Face Emotion Recognition

Sourabh Pramanik

Data science trainee,

AlmaBetter, Bangalore.

Introduction:

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 . But there are some challenges associated with digital learning. In tis project we will try to deal with those challenges and will create a model which will solve the problems related to digital learning.

Problem Statement:

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.

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. Digital classrooms are conducted via video telephony software program (exZoom) where it’s not possible for medium scale class (25-50) to see all students and access the mood. Because of this drawback, students are not focusing on content due to lack of surveillance. While digital platforms have limitations in terms of physical surveillance but it comes with the power of data and machines which can work for you. It provides data in the form of video, audio, and texts which can be analysed 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 analysed and tracked.

Data Description:

Here in this project we are using FER dataset to train our model. In this dataset we have images of some different categories of emotions.

The categories are-

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

Objective of our Project:

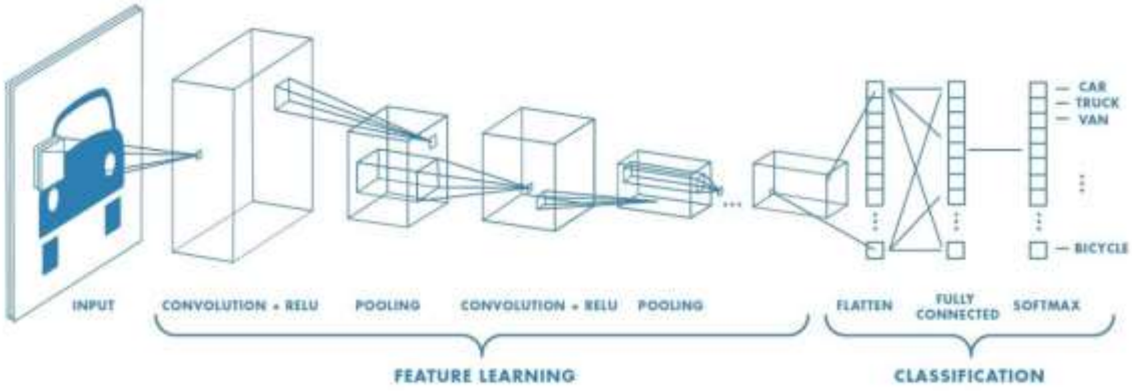
The main objective our project is to build a Face Emotion Recognition model which can detect the Facial emotion of every student. It will help to judge the understanding of students.

Steps Involved:

- After doing the scaling of the input images we will apply Inceptiov3 CNN model. But we will remove the last layer of Inceptionv3 so that we can add a layer according our need.
- We will set the final layer and activation function ‘Softmax’. And will see the layers by using ‘ model.summary() ’.
- Now we will set the optimizer ‘Adam’ and will do Data Augmentation.
- Now by setting some ‘callbacks’ we will train our model and will check accuracy. We will save our model in ‘model.H5’ file.
- Now for Deployment purpose we will Flask web application.
- We will deploy our model in AWS EC2 platform.

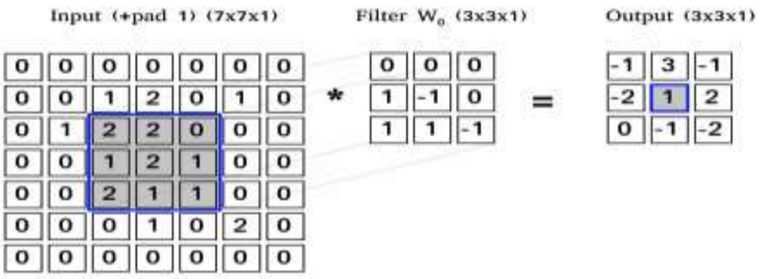
Convolution Neural Network-

A **convolutional neural network (CNN)** is a type of artificial neural network used in image recognition and processing that is specifically designed **to process pixel data**.



