Capstone Project - 5

Face Emotion Recognition

Navin kodam

Content :

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.

Dataset (FER 2013)

This model is capable of recognizing six basic emotions as following:

1. **Happy**
2. **Sad**
3. **Angry**
4. **Surprise**
5. **Fear**
6. **Neutral**

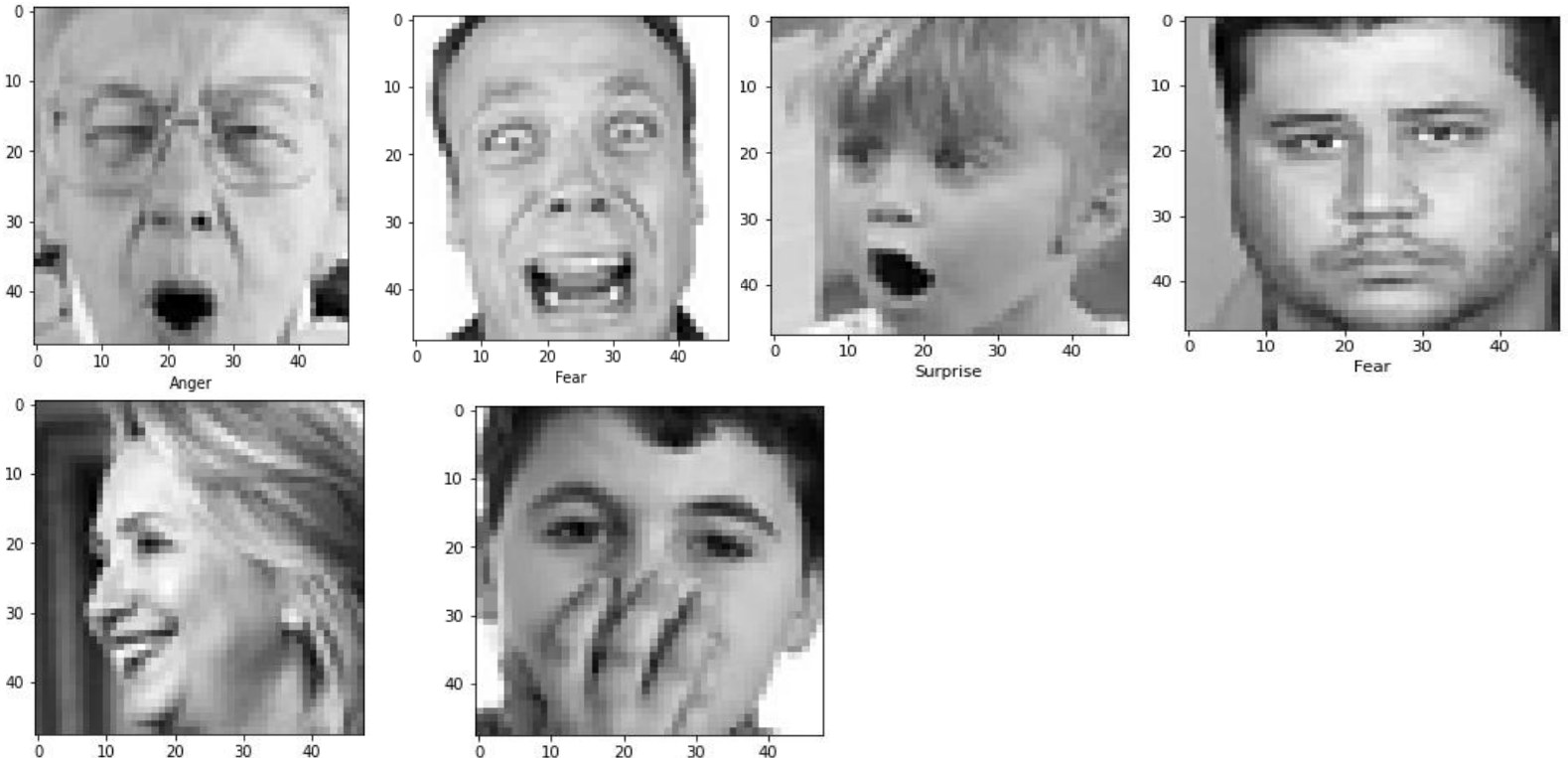
Preparing dataset for modeling

FER 2013 dataset have 6 different
Emotions and images in pixels
Form, converted pixels into images
And the dataset contains
28,709 images for training and
7,178 images for test purpose

	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 :

