Configuration Manual

MSc Research Project
Data Analytics

Vikalp Singh Tomar
Student ID: x20239441

School of Computing
National College of Ireland

Supervisor: Dr. Giovani Estrada

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Vikalp Singh Tomar
Student ID:	X20239441
Programme:	M.Sc. Data Analytics **Year:** 2022
Module:	Research Project
Lecturer:	Dr. Giovani Estrada
Submission Due Date:	15/08/2022
Project Title:	Configuration Manual
Word Count:	10

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: Vikalp Singh Tomar

Date: ……………………………………………………………………………………………………

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

Vikalp Singh Tomar
X20239441

1 Introduction

The document contains detailed information on the tool and softwares that aid in the construction of the project from start to finish. This configuration manual document is included with the research project report, allowing for a better understanding of the study. As a result, all technical information essential for project completion that cannot be shared with the report is explained here.

2 Hardware and Software requirement.

To study the activation function and architecture of autoencoder neural network we need to build and run the neural network also the image classification problem that we choose needed to be run. So the hardware and the software that are used to perform the task are describe below.

2.1 Software used

Table 1: Software used

Tools used for programming	Anaconda navigator,Jupyter Notebook, Google Colab
Tools used to build the report	MS. Excel, MS. PowerPoint and MS. Word.
Programing Language used	Python.
Data storage	Google Drive, GitHub, Local system

2.2 Hardware required

Table 2: Hardware used

System	**Specification**
Operating System	Windows 10 pro
Processor	Intel core i5-7th Gen
RAM	8 GB
System type	64-bit OS, x64-based processor
Graphic card	NVidia GeForce

3 Software installation

3.1 Process to install Anaconda navigator and Jupyter Notebook On windows.

1) Visit the Anaconda downloads page[1]. Go to the following link: Anaconda.com/downloads

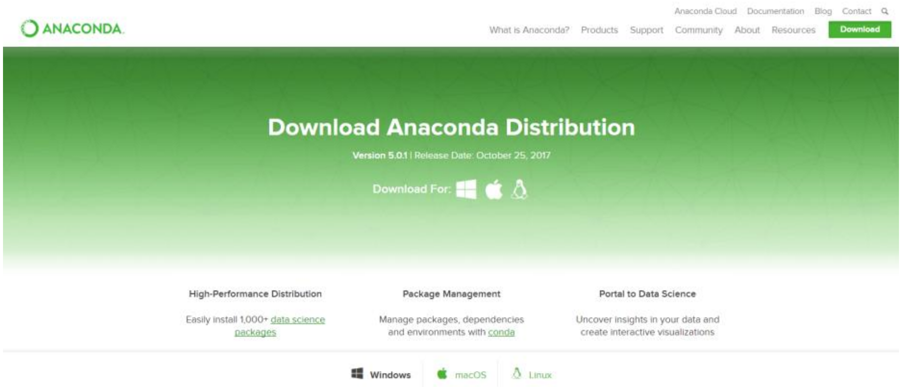


Figure 1: The Anaconda Downloads Page will look something like this

2) Select Windows. Select Windows where the three operating systems are listed shown in figure 2 below.

Figure 2: Select window option

3) Then the .Exe file gets downloaded.
4) After downloading the .exe file, we need to install the anaconda in the system. To do this we follow the below step

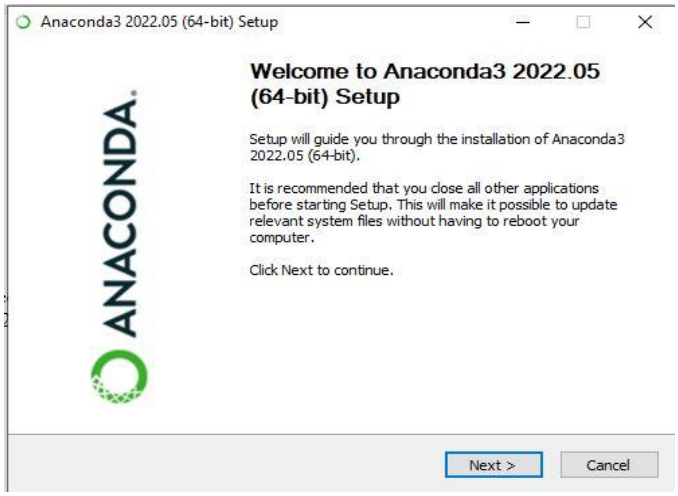


Figure 3: Installation window

[1] https://problemsolvingwithpython.com/01-Orientation/01.03-Installing-Anaconda-on-Windows/

