation

Competition: https://codalab.lisn.upsaclay.fr/competitions/333?secret_key=3b31d945-289d-4da6-939d-39435b506ee5

Table of Contents

Introduction

This challenge is a Nuclei segmentation task with dataset of 24 training and 6 testing images(1000 x 1000)
The difficulty to this challenge is the image size.
The images are too large to train, especially in segmentation problem which needs a large model architecture.
But the Nuclei are too small compare to original image, result will be bad if I reduce the input size.
So I finally set hyperparameter: IMS_PER_BATCH to 1 to prevent OOM.

Result

score: **0.24303**

Data Pre-Processing

In this challenge I only do the label format transform from many 0/1 mask images to RLE form, and place all the information correctly to fit COCO format.
with help of these websites:

- https://www.immersivelimit.com/tutorials/create-coco-annotations-from-scratch
- https://www.gushiciku.cn/pl/gavr/zh-tw
- https://www.aiuai.cn/aifarm1578.html

Architecture:

```
train_image_root/
    ├── 1.png
    └── 2.png...
test_image_root/
    ├── 3.png
    └── 4.png...
```

and a config file which describes image paths and masks(RLE form or Polygon)
https://cocodataset.org/#format-data

Model architecture

I use Detectron2, which is a platform for object detection, segmentation and other visual recognition tasks, to help me combine the environment and models.
And I use X101-FPN ResNeXt-101 + FPN in the end which gives me the highest score after many times of training.

Hyperparameters

```
INPUT.MIN_SIZE_TRAIN = 1000
DATALOADER.NUM_WORKERS = 2
SOLVER.IMS_PER_BATCH = 1
SOLVER.BASE_LR = 0.00025
SOLVER.STEPS = (3000, 3500, 4500)
SOLVER.MAX_ITER = 5000
MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 256
MODEL.ROI_HEADS.NUM_CLASSES = 1
```

Firstly I use R101-FPN to train on this task.

I try to train longer and use larger batch size per image, get a better result.

4	0.167375	answer.zip	12/13/2021 16:17:43	85969	Finished		−

Description:

```
1st
mask_rcnn_R_101_FPN_3x
cfg.DATALOADER.NUM_WORKERS = 2
cfg.SOLVER.IMS_PER_BATCH = 1
cfg.SOLVER.MAX_ITER = 400
cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 128
```

5	0.201543	answer.zip	12/14/2021 05:41:24	92749	Finished		−

Description:

```
2nd
mask_rcnn_R_101_FPN_3x
cfg.DATALOADER.NUM_WORKERS = 2
cfg.SOLVER.IMS_PER_BATCH = 1
cfg.SOLVER.MAX_ITER = 1200
cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 512
```

Try to restrict the min value of image sizes, get a better result.

6	0.220453	answer.zip	12/14/2021 09:37:40	96524	Finished		−

Description:

```
3rd
cfg.DATALOADER.NUM_WORKERS = 2
cfg.SOLVER.IMS_PER_BATCH = 1
cfg.SOLVER.MAX_ITER = 1200
cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 512
cfg.INPUT.MIN_SIZE_TRAIN = 1000
```

Try to reduce lr at certain steps [6000, 8000] (total: 9000), get a slightly better result.

12	0.230469	answer.zip	12/15/2021 05:13:18	118223	Finished		−

Description:

```
exp5
cfg.INPUT.MIN_SIZE_TRAIN = 1000

cfg.DATALOADER.NUM_WORKERS = 2
cfg.SOLVER.IMS_PER_BATCH = 1
cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 512

cfg.SOLVER.STEPS = (6000, 8000)
cfg.SOLVER.MAX_ITER = 9000
```

Try not to use the final weights to prevent overfitting,

20	0.232164	answer.zip	12/15/2021 13:36:30	132094	Finished	

Description:

