Configuration Manual

MSc Research Project
Data Analytics

Rakhi Ashok Sonkusare
Student ID: x20122314

School of Computing
National College of Ireland

Supervisor: Aaloka Anant

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Rakhi Ashok Sonkusare
Student ID:	x20122314
Programme:	Data Analytics
Year:	2021
Module:	MSc Research Project
Supervisor:	Aaloka Anant
Submission Due Date:	16/12/2021
Project Title:	Configuration Manual
Word Count:	907
Page Count:	6

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	16th December 2021

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Rakhi Ashok Sonkusare
x20122314

1 Introduction

The below configuration manual gives an overview of all the required hardware and software configurations for the code run and provides step-by-step guidance for the required dataset upload and relevant code changes to execute the models and obtain results.

2 System Specifications

2.1 Hardware Configurations

The code implementation is done on Windows 10 with 64-bit operating system and an Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz 2.11 GHz processor and 8 GB of RAM. The Graphics Card (GPU) is NVIDIA GEFORCE RTX 2060.

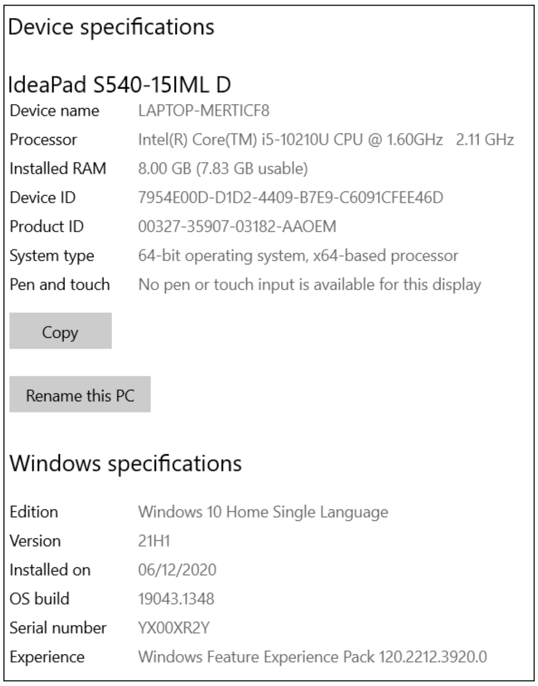

Figure 1: Hardware Configurations

2.2 Software Configuration

We have used Python 3.7 version for writing the code for this research. The code was implemented and run using Jupyter notebook on the Anaconda Navigator. The execution of deep learning models require a dedicated GPU for faster execution and thus, we used the Google Colab IDE which is a cloud-based open-source publicly accessible platform that enables the users to execute Python code and provides free GPU and TPU. It also allows saving the outputs of the code for easy sharing.

2.2.1 Anaconda Navigator

1. The Anaconda Navigator was installed from the Anaconda Individual Edition through the link: `https://www.anaconda.com/products/individual`
2. After the installation is completed, we need to set up a new environment through the Environment tab on the Navigator for TensorFlow and load the required packages.
3. To open the Jupyter Notebook, go to the Navigator and click on the 4th tab as in Figure 2 to Launch.

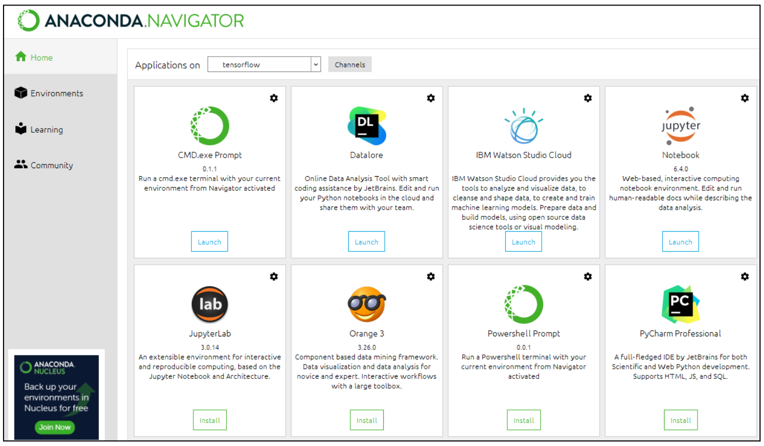

Figure 2: Anaconda Navigator

2.2.2 Google Colab IDE

1. Golab Colab IDE is publicly accessible through the link: `https://colab.research.google.com/?utm_source=scs-index`
2. To open or upload a Python notebook, you first need to log in to Colab with a valid Google account. For this research, we created a new account to use 16GB of the space for dataset upload and execution of Python code.
3. The Python notebook can be uploaded through the File tab as shown in Figure 3.