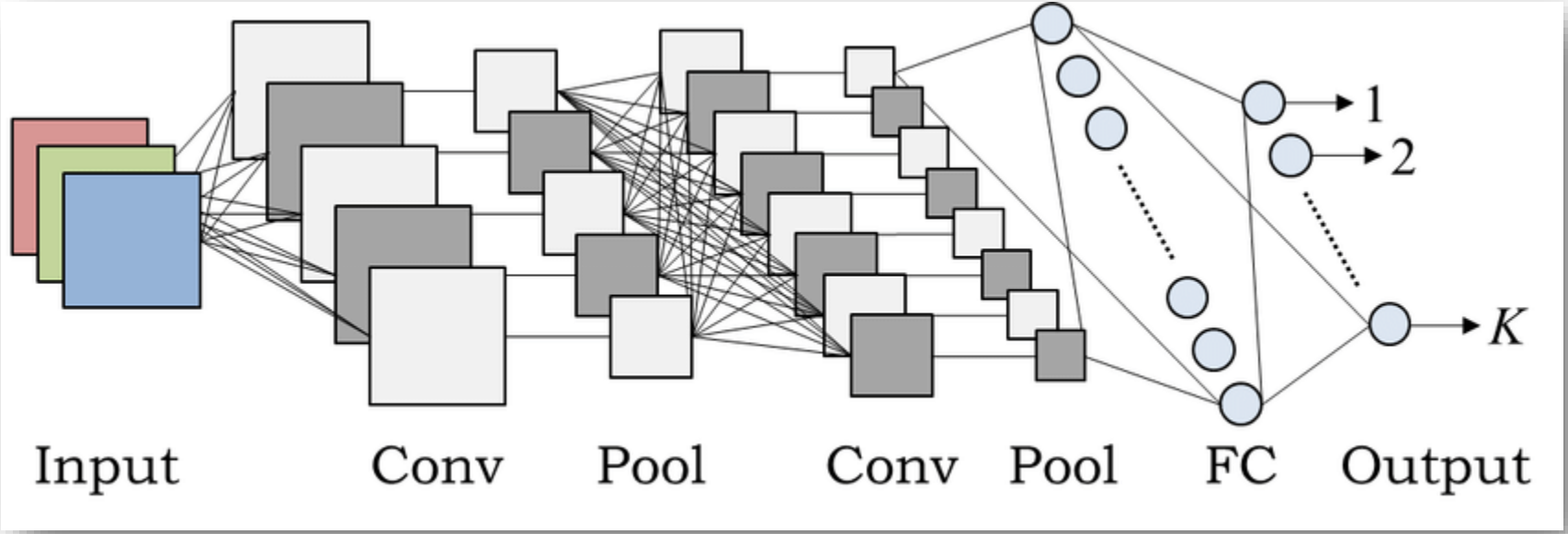


Dependencies

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 :

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

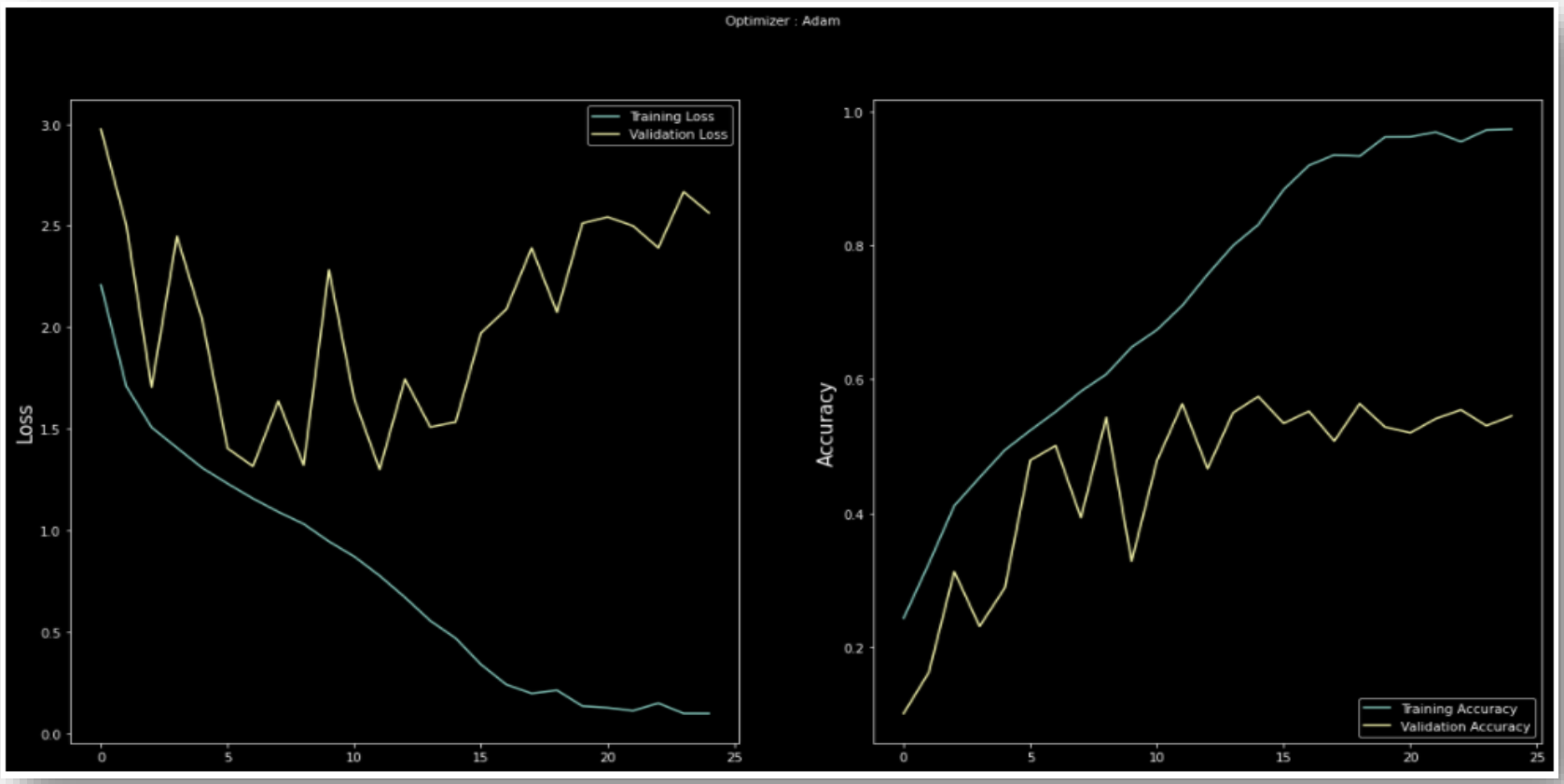
```
model = Sequential()


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')
```

