User Emotion Detection for News Article Impact Analysis

Group – 9

Mohit Mendi – 100800234
Ashika R Ezhuva - 100761360

Date

04 – April – 2021

Course title

AI in Enterprise Analytics

OVERVIEW

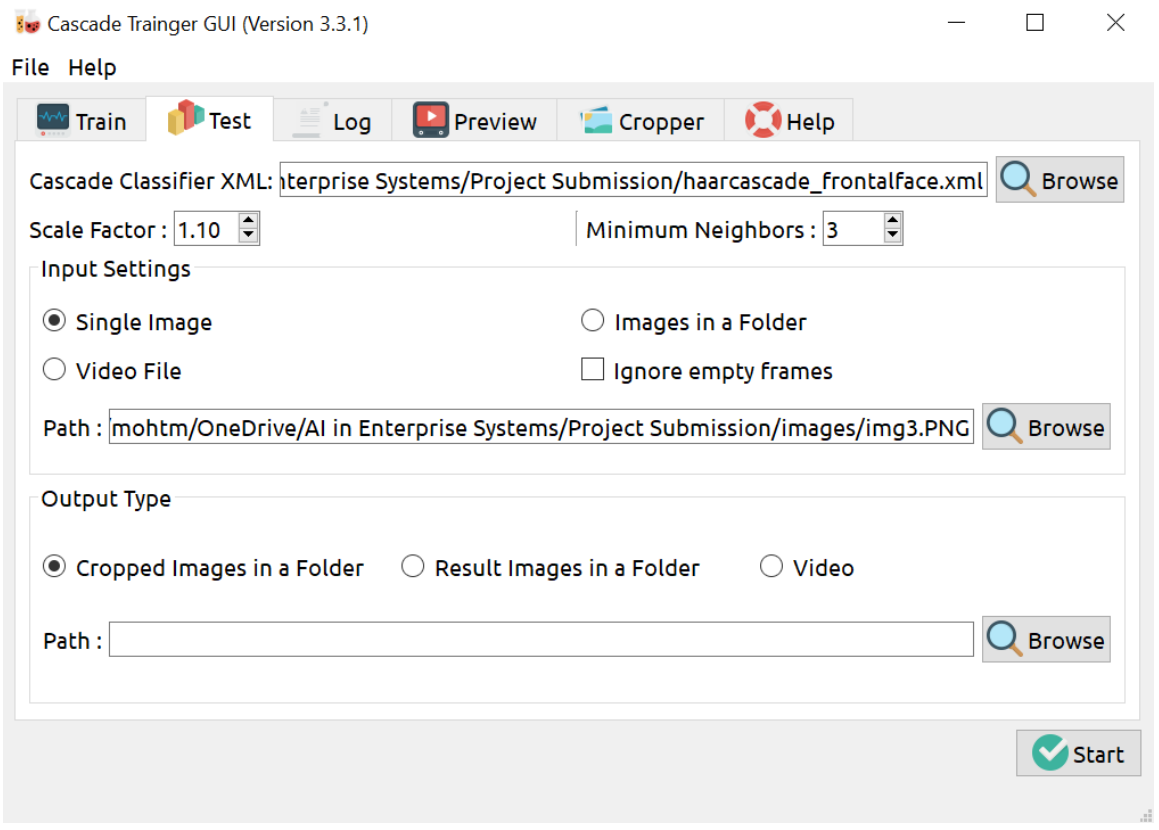
The objective of the project is to identify the sentiment behind the context of a news article and compare that to the facial expression in the same article. This is done using multiple training and testing datasets which consists to various images and their emotions. We have used Spacy model to perform sentiment from text.

