National College of Ireland

Configuration Manual

MSc Research Project
MSc in Data Analytics

Ankur Ghogale
Student ID: X19193866

School of Computing
National College of Ireland

Supervisor: Prof. Noel Cosgrave

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Ankur Ghogale
Student ID:	X19193866
Programme:	MSc in Data Analytics
Year:	2020
Module:	MSc Research Project
Supervisor:	Prof. Noel Cosgrave
Submission Due Date:	17/12/2020
Project Title:	Configuration Manual
Word Count:	919
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:	1st February 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

Ankur Ghogale
X19193866

1 Introduction

This configuration manual present all the step involved for the implementation of our thesis project. The below section 2 shows the system configuration needed to run this project, section 3 shows system setup for the project, section 4 describes all the modules and libraries used and installed, section 5 presents EDA on the data in csv file, section 6 mentions data collection and pre-processing procedures and section 7 and 8 shows modelling and evaluation. Screenshots and output snippets are mentioned to support our claim.

2 System Configuration

- **Programming Language:** Python version 3.6.9
- **Integrated Development Environment (IDE):** Google Colab Pro
- **IDE Specification:** GPU offered by Google Colab Pro are Nvidia K80, Tesla T4 and P100 with maximum limit of 24GB RAM.
- **Storage:** Google Drive One with 100GB storage space
- **Local System Configuration:** i5 6th generation, 64-bit Mac OS Catalina (version 10.15.2), 8GB RAM and 250GB HD
- **Browser:** Safari version 13.0.4

3 System Setup

Firstly, Google Colab Pro is setup for GPU use and Google drive is integrated using google.colab library. This is done by clicking runtime -¿ change runtime type then select GPU or TPU in hardware accelerator section and finally select runtime shape as standard or High-Ram depending upon the usage. While running CNN model we used High-Ram and for other purposes we used standard. The below figure 1 shows GPU setup. Then integrate Google drive using mount function. Once it is done we have access to all the files present in the drive using Colab notebook. The figure 2 shows the function and its output once it is successfully integrated. We have made two different notebook files for our project. Emotion_Recognition_Data_Preprocessing.ipynb contains all the pre-processing techniques implemented on the data and Emotion_Recognition_modelling.ipynb contains modelling and evaluation part of the project.

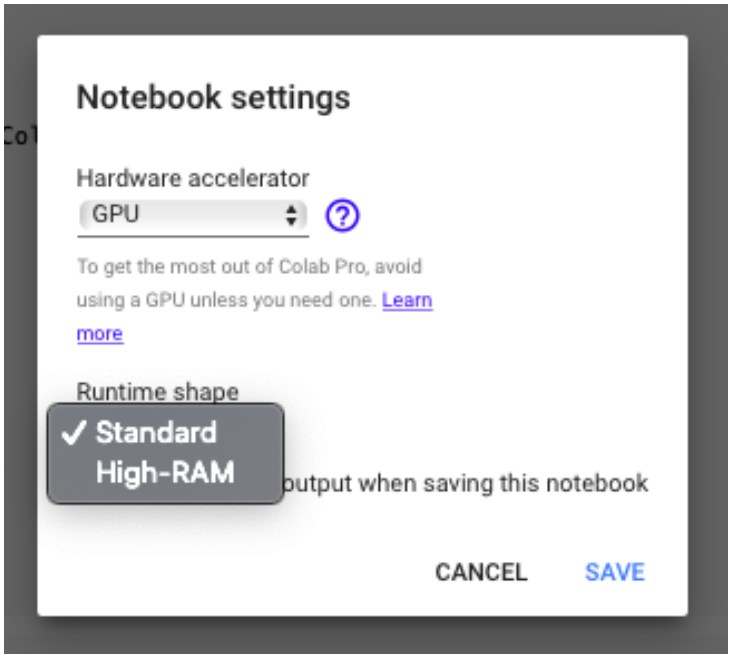

Figure 1: GPU Setup

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

Mounted at /content/gdrive
```

Figure 2: Integrating google drive

4 Libraries

Figure 3 shows necessary libraries for the pre-processing of the three types of data i.e. text, audio, and video. This libraries were used for cleaning textual data, performing techniques like tokenization, lemmatization, etc. It also comprises of libraries like moviepy and ffmpeg for editing video and audio data.

