National College of Ireland

Configuration Manual

MSc Research Project
MSC in Data Analytics

Aiswarian Sebastian Varghese
Student ID: 20207816

School of Computing
National College of Ireland

Supervisor: Mr. Hicham Rifai

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Aiswarian Sebastian Varghese
Student ID:	20207816
Programme:	MSC in Data Analytics
Year:	2021-'22
Module:	MSc Research Project
Supervisor:	Mr. Hicham Rifai
Submission Due Date:	15/08/2022
Project Title:	Configuration Manual
Word Count:	888
Page Count:	8

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:	15th August 2022

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

Aiswarian Sebastian Varghese
20207816

1 Introduction

The configuration manual is completed as per the as per the requirement of the National College of Ireland for implementing final year research Project. The following configuration manual specifies all the software and hardware tools that are utilised for optimising Traffic sign Classification system..

2 System Requirements

The software and Hardware requirements that are utilised for implementation of the presented work is discussed in this section.

2.1 Hardware Specification

The Hardware requirement is the minimum system requirements like RAM and OS used for running the project. Processor: Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz 2.90 GHz Random Access Memory: 16.0 GB Storage: 256 ssd/ 500 GB HDD Operating System: 64-bit operating system, x64-based Operating System

2.2 Software Specification

The programming tools and development environments used in the research work implementation is explained below.

- Jupyter Notebook
- Google Colaboratory
- Python Version 3
- Overleaf

3 Environment Configuration

3.1 Anaconda Navigator

The presented CNN model was built and run in both jupyter notebook and in google colaboratory. For the jupyter notebook requirement Anaconda navigator and the latest python version was installed.

Anaconda navigat was installed from https://www.anaconda.com/distribution/#download-section and latest python version "python 3.7" as also installed. The anaconda interface was run in default settings for implementing the project.

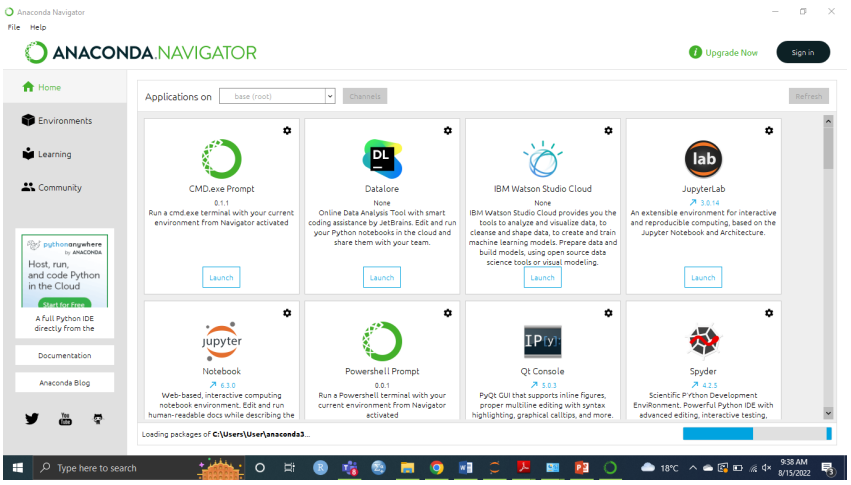


Figure 1: Anaconda Navigator Interface

3.2 Jupyter Notebook

The work was built and run in the jupyter notebook that was available in the anaconda interface. jupyter notebook is an integrated development enviornment to creating and sharing computatinal documents.

Figure 2: Jupyter Notebook

3.3 Google Colaboratory

Image data takes more processing time when we run it on jupyter notebook so to overcome this situation we are using Google colab. its is an open source platform or a free jupyter notebook env which completely runs on cloud.

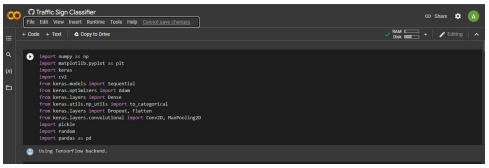


Figure 3: google colaboratory

4 Dataset

The dataset for the Traffic sign Classification system is taken from the Kaggle open source repository. the dataset considered for building the model is the German Traffic sign Recognition Benchmarks. the dataset contains more than 45000 samples of traffic sign images with 43 different traffic sign classes.

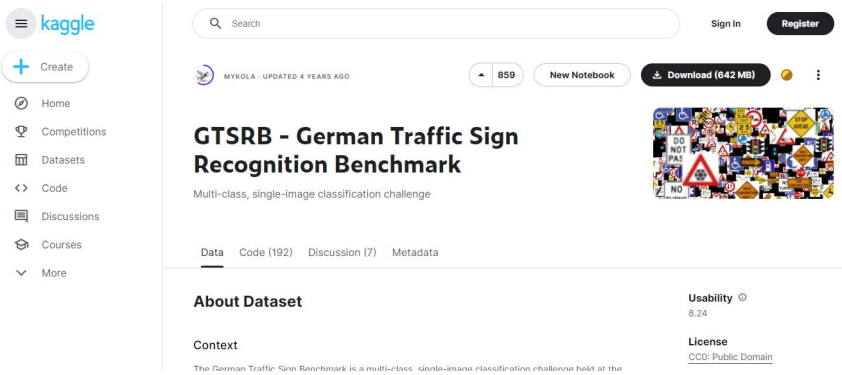


Figure 4: Kaggle Repository

The dataset contains 43 classes of traffic signs images. Each class contains different frequencies of the samples for better training of the model. The dataset itself is splitted to training, Testing and validation set which is useful for evaluating the performance during model implementation. this dataset can be directly downloaded from the repository or it can be cloned for google colaboratory. The dataset is explained in 5

5 Implementation

The presented work is implemented using the python programming language. The proposed work is the traffic sign classification and deals with the images all the supporting library packages are installed for the implementation. these packages are listed below.