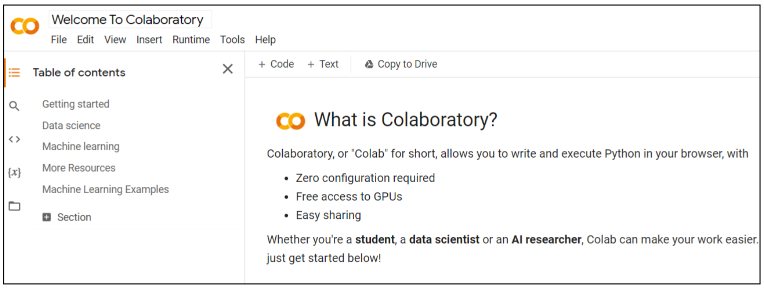

Figure 3: Google Colaboratory

3 Packages and Libraries Used for Code Implementation

The code implementation depends upon the installation of appropriate packages that are available in the Anaconda Navigator. The below packages have been installed using the pip command for this research:

- keras
- tensorFlow
- numpy
- pandas
- shutil
- matplotlib
- sklearn
- skimmage
- seaborn

4 Dataset Description

1. The dataset is taken from the 8th FGCV Plant Pathology competition available through the link: `https://www.kaggle.com/c/plant-pathology-2021-fgvc8/data`

2. The dataset is downloaded in Zip format and can be unzipped and used in the local environment. The downloaded folder consists of two subfolders: train_images and test_images, and a train.csv file with labels assigned to each training image. The dataset can be uploaded on Google drive to run on Google Colab. The code is updated to point the dataset which is explained in the next section 5.

5 Steps for Code Execution

For running the Python code on Google Colab IDE, we need to follow two of the pre-requisites:

1. The dataset needs to be uploaded on Google Drive and then mounted on drive using the below command.

```
from google.colab import drive
drive.mount('/content/drive')
```

Figure 4: Mounting drive on Google Colab

2. Once the drive is mounted, the code should be updated with the directory path to be able to read the image files and .csv file with labels.

```
#IMAGE PATH & DATAFRAME:

TRAIN_PATH = "/content/drive/MyDrive/plant-pathology-2021-fgvc8/train_images"
train_df = pd.read_csv("/content/drive/MyDrive/plant-pathology-2021-fgvc8/train.csv")
```

```
[ ] from pathlib import Path
    dataset_path = Path('/content/drive/MyDrive/plant-pathology-2021-fgvc8')
```

```
[ ] train_df = pd.read_csv(dataset_path/'train.csv')
    train_df.head()
```

	image	labels
0	800113bb65efe69e.jpg	healthy
1	8002cb321f8bfcdf.jpg	scab frog_eye_leaf_spot complex
2	80070f7fb5e2ccaa.jpg	scab
3	80077517781fb94f.jpg	scab
4	800cbf0ff87721f8.jpg	complex

Figure 5: Changing the file directory path

5.1 Baseline EfficientNet-B3 and ResNet152 CNN models Training

- Once the dataset is available, we use the ImageDataGenerator for image pre-processing like Resizing the image sizes to 512 * 512 and performing data augmentation methods like Rotation, Horizontal Flipping, Shifting, Zooming and Shearing transformations on 32 images.
- Splitting the train images into training and validation in 80:20 split ratio.
- The models are defined and compiled using Adam Optimizer and trained with a batch size of 32 and number of epochs as 8.

- The models are rebuilt to perform fine-tuning of hyperparameters like learning rate and the number of units in the Dense Layer. The optimal values obtained for EfficientNet-B3 are 0.001 for learning rate and 768 number of units, whereas, for ResNet152, the learning rate is 0.0001 and 512 number of units in the Dense layer. The models are trained with a batch size of 32 and number of epochs as 8.

```
tuner = kt.RandomSearch(
    modelEfficientNetB3,
    objective='val_loss',
    max_trials=12,
    directory  = LOG_DIR)

[ ] stop_early = tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=5)

[ ] tuner.search(x_train, y_train, epochs=12, batch_size = 64, validation_data=(x_val, y_val), callbacks=[stop_early])

Trial 4 Complete [00h 03m 22s]
val_loss: 2.3076694011688232

Best val_loss So Far: 2.3076694011688232
Total elapsed time: 00h 10m 39s
INFO:tensorflow:Oracle triggered exit

best_hps=tuner.get_best_hyperparameters(num_trials=1)[0]
print(f"""
The hyperparameter search is complete. The optimal number of units in the first densely-connected
layer is {best_hps.get('units')} and the optimal learning rate for the optimizer
is {best_hps.get('learning_rate')}.
""")

The hyperparameter search is complete. The optimal number of units in the first densely-connected
layer is 768 and the optimal learning rate for the optimizer
is 0.001.
```