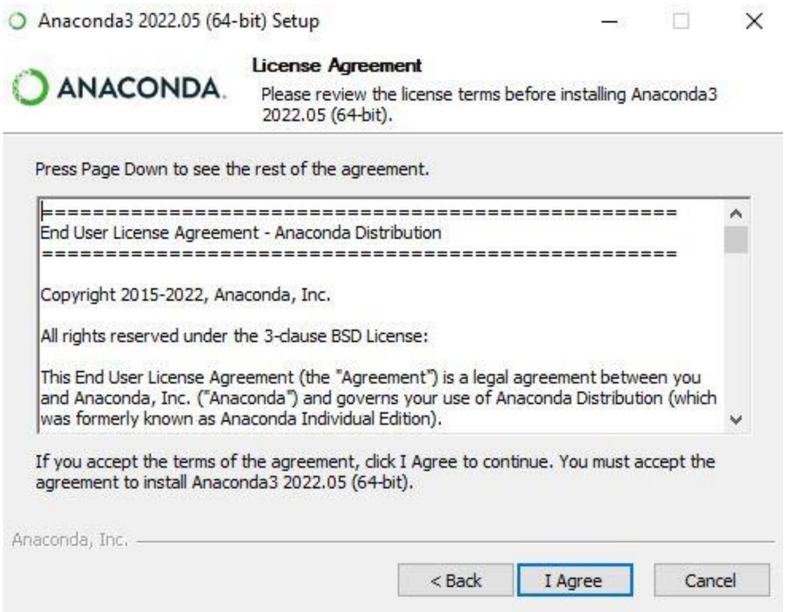


Figure 4: License agreement window

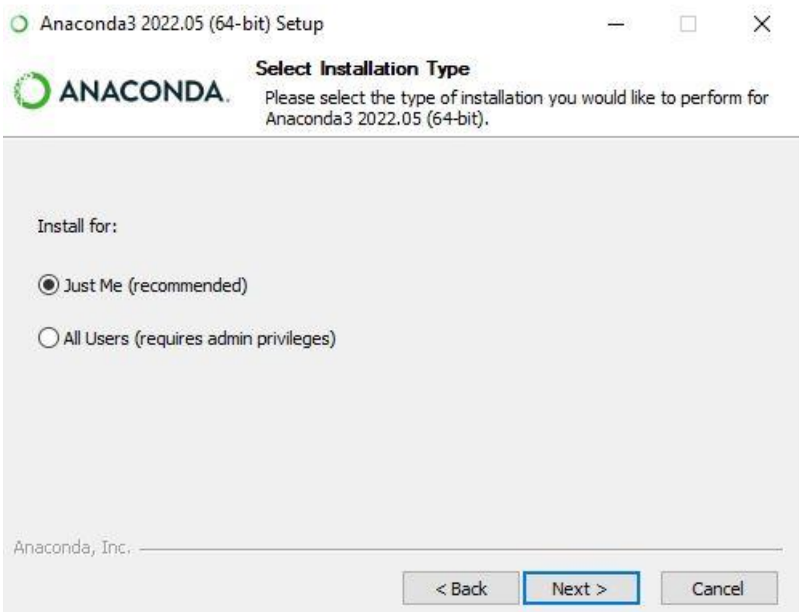


Figure 5: Selection of installing type window

5) After selecting the user in the above figure press next and wait for the complete installation of anaconda in your system.
6) When the installation complete, launch the application from start menu then one should see something similar to the following figure. JupyterLab and Jupyter Notebook are installed by default.