As mentioned earlier, the programming language used in Python version 3. Hence, the libraries associated with Python is used in the implementation of the model for analysis of deception by the application of CNN. The libraries used are mentioned below.

- NumPy
- Keras
- CV2

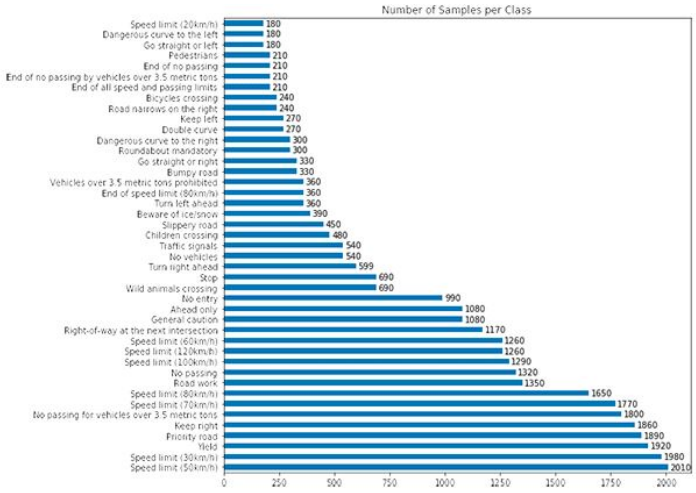


Figure 5: Dataset

- Pickle
- matplotlib
- Plotly
- TensorFlow

```
import numpy as np
import matplotlib.pyplot as plt
import keras
import cv2
from keras.models import Sequential
from keras.optimizers import Adam
from keras.layers import Dense
from keras.utils.np_utils import to_categorical
from keras.layers import Dropout, Flatten
from keras.layers.convolutional import Conv2D, MaxPooling2D
import pickle
import random
import pandas as pd

Using TensorFlow backend.
```

Figure 6: Importing Necessary packages

5.1 Preprocessing of Data

The preprocessing of the data is an important phase in classification of the traffic signs. As the work deals with images certain preprocessing should be done in order to improve the computational speed and information extraction from image (Saritha and Kumar; October 2020). the major pre processing steps are Conversion of images to gray scale, Histogram Equalization and normalization of images.

The conversion of the data points into grayscale is because the RGB data is very difficult to compute also the images are normalized to standard pixel rate for better processing.

```
$ workon traffic_signs
$ pip install opencv-contrib-python
$ pip install numpy
$ pip install scikit-learn
$ pip install scikit-image
$ pip install imutils
$ pip install matplotlib
$ pip install tensorflow==2.0.0 # or tensorflow-gpu
```

Figure 7: Importing Necessary Packages

Data Preprocessing

```
[ ]  plt.imshow(X_train[1000])
     plt.axis('off')
     print(X_train[1000].shape)
     print(y_train[1000])
```

Figure 8: Data Preprocessing

```
def grayscale(img):
    image = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    plt.axis('off')
    return image

[ ]  img = grayscale(X_train[1000])
     plt.imshow(img, cmap = 'gray')
     print(img.shape)
```

Figure 9: Gray Scale Conversion

5.2 Data Labelling

The entire dataset consists of 43 different classes and all these classes are labelled. the labelling of the data is done for the accurate recognition of the traffic sign and it implies what traffic sign is predicted.

```
def preprocessing(img):
    img = grayscale(img)
    img = equalize(img)
    img = img/255
    return img
```

Figure 10: Data Labelling

5.3 Model Implementation

The traffic sign classification model is implemented using the deep convolutional neural network. the GTSRB dataset is is preproceesed and the images are passed to the CNN network for the classification and recognition of the traffic signs (Muhammad Zain Amin; October 2019).

- Convolutional Neural Network model: we have implemented a sequential convolutional neural network model for the presented work the developed CNN model consists of four convolutional layer, two max pooling layer, flatten and dense layers and four fully connected layers.

