Capstone Project - 5

Face Emotion Recognition

Jyoti Chiluka

Content :

1. Defining problem statement
2. Data Description
3. Exploratory Data Analysis (EDA)
4. Dependencies
5. Creating model
6. Fitting the model with training and validation data
7. Testing the model
8. Detected images
9. Conclusion

The Dilemma :

The Indian education landscape has been undergoing rapid changes for the past 10 years owing to the advancement of web-based learning services, specifically, eLearning platforms.

Global E-learning is estimated to witness an 8X over the next 5 years to reach USD 2B in 2021. India is expected to grow with a CAGR of 44% crossing the 10M users mark in 2021. Although the market is growing on a rapid scale, there are major challenges associated with digital learning when compared with brick and mortar classrooms. One of many challenges is how to ensure quality learning for students. Digital platforms might overpower physical classrooms in terms of content quality but when it comes to understanding whether students are able to grasp the content in a live class scenario is yet an open-end challenge.

The Dilemma contd..

In a physical classroom during a lecturing teacher can see the faces and assess the emotion of the class and tune their lecture accordingly, whether he is going fast or slow. He can identify students who need special attention. It provides data in the form of video, audio, and texts which can be analyzed using deep learning algorithms. Deep learning backed system not only solves the surveillance issue, but it also removes the human bias from the system, and all information is no longer in the teacher's brain rather translated in numbers that can be analyzed and tracked.

We will solve the above-mentioned challenge by applying deep learning algorithms to live video data. The solution to this problem is by recognizing facial emotions. The model should be able to real-time identify the emotions of students in a live class.

Data Description (Fer 2013)

Dataset Fer 2013 taken form Kaggle. The dataset consists of 48*48 pixel grayscale images of faces. The faces have been categorized into facial expressions into one of six categories (0=Angry, 1=Fear, 2=Sad, 3=Happy, 4=Surprise, 5=Neutral)

