**Documentation: Implementation of Mask-RCNN using TensorFlow API**

***Prepared By: Lim Jun Jie***

**Environment Set-Up**

* Refer to my github link for the instruction to implement TensorFlow GPU (CUDA) in Windows <https://github.com/JJLim99/Implementation-of-TensorFlow-GPU-CUDA-in-Windows.git>
* If you tend to train the model using your own graphic card, make sure you have enough GPU, otherwise you can use Google Colab (GPU runtime).

**Mask-RCNN Project Set-up**

1. Download the project repository from [*https://github.com/JJLim99/SDD*](https://github.com/JJLim99/SDD).
2. Download the latest tensorflow models from [*https://github.com/tensorflow/models*](https://github.com/tensorflow/models) and replace it into the project repository.
3. In your current project repository, you must have following file:
4. CP (CheckPoint for your train model)
5. dataset

- annotations

- masks (annotated masks images in png)

- xmls (xmls files for the annotated images)

* JPEGImages (train images in JPEG)
* TestImages (test images in JPEG)
* label.pbtxt

1. IG (To save inference graph of your train model)
2. models
3. pre\_trained\_models (To store pre-trained models, can try more models from [*https://github.com/tensorflow/models/blob/master/research/object\_detection/g3doc/detection\_model\_zoo.md*](https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md))
4. Supporting\_Script (Contains resize\_image.py to resize and rename your dataset)
5. mask\_rcnn\_inception\_v2\_coco (or any other .config file that u want to use, search more .config files in *PATHTOPROJECT\models\research\object\_detection\samples\configs*)
6. mask\_rcnn\_eval.ipnyb (To test your train model)
7. create\_mask\_rcnn\_tf\_record.py (Copy this file into *PATHTOPROJECT\models\research\object\_detection\dataset\_tools*)

**Implementation of Mask-RCNN**

1. Preparing datasets
2. Obtain your dataset. You can resize and rename your dataset using resize\_images.py in Supporting\_Script file.
3. Preparing annotated masks images using PixelAnnotationTool

-Download PixelAnnotationTool from [*https://github.com/abreheret/PixelAnnotationTool*](https://github.com/abreheret/PixelAnnotationTool)

-Save your annotated masks images in *PATHTOPROJECT/annotations/masks*

1. Preparing xml files for annotated images using labelImg

-Download labelImg from [*https://github.com/tzutalin/labelImg*](https://github.com/tzutalin/labelImg)

-Save your xml files in *PATHTOPROJECT/annotations/xmls*

1. Run [Check\_pixel\_values.ipynb](https://github.com/vijendra1125/Custom-Mask-RCNN-using-Tensorfow-Object-detection-API/blob/master/extra/Check_pixel_values.ipynb) to check the pixel of certain object.
2. Go to label.pbtxt and create\_mask\_rcnn\_tf\_record.py, label the object with its corresponding pixel value. In label.pbtxt, the last 3 digits refer to the pixel value of the object.
3. Generating tf record
4. Before generating tf record, in cmd, go to *PATHTOPROJECT/models/research* and run the following command to prevent error.

*set PYTHONPATH=PATHTOPROJECT\models;PATHTOPROJECT\models\research;PATHTOPROJECT\models\research\slim  
set PATH=%PATH%;%PYTHONPATH%*

