Configuration Manual

MSc Research Project
Programme Name

Karthi Mahendran
Student ID: X20118198

School of Computing
National College of Ireland

Supervisor: Jorge Basilio

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Karthi Mahendran		
Student ID:	X20118198		
Programme:	Data Analytics	**Year:**	2021
Module:	MSc Research Project		
Lecturer:	Jorge Basilio		
Submission Due Date:	16\08\21		
Project Title:	Configuration Manual		
Word Count:	941	**Page Count:**	6

Signature: ……

Date: 16th August 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.

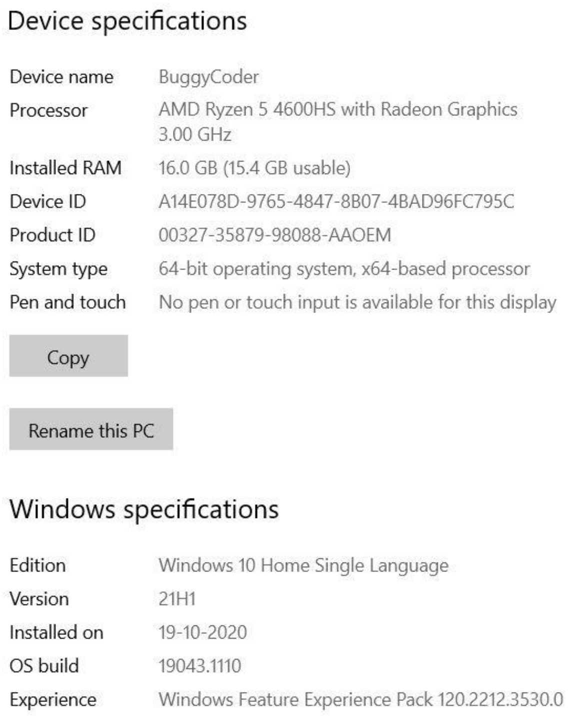
Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Forename Surname
Student ID:

1 Hardware Requirements

Windows 10 with a 64-bit operating system, an AMD Ryzen 5 4600HS with Radeon Graphics 3.00 GHz and 16 GB of RAM were the computer configurations used for project the implementation.



Device specifications

Device name	BuggyCoder
Processor	AMD Ryzen 5 4600HS with Radeon Graphics 3.00 GHz
Installed RAM	16.0 GB (15.4 GB usable)
Device ID	A14E078D-9765-4847-8B07-4BAD96FC795C
Product ID	00327-35879-98088-AAOEM
System type	64-bit operating system, x64-based processor
Pen and touch	No pen or touch input is available for this display

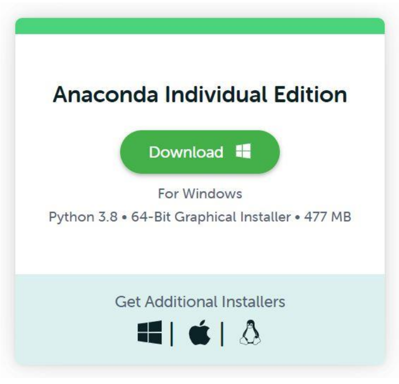
Windows specifications

Edition	Windows 10 Home Single Language
Version	21H1
Installed on	19-10-2020
OS build	19043.1110
Experience	Windows Feature Experience Pack 120.2212.3530.0

Figure 1: Hardware Configuration

2 Software Requirement

Throughout the research, we used the Python 3.8 to write all of the coding sections. We introduced Python codes using the Jupyter Lab on the Anaconda Navigator platform. The first thing we need to do is install the Anaconda App. The 64-bit version of the configuration file is imported as requirements change to be consistent with the 64-bit Windows OS.



Anaconda Individual Edition
Download
For Windows
Python 3.8 • 64-Bit Graphical Installer • 477 MB
Get Additional Installers

Figure 2: Anaconda Specification

The Anaconda Navigator opens from Start -> Anaconda Navigator after successful installation. The Jupyter Lab is installed in Anaconda Navigator and it can be launched from there itself.

