6-Sept-24

Tran Duc Anh Dang | 103995439

Studio Class: Studio 1-3

PORTFOLIO

WEEK 5 - DOCUMENTATION

# Abstract

This portfolio submission demonstrates the development and evaluation of deep learning models for image classification and object detection tasks using CNN, ResNet50, and Mask R-CNN. The submission includes labeled datasets, model outcomes, and source code structured as follows:

* Labeled Log Dataset: A set of 10 annotated images using the LabelMe tool and the corresponding JSON file.
* CNN Model: Test outcomes of a basic CNN model trained to classify images into "rust" and "no rust," with images and results saved in the folder “cnn\_test.”
* ResNet50 Model: Test outcomes of a ResNet50 model for the same classification task, with results saved in “resnet50\_test.”
* Mask R-CNN Model: Developed for detecting log objects in images. The model outputs detection results, including bounding boxes, confidence scores, and segmentation masks. The results are saved in the “rcnn\_test” folder.
* Source Code: The source code for all models is located in the folder “code.”

You can find the requirements, documentation and source code file at:

**Requirements:** https://github.com/kinqsradio/COS40007-Artificial-Intelligence-for-Engineering/tree/main/week-05-portfolio/requirements

**Documentation:** https://github.com/kinqsradio/COS40007-Artificial-Intelligence-for-Engineering/tree/main/week-05-portfolio/docs

**Code:** https://github.com/kinqsradio/COS40007-Artificial-Intelligence-for-Engineering/tree/main/week-05-portfolio/code

**dataset.json:** https://github.com/kinqsradio/COS40007-Artificial-Intelligence-for-Engineering/tree/main/week-05-portfolio/my\_coco\_annotations

**cnn\_test:** https://github.com/kinqsradio/COS40007-Artificial-Intelligence-for-Engineering/tree/main/week-05-portfolio/cnn\_test

**resnet50\_test:** https://github.com/kinqsradio/COS40007-Artificial-Intelligence-for-Engineering/tree/main/week-05-portfolio/resnet50\_test

**rcnn\_test:** https://github.com/kinqsradio/COS40007-Artificial-Intelligence-for-Engineering/tree/main/week-05-portfolio/rnn\_test

Table of Contents

[Abstract 1](#_Toc176551424)