Fitting the model with Training and Validation

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 15/25
301/301 [==============================] - 31s 103ms/step - loss: 0.4670 - accuracy: 0.8307 - val_loss: 1.5320 - val_accuracy: 0.5741
```

EDA(plotting loss and accuracy)



Testing the model contd... :

Testing the model by uploading images

Actual image is angry and prediction is also angry

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

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

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

Actual image is happy and prediction image is also happy

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

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

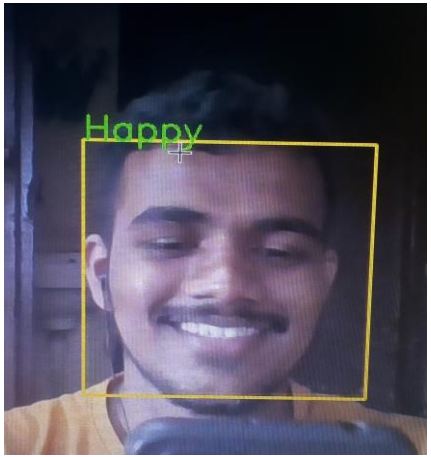
Testing the model contd... :

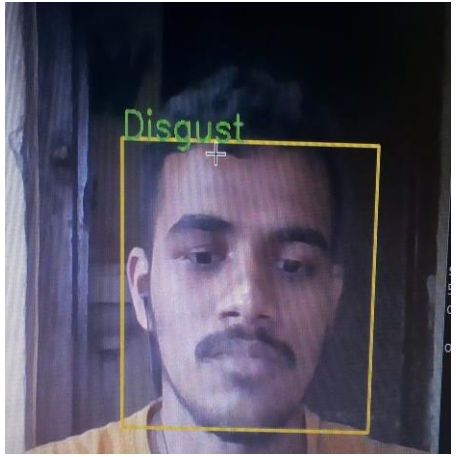
1. Testing the model in Real-time using OpenCV and WebCam
2. Now we will test the model that we build for emotion detection in real-time using OpenCV and webcam.
3. We will save the model weights in json file and h5 file
4. After importing the model weights we have imported a haar cascade file that is designed by open cv to detect the frontal face.
5. After importing the haar cascade file we will have written a code to detect faces and classify the desired emotions
6. We have assigned the labels that will be different emotions like angry, happy, sad, surprise, neutral

Testing the model contd...

1. As soon as you run the code a new window will pop up and your webcam will turn on.
2. 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
3. Deployed it on stream-lit platform using Streamlit-Webrtc for the front end application

Detected images :





Challenges:

1. Large image dataset to handle
2. Couldn’t able to connect GPU with Jupyter notebook
3. Tired of using different models, finally found the best one
4. Continuous Runtime and Ram Crash due to large dataset
5. Deploying project at Heroku platform (tensorflow version is not supporting that’s why deployed on stream lit and taken a video for reference purpose)
6. Carefully tuned hyperparameters

Conclusion:

- I conclude this project by hoping that you got a fair idea and understood the whole pipeline on how you can make an emotion detection model
- We trained our 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.
- we also used Computer Vision as part of this model. Haarcascade is the package used from OpenCV to detect objects in other images.
- We trained the model with several images and then used the test images to see how the results match up. and we have taken epochs as 25 and we got the optimum score at 15th epoch
- For this model, the accuracy that we achieved for the validation set is 57%. To further increase the accuracy of the model, we can either expand the training dataset we have or increase the batch size for the model. Through these parameters, we can increase the model accuracy for this model.
- Model is identifying students emotions using minimum reference images and Successfully deployed web app of real-time webcam video feed on streamlit platform