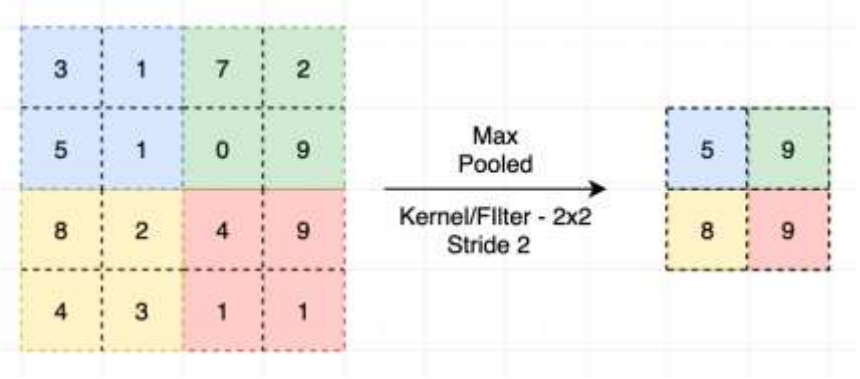
Convolution-

The term convolution refers to **the mathematical combination of two functions to produce a third function**. It merges two sets of information. In the case of a CNN, the convolution is performed on the input data with the use of a filter or kernel (these terms are used interchangeably) to then produce a feature map.



Pooling Layer-

Pooling layers are **used to reduce the dimensions of the feature maps**. Thus, it reduces the number of parameters to learn and the amount of computation performed in the network. The pooling layer summarises the features present in a region of the feature map generated by a convolution layer



Activation Function-

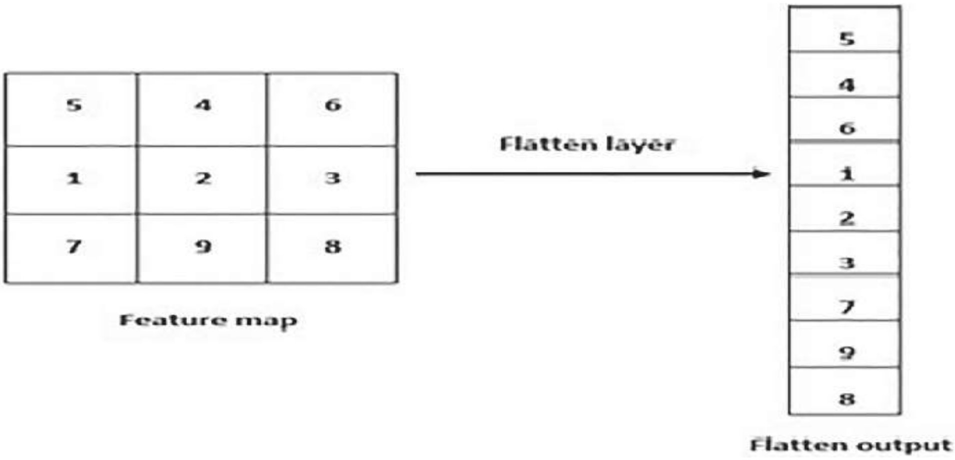
An Activation Function **decides whether a neuron should be activated or not**. This means that it will decide whether the neuron's input to the network is important or not in the process of prediction using simpler mathematical operations. Here are some activation functions-

- Sigmoid
- Tanh

- ReLU
- Leaky ReLU
- Softmax

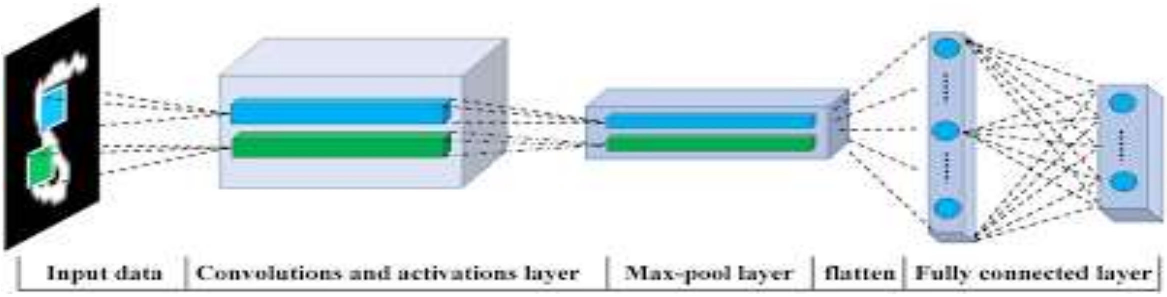
Flatten Layer-

Flattening is converting the data into a **1-dimensional array** for inputting it to the next layer. We flatten the output of the convolutional layers to create a single long feature vector. And it is connected to the final classification model, which is called a fully-connected layer.