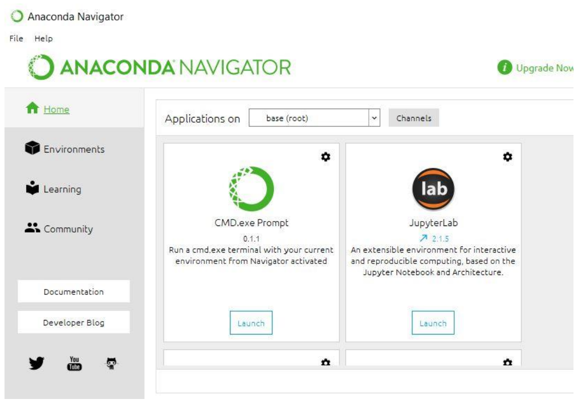


Figure 3: Overview of Anaconda Navigator

3 Python Library Package Requirement

The list of python package are installed using pip command in the python environmental command prompt,

- tensorflow == 1.15.0
- keras == 2.2.4
- numpy
- pandas
- piglet
- chatterbot
- SpeechRecognition == 2.13
- Scikit-learn
- Matplotlib
- Nltk
- Gtts
- Pickle
- Seaborn

- PIL – Pillow
- Np_utils
- Joblib
- Textblob
- Nfx

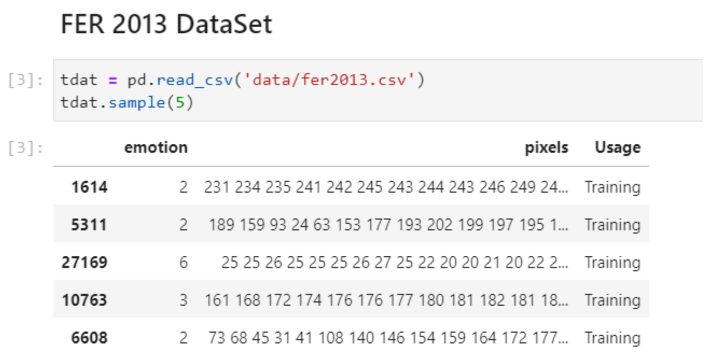
4 Dataset Description

- The Fer2013 dataset can be download in Kaggle repository. Download URL – https://www.kaggle.com/ashishpatel26/facial-expression-recognitionferchallenge
- Extract the FER2013.csv from the fer2013dataset.jar to data folder
- DialyDialog can be download from http://yanran.li/dailydialog.html
- Extract the files from ijcnlp_dailydialog.zip to data folder

5 Data Preparation

The notebook file 1.datapreparation.ipynb will be present in the same artifacts folders. In this file we can prepare the data transformation for input to model.

- The fer.2013csv file read into data frame and examined some data transformations
- Saved all the emotion images into data\test and data\train
- For text analysis data/dialogues_text.txt and data/dialogues_emotion.txt are merged and save in data/dailydialog.csv



FER 2013 DataSet

```
[3]: tdat = pd.read_csv('data/fer2013.csv')
     tdat.sample(5)
```

[3]:

	emotion	pixels	Usage
1614	2	231 234 235 241 242 245 243 244 243 246 249 24...	Training
5311	2	189 159 93 24 63 153 177 193 202 199 197 195 1...	Training
27169	6	25 25 26 25 25 25 26 27 25 22 20 20 21 20 22 2...	Training
10763	3	161 168 172 174 176 176 177 180 181 182 181 18...	Training
6608	2	73 68 45 31 41 108 140 146 154 159 164 172 177...	Training

Figure 4: Fer2013 CSV data

6 Model Preparation

The notebook file 2.face-cnn-model.ipynb, 3.CNN-CapsNet.ipynb, 4.NaiveBayes-test.ipynb will be present in the same artifacts folders. These files train the dataset according to their models.

6.1 CNN Model Training

- To train the fer2013 data, data understating is done again.
- Data augmentation is done in the section.