OBJECTIVES

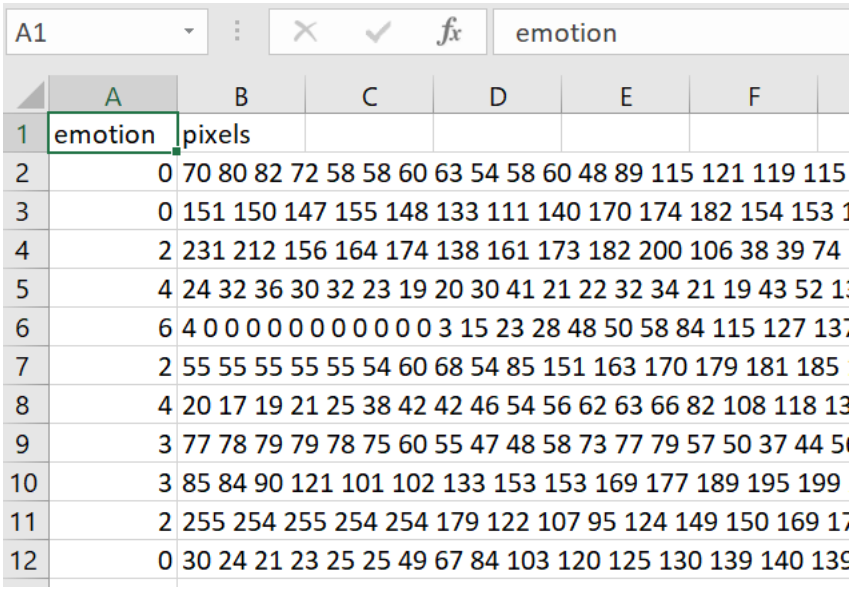
There is total 5 objectives for the project listed below,

- To train the Haar Model for Facial Area Detection.
- Facial Sentiment Analysis using CNN Model for the picture in the article.
- Using Tesseract OCR (Optical Character Recognition) in the associate text with the article.
- Using Spacy(spaCy) Model for text sentiment analysis.
- Finally comparing the output of the image and the text to see if they both matches.

Train the Haar Model for Facial Area Detection.



The Haar Model has been trained using the training data set in Cascade Trainger GUI tool.