Fully Connected Layer-

Fully Connected Layer is simply, **feed forward neural networks**. Fully Connected Layers form the last few layers in the network. The input to the fully connected layer is the output from the final Pooling or Convolutional Layer, which is flattened and then fed into the fully connected layer.



ImageNet Competition-

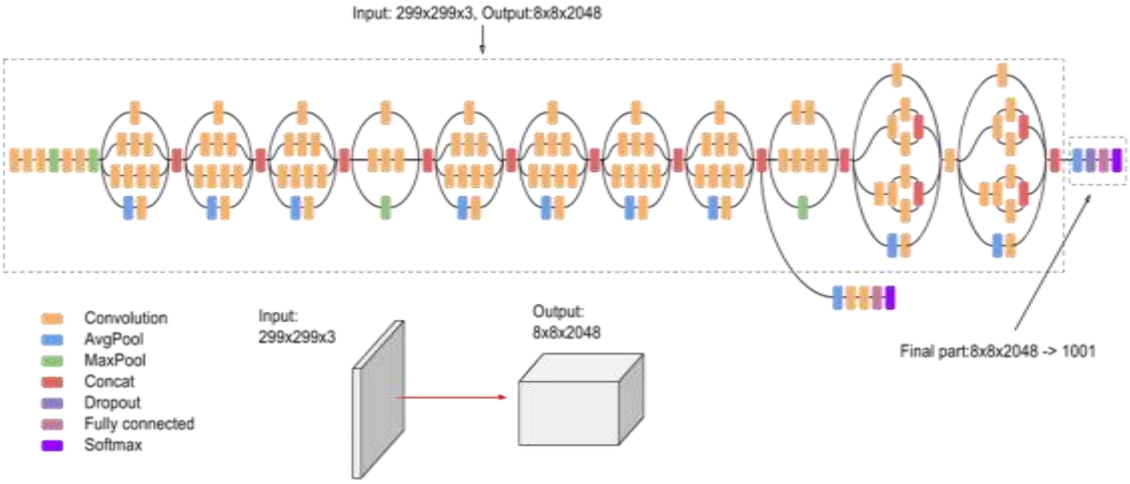
The ImageNet Large Scale Visual Recognition Challenge is **a benchmark in object category classification and detection on hundreds of object categories and millions of images**. The challenge has been run annually from 2010 to present, attracting participation from more than fifty institutions. In this competition people make different different types of CNN models. Whichever model gives high accuracies those models are are present in Keras Library. These are some popular models-

LeNet5 , AlexNet, vgg16, InceptionV3 restnet50, , resnet v2 , ResNeXt 50.

In our project we will use **InceptionV3** CNN model.

InceptionV3-

Inception v3 is a widely-used image recognition model that has been shown to attain greater than 78.1% accuracy on the ImageNet dataset.



The model itself is made up of symmetric and asymmetric building blocks, including convolutions, average pooling, max pooling, concats, dropouts, and fully connected layers. BatchNormalization is used extensively throughout the model and applied to activation inputs. Loss is computed via Softmax.

BatchNormalization- Batch normalization is a technique designed to automatically standardize the inputs to a layer in a deep learning neural network.

Once implemented, batch normalization has the effect of dramatically accelerating the training process of a neural network, and in some cases improves the performance of the model via a modest regularization effect. The BatchNormalization layer can be added to your model to standardize raw input variables or the outputs of a hidden layer.

Data Augmentation-

Data augmentation is **a strategy that enables practitioners to significantly increase the diversity of data available for training models, without actually collecting new data**. Data augmentation techniques such as cropping, padding, and horizontal flipping are commonly used to train large neural networks.

Adam Optimizer-

Adaptive Moment Estimation combines the power of RMSProp (root-mean-square prop) and momentum-based GD. In Adam optimizers, the power of momentum GD to hold the history of updates and the adaptive learning rate provided by RMSProp makes Adam optimizer a powerful method.