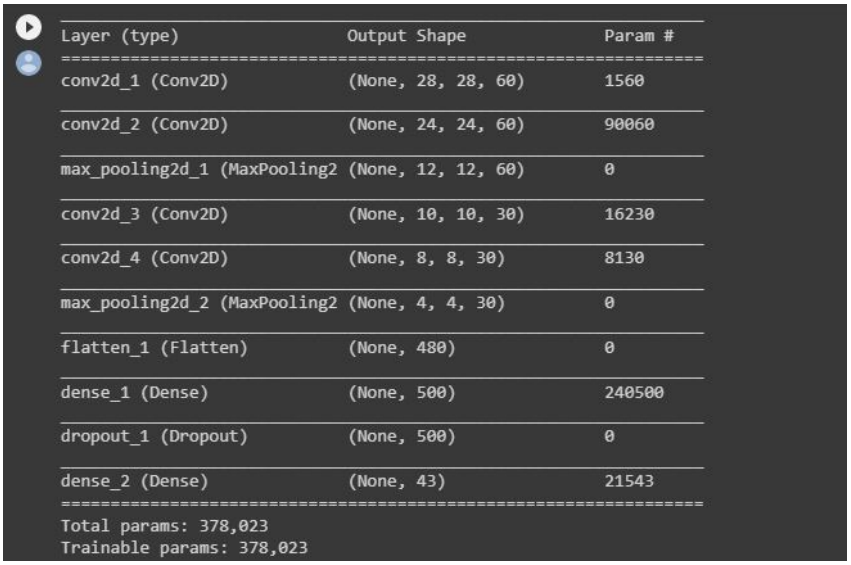
```
def neural_model():
    model = Sequential()
    model.add(Conv2D(60, (5, 5), input_shape = (32, 32, 1), activation = 'relu'))
    model.add(Conv2D(60, (5, 5), input_shape = (32, 32, 1), activation = 'relu'))
    model.add(MaxPooling2D(pool_size = (2,2)))

    model.add(Conv2D(30, (3, 3), activation = 'relu'))
    model.add(Conv2D(30, (3, 3), activation = 'relu'))
    model.add(MaxPooling2D(pool_size = (2, 2)))

    #model.add(Dropout(0.5))

    model.add(Flatten())
    model.add(Dense(500, activation = 'relu'))
    model.add(Dropout(0.5))
    model.add(Dense(num_classes, activation = 'softmax'))
    model.compile(Adam(lr = 0.001), loss = 'categorical_crossentropy', metrics = ['accuracy'])
    return model
```

Figure 11: Convolutional Neural Network

After developing the first phase of the model the hyperparameter tuning is done for improving the performance of the model. the model performed ell in 50 and 10 epochs and also the optimizer used was the Adam optimizer and the loss was claculated using categorical cross entropy.

6 Evaluation

The performance of the model is evaluated by the performance metrics and the presented model is compared with other existing models in the domain (Islam and Raj; April 2017). the performance metrics such as precision, accuracy, recall and f1 score is evaluated.



Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 28, 28, 60)	1560
conv2d_2 (Conv2D)	(None, 24, 24, 60)	90060
max_pooling2d_1 (MaxPooling2	(None, 12, 12, 60)	0
conv2d_3 (Conv2D)	(None, 10, 10, 30)	16230
conv2d_4 (Conv2D)	(None, 8, 8, 30)	8130
max_pooling2d_2 (MaxPooling2	(None, 4, 4, 30)	0
flatten_1 (Flatten)	(None, 480)	0
dense_1 (Dense)	(None, 500)	240500
dropout_1 (Dropout)	(None, 500)	0
dense_2 (Dense)	(None, 43)	21543

Total params: 378,023
Trainable params: 378,023

Figure 12: Summary Of the model

7 Overleaf for Documentation

For documenting the presented work overleaf is selected. all the alignment and standardisation is done by the conduct od national college of Ireland.

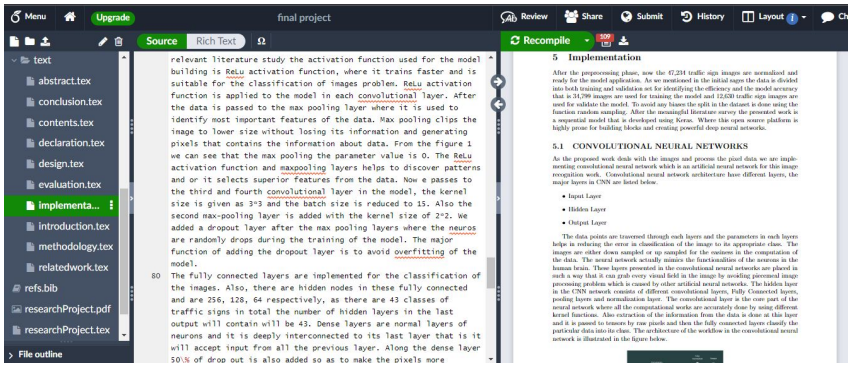


Figure 13: latex interface

References

Islam, K. T. and Raj, R. G. (April 2017). Real-time (vision-based) road sign recognition using an artificial neural network, *www.mdpi.com/journal/sensors* (28406471).

Muhammad Zain Amin, N. N. (October 2019). Convolutional neural network: Text classification model for open domain question answering system, *Journal of Advanced Research in Applied Sciences and Engineering Technology* .

Saritha and Kumar, D. A. (October 2020). The role of artificial intelligence in automatic traffic light detection system, *International Journal of Scientific Technology research* **9**(10): 4006 – 4015.