*protoc object\_detection/protos/\*.proto --python\_out=.*

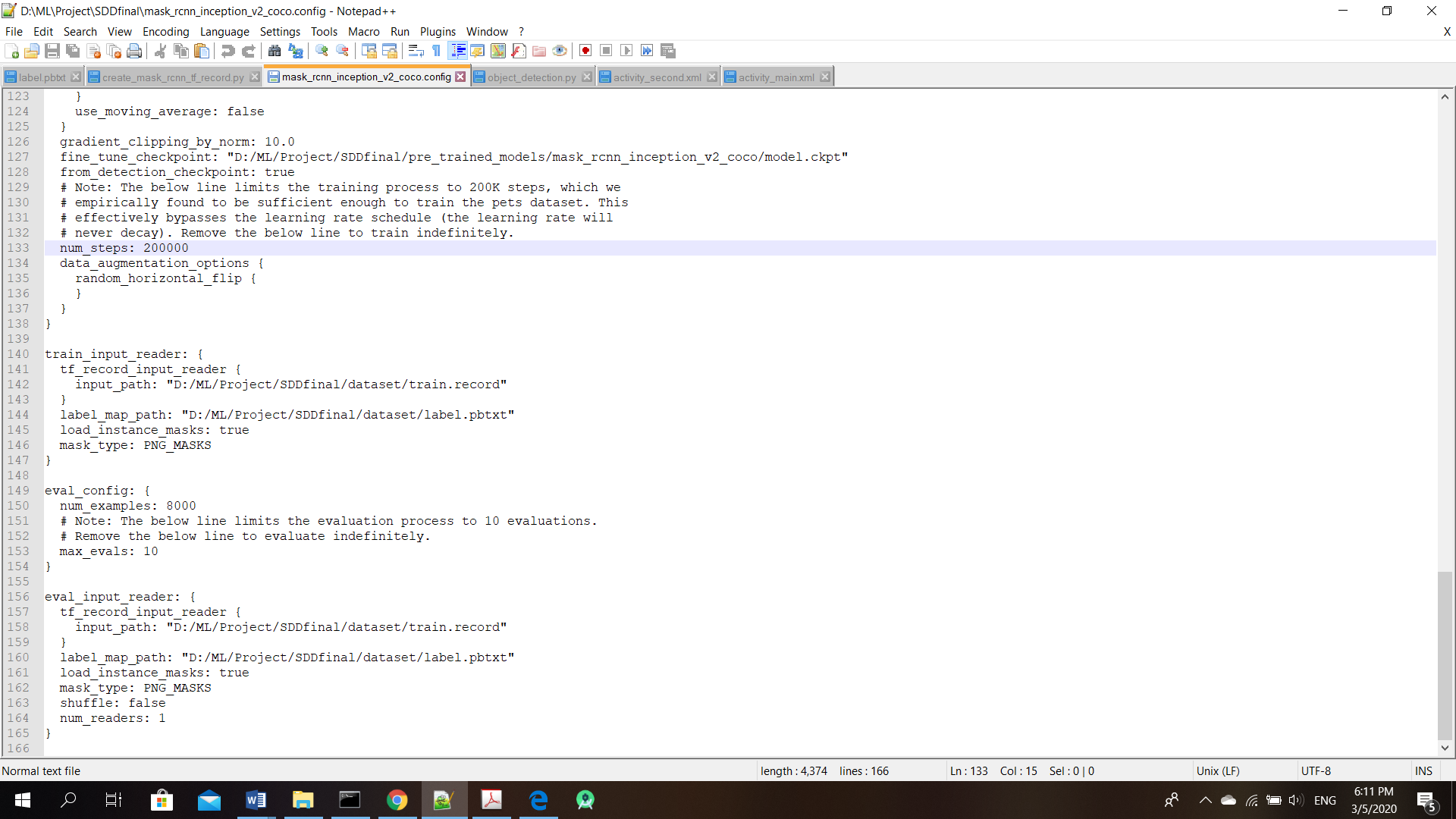
*python setup.py build*

*python setup.py install*

1. Then, at the *PATHTOPROJECT\models\research*, run the following command to generate tf record.

*python object\_detection/dataset\_tools/create\_mask\_rcnn\_tf\_record.py --data\_dir=PATHTOPROJECT/dataset --annotations\_dir=Annotations --image\_dir=JPEGImages --output\_dir=PATHTOPROJECT/dataset/train.record --label\_map\_path=PATHTOPROJECT/dataset/label.pbtxt*

1. Training
2. Open the .config file in your project repository and modify 5 highlighted parts as shown in the image below according to your path. (I used mask\_rcnn\_inception\_v2\_coco.config)



1. Download the Mask-RCNN pre-trained models from <https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md> (I used mask\_rcnn\_inception\_v2\_coco)
2. Run the following command to start training.

*python object\_detection/legacy/train.py --train\_dir=PATHTOPROJECT/CP --pipeline\_config\_path=PATHTOPROJECT/mask\_rcnn\_inception\_v2\_coco.config*

# Train the models until the loss is below 0.2 or lower.

1. Generating Inference\_Graph

Once the training had been done, run the following command to generate inference\_graph.

*python object\_detection/export\_inference\_graph.py --input\_type=image\_tensor --pipeline\_config\_path=PATHTOPROJECT/mask\_rcnn\_inception\_v2\_coco.config --trained\_checkpoint\_prefix=PATHTOPROJECT/CP/model.ckpt-****20000*** *--output\_directory=PATHTOPROJECT/IG*

# The highlighted test in the command above should be changed according to your steps of training

1. Test the trained models.

Open jupyter notebook and run mask\_rcnn\_eval.ipnyb to test the trained models. Change the PATH as commented in the script.