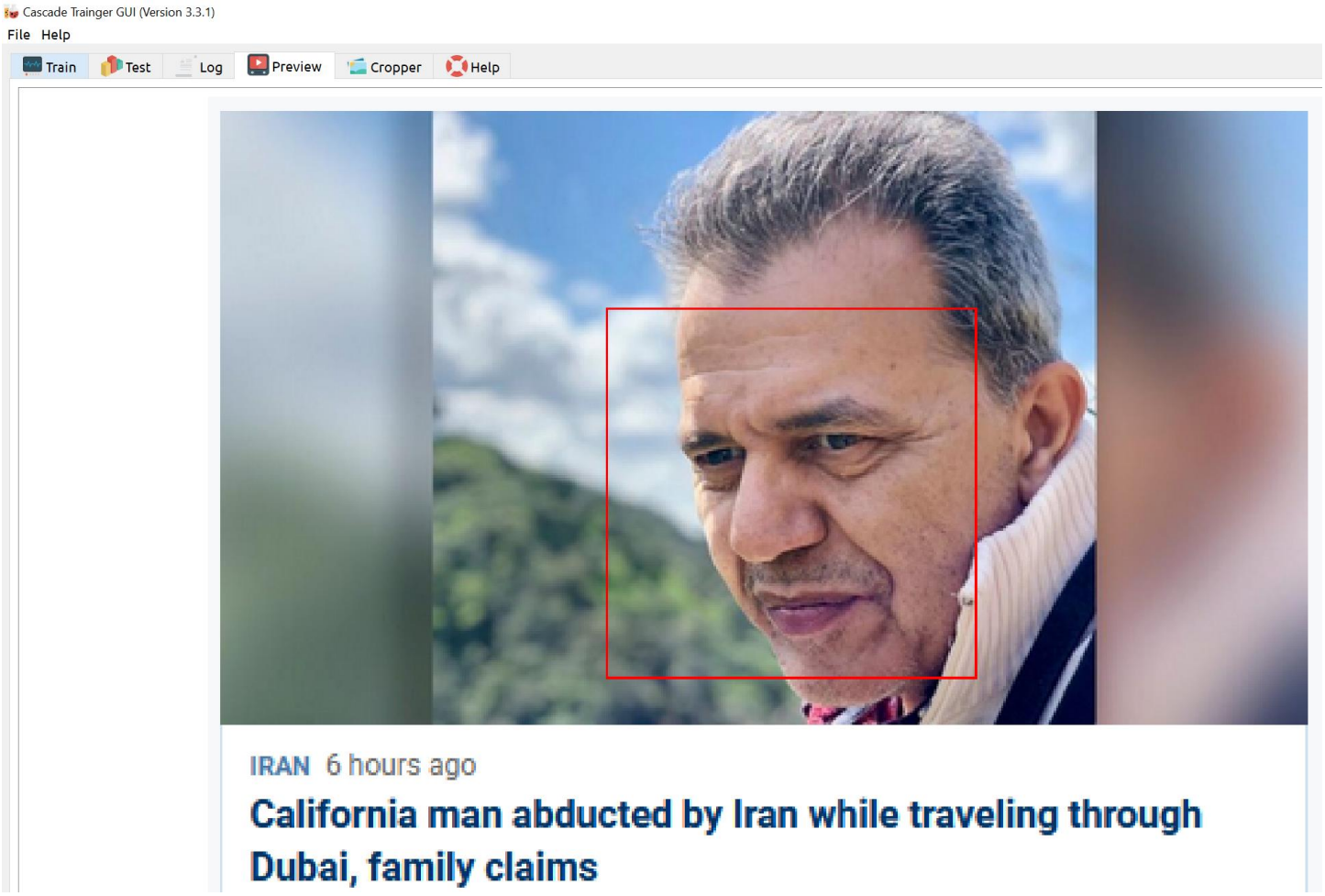
A1 | emotion

	A	B	C	D	E	F
1	emotion	pixels				
2	0	70 80 82 72 58 58 60 63 54 58 60 48 89 115 121 119 115				
3	0	151 150 147 155 148 133 111 140 170 174 182 154 153 1				
4	2	231 212 156 164 174 138 161 173 182 200 106 38 39 74				
5	4	24 32 36 30 32 23 19 20 30 41 21 22 32 34 21 19 43 52 1				
6	6	4 0 0 0 0 0 0 0 0 0 0 0 3 15 23 28 48 50 58 84 115 127 13				
7	2	55 55 55 55 55 54 60 68 54 85 151 163 170 179 181 185				
8	4	20 17 19 21 25 38 42 42 46 54 56 62 63 66 82 108 118 13				
9	3	77 78 79 79 78 75 60 55 47 48 58 73 77 79 57 50 37 44 5				
10	3	85 84 90 121 101 102 133 153 153 169 177 189 195 199				
11	2	255 254 255 254 254 179 122 107 95 124 149 150 169 17				
12	0	30 24 21 23 25 25 49 67 84 103 120 125 130 139 140 139				

As we can observe, the train dataset consists of 2 columns, namely 'emotion' and 'pixels'. 'Emotion' consists of the category of which the certain picture falls under and the 'pixels' consists of individual pixels of each picture that is mentioned in the training dataset. This helps the Sentimental Analysis to work very fast.

Testing in Cascade Trainger GUI:

The testing is done by selecting the particular trained model and uploading a image into it from the tool.



As you can see the model is able to identify the face very precisely in the testing phase.

Facial Sentimental Analysis

The Facial Sentimental Analysis is done by training the model with the trained and validation dataset that has been mentioned above. This is performed on Python. This has the accuracy of about 60-70% currently. The Haar Model is used and implemented into the CNN Model to perform the facial sentimental analysis much precisely and quickly. Haar is used by the CNN Model to find the region of the detection quickly.