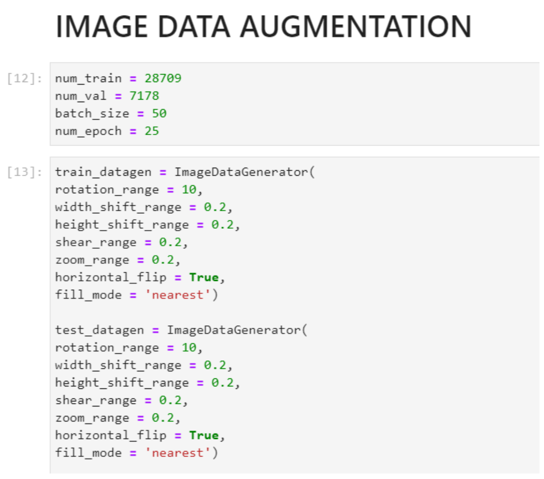


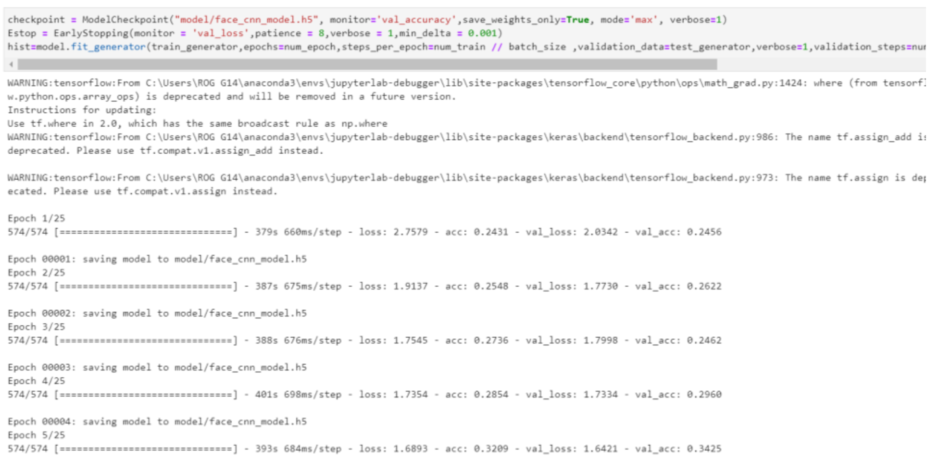
IMAGE DATA AUGMENTATION

```
[12]: num_train = 28709
      num_val = 7178
      batch_size = 50
      num_epoch = 25
```

```
[13]: train_datagen = ImageDataGenerator(
      rotation_range = 10,
      width_shift_range = 0.2,
      height_shift_range = 0.2,
      shear_range = 0.2,
      zoom_range = 0.2,
      horizontal_flip = True,
      fill_mode = 'nearest')

      test_datagen = ImageDataGenerator(
      rotation_range = 10,
      width_shift_range = 0.2,
      height_shift_range = 0.2,
      shear_range = 0.2,
      zoom_range = 0.2,
      horizontal_flip = True,
      fill_mode = 'nearest')
```

Figure 5: Image Data Augmentation

- CNN model is defined and compiled.
- Model is trained with 25 Epoch and 50 batch sizes.
- Model is saved in “model/cnn_model.h5”
- Accuracy of 52% is achieved
- Accuracy plot is plotted.



```
checkpoint = ModelCheckpoint("model/face_cnn_model.h5", monitor='val_accuracy',save_weights_only=True, mode='max', verbose=1)
Estop = EarlyStopping(monitor = 'val_loss',patience = 8,verbose = 1,min_delta = 0.001)
hist=model.fit_generator(train_generator,epochs=num_epoch,steps_per_epoch=num_train // batch_size ,validation_data=test_generator,verbose=1,validation_steps=nu

WARNING:tensorflow:From C:\Users\ROG G14\anaconda3\envs\jupyterlab-debugger\lib\site-packages\tensorflow_core\python\ops\math_grad.py:1424: where (from tensorf
w.python.ops.array_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From C:\Users\ROG G14\anaconda3\envs\jupyterlab-debugger\lib\site-packages\keras\backend\tensorflow_backend.py:986: The name tf.assign_add i
deprecated. Please use tf.compat.v1.assign_add instead.

WARNING:tensorflow:From C:\Users\ROG G14\anaconda3\envs\jupyterlab-debugger\lib\site-packages\keras\backend\tensorflow_backend.py:973: The name tf.assign is de
ecated. Please use tf.compat.v1.assign instead.

Epoch 1/25
574/574 [==============================] - 379s 660ms/step - loss: 2.7579 - acc: 0.2431 - val_loss: 2.0342 - val_acc: 0.2456

Epoch 00001: saving model to model/face_cnn_model.h5
Epoch 2/25
574/574 [==============================] - 387s 675ms/step - loss: 1.9137 - acc: 0.2548 - val_loss: 1.7730 - val_acc: 0.2622

Epoch 00002: saving model to model/face_cnn_model.h5
Epoch 3/25
574/574 [==============================] - 388s 676ms/step - loss: 1.7545 - acc: 0.2736 - val_loss: 1.7998 - val_acc: 0.2462

Epoch 00003: saving model to model/face_cnn_model.h5
Epoch 4/25
574/574 [==============================] - 401s 698ms/step - loss: 1.7354 - acc: 0.2854 - val_loss: 1.7334 - val_acc: 0.2960

Epoch 00004: saving model to model/face_cnn_model.h5
Epoch 5/25
574/574 [==============================] - 393s 684ms/step - loss: 1.6893 - acc: 0.3209 - val_loss: 1.6421 - val_acc: 0.3425
```