```
[ ] import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    import seaborn as sns
    import nltk
    from nltk import ne_chunk
    import plotly.express as px
    from collections import Counter
    import matplotlib.pyplot as plt
    from wordcloud import WordCloud, ImageColorGenerator
    import pafy
    from moviepy.editor import *
    import os
    import cv2
    import shutil
    import librosa, librosa.display
    import IPython.display as ipd
    import math
    import glob
    from PIL import Image
    import mutagen
    from mutagen.mp3 import MP3
    import time
    import spacy
```

Figure 3: Libraries used for pre-processing

Before importing the mentioned libraries it is necessary to install the libraries given in the figure 4. Libraries like pafy (built on youtube-dl framework) and mutagen are not pre-installed on colab, therefore it is needed to install them from our side.

```
[3] pip install youtube-dl

    Collecting youtube-dl
      Downloading https://files.pythonhosted.org/packages/46/9c/69f5ede4f4b3e01390a9e9b355cb3bbe4e7550439bd0c3
         |████████████████████████████████| 1.9MB 4.3MB/s
    Installing collected packages: youtube-dl
    Successfully installed youtube-dl-2020.12.14

    pip install pafy

    Collecting pafy
      Downloading https://files.pythonhosted.org/packages/74/69/829919eeadff695338f98fa12bb99e45490761a2010c8d
    Installing collected packages: pafy
    Successfully installed pafy-0.5.5

[5] pip install mutagen

    Collecting mutagen
      Downloading https://files.pythonhosted.org/packages/16/b3/f7aa8edf2ff4495116f95fd442b2a346aa55d1d4631314
         |████████████████████████████████| 225kB 4.3MB/s
    Installing collected packages: mutagen
    Successfully installed mutagen-1.45.1
```

Figure 4: Libraries to be installed

The figure 5 shows the libraries needed for building machine learning and neural network models. It also contains libraries like SMOTE which is used for oversampling and balancing data.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten
from tensorflow.keras.layers import Conv2D
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from imblearn.over_sampling import SMOTE
from collections import Counter
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import MultinomialNB
```

Figure 5: Libraries used for modelling

5 EDA on CSV data

This section shows the exploratory data analysis on the csv file. It represents the total number of rows and distribution of all the seven emotion categories. Figure 5 shows the output data types, number of null values present in each column and number of observations present. The figure 6 shows the number of observations present for each class.

```
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   link               4513 non-null   object
 1   start              4513 non-null   float64
 2   end                4513 non-null   float64
 3   video              4513 non-null   object
 4   utterance          4513 non-null   object
 5   arousal            4513 non-null   float64
 6   valence            4513 non-null   float64
 7   EmotionMaxVote     4513 non-null   object
 8   transcript         4513 non-null   object
 9   preprocessed_text  4513 non-null   object
dtypes: float64(4), object(6)
memory usage: 387.8+ KB
```

Figure 6: Dataset information

6 Data Collection and Pre-processing

This section contains code and output snippets of data collection and pre-processing techniques we used in the project. Figure 8 shows the calling of a download_vid custom method which is used to download videos for all the three datasets (training, testing, validation). The figure also shows the output with the video format, video quality, and video id. It also gives in the output if any video is unavailable or cannot be fetched. It is one of the most time consuming process of the project.