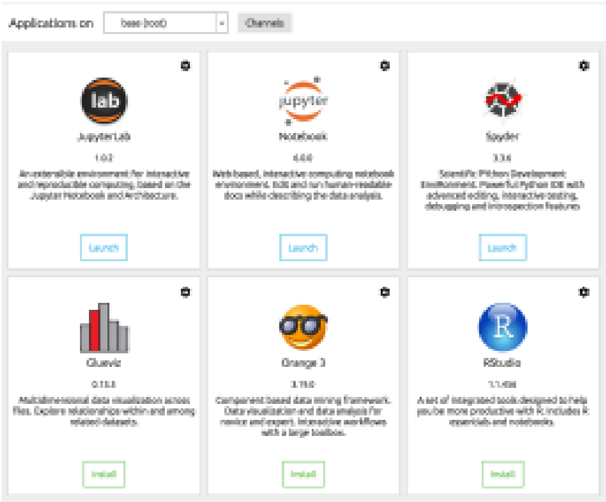


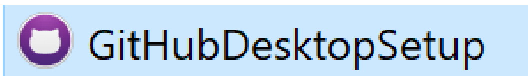
Figure 6: Anaconda interface

3.2 GitHub desktop installation

i. Visit this link https://desktop.github.com/ and Click Download for Windows[2].

Figure 6: GitHub download page

ii. Once the setup file download, we need to install the setup.



iii. After completion of GitHub installation. The application launch. And need to make the repository where we can save the all file and share it with through any one

3.3 Google colaboratory

For more programing part google colab has been used. Because of its benefits such as, free GPU, Store Notebooks on Google Drive. It can also combine with Github and local memory, which comes in useful while using it. Also no need to install it on the system and the programming notebook can be saved directly to.

[2] https://www.techrepublic.com/article/how-to-install-github-desktop/

To start a new notebook on Colab, go to https://colab.research.google.com/, and it will automatically show your previous notebooks and give you the opportunity to start a new one[3].

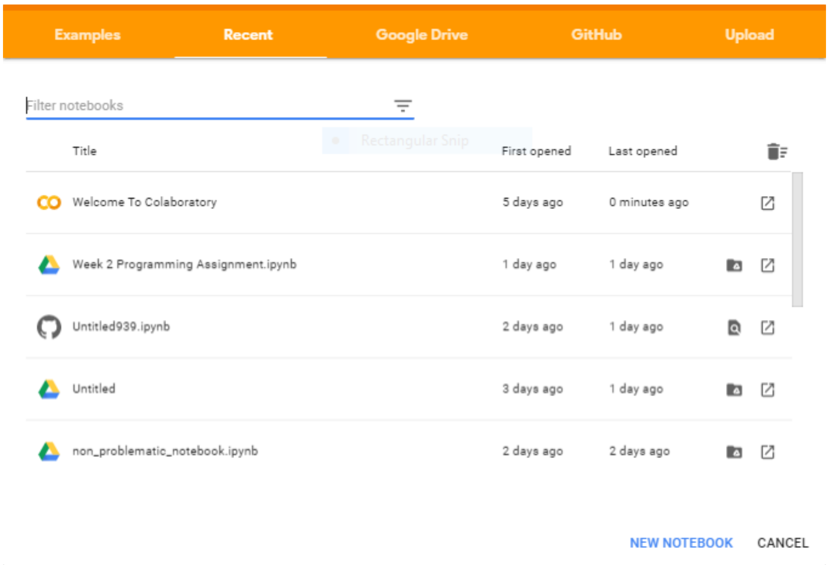


Figure 6: google colab interface to for new working notebook

The most significant advantage of Colab is that it offers free GPU and TPU support. Runtime > Change runtime type allows you to simply pick GPU or TPU for your program. Which help to increase the speed of the runtime.

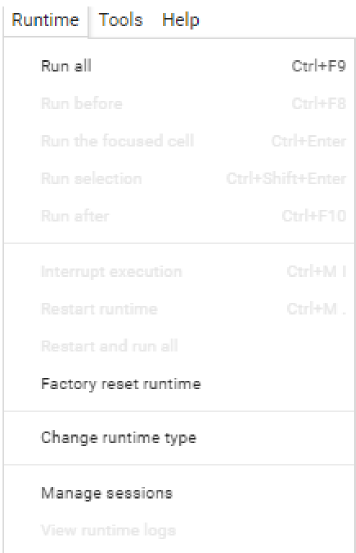


Figure 7: Changing runtime

3.4 Microsoft Word, Microsoft PowerPoint, Microsoft excel

Ms. Word, Ms. PowerPoint, and Ms. Excel are all employed to construct the report and assist in the creation of graphs as well as the presentation of the research project.All these software are handy to use and easy to understand which help to do the writing part of the research.

