National College of Ireland

Configuration Manual

MSc Research Project
MSC in Data Analytics

Geethu Issac
Student ID: 20210515

School of Computing
National College of Ireland

Supervisor: Mr. Jorge Basilio

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Geethu Issac
Student ID:	20210515
Programme:	MSC in Data Analytics
Year:	2021-'22
Module:	MSc Research Project
Supervisor:	Mr. Jorge Basilio
Submission Due Date:	19/09/2022
Project Title:	Configuration Manual
Word Count:	799
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:	16th September 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

Geethu Issac
20210515

1 Introduction

As per the requirements of MSc Research Project submission of National College of Ireland, the Configuration Manual is concluded in relation to implementation of work. This manual reports the software used and settings considered to analyse the Deception in the audio signal using Convolutional Neural Network..

2 System Specification

The hardware and software requirements for the implementation of the model is discussed in this section.

2.1 Hardware Requirements

The hardware requirement in the development of the model is covered here detailing processor, RAM, storage and OS. Processor: 11th Gen Intel(R) Core(TM) i5-1125G7 @2.40GHz 2.42GHz Random Access Memory: 8 Giga Bytes Storage: 256GB SSD/1TB HDD Operating System: 64-bit operating system, x64-based Operating System

2.2 Software Requirements

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

- Jupyter Notebook
- Python Version 3
- Overleaf

3 Environment Setup

3.1 Anaconda Installation

Anaconda was installed from https://www.anaconda.com/distribution/#download-section and the Python 3.7 version of Anaconda was installed for the operating system. The default settings for the same must be retained.

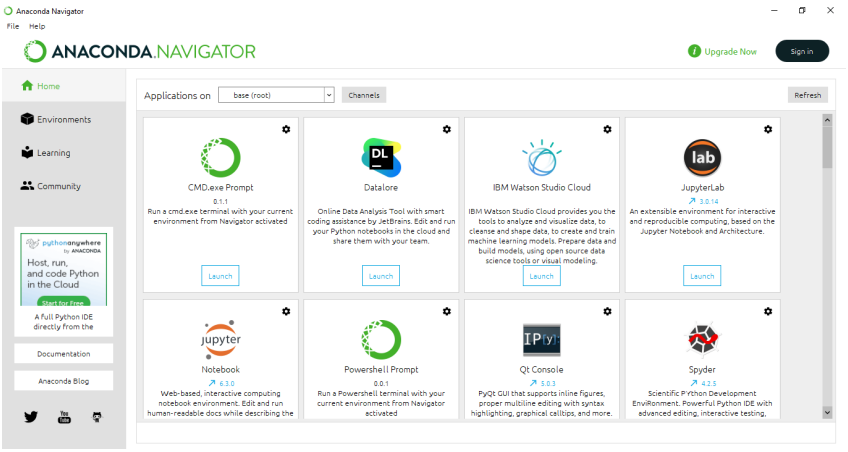


Figure 1: Anaconda Navigator

3.2 Jupyter Notebook

After starting Anaconda Navigator, extra optional tools can be installed. Jupyter Notebook was the Integrated Development Environment (IDE) used in this work and was installed by default.

