*School of Science, Computing and Engineering Technologies*

A black and red square with white text

Description automatically generated

**COS40007**

**Artificial Intelligence for Engineering**

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

Studio 5 – Portfolio 5

**Students’ Name:** Duc Thinh Pham - 104169675

**Lecturer:** Dr. Trung Luu

**Date:** October 11th, 2024

Contents

[Abstract 3](#_Toc179752870)

[Task 1: Develop CNN and Resnet50 3](#_Toc179752871)

[Task 2: Develop Mask RCNN for Detecting Log 6](#_Toc179752872)

[Task 3: Extending Log Labelling to Another Class 8](#_Toc179752873)

[Appendix 9](#_Toc179752874)

# Abstract

This portfolio demonstrates the application of deep learning techniques for image classification and object detection using TensorFlow and Keras. The project comprises three primary tasks: developing and evaluating a Convolutional Neural Network (CNN) and ResNet50 model for classifying corrosion in structural assets, implementing a Mask R-CNN model for detecting wooden logs in images, and extending the labeling process to incorporate additional classes. The CNN models trained on a labeled corrosion dataset are evaluated for accuracy on a test set, comparing performance between a simple CNN and a more complex ResNet50 architecture. The Mask R-CNN model is trained on a labeled wooden log dataset to detect log objects, with a post-processing step to count detected logs in the test images. This portfolio also includes updated annotations to identify broken logs, highlighting the potential for extending detection capabilities to multi-class object recognition. The results are validated with accuracy metrics and visual outputs, demonstrating the effectiveness of these models in handling classification and detection tasks in engineering contexts.

# Task 1: Develop CNN and Resnet50

1. Start by selecting 10 images of rust and 10 images of no rust at random to create a Test Set, ensuring these images are excluded from the training dataset.

A screenshot of a computer program

Description automatically generated

1. Develop a simple CNN model similar to the MNIST classification architecture. Train this model using the corrosion dataset with two classes: "rust" and "no rust" (excluding the Test Set). After training and saving the model, evaluate its accuracy by testing it with the 20 images in the Test Set, measuring the correct classifications.

A screenshot of a computer program

Description automatically generated

A computer screen shot of a program

Description automatically generated

Result table:

|  |  |
| --- | --- |
| 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 | 0 |
| 1 | 0 |
| 1 | 0 |
| 1 | 0 |
| 1 | 0 |
| 1 | 0 |
| 1 | 0 |
| 1 | 0 |
| 1 | 0 |
| 1 | 0 |

Accuracy of the Simple CNN model on the test set: 55.00%

You can find the test results with images in the **cnn\_test** folder.

1. Next, implement a more advanced CNN model using the ResNet50 architecture, training it with the same dataset as in step 2. Test this model with the same Test Set to measure its accuracy.

A screenshot of a computer program

Description automatically generated

Result table:

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

Accuracy of the ResNet50 model on the test set: 50.00%

You can find the test results with images in the **restnet50\_test** folder.

# Task 2: Develop Mask RCNN for Detecting Log

1. Begin by randomly selecting 10 images to form a Test Set.

Due to hardware limitations on my local machine, I split the training and testing datasets evenly using **labelme2coco** for this task.

A screenshot of a computer

Description automatically generated

1. Create a Mask R-CNN model and train it using the labeled log images, excluding those in the Test Set.

A screenshot of a computer program

Description automatically generated

1. Evaluate the model on the Test Set, generating images that display detected log objects along with their confidence scores.
2. Write a python program that count number of detected logs in each output image (Log counting)

For question 3 and 4, generating both log counts and detected objects with confidence scores and segmentation, as illustrated in the portfolio requirements PDF. Furthermore, I included a threshold to filter out inaccurate or low-confidence detections.

A stack of logs with numbers and text

Description automatically generated with medium confidence

# Task 3: Extending Log Labelling to Another Class

I update the labels for those 10 images using labelme, replacing the label of any broken logs with "detected\_log"? Then, save the updated labels for all 10 images.

A stack of logs with green circles

Description automatically generated

# Appendix

My GitHub repository: [thinhpham1807/COS40007---Artificial-Intelligence-for-Engineering (github.com)](https://github.com/thinhpham1807/COS40007---Artificial-Intelligence-for-Engineering)

Data files: <https://drive.google.com/drive/folders/1fyCmJYQNspCvwO_vfwarsQs6x0MKW4cP?usp=sharing>