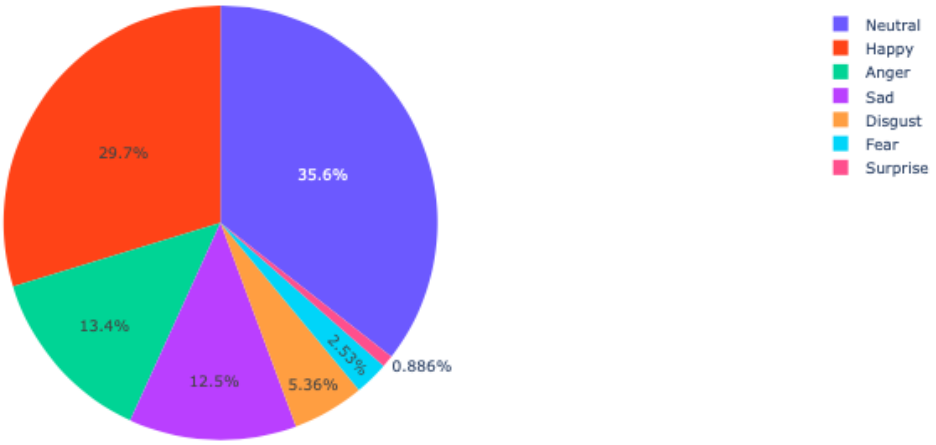

Figure 7: Distribution of emotion categories

```
#For Testing data
path='/content/gdrive/MyDrive/Colab Notebooks/Test_Videos'
download_vid(path, data_test)
```

```
1280x720 mp4
normal:mp4@640x360
normal:mp4@1280x720
4e03b2993_1.mp4

1280x720 mp4
normal:mp4@640x360
normal:mp4@1280x720
5b1fe4bd9_1.mp4

1280x720 mp4
ERROR: Video unavailable
something went wrong
normal:mp4@640x360
normal:mp4@1280x720
5db2d5e44_1.mp4
```

Figure 8: Fetching Videos from Youtube

Once the videos are fetched, next step is to split them as per the timestamp and figure 9 shows the output from the training set and it also shows the time taken to split the videos in training set.

```
100%|██████████| 131/131 [00:06<00:00, 19.86it/s]
[MoviePy] Done.
[MoviePy] >>>> Video ready: /content/gdrive/MyDrive/Colab Notebooks/balance_train_clips/Disgust/610fd0cfb_1.mp4utterance_7.mp4.mp4

[MoviePy] >>>> Building video /content/gdrive/MyDrive/Colab Notebooks/balance_train_clips/Happy/610fd0cfb_1.mp4utterance_8.mp4.mp4
[MoviePy] Writing audio in 610fd0cfb_1.mp4utterance_8.mp4TEMP_MPY_wvf_snd.mp3
100%|██████████| 219/219 [00:00<00:00, 302.56it/s][MoviePy] Done.
[MoviePy] Writing video /content/gdrive/MyDrive/Colab Notebooks/balance_train_clips/Happy/610fd0cfb_1.mp4utterance_8.mp4.mp4

100%|██████████| 297/297 [00:21<00:00, 13.71it/s]
[MoviePy] Done.
[MoviePy] >>>> Video ready: /content/gdrive/MyDrive/Colab Notebooks/balance_train_clips/Happy/610fd0cfb_1.mp4utterance_8.mp4.mp4

file not found
--- Data loaded. Loading time: 10727.424177646637 seconds ---
```

Figure 9: Clipping fetched videos

Then the clipped videos are converted to frames and then these frames are again treated by opencv to crop the face. The output snippet shown in the figure 10 describes how we tried remove distorted images from our set. We used a function which would save the image if only a face is identified, otherwise it would ignore it.

```
Image faces_detected written to filesystem:  True
Found 0 Faces! file ignored
Found 1 faces and saved
Image faces_detected written to filesystem:  True
Found 2 Faces! file ignored
Found 1 faces and saved
Image faces_detected written to filesystem:  True
Found 1 faces and saved
Image faces_detected written to filesystem:  True
Found 1 faces and saved
Image faces_detected written to filesystem:  True
Found 0 Faces! file ignored
Found 1 faces and saved
```

Figure 10: Cropping faces

The figure 11 shows the extraction of audio data from the video files. As the names of videos are taken from csv file, it also mentions if the file is not present.

Once the audio data is extracted and sorted based on emotions, feature extraction process is implemented. Figure 12 shows the feature extracted and time taken to extract features from the training set.

