Capstone Project

Live Class Monitoring System(Face Emotion Recognition)

Sumaya Bai
Het Kothari

Contents

Introduction

Facial Emotion Recognition (FER) is the technology that analyses facial expressions from both static images and videos in order to reveal information of one's emotional state.
It is the process of detecting human emotions from facial expressions. The human brain recognizes emotions automatically, and software has now been developed that can recognize emotions as well. This technology is becoming more accurate all the time, and will eventually be able to read emotions as how our brains do.

Problem Statement

- One of the biggest problems of E-learning systems is to maintain the motivation of the students in the virtual classrooms.Unlike the formal classroom, a tutor when teaching online cannot observe a student's emotional states.
- The purpose of the project is to develop a Facial Emotion Recognition System (FERS), which recognize the emotional states of students in video-conference type E-learning systems.
- In order to create a more interactive educational environment, this system transfers the emotional states of the students to the educator instantaneously.
- Our study is supportive of the studies that make possible to observe the motivation level of both the individual and the virtual classroom in the e-learning systems.

Data Summary

We've built a deep learning model which detects the real time emotions of students through a webcam so that teachers can understand if students are able to grasp the topic according to students' expressions or emotions and then deploy the model. The model is trained on the FER-2013 dataset .

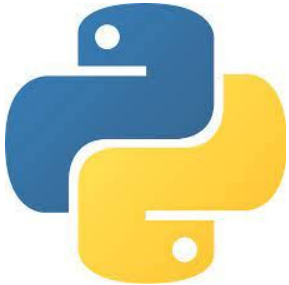
This dataset consists of 35887 grayscale, 48x48 sized face images with seven emotions - angry, disgusted, fearful, happy, neutral, sad and surprised.

dataset link:- https://www.kaggle.com/msambare/fer2013

Label	Emotion	Number of images for Training	Number of images for Testing
0	angry	3995	958
1	disgust	436	111
2	fear	4097	1024
3	happy	7215	1774
4	sad	4830	1247
5	surprised	3171	831
6	neutral	4965	1233

Dependencies

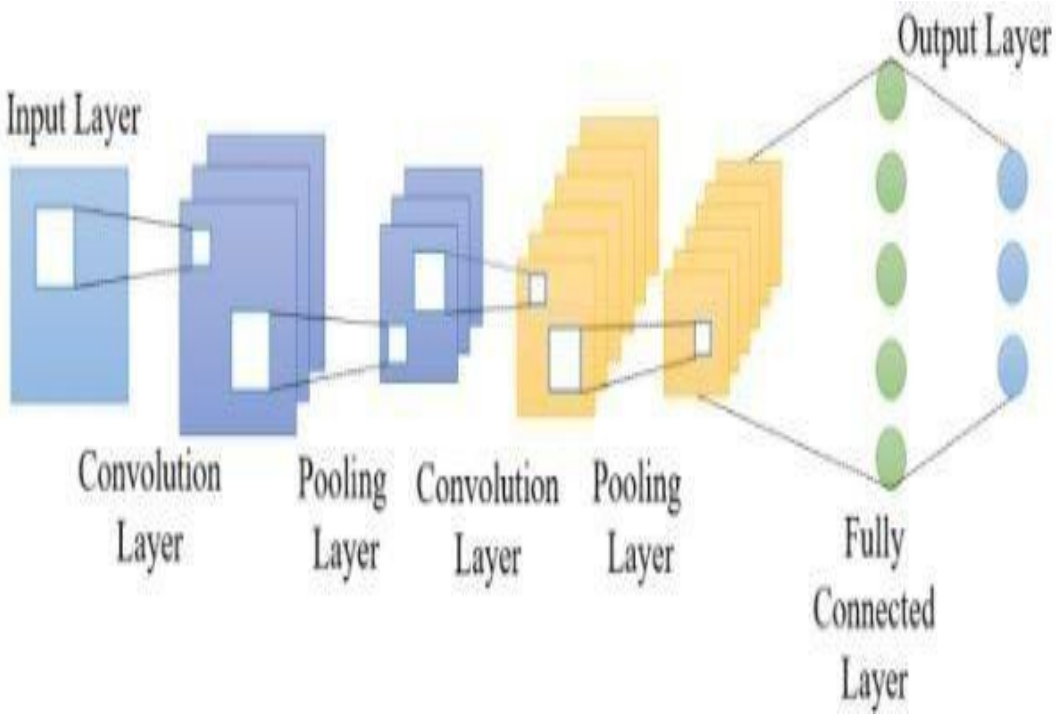
1. Python
2. Tensorflow
3. Streamlit
4. Streamlit-webrtc
5. Opencv



CNN Model

CNNs are a class of Deep Neural Networks that can recognize and classify particular features from images and are widely used for analyzing visual images. Their applications range from image and video recognition, image classification, medical image analysis, computer vision and natural language processing.
A CNN uses a system much like a multilayer perceptron that has been designed