The Training CNN Model code has been mentioned below.

The necessary packages such as Keras, Sequential, Convolutional etc. The classes using that we mentioned is 5 and the image size is 48 * 48 and batch size is 32. The training and validation data set path has been mentioned in the code as well. The rest of the code is for augmenting the image for the input.

Train_Model.py

C: > Users > mohtm > OneDrive > AI in Enterprise Systems > Project Submission > Train_Model.py > ...

```python
from __future__ import print_function
import keras
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dense,Dropout,Activation,Flatten,BatchNormalization
from keras.layers import Conv2D,MaxPooling2D
import os

num_classes = 5
img_rows,img_cols = 48,48
batch_size = 32

train_data_dir = '/Emotion Classification/train'
validation_data_dir = '/Emotion Classification/validation'

train_datagen = ImageDataGenerator(
                    rescale=1./255,
                    rotation_range=30,
                    shear_range=0.3,
                    zoom_range=0.3,
                    width_shift_range=0.4,
                    height_shift_range=0.4,
                    horizontal_flip=True,
                    fill_mode='nearest')

validation_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
                    train_data_dir,
                    color_mode='grayscale',
                    target_size=(img_rows,img_cols),
                    batch_size=batch_size,
                    class_mode='categorical',
                    shuffle=True)

validation_generator = validation_datagen.flow_from_directory(
                            validation_data_dir,
                            color_mode='grayscale',
                            target_size=(img_rows,img_cols),
                            batch_size=batch_size,
                            class_mode='categorical',
                            shuffle=True)
```

Train_Model.py

C: > Users > mohtm > OneDrive > AI in Enterprise Systems > Project Submission > Train_Model.py > ...

```
46
47  # Block-1
48
49  model.add(Conv2D(32,(3,3),padding='same',kernel_initializer='he_normal',input_s
50  model.add(Activation('elu'))
51  model.add(BatchNormalization())
52  model.add(Conv2D(32,(3,3),padding='same',kernel_initializer='he_normal',input_s
53  model.add(Activation('elu'))
54  model.add(BatchNormalization())
55  model.add(MaxPooling2D(pool_size=(2,2)))
56  model.add(Dropout(0.2))
57
58  # Block-2
59
60  model.add(Conv2D(64,(3,3),padding='same',kernel_initializer='he_normal'))
61  model.add(Activation('elu'))
62  model.add(BatchNormalization())
63  model.add(Conv2D(64,(3,3),padding='same',kernel_initializer='he_normal'))
64  model.add(Activation('elu'))
65  model.add(BatchNormalization())
66  model.add(MaxPooling2D(pool_size=(2,2)))
67  model.add(Dropout(0.2))
68
69  # Block-3
70
71  model.add(Conv2D(128,(3,3),padding='same',kernel_initializer='he_normal'))
72  model.add(Activation('elu'))
73  model.add(BatchNormalization())
74  model.add(Conv2D(128,(3,3),padding='same',kernel_initializer='he_normal'))
75  model.add(Activation('elu'))
76  model.add(BatchNormalization())
77  model.add(MaxPooling2D(pool_size=(2,2)))
78  model.add(Dropout(0.2))
79
80  # Block-4
81
82  model.add(Conv2D(256,(3,3),padding='same',kernel_initializer='he_normal'))
83  model.add(Activation('elu'))
84  model.add(BatchNormalization())
85  model.add(Conv2D(256,(3,3),padding='same',kernel_initializer='he_normal'))
86  model.add(Activation('elu'))
87  model.add(BatchNormalization())
88  model.add(MaxPooling2D(pool_size=(2,2)))
89  model.add(Dropout(0.2))
```