3 https://www.kdnuggets.com/2020/06/google-colab-deep-learning.html

Figure 7: Microsoft tool used for report building

4 Python Libraries used

For the deep learing task and aslo for the visualization task different libraries are used in python that are showed in figure 8 below.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, UpSampling2D
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import os, ast
import numpy as np
import pandas as pd

from matplotlib import pyplot as plt
import matplotlib.patches as patches
import seaborn as sns
from keras import keras
from tensorflow.keras.utils import img_to_array,array_to_img, load_img
from PIL import Image, ImageChops
from sklearn.neighbors import KernelDensity
import random
```

Figure 8: Python libraries used

Table 3: Python Libraries version

Libraries	Version
Sklearn	1.0.2
Pandas	1.41
Matplotlib	3.5.1
Cv2	4.6.0
Tensorflow	2.8.0
Numpy	1.12.2
Plotly	5.9.0
Seaborn	0.11.2

5 Data Understanding and Pre-processing Step

The data understanding plays an important role in the model building part (David et al. 2020), So it's important to get familiar with the data very well. To read the pandas libraries come into play As shown in below figure.

```
#Read Data
#Read the train.csv data:
train_df = pd.read_csv(f'{DIR_INPUT}/train.csv')
sample_sub_df = pd.read_csv(f'{DIR_INPUT}/sample_submission.csv')
train_df.head()
```

	image_id	width	height	bbox	source
0	b6ab77fd7	1024	1024	[834.0, 222.0, 56.0, 36.0]	usask_1
1	b6ab77fd7	1024	1024	[226.0, 548.0, 130.0, 58.0]	usask_1
2	b6ab77fd7	1024	1024	[377.0, 504.0, 74.0, 160.0]	usask_1
3	b6ab77fd7	1024	1024	[834.0, 95.0, 109.0, 107.0]	usask_1
4	b6ab77fd7	1024	1024	[26.0, 144.0, 124.0, 117.0]	usask_1

Figure 9: Extraction of data

For applying the autoencoder the images are separated into 2 folder one is images having a wheat head and second without a wheat some of the images are show in figure 10 and figure 11 respectively.

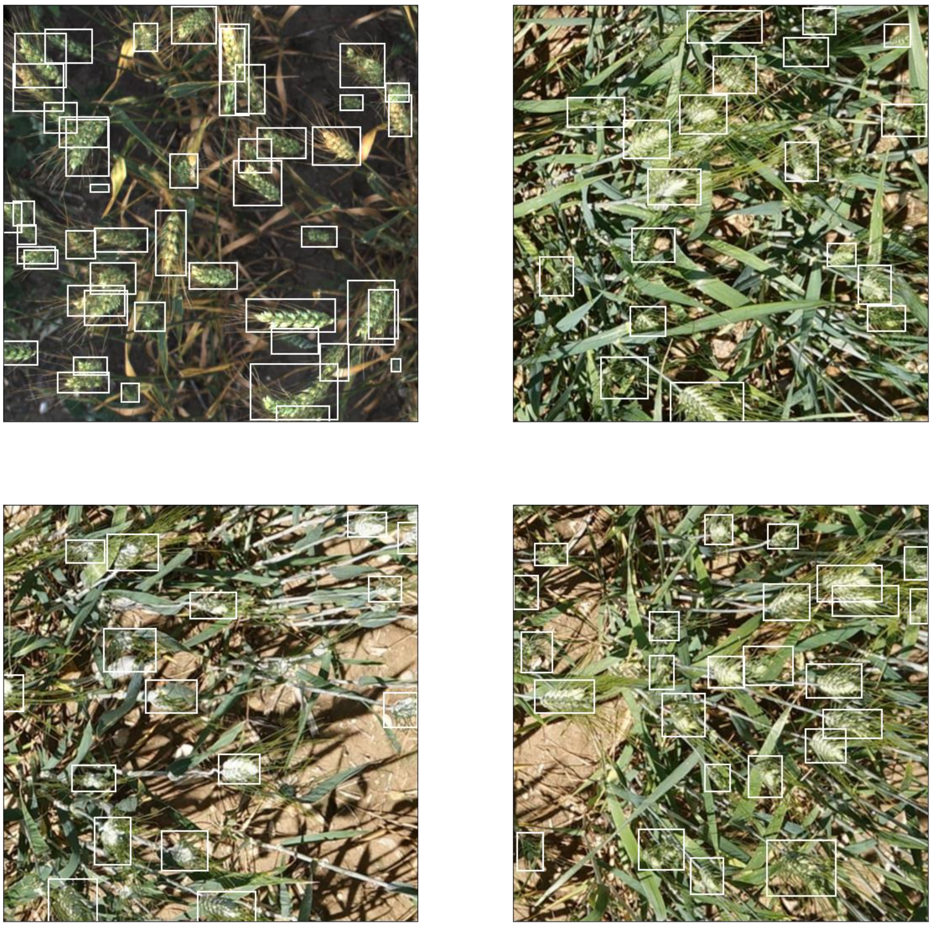


Figure 9: Images with the wheat head

Figure 9: Images without the wheat head

Convolution Autoencoder Network:

```
#Encoder
model = Sequential()
model.add(Conv2D(64, (3,3 ), activation='ReLU', padding='same', input_shape=(SIZE, SIZE, 3)))
model.add(MaxPooling2D((2, 2), padding='same'))
model.add(Conv2D(32, (3, 3), activation='PReLU', padding='same'))
model.add(MaxPooling2D((2, 2), padding='same'))
model.add(Conv2D(16, (3, 3), activation='ELU', padding='same'))
model.add(MaxPooling2D((2, 2), padding='same'))

#Decoder
model.add(Conv2D(16, (3, 3), activation='ELU', padding='same'))
model.add(UpSampling2D((2, 2)))
model.add(Conv2D(32, (3, 3), activation='PReLU', padding='same'))
model.add(UpSampling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='ReLU', padding='same'))
model.add(UpSampling2D((2, 2)))


model.add(Conv2D(3, (3, 3), activation='sigmoid', padding='same'))

model.compile(optimizer='adam', loss='mean_squared_error', metrics=['mse'])
model.summary()
```

Figure 9: Convolution Autoencoder Using Multiple activation function.

```
7/7 [==============================] - 55s 8s/step - loss: 0.0304 - mse: 0.0304 - val_loss: 0.0299 - val_mse: 0.0299
Epoch 15/25
7/7 [==============================] - 55s 8s/step - loss: 0.0301 - mse: 0.0301 - val_loss: 0.0308 - val_mse: 0.0308
Epoch 16/25
7/7 [==============================] - 55s 8s/step - loss: 0.0291 - mse: 0.0291 - val_loss: 0.0289 - val_mse: 0.0289
Epoch 17/25
7/7 [==============================] - 55s 8s/step - loss: 0.0287 - mse: 0.0287 - val_loss: 0.0279 - val_mse: 0.0279
Epoch 18/25
7/7 [==============================] - 55s 8s/step - loss: 0.0282 - mse: 0.0282 - val_loss: 0.0281 - val_mse: 0.0281
Epoch 19/25
7/7 [==============================] - 55s 8s/step - loss: 0.0285 - mse: 0.0285 - val_loss: 0.0270 - val_mse: 0.0270
Epoch 20/25
7/7 [==============================] - 55s 8s/step - loss: 0.0272 - mse: 0.0272 - val_loss: 0.0306 - val_mse: 0.0306
Epoch 21/25
7/7 [==============================] - 55s 8s/step - loss: 0.0275 - mse: 0.0275 - val_loss: 0.0296 - val_mse: 0.0296
Epoch 22/25
7/7 [==============================] - 55s 8s/step - loss: 0.0278 - mse: 0.0278 - val_loss: 0.0295 - val_mse: 0.0295
Epoch 23/25
7/7 [==============================] - 55s 8s/step - loss: 0.0278 - mse: 0.0278 - val_loss: 0.0251 - val_mse: 0.0251
Epoch 24/25
7/7 [==============================] - 55s 8s/step - loss: 0.0270 - mse: 0.0270 - val_loss: 0.0282 - val_mse: 0.0282
Epoch 25/25
7/7 [==============================] - 57s 8s/step - loss: 0.0262 - mse: 0.0262 - val_loss: 0.0281 - val_mse: 0.0281
```

Figure 9: Output of the best model when trained for 25 Epochs.

Reference

David, Etienne et al. (2020). “Global Wheat Head Detection (GWHD) dataset: *a large and diverse dataset of high-resolution RGB-labelled images to develop and benchmark wheat head detection methods*”. In: Plant Phenomics 2020