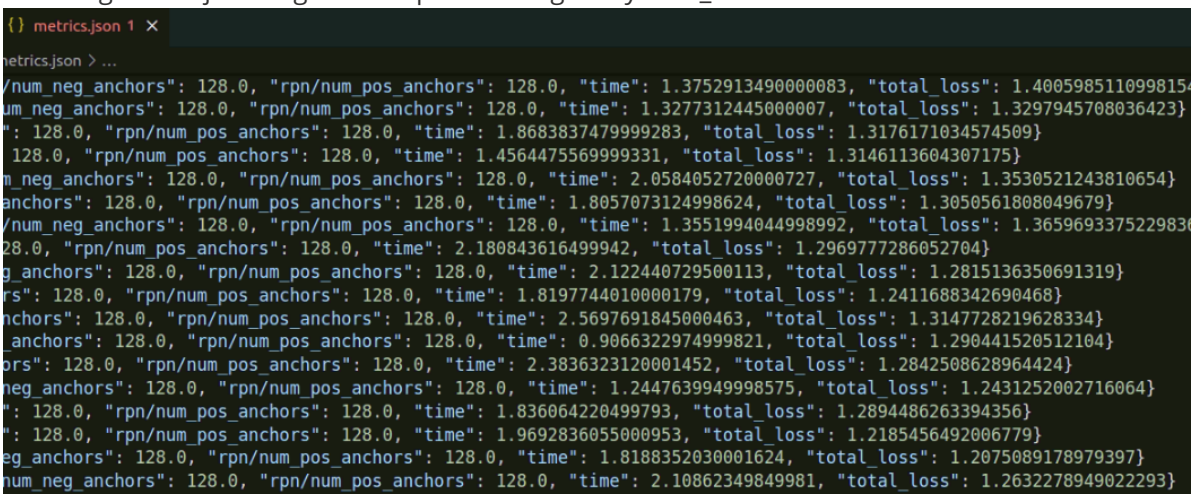
```
exp7
cfg.INPUT.MIN_SIZE_TRAIN = 1000

cfg.DATALOADER.NUM_WORKERS = 2
cfg.SOLVER.IMS_PER_BATCH = 1
cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 256

cfg.SOLVER.STEPS = (2000, 2500)
cfg.SOLVER.MAX_ITER = 3000
```

23	0.232837	answer.zip	12/15/2021 14:39:48	133582	Finished	

Description:

```
exp7 epoch 0002099
```

Checking metric.json to guess the perfect weights by total_loss



```
{} metrics.json 1 ×
metrics.json > ...
/num_neg_anchors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 1.3752913490000083, "total_loss": 1.400598511099815
um_neg_anchors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 1.3277312445000007, "total_loss": 1.3297945708036423}
": 128.0, "rpn/num_pos_anchors": 128.0, "time": 1.8683837479999283, "total_loss": 1.3176171034574509}
 128.0, "rpn/num_pos_anchors": 128.0, "time": 1.4564475569999331, "total_loss": 1.3146113604307175}
m_neg_anchors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 2.0584052720000727, "total_loss": 1.3530521243810654}
anchors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 1.8057073124998624, "total_loss": 1.3050561808049679}
/num_neg_anchors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 1.3551994044998992, "total_loss": 1.365969337522983
28.0, "rpn/num_pos_anchors": 128.0, "time": 2.180843616499942, "total_loss": 1.2969777286052704}
g_anchors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 2.122440729500113, "total_loss": 1.2815136350691319}
rs": 128.0, "rpn/num_pos_anchors": 128.0, "time": 1.8197744010000179, "total_loss": 1.2411688342690468}
nchors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 2.5697691845000463, "total_loss": 1.3147728219628334}
_anchors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 0.9066322974999821, "total_loss": 1.290441520512104}
ors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 2.3836323120001452, "total_loss": 1.2842508628964424}
neg_anchors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 1.2447639949998575, "total_loss": 1.2431252002716064}
": 128.0, "rpn/num_pos_anchors": 128.0, "time": 1.836064220499793, "total_loss": 1.2894486263394356}
": 128.0, "rpn/num_pos_anchors": 128.0, "time": 1.9692836055000953, "total_loss": 1.2185456492006779}
eg_anchors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 1.8188352030001624, "total_loss": 1.2075089178979397}
num_neg_anchors": 128.0, "rpn/num_pos_anchors": 128.0, "time": 2.10862349849981, "total_loss": 1.2632278949022293}
```

I also try many combinations of hyperparameters, different weights collected at different epoch, the value of predict threshold, ... but not very useful. I'm stuck at behind the baseline a little bit, but always can not over it.

Finally I use X101-FPN to train my model with hyperparameters mentioned before and get best result at epoch 2099.

After getting this perfect result, I still tried other weights from different epochs, but no one is better than 2099. When I tries to submit the epochs before 2099, system tells me that I met the

max submission counts...

30	0.243034	answer.zip	12/16/2021 09:21:22	154526	Finished	✔	−

Description:

```
exp9 epoch 2099
cfg.SOLVER.STEPS = (3000, 3500, 4500)
cfg.SOLVER.MAX_ITER = 5000

-----------------------------------------------

mask_rcnn_X_101_32x8d_FPN_3x
```

31	0.230067	answer.zip	12/16/2021 09:23:49	154776	Finished		−

Description:

```
exp9 final.pth
```

32	0.231201	answer.zip	12/16/2021 09:27:24	154902	Finished		−

Description:

```
exp9 epoch 2999
```

33	0.2306	answer.zip	12/16/2021 09:32:10	155153	Finished		−

Description:

```
exp9 epoch 2549
```

Summary

In this challenge, I use Detectron2 platform and X101-FPN as my instance segmentation model and get score **0.24303** on test data.

During the fine-tuning, I struggle with so many parameters to edit, and many edits looks useless in experiments. It really brought me a lot of Stress. It's really hard to fine-tune a model not only technically but also mentally. Luckily I find a pair that cross the bassline.