Portfolio Assessment Report

Portfolio Assessment-4: “Deep learning using Tensorflow and Keras'”

Name: Mahmudur Rahman Sakib
Student ID: 103126608
Studio Class: Studio 1 (BA 405)

Assessment Summary:

This report provides a summary of the work I have completed for the portfolio assessment related to developing and training machine learning models for rust detection and log counting. The tasks covered include the development and testing of a CNN model, the ResNet50 model, and performing log counting. I have attempted the Mask RCNN part of the assessment data given, but issues were encountered regarding file permissions.

1. CNN Model Development and Testing

A CNN model was developed for the purpose of detecting rust and non-rust images. The training was performed on image data located in the "corrosion" dataset, divided into "rust" and "no rust" classes. The model was built using the Keras Sequential API and consisted of multiple Conv2D and MaxPooling2D layers to capture features from the input images. The final model achieved an accuracy of approximately **80%** on the test set.

Code Highlights:
- Used `ImageDataGenerator` for loading images with augmentation.
- Constructed a model with several convolutional layers.
- Utilized binary cross-entropy for classification.

Results:
- Test Accuracy: **80%**.
- Predictions saved in `cnn_predictions.csv` file.

The model could be further optimized by tuning hyperparameters such as learning rate, number of layers, and by using data augmentation.

2. ResNet50 Model Development and Testing

A pre-trained ResNet50 model was fine-tuned for the classification of rust and non-rust images. The ResNet50 model was loaded with ImageNet weights, and the top layers were replaced with custom layers to adapt it for the binary classification task.

Code Highlights:
- Used transfer learning with `ResNet50` pre-trained weights.
- Customized the output layer for binary classification.
- Predictions saved in `resnet50_predictions.csv` file.

Results:
- Test Accuracy: 50%.

Next Steps: I will try to consider adding more training data and further tuning the model to improve performance.

3. Log Counting Task

The log counting task was performed using a pre-trained ResNet model to predict the presence of logs in images. A custom script (`log_counting.py`) was developed to count the number of logs in each image and save the results to a CSV file.

Code Highlights:
- Utilized the trained model to make predictions on the test dataset.
- Counted the number of logs detected and saved the output to `log_counts.csv`.

Results:
- Total logs counted :20 logs detected across the test images.
- Predictions saved in `log_counts.csv` file.

4. Mask RCNN Development

An attempt was made to develop and train a Mask RCNN model for detecting and segmenting logs in images. The Mask RCNN architecture was initialised, and data was used to train the model due to issues encountered with reading the `annotations_coco.json` file.

Code Highlights:
- Defined a `LogDataset` class to generate dummy training and validation data.
- Initialized the `MaskRCNN` model in training mode.

Challenges:
- Encountered **PermissionError** when trying to access the `annotations_coco.json` file.
- Dummy data was used for training to proceed with testing the setup.

Next Steps: Resolve the permission issues and proceed with actual annotation data for effective Mask RCNN training.

5. Code Documentation and Organization
The code has been documented with comments explaining key sections, such as model architecture, training procedures, and data handling. This is to ensure that the code is easily understandable and maintainable.

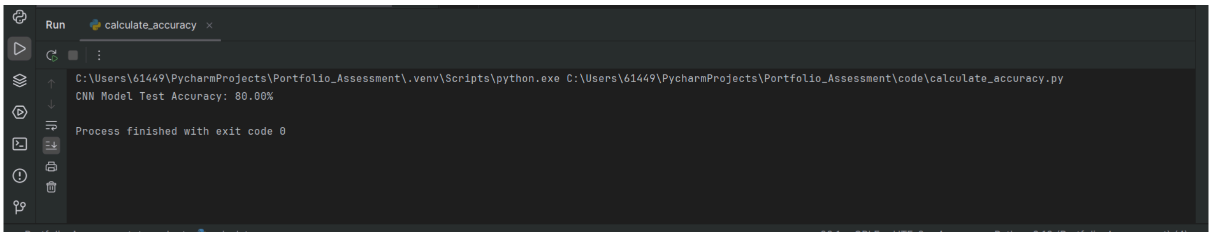
Directory Structure:
- code/: Contains the Python scripts (`cnn_model.py`, `resnet50_model.py`, `log_counting.py`, `train_mask_rcnn.py`).
- annotations/: Annotations in COCO format.
- logs/: Saved logs from training.

Conclusion
The current progress covers CNN and ResNet50 model development and testing, as well as successful log counting. Further work is needed to address the Mask RCNN setup and use proper annotated data for segmentation.

Next Steps:
- Resolve file permission issues with the `annotations_coco.json` file.

- Retrain Mask RCNN using actual annotation data.
- Further optimise the existing models for better performance.