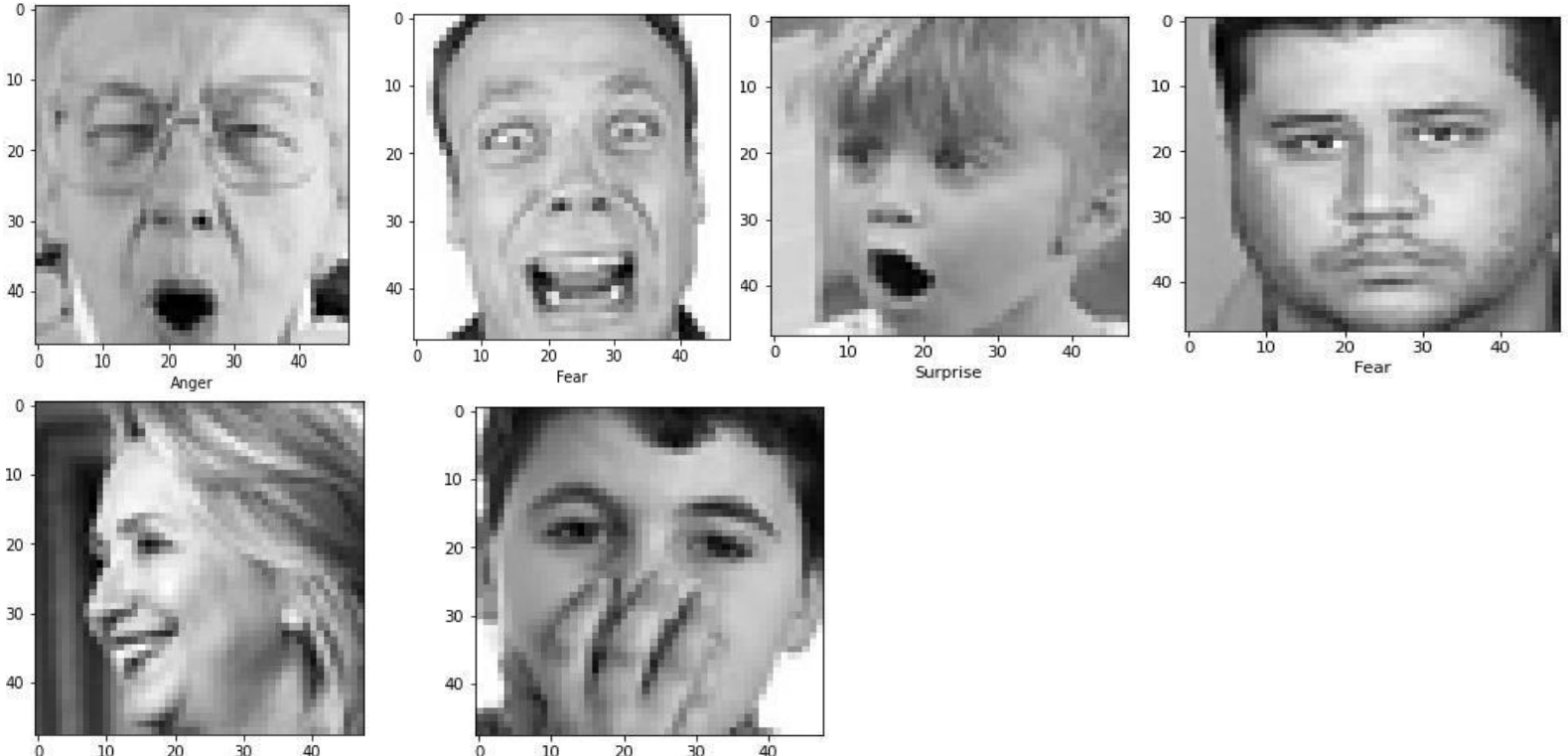
Data pre-processing for modeling

The fer2013.csv consists of three different columns namely emotion, pixels and usage and this dataset have 6 different emotions and images in pixels form, converted pixels into images. The dataset contains 28,709 images for training .

	emotion	pixels	Usage
0	0	70 80 82 72 58 58 60 63 54 58 60 48 89 115 121...	Training
1	0	151 150 147 155 148 133 111 140 170 174 182 15...	Training
2	2	231 212 156 164 174 138 161 173 182 200 106 38...	Training
3	4	24 32 36 30 32 23 19 20 30 41 21 22 32 34 21 1...	Training
4	6	4 0 0 0 0 0 0 0 0 0 0 0 0 3 15 23 28 48 50 58 84...	Training

EDA

Plotted images with their corresponding emotions:

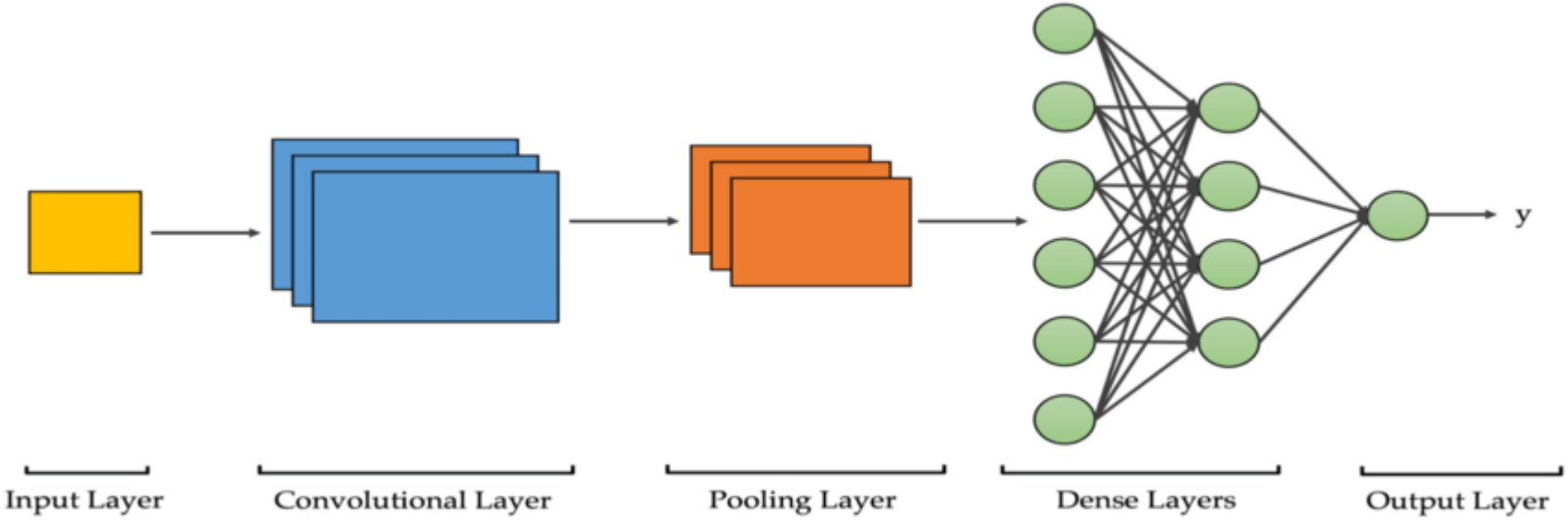


Dependencies

1. Python 3
2. Tensorflow 2.0
3. Stream-lit
4. Streamlit-Webrtc
5. OpenCV

Using CNN model

What is CNN?



Building the CNN model :

```
model = Sequential()


input_shape = (48,48,1)


model.add(Conv2D(64, (5, 5), input_shape=input_shape,activation='relu', padding='same'))
model.add(Conv2D(64, (5, 5), padding='same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))


model.add(Conv2D(128, (5, 5),activation='relu',padding='same'))
model.add(Conv2D(128, (5, 5),padding='same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(256, (3, 3),activation='relu',padding='same'))
model.add(Conv2D(256, (3, 3),activation='relu',padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))

## (15, 15) --->  30
model.add(Flatten())
model.add(Dense(6, activation='softmax'))

model.compile(loss='categorical_crossentropy', metrics=['accuracy'],optimizer='adam')
```

Batch normalization : It reduces the amount by what the hidden unit values shift around (covariance shift).

Max pooling: in this case, a kernel of size n*n is moved across the matrix and for each position the max value is taken and put in the corresponding position of the output matrix.

Total params: 1,661,126
Trainable params: 1,660,230
Non-trainable params: 896

Loss function, Optimizer

- Loss function used is categorical cross entropy.
- Cross entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1.
- Optimizer used is the Adam optimizer.
- Adam is the method that computes adaptive learning rates for each parameter.