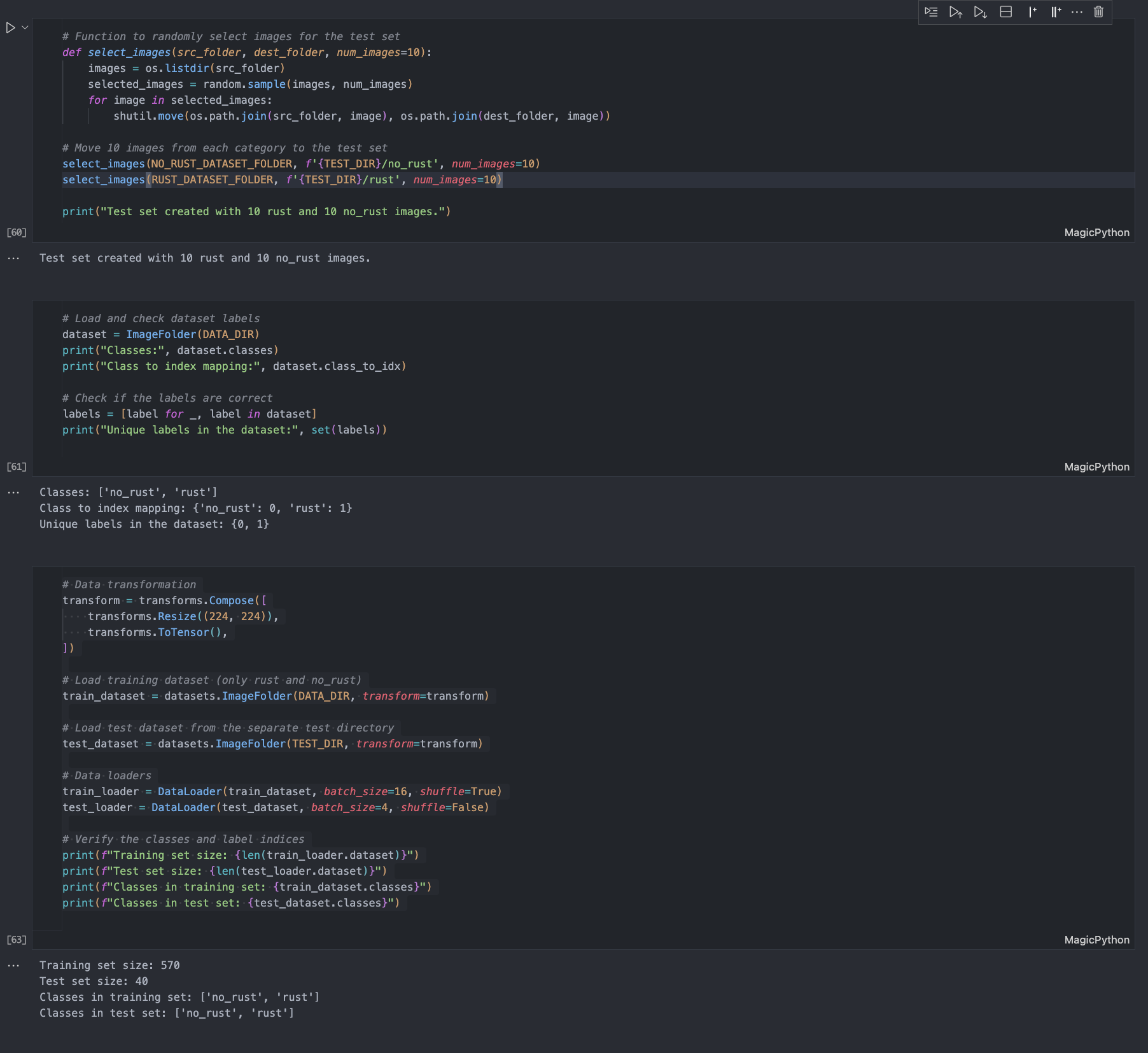
[Task 1: Develop CNN and Resnet50 3](#_Toc176551425)

[Task 2: Develop Mask RCNN for Detecting Log 6](#_Toc176551426)

[Task 3: Extending Log Labelling to Another Class 8](#_Toc176551427)

# Task 1: Develop CNN and Resnet50

1. First randomly take out 10 rust and 10 no rust images for testing (We call this as Test Set). So, your training set will not contain these 20 images. yes



1. Develop a simple CNN model similar to minst classification but train with the provided corrosion data with class “rust” and “no rust” (that excludes Test Set). Once the model is trained and saved test with your Test dataset and measure the accuracy (using correct classification of 20 images in the test set)