Figure 6: Training of CNN Model

6.2 CNN-CapsNet Model Training

- Capsule Layer is defined through several classes
- Defining 2 Conv2d layer as CNN and append it in Capsule Layer
- Split the data into Training and Testing
- Augment the image before the model get its input
- Model is trained and saved in data/cnn-capsnet_w.h5
- Model accuracy is 85% with 5 epoch
- Plotting is done after the model trained.

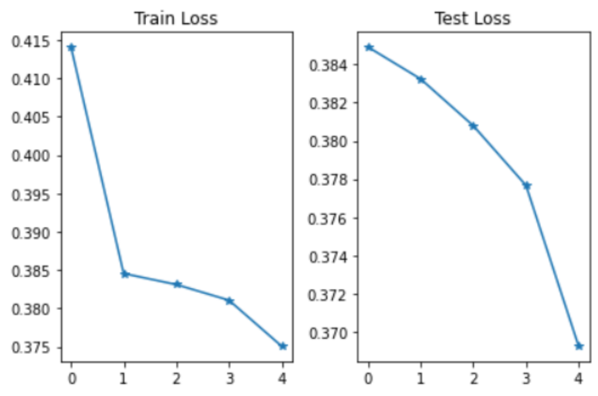
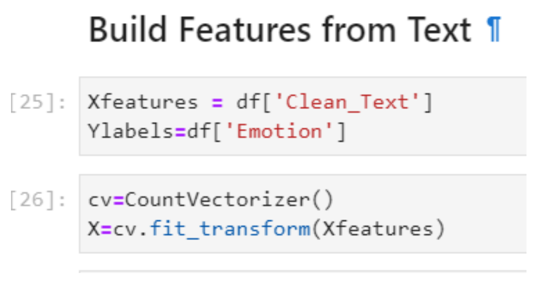


Figure 7: Loss Accuracy Plot for Train and Test Data

6.3 Naive Bayes

- Data imported and Exploration is done
- Data cleaning, Sentiment Analysis, Keyword Extraction and Word Cloud
- Build the feature from Test

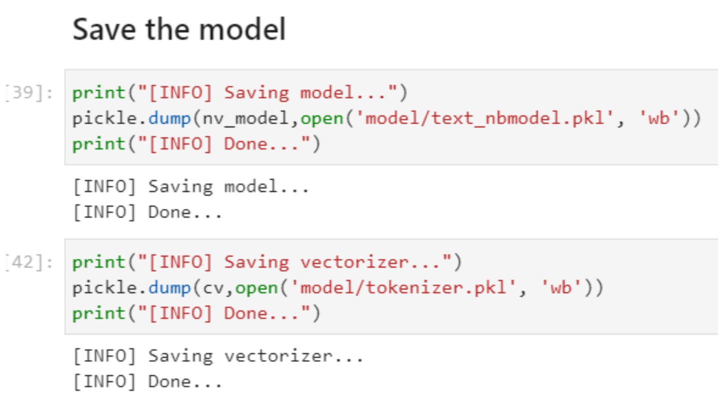


Build Features from Text ¶

```
[25]: Xfeatures = df['Clean_Text']
      Ylabels=df['Emotion']

[26]: cv=CountVectorizer()
      X=cv.fit_transform(Xfeatures)
```

Figure 8: Build Feature from Text

- Build the model and Evaluate
- Model has accuracy 59%
- Save the model and tokenizer