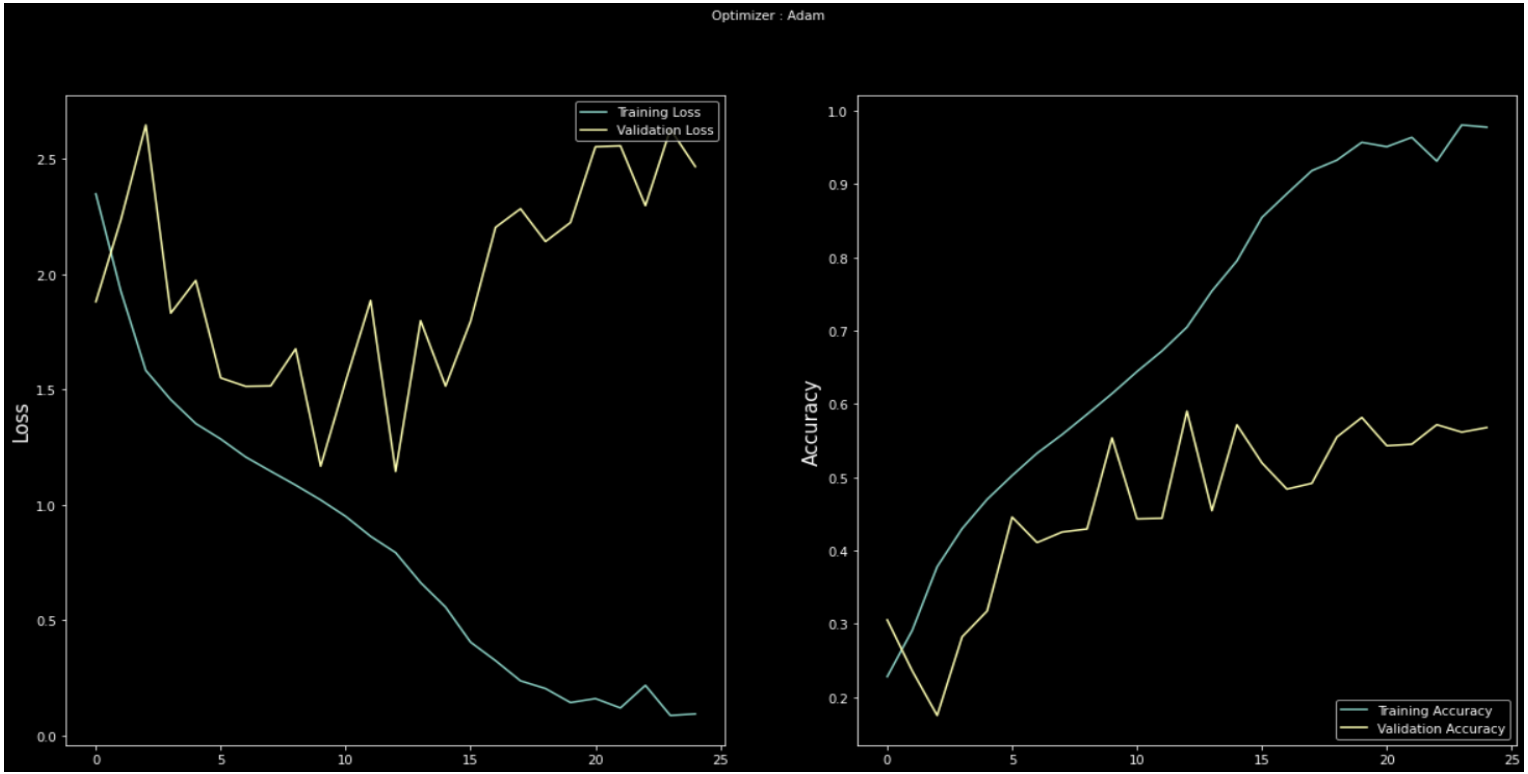
```
model.compile(loss='categorical_crossentropy', metrics=['accuracy'],optimizer='adam')
```

Fitting the model with Training and Validation data

In this model
we are using Adam optimizer.
The epochs used in the model are 25
batch size of 64
shuffling =True,
Validation split =0.2

```
Epoch 10/25
301/301 [==============================] - 25s 84ms/step - loss: 1.0210 - accuracy: 0.6139 - val_loss: 1.1671 - val_accuracy: 0.5535
```

EDA(plotting loss and accuracy)



Testing the model

Testing the model by uploading images

```
#Surprise face predicting
model.predict(test_img)
```

```
array([[0., 0., 0., 0., 1., 0.]], dtype=float32)
```

label_map = ['Anger', 'Fear', 'Sad', 'Happy', 'Surprise', 'Neutral']

Actual image is suprise and prediction is also surprise

```
#Neutral face predicting
model.predict(test_img2)
```

```
array([[0., 0., 0., 0., 0., 1.]], dtype=float32)
```

label_map = ['Anger', 'Fear', 'Sad', 'Happy', 'Surprise', 'Neutral']

Actual image is neutral and prediction is also neutral

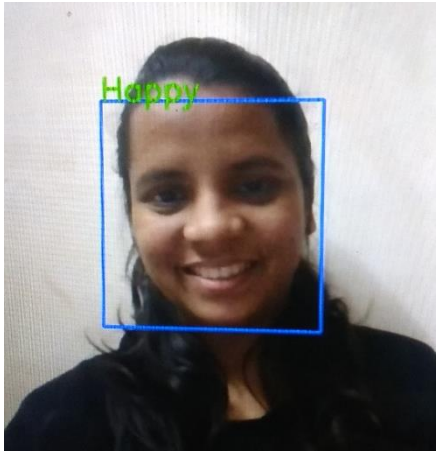
Testing the model contd..