A screenshot of a computer program

Description automatically generated

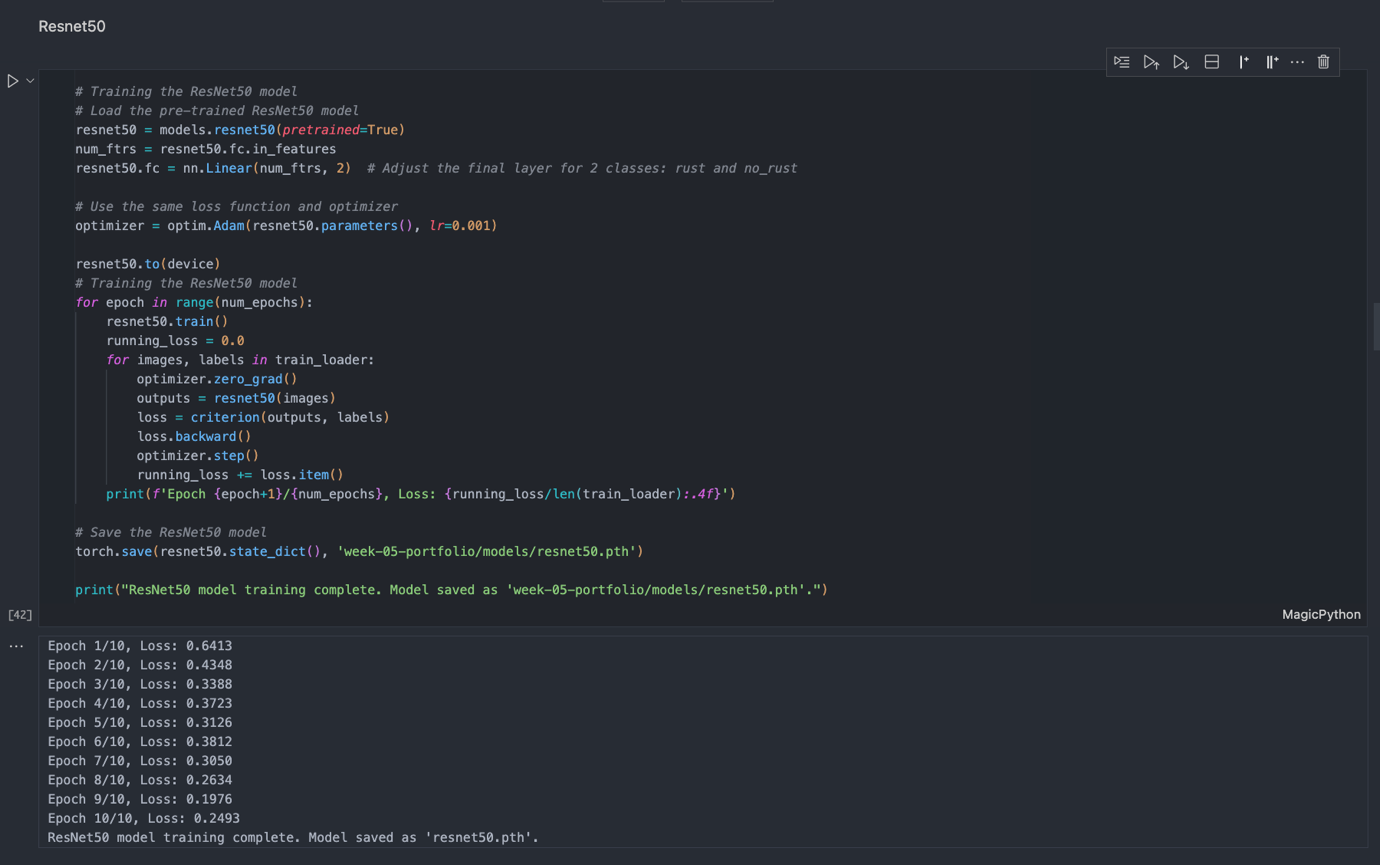
Result Tables:

|  |  |
| --- | --- |
| True Class | Predicted Class |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |

Accuracy: 100%

The result for **test outcome** containing **images** can be found at **cnn\_test** folder!

1. Now develop a more complex CNN, Restnet50 and train with the same dataset as in step 2 and test with Test dataset and measure the accuracy (using 20 images in the test set)



Result Tables:

|  |  |
| --- | --- |
| True Class | Predicted Class |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |

Accuracy: 95%

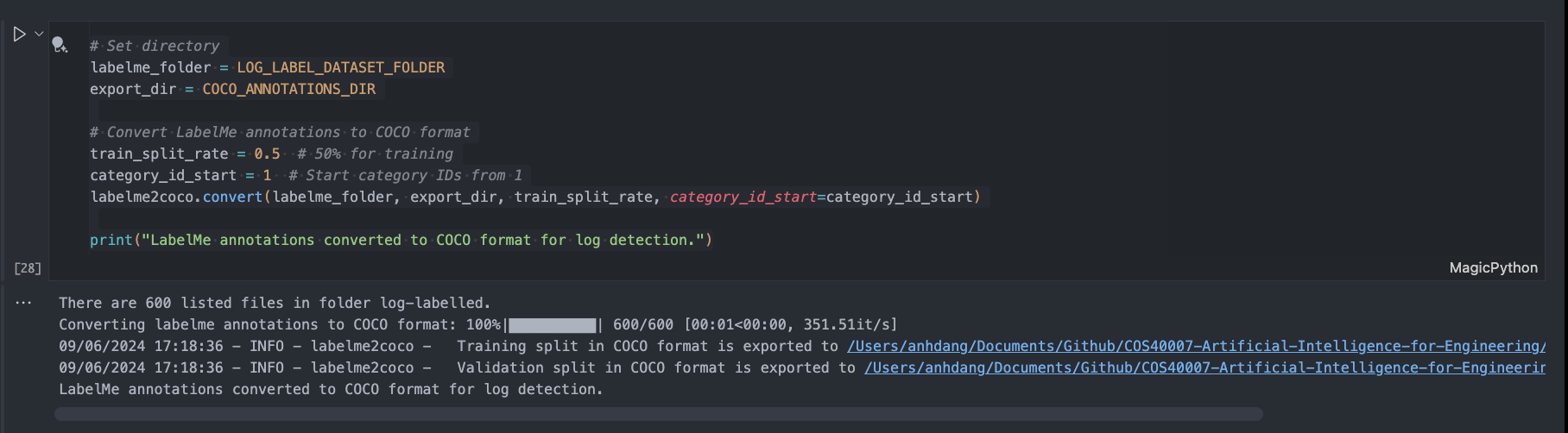
The result for **test outcome** containing **images** can be found at **resnet50\_test** folder!

# Task 2: Develop Mask RCNN for Detecting Log

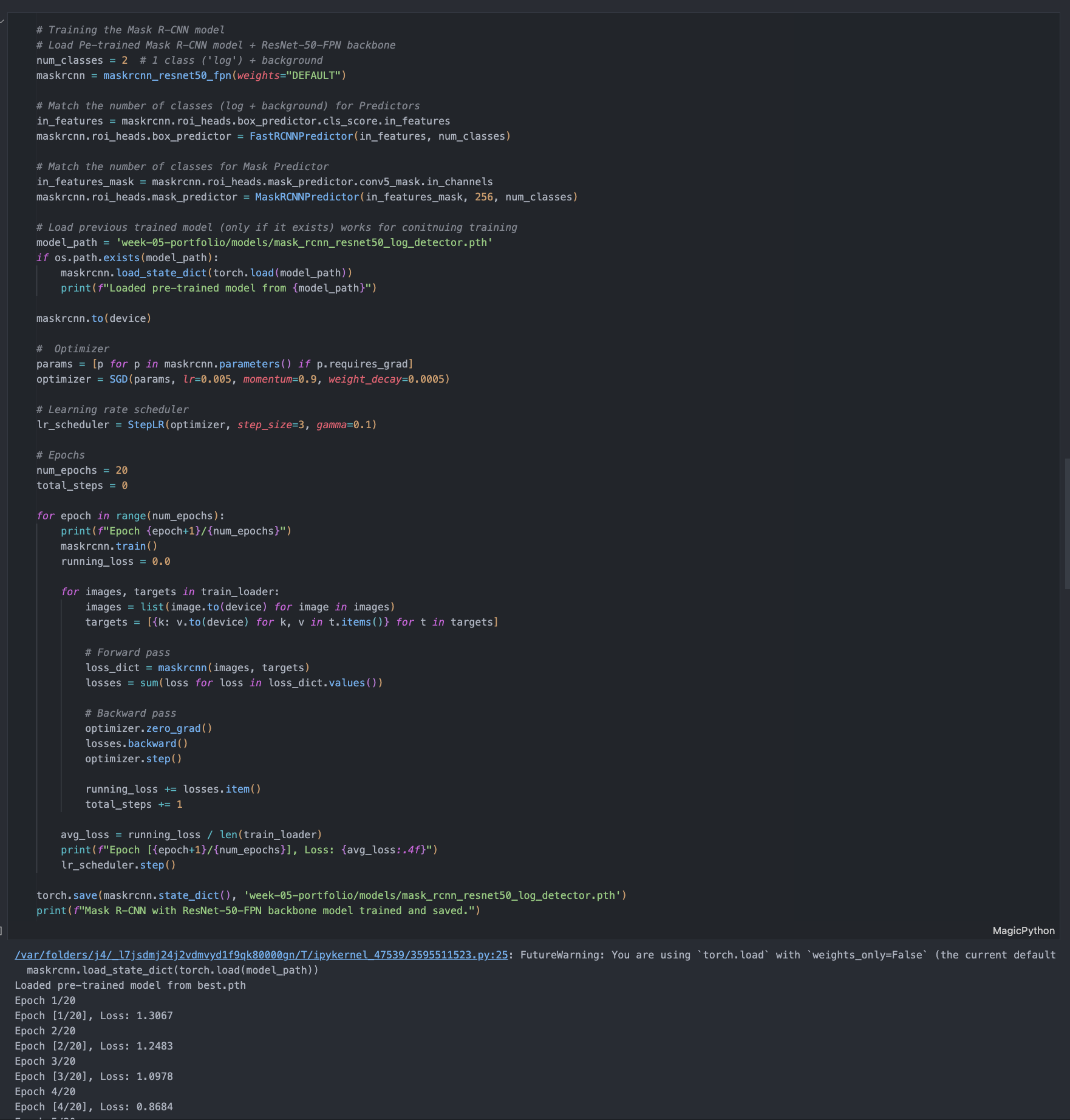
1. First randomly take out 10 images for testing (We call this as Test Set)