7 Modelling

This section describes the models implemented and shows the output given while running these models. In our project, we used a 2D CNN model for emotion recognition using images. It is represented in the figure 13. It also shows the number of epochs running along with the accuracy of training and validation data.

```
file not found
5b44393ed.mp4
[MoviePy] Writing audio in /content/gdrive/MyDrive/Colab Notebooks/Train_AUDIO/8c56c5ac5/8c56c5ac5.mp3
100%|██████████| 2494/2494 [00:02<00:00, 832.04it/s][MoviePy] Done.

extracted {} {'8c56c5ac5.mp4'}
[MoviePy] Writing audio in /content/gdrive/MyDrive/Colab Notebooks/Train_AUDIO/76eecf968/76eecf968.mp3
100%|██████████| 1997/1997 [00:02<00:00, 858.62it/s][MoviePy] Done.

extracted {} {'76eecf968.mp4'}
file not found
243cff084.mp4
[MoviePy] Writing audio in /content/gdrive/MyDrive/Colab Notebooks/Train_AUDIO/7459327df/7459327df.mp3
100%|██████████| 1536/1536 [00:02<00:00, 713.68it/s][MoviePy] Done.

extracted {} {'7459327df.mp4'}
[MoviePy] Writing audio in /content/gdrive/MyDrive/Colab Notebooks/Train_AUDIO/a3f32e3a3/a3f32e3a3.mp3
100%|██████████| 1223/1223 [00:01<00:00, 830.33it/s][MoviePy] Done.

extracted {} {'a3f32e3a3.mp4'}
[MoviePy] Writing audio in /content/gdrive/MyDrive/Colab Notebooks/Train_AUDIO/b32a6ac3b/b32a6ac3b.mp3
100%|██████████| 2761/2761 [00:04<00:00, 591.89it/s][MoviePy] Done.

extracted {} {'b32a6ac3b.mp4'}
[MoviePy] Writing audio in /content/gdrive/MyDrive/Colab Notebooks/Train_AUDIO/c9f0e3623/c9f0e3623.mp3
100%|██████████| 1692/1692 [00:01<00:00, 904.52it/s][MoviePy] Done.
```

Figure 11: Extraction Audio from Video

```
[-282.14790471   94.50092837   31.01372358   29.16079713   10.58179149
   20.07226562    1.03472449   17.32125863    5.39681626    9.98538719
    1.03348052    6.68837352    0.86597734] 6
[-286.41019482   67.02629141  -18.14577301    1.32512847  -31.43008227
   -0.7239889   -13.30067547  -12.40417458  -10.766134    -16.41091043
  -12.77570872  -12.68259906  -11.00992024] 6
[-156.6802438    83.37258712  -23.11137095   25.92980508  -10.75454732
    4.81199085  -14.30532429    2.46596802   -7.66027978    0.37026314
  -11.81843563    8.5332777   -12.58669751] 6
[-274.95648259   95.86972509  -24.17061238   25.07780821  -18.77513
    0.35405332  -14.37642924   -1.32008051  -11.5051059     3.61923987
  -17.04482453   -0.59090594  -15.52890236] 6
--- Data loaded. Loading time: 713.1590411663055 seconds ---
```

Figure 12: MFCC feature and time taken for this process

```
40/40 [==============================] - 6s 162ms/step - loss: 0.4665 - accuracy: 0.8378 - val_loss: 3.8947 - val_accuracy: 0.1496
Epoch 43/50
40/40 [==============================] - 7s 163ms/step - loss: 0.4655 - accuracy: 0.8342 - val_loss: 3.7532 - val_accuracy: 0.1592
Epoch 44/50
40/40 [==============================] - 6s 161ms/step - loss: 0.4503 - accuracy: 0.8393 - val_loss: 3.9593 - val_accuracy: 0.1451
Epoch 45/50
40/40 [==============================] - 6s 161ms/step - loss: 0.4247 - accuracy: 0.8501 - val_loss: 3.7478 - val_accuracy: 0.1607
Epoch 46/50
40/40 [==============================] - 6s 160ms/step - loss: 0.4296 - accuracy: 0.8524 - val_loss: 3.8425 - val_accuracy: 0.1562
Epoch 47/50
40/40 [==============================] - 6s 161ms/step - loss: 0.4196 - accuracy: 0.8473 - val_loss: 4.0267 - val_accuracy: 0.1518
Epoch 48/50
40/40 [==============================] - 6s 161ms/step - loss: 0.4261 - accuracy: 0.8493 - val_loss: 3.8041 - val_accuracy: 0.1443
Epoch 49/50
40/40 [==============================] - 6s 161ms/step - loss: 0.3973 - accuracy: 0.8556 - val_loss: 3.8293 - val_accuracy: 0.1570
Epoch 50/50
40/40 [==============================] - 6s 161ms/step - loss: 0.3961 - accuracy: 0.8655 - val_loss: 3.9877 - val_accuracy: 0.1763
```

Figure 13: CNN model output

8 Evaluation

Further the results given by the models are evaluated using confusion matrix and classification report. The confusion matrix is displayed using heat-map to understand the results easily.