Callbacks-

A callback is an object that can perform actions at various stages of training (e.g. at the start or end of an epoch, before or after a single batch, etc).

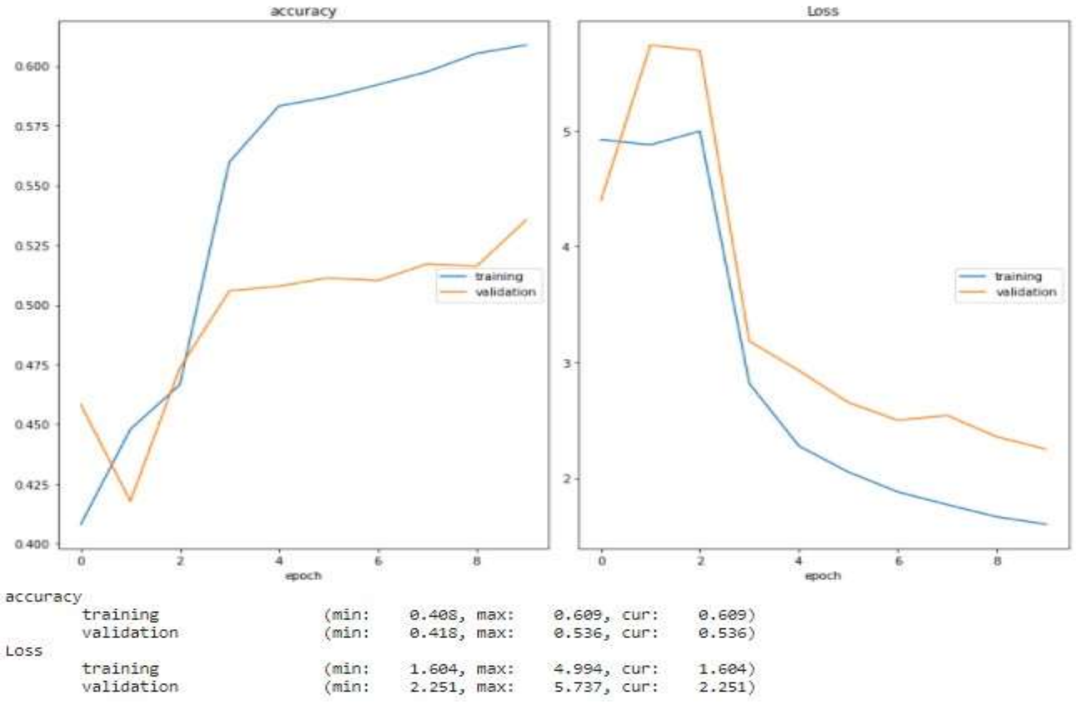
Reduce Learning Rate-

Reduce learning rate when a metric has stopped improving.Models often benefit from reducing the learning rate by a factor of 2-10 once learning stagnates. This callback monitors a quantity and if no improvement is seen for a 'patience' number of epochs, the learning rate is reduced.

Checkpoint-

Callback to save the Keras model or model weights at some frequency.'ModelCheckpoint' callback is used in conjunction with training using model.fit() to save a model or weights (in a checkpoint file) at some interval, so the model or weights can be loaded later to continue the training from the state saved.

Now finally after setting some Callbacks we trained our model and checked the training and validation accuracy and saved our model in a H5 file.



```
accuracy
	training                   (min:    0.408, max:    0.609, cur:    0.609)
	validation                 (min:    0.418, max:    0.536, cur:    0.536)
Loss
	training                   (min:    1.604, max:    4.994, cur:    1.604)
	validation                 (min:    2.251, max:    5.737, cur:    2.251)
```