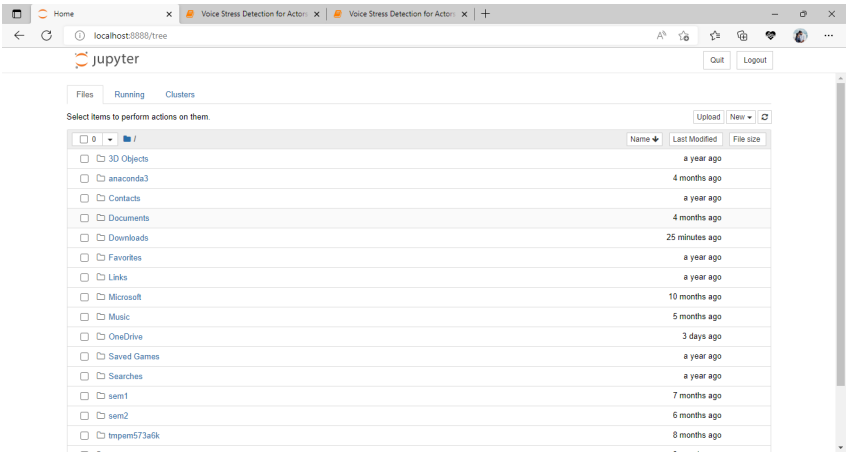


Figure 2: Jupyter Notebook

4 Dataset

The dataset taken for analysis was The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) taken from Steven R. Livingstone (2018). The dataset was taken from an article taken from PLUS One. It was a publically available data with audio as well as video content. For this work only the audio signals are taken into consideration. The article allows to get to know the privacy and licence linked to the content and the recordings.

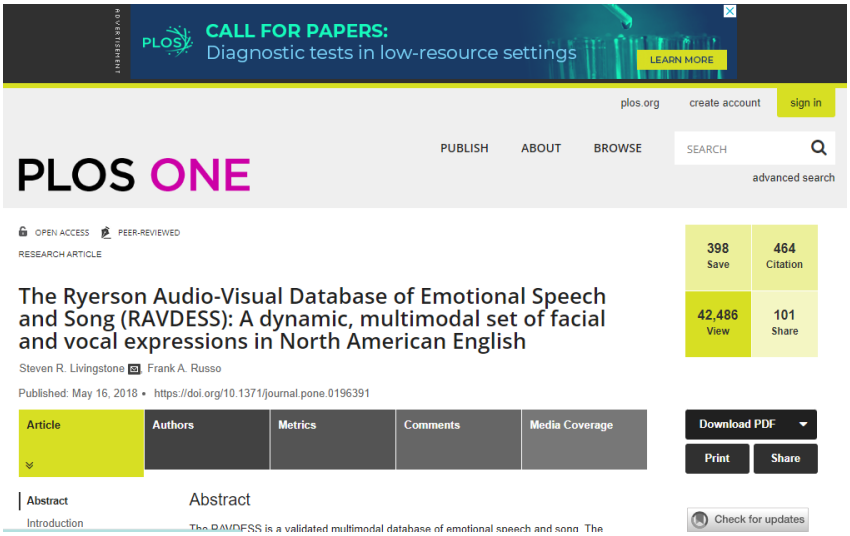


Figure 3: Citation of Dataset

The dataset include recordings of 24 different actors among which 12 are male subjects and 12 are female subjects. The file name convention shows the recording details such as Modality, Vocal channel, Emotion, Emotional intensity, Statement , Repetition and Actor gender. This can distinguish the unisexual recordings. There are 1440 recordings of speech in the dataset. This can be directly downloaded into the local PC and was directly used in the IDE using Python programming.

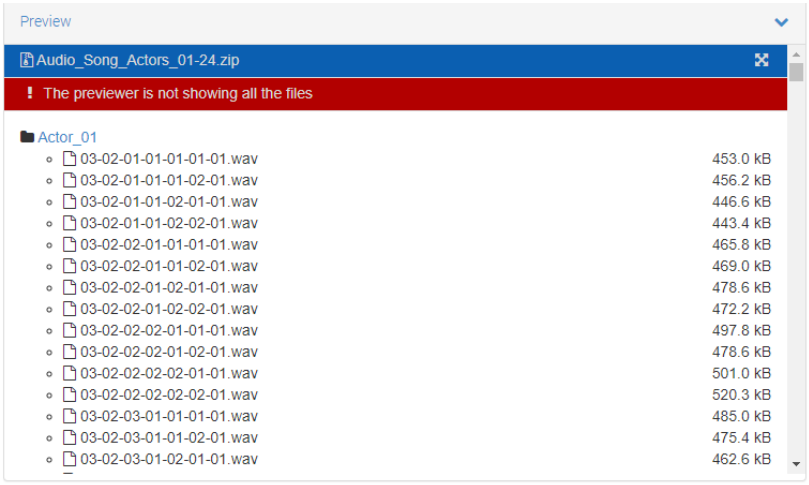


Figure 4: Dataset

5 Implementation

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
- Matplotlib
- Pandas
- Librosa
- Sklearn
- Plotly
- TensorFlow
- Keras

```
import os
import random
import sys


## Package
import glob
import keras
import IPython.display as ipd
import librosa
import librosa.display
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import plotly
```

```
plotly.graph_objs as go
import plotly.offline as py
import plotly.tools as tls
import seaborn as sns
import scipy.io.wavfile
import tensorflow
py.init_notebook_mode(connected=True)
```

Figure 5: Package Import

```
## Keras
from tensorflow.keras import regularizers
from tensorflow.keras.callbacks import ModelCheckpoint, LearningRateScheduler, EarlyStopping
from tensorflow.keras.callbacks import  History, ReduceLROnPlateau, CSVLogger
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.layers import Dense, Embedding, LSTM
from tensorflow.keras.layers import Input, Flatten, Dropout, Activation, BatchNormalization
from tensorflow.keras.layers import Conv1D, MaxPooling1D, AveragePooling1D
from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.preprocessing.text import Tokenizer
from keras.utils import np_utils
from tensorflow.keras.utils import to_categorical


## Sklearn
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import LabelEncoder


## Rest
from scipy.fftpack import fft
from scipy import signal
from scipy.io import wavfile
from tqdm import tqdm

input_duration=3
# % pylab inline
```

Figure 6: Package Import

5.1 Preprocessing of Data

The preprocessing done on recording include pitch tuning, speed tuning, stretching, silencing and white noise addition. The code performed is given below.

Data Making/Processing

In [37]:
```
def plot_time_series(data):
    """
    Plot the Audio Frequency.
    """
    fig = plt.figure(figsize=(14, 8))
    plt.title('Raw wave ')
    plt.ylabel('Amplitude')
    plt.plot(np.linspace(0, 1, len(data)), data)
    plt.show()


def noise(data):
    """
    Adding White Noise.
    """
    noise_amp = 0.005*np.random.uniform()*np.amax(data)
    data = data.astype('float64') + noise_amp * np.random.normal(size=data.shape[0])
    return data

def shift(data):
    """
    Random Shifting.
    """
    s_range = int(np.random.uniform(low=-5, high = 5)*500)
    return np.roll(data, s_range)

def stretch(data, rate=0.8):
    """
    Streching the Sound.
    """
    data = librosa.effects.time_stretch(data, rate)
    return data

def pitch(data, sample_rate):
    """
    Pitch Tuning.
    """
    bins_per_octave = 12
    pitch_pm = 2
    pitch_change =  pitch_pm * 2*(np.random.uniform())
    data = librosa.effects.pitch_shift(data.astype('float64'),
```

Figure 7: Data Preprocessing

Spectrograms are the main visualizations used in the analysis of audio signal in order to determine the amplitude and frequency of audio signal. The code of visualizing the spectrograms are discussed below.

[9]:
```
# Plotting Wave Form and Spectrogram
freqs, times, spectrogram = log_specgram(samples, sample_rate)

fig = plt.figure(figsize=(14, 8))
ax1 = fig.add_subplot(211)
ax1.set_title('Raw wave of ' + filename)
ax1.set_ylabel('Amplitude')
librosa.display.waveshow(samples, sr=sample_rate)

ax2 = fig.add_subplot(212)
ax2.imshow(spectrogram.T, aspect='auto', origin='lower',
           extent=[times.min(), times.max(), freqs.min(), freqs.max()])
ax2.set_yticks(freqs[::16])
ax2.set_xticks(times[::16])
ax2.set_title('Spectrogram of ' + filename)
ax2.set_ylabel('Freqs in Hz')
ax2.set_xlabel('Seconds')
```

Figure 8: Spectrogram

5.2 Data Labelling

The six different emotions are labelled to positive, negative and neutral deception audio signals. Thus the overall model becomes a three class problem.

```
#3 class: Positive, Neutral & Negative

# Positive:  Happy
# Negative: Angry, Fearful, Sad
# Neutral: Calm, Neutral

label3_list = []
for i in range(len(data_df)):
    if data_df.emotion[i] == 1: # Neutral
        lb = "_neutral"
    elif data_df.emotion[i] == 2: # Calm
        lb = "_neutral"
    elif data_df.emotion[i] == 3: # Happy
        lb = "_positive"
    elif data_df.emotion[i] == 4: # Sad
        lb = "_negative"
    elif data_df.emotion[i] == 5: # Angry
        lb = "_negative"
    elif data_df.emotion[i] == 6: # Fearful
        lb = "_negative"
    else:
        lb = "_none"

    # Add gender to the label
    label3_list.append(data_df.gender[i] + lb)

len(label3_list)
```

Figure 9: Data Labelling

5.3 Models Implemented

For the model implementation, data from a single actor was used to compare Machine Learning model with Deep Learning model. In this work, Decision Tree and Random Forest model was implemented on single actor data.

- Decision Tree: The implementation of Decision Tree is discussed with the figure as discussed in Bento (2021). As the precision of the model obtained is 57.5%, it is not further taken for analysis.

```
DecisionTreeClassifier(criterion="gini", max_depth=10, max_features="log2",
                       max_leaf_nodes = 10, min_samples_leaf = 3, min_samples_split = 20
                       n_estimators= 200, random_state= 5)

.t(X_train, y_train)
```

Figure 10: Decision Tree

- Random Forest: The implementation of Random Forest model for a single actor audio data is discussed with the figure below and cited in Yiu (2019). As the precision of the model obtained is just 62.34%, it is also not considered for analysis. As the problem is a multiclass classification problem, Decision Trees and Random Forests are the best possible machine learning solutions.

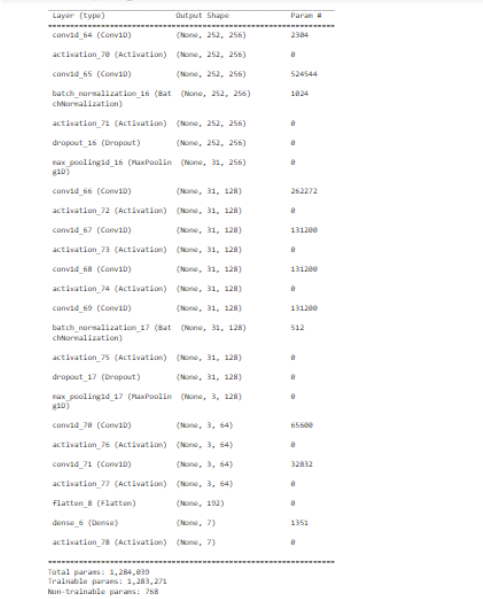
```
]: from sklearn.ensemble import RandomForestClassifier

]: rforest = RandomForestClassifier(criterion="gini", max_depth=10, max_features="log2",
                                    max_leaf_nodes = 10, min_samples_leaf = 3, min_samples_split = 20,
                                    n_estimators= 200, random_state= 5)

]: rforest.fit(X_train, y_train)
```

Figure 11: Random Forest

- Convolutional Neural Network: The implementation of Convolutional Neural network as covered in Mandal (2021)for the research work is as discussed below. Keras was used for implementation with Sequential model.



Layer (type)	Output Shape	Param #
conv1d_64 (Conv1D)	(None, 252, 256)	2304
activation_70 (Activation)	(None, 252, 256)	0
conv1d_65 (Conv1D)	(None, 252, 256)	524544
batch_normalization_16 (BatchNormalization)	(None, 252, 256)	1024
activation_71 (Activation)	(None, 252, 256)	0
dropout_16 (Dropout)	(None, 252, 256)	0
max_pooling1d_16 (MaxPooling1D)	(None, 31, 256)	0
conv1d_66 (Conv1D)	(None, 31, 128)	262272
activation_72 (Activation)	(None, 31, 128)	0
conv1d_67 (Conv1D)	(None, 31, 128)	131200
activation_73 (Activation)	(None, 31, 128)	0
conv1d_68 (Conv1D)	(None, 31, 128)	131200
activation_74 (Activation)	(None, 31, 128)	0
conv1d_69 (Conv1D)	(None, 31, 128)	131200
batch_normalization_17 (BatchNormalization)	(None, 31, 128)	512
activation_75 (Activation)	(None, 31, 128)	0
dropout_17 (Dropout)	(None, 31, 128)	0
max_pooling1d_17 (MaxPooling1D)	(None, 3, 128)	0
conv1d_70 (Conv1D)	(None, 3, 64)	65600
activation_76 (Activation)	(None, 3, 64)	0
conv1d_71 (Conv1D)	(None, 3, 64)	32832
activation_77 (Activation)	(None, 3, 64)	0
flatten_8 (Flatten)	(None, 192)	0
dense_6 (Dense)	(None, 7)	1351
activation_78 (Activation)	(None, 7)	0

Total params: 1,284,039
Trainable params: 1,283,271
Non-trainable params: 768

Figure 12: CNN

6 Evaluation

The evaluation was done by mainly using the loss graph as the dataset considered is discontinuous. The evaluation metrics like precision and recall was also identified in the results section to explain the integrity of the work.

7 Overleaf for Documentation

For the documentation purpose, the overleaf or latex was used. This is allows ease in alignment as per the referencing style suggested by NCI.

```
[61]:
# Plotting the Train Valid Loss Graph

plt.plot(cnnhistory.history['loss'])
plt.plot(cnnhistory.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```

Figure 13: Loss Plot

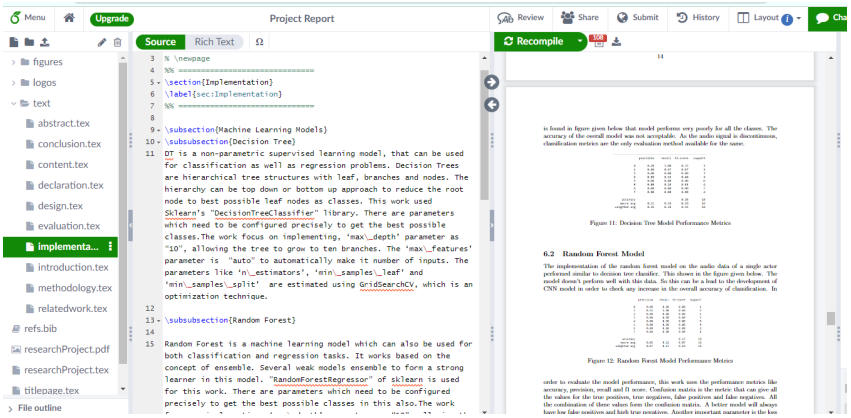


Figure 14: UI of Overleaf

References

Bento, C. (2021). Decision tree classifier explained in real-life: picking a vacation destination, *Towards Data Science* .

Mandal, M. (2021). Introduction to convolutional neural networks (cnn), *Data Science Blogathon* .

Steven R. Livingstone, F. A. R. (2018). The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english, *PLUS One 13(5): e0196391* .

Yiu, T. (2019). Understanding random forest how the algorithm works and why it is so effective, *Towards Data Science* .