The above code is the basic structure for the CNN Model. This consists of convolutional (Conv2D), padding, kernel. We are using 'elu' and 'softmax' as the activation functions. The model as been trained in the below code. We have initiated an early stopping point; it would stop is training and testing accuracy is heading far from each other.

```
# Block-5

model.add(Flatten())
model.add(Dense(64,kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))

# Block-6

model.add(Dense(64,kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))

# Block-7

model.add(Dense(num_classes,kernel_initializer='he_normal'))
model.add(Activation('softmax'))

print(model.summary())

from keras.optimizers import RMSprop,SGD,Adam
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau

checkpoint = ModelCheckpoint('Emotion_little_vgg.h5',
                             monitor='val_loss',
                             mode='min',
                             save_best_only=True,
                             verbose=1)

earlystop = EarlyStopping(monitor='val_loss',
                          min_delta=0,
                          patience=3,
                          verbose=1,
                          restore_best_weights=True
                          )
```

Tesseract OCR (Optical Character Recognition)

We have installed tesseract tool on the windows and used pytesseract in Python to perform the OCR on the images that are given as an input. The example can be shown below:

Example 1

```
[6]: img_path = r"images\img1.PNG"
     img = cv2.imread(img_path)
     plt.imshow(img)
```

```
[6]: <matplotlib.image.AxesImage at 0x2131cdd06a0>
```

Reading text from image

```
[7]: text = pytesseract.image_to_string(img)
     text
```

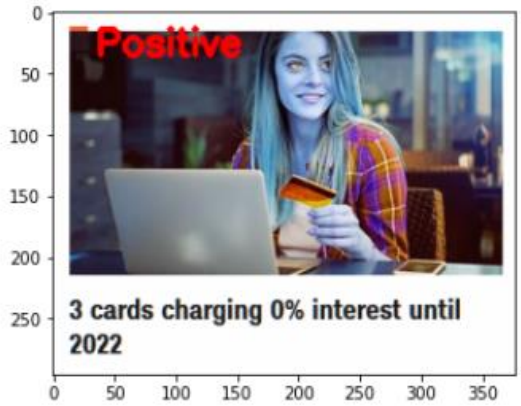
```
[7]: '3 cards charging 0% interest until\n2022\n\n \n\x0c'
```

As you can see the text from the image is same as the text that has been retrieved by the tesseract.

Spacy (spaCy) Model for text sentimental analysis

The spacy model has been used to perform the text analysis in the example below.

```
Image_Sentiment(img_path)
```

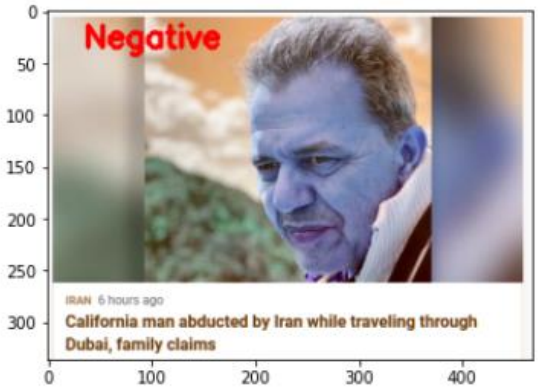


Text Sentiment

```
Text_Sentiments(text)
```

```
Positive
```

```
Image_Sentiment(img_path)
```



Text Sentiment

```
Text_Sentiments(text)
```

```
Negative
```

As you can see, the text analysis is performed in the text that is been retrieved from the OCR and stored in the variable 'text'.

Comparing the output of image and text from a news article.

Example 1:

```
Image_Sentiment(img_path)
```

Text Sentiment

```
Text_Sentiments(text)
```

```
Positive
```

In this example, we can observe that the output given on the image is 'Positive' and the output from the analysis of the text is given as 'Positive' too. So, the emotion of the image is equal to the emotion in the text below it.

Example 2:

```
: Image_Sentiment(img_path)
```

Text Sentiment

```
: Text_Sentiments(text3)
```

```
Negative
```

In this example, we can observe that the output given on the image is 'Positive' and the output from the analysis of the text is given as 'Negative'. So, the emotion of the image is not equal to the emotion in the text below it.