Deployment

Flask framework-

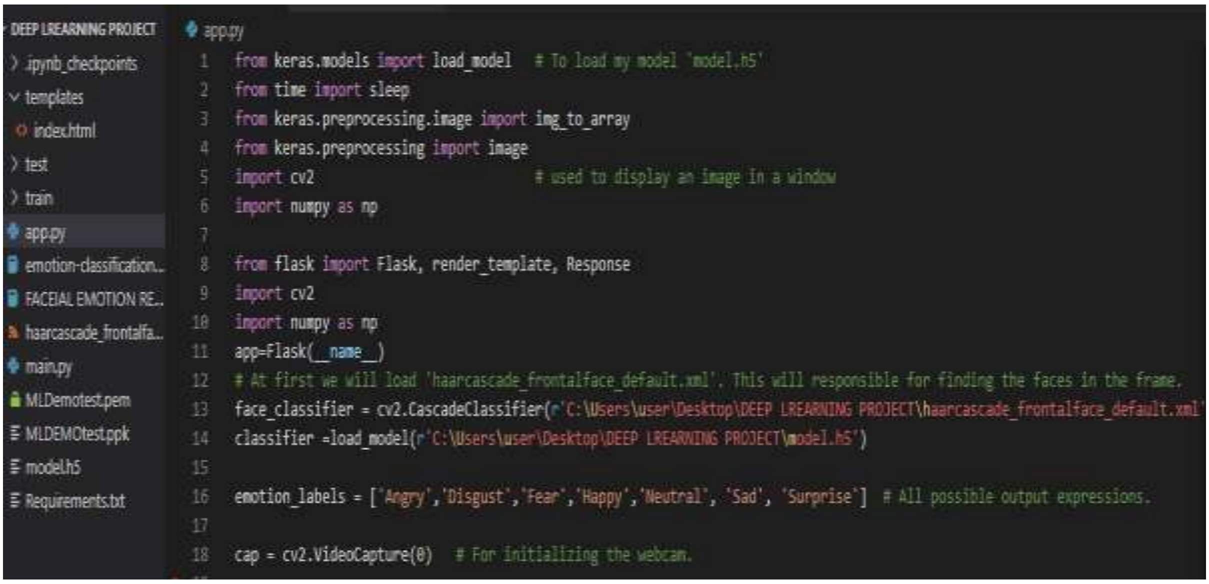
Flask is a web framework for Python, meaning that it provides **functionality for building web applications**, including managing HTTP requests and rendering templates.

Haar cascade Frontal face XML file-

It is an Object Detection Algorithm used to identify faces in an image or a real time video. The algorithm uses edge or line detection features proposed by Viola and Jones in their research paper “Rapid Object Detection using a Boosted Cascade of Simple Features” published in 2001. The algorithm is given a lot of positive images consisting of faces, and a lot of negative images not consisting of any face to train on them.

OpenCv-

OpenCV is a great tool **for image processing and performing computer vision tasks**. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more. It supports multiple languages including python, java C++.



```python
from keras.models import load_model   # To load my model 'model.h5'
from time import sleep
from keras.preprocessing.image import img_to_array
from keras.preprocessing import image
import cv2                            # used to display an image in a window
import numpy as np

from flask import Flask, render_template, Response
import cv2
import numpy as np
app=Flask(__name__)
# At first we will load 'haarcascade_frontalface_default.xml'. This will responsible for finding the faces in the frame.
face_classifier = cv2.CascadeClassifier(r'C:\Users\user\Desktop\DEEP LREARNING PROJECT\haarcascade_frontalface_default.xml'
classifier =load_model(r'C:\Users\user\Desktop\DEEP LREARNING PROJECT\model.h5')

emotion_labels = ['Angry','Disgust','Fear','Happy','Neutral', 'Sad', 'Surprise']  # All possible output expressions.

cap = cv2.VideoCapture(0)   # For initializing the webcam.
```

AWS EC2 Instance-

An Amazon EC2 instance is a virtual server in Amazon's Elastic Compute Cloud (EC2) for running applications on the Amazon Web Services (AWS) infrastructure. AWS is a comprehensive, evolving cloud computing platform; EC2 is a service that enables business subscribers to run application programs in the computing environment.

Application Link-

ec2-3-141-11-173.us-east-2.compute.amazonaws.com:8080

Conclusion-

- Our model is performing well with using InceptionV3 CNN model.
- So now this application can be used during online classrooms.
- We can use the same concept for Face detection, Object detection etc.

References-

1. Analytics Vidhya

2. Towards data Science

3. Stackoverflow