Figure 6: Hyper Parameter tuning for EfficientNet-B3 using Random Search Optimization

- Models are saved using checkpoints and reloaded to make predictions on unlabeled testing data.
- Accuracy, Classification report for Precision, Recall and F1-score, Confusion Matrix is calculated and plots are displayed.

5.2 AppleCaps - Capsule Network Model Training

- The dataset is mounted on drive and the image files are read. Pre-processing is done on the input train images like Resizing to 224 * 224 dimension, Gaussian Blurring, Augmentation techniques like Rotation, Horizontal Flipping, Shifting, Zooming and Shearing transformations on 1000 images. RGB images are converted to grayscale images to highlight the features on the leaves.
- Defined two convolutional layers, reshaping and squashing function using several classes.
- Decoder is built to calculate reconstruction and margin loss.
- Splitting the data into train and validation set in 80:20 split ratio.
- The model is trained with a batch size of 32 and number of epochs as 8.

```
WARNING:tensorflow:From <ipython-input-89-75bced9b14b6>:11: checkpoint_exists (from tensorflow.python.training.
Instructions for updating:
Use standard file APIs to check for files with this prefix.
Epoch: 1  accuracy: 25.5852%  loss: 2.278965 val_accuracy: 25.4863% val_loss: 2.609553(improved)
Epoch: 2  accuracy: 41.7654%  loss: 2.208561 val_accuracy: 30.6523% val_loss: 2.109549(improved)
Epoch: 3  accuracy: 58.1542%  loss: 2.156423 val_accuracy: 51.7345% val_loss: 2.409537
Epoch: 4  accuracy: 65.4857%  loss: 2.129485 val_accuracy: 44.6589% val_loss: 2.294850(improved)
Epoch: 5  accuracy: 80.2657%  loss: 2.109434 val_accuracy: 60.6752% val_loss: 2.185632(improved)
Epoch: 6  accuracy: 84.9156%  loss: 2.099431 val_accuracy: 75.2546% val_loss: 2.159951(improved)
Epoch: 7  accuracy: 86.3745%  loss: 2.079331 val_accuracy: 87.6845% val_loss: 2.062598(improved)
Epoch: 8  accuracy: 87.0648%  loss: 2.062510 val_accuracy: 88.8560% val_loss: 2.042510(improved)
```

Figure 7: Output of AppleCaps model

- Accuracy, Classification report for Precision, Recall and F1-score, Confusion Matrix is calculated and the plots are displayed.

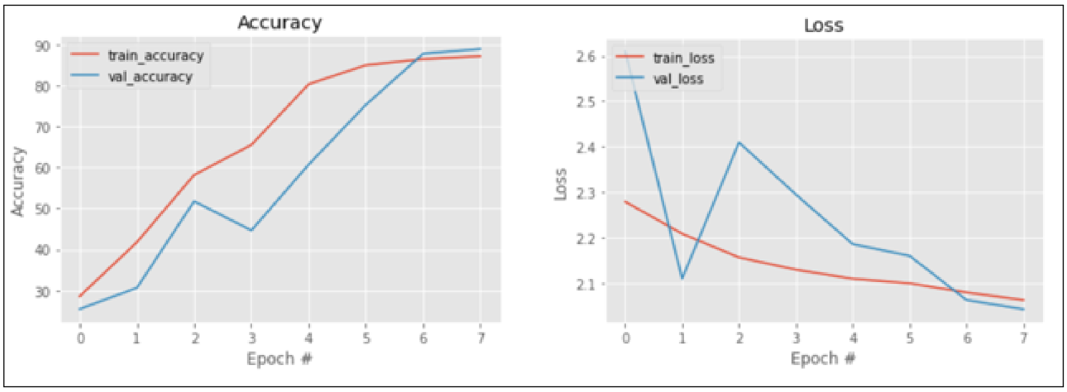

Figure 8: Accuracy and Loss plots of AppleCaps model

6 Prediction Result for AppleCaps model

The trained AppleCaps model is saved and reloaded to make predictions on an unlabeled dataset.

Figure 9: Prediction result of AppleCaps model