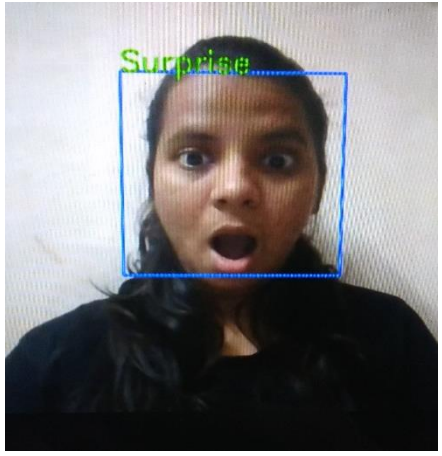
1. Testing the model in Real-time using OpenCV and WebCam
2. Tested the model that we build for emotion detection in real-time using OpenCV and webcam.
3. Saved the model weights in json file and h5 file.
4. After importing the model weights we have imported a haar cascade file that is designed by open cv to detect the frontal face.
5. The image frame is converted into grayscale and resized and reshaped with the help of NumPy. Detect faces and classify the desired emotions.
6. Labels are assigned with different emotions like angry, fear, sad, happy, surprise, neutral.

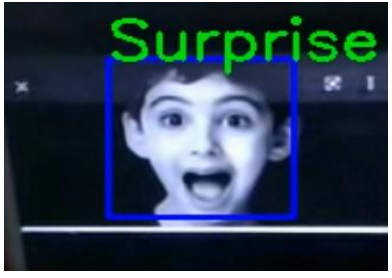
Testing the model contd…

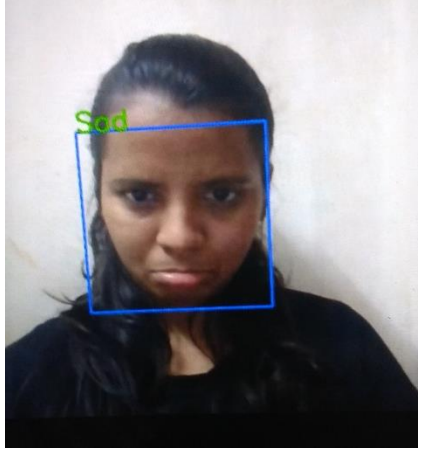
- After running the code a new window will pop up with web application then we have to start the web cam.
- It will then detect the face of the person, draw a bounding box over the detected person, and then convert the RGB image into grayscale & classify it in real-time.
- Deployed it on Heroku and stream-lit platform using Streamlit-Webrtc for the front end application.

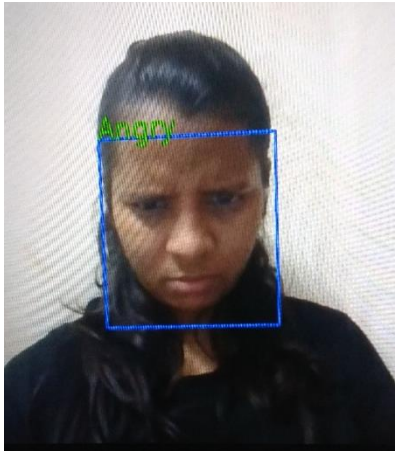
Detected images :











Challenges:

1. Jupyter notebook was not able to connect to the GPU.
2. Tired of using different models, finally found the best one.
3. Continuous Runtime and Ram Crash due to large dataset.
4. Deploying project at Heroku platform (TensorFlow version is not supporting most of the times but successfully deployed)
5. Hyperparameters are tuned carefully

Conclusion:

- Trained the model using Convolutional Neural Network (CNN) we just added layers with a channels and padding requirement in a sequential model just by calling add method.
- Haarcasacade is the package used from OpenCV to detect objects in other images. Initially the video frame is stored in the video object.
- The epochs as 25 and the optimum score is at 10^{th} epoch.
- Trained the model with several images and then used the test images to see how the results match up.
- The accuracy that achieved for the validation set is 56% for further increase the accuracy of the model, we can either expand the training dataset or we have to increase the batch size for the model. Through these parameters, we can increase the model accuracy.
- Model is identifying students emotions using minimum reference images and Successfully deployed web app of real-time webcam video feed on Heroku and streamlit platform.

Q & A