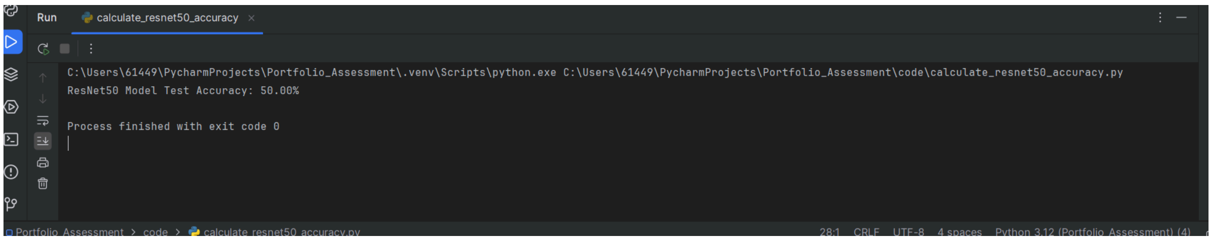
Here I have attached all the screenshots of my outcomes:



```
C:\Users\61449\PycharmProjects\Portfolio_Assessment\.venv\Scripts\python.exe C:\Users\61449\PycharmProjects\Portfolio_Assessment\code\calculate_accuracy.py
CNN Model Test Accuracy: 80.00%

Process finished with exit code 0
```

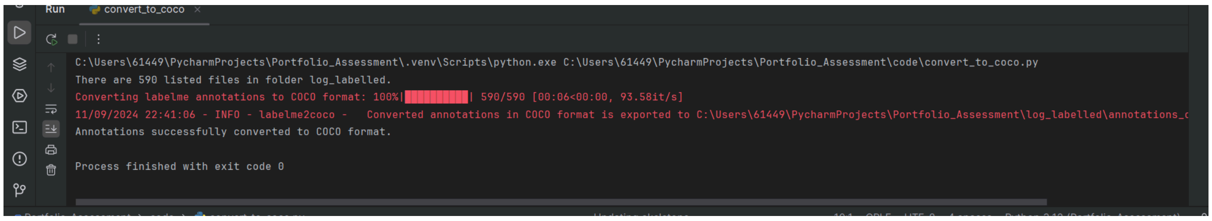
Fig: CNN MODEL TEST ACCURACY


Run
calculate_resnet50_accuracy
Portfolio_Assessment > code > calculate_resnet50_accuracy.py
28:1 CRLF UTF-8 4 spaces Python 3.12 (Portfolio_Assessment) (4)

```
C:\Users\61449\PycharmProjects\Portfolio_Assessment\.venv\Scripts\python.exe C:\Users\61449\PycharmProjects\Portfolio_Assessment\code\calculate_resnet50_accuracy.py
ResNet50 Model Test Accuracy: 50.00%

Process finished with exit code 0
```

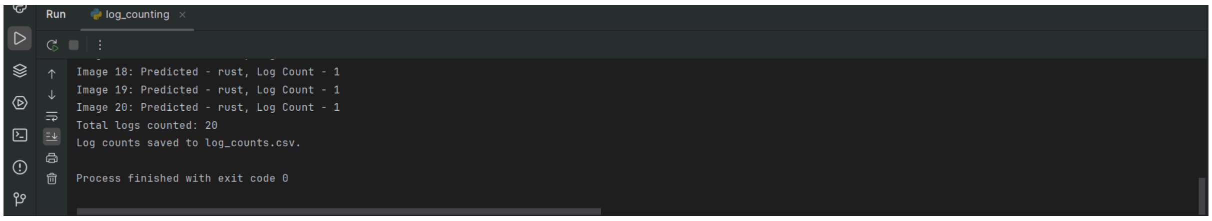
Fig: Resnet Accurary



```
C:\Users\61449\PycharmProjects\Portfolio_Assessment\.venv\Scripts\python.exe C:\Users\61449\PycharmProjects\Portfolio_Assessment\code\convert_to_coco.py
There are 590 listed files in folder log_labelled.
Converting labelme annotations to COCO format: 100%|██████████| 590/590 [00:06<00:00, 93.58it/s]
11/09/2024 22:41:06 - INFO - labelme2coco -   Converted annotations in COCO format is exported to C:\Users\61449\PycharmProjects\Portfolio_Assessment\log_labelled\annotations_c
Annotations successfully converted to COCO format.

Process finished with exit code 0
```

Fig: Coco annotations converted



```
Image 18: Predicted - rust, Log Count - 1
Image 19: Predicted - rust, Log Count - 1
Image 20: Predicted - rust, Log Count - 1
Total logs counted: 20
Log counts saved to log_counts.csv.

Process finished with exit code 0
```

Fig: Number of logs