- For this task as I'm facing some hardware limitation on my Macbook, I have actually

split the training and testing in half straight through **labelme2coco.**



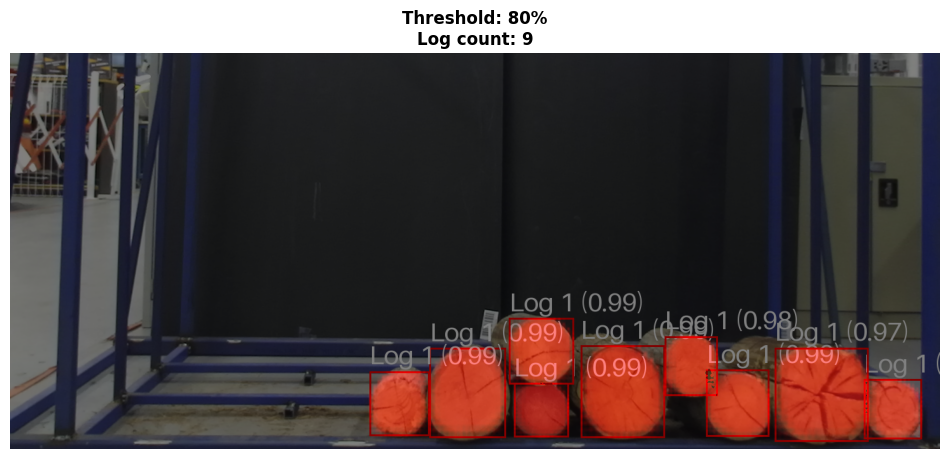
1. Develop a Mask RCNN model using the labelled log as training data (excludes test set).



Initialially I have trained the model around 17 epochs while having the train loss down to aproximately around 2.6 and then I further training the model for another 20 epochs and able to get the train loss down to around 0.6 which is a significantly improving from 2.6. This process took around 6-7 hours.

1. Test the model with Test set and generate images of detected log objects along with confidence score. For example, the test outcome of one image will look like similar to the following image. You will need to use OpenCV to produce such image
2. Write a python program that count number of detected logs in each output image (Log counting)

I have combined the answer for requirement 3 and 4 together and producing log counting and detected objects along with confidence score as well as segmentation which has been shown in portfolio requirements PDF as example. Additionally I have added threshold to filtered out bad or inaccurate results detected by the models. Below is an example of the output of the model.



The result for **test outcome** containing **images** can be found at **rcnn\_test** folder!

# Task 3: Extending Log Labelling to Another Class

The result this tasks you can find at **my\_coco\_annotations** and the json file required to submit is **dataset.json**

A screenshot of a computer

Description automatically generated