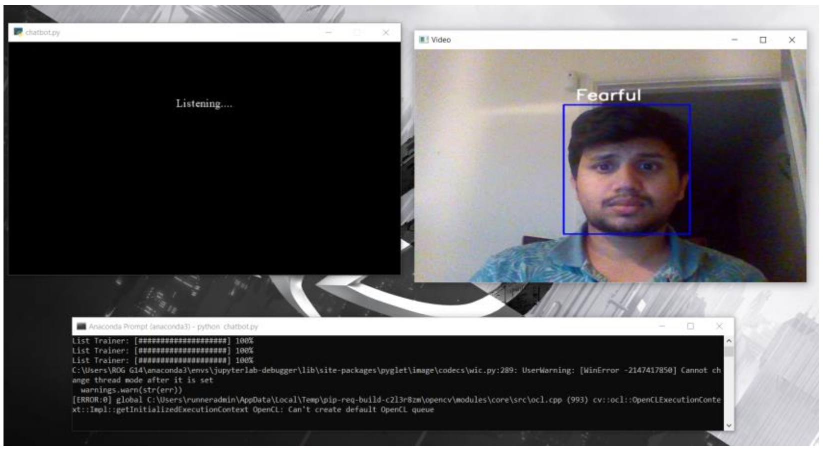
Save the model

```
[39]: print("[INFO] Saving model...")
      pickle.dump(nv_model,open('model/text_nbmodel.pkl', 'wb'))
      print("[INFO] Done...")

      [INFO] Saving model...
      [INFO] Done...

[42]: print("[INFO] Saving vectorizer...")
      pickle.dump(cv,open('model/tokenizer.pkl', 'wb'))
      print("[INFO] Done...")

      [INFO] Saving vectorizer...
      [INFO] Done...
```

Figure 9: Saving the model and vectorizer

7 Audio Chat Bot

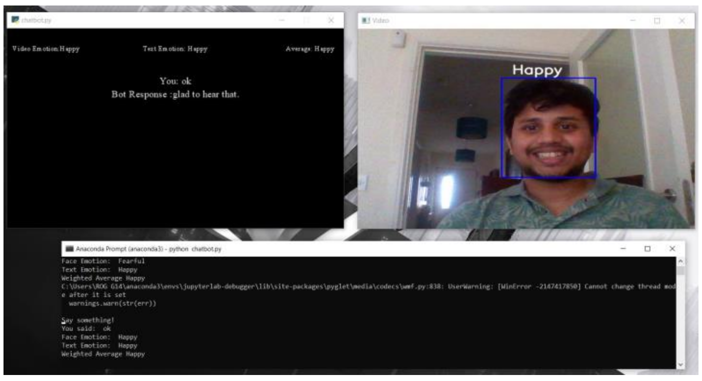
- Go to demo folder in the artifacts
- There will many python files for the execution of the chatbot
    - Botcontroller.py – has the chatter library for chatbot
    - Chatbot.py – main file to run
    - Speech_to_text.py – speech convert to text
    - user_interface.py – pyglet UI design
- Open “cmd” with admin
- Install all the required python package listed before
- Go to the artifact folder path and “cd demo”
- Now execute the command “python chatbot.py”
- It automatically takes the model from the model folder and start executing
- Now you will be seeing three windows i.e cmd prompt, UI window, OpenCV camera Window.
- Arrange the UI, camera window side by side to view it properly this Figure 10



```
List Trainer: [####################] 100%
List Trainer: [####################] 100%
List Trainer: [####################] 100%
C:\Users\ROG G14\anaconda3\envs\jupyterlab-debugger\lib\site-packages\pyglet\image\codecs\wic.py:289: UserWarning: [WinError -2147417850] Cannot ch
ange thread mode after it is set
  warnings.warn(str(err))
[ERROR:0] global C:\Users\runneradmin\AppData\Local\Temp\pip-req-build-c2l3r8zm\opencv\modules\core\src\ocl.cpp (993) cv::ocl::OpenCLExecutionConte
xt::Impl::getInitializedExecutionContext OpenCL: Can't create default OpenCL queue
```

Figure 10: Staring of Audio Chat Bot

- Now to speak with chatbot, you have to click ENTER key by selecting the cmd prompt. (ENTER key would trigger the mic to listen the user voice and take it as a input)
- If u want close the you say “BYE”, chatbot will be closed.
- Once the input is taken you might see the result in the UI screen like Figure 11.


chatbot.py
Video Emotion:Happy
Text Emotion: Happy
Average: Happy
You: ok
Bot Response :glad to hear that.
Video
Happy
Anaconda Prompt (anaconda3) - python chatbot.py

```
Face Emotion:  Fearful
Text Emotion:  Happy
Weighted Average Happy
C:\Users\ROG G14\anaconda3\envs\jupyterlab-debugger\lib\site-packages\pyglet\media\codecs\wmf.py:838: UserWarning: [WinError -2147417850] Cannot change thread mod
e after it is set
  warnings.warn(str(err))

Say something!
You said:  ok
Face Emotion:  Happy
Text Emotion:  Happy
Weighted Average Happy
```

Figure 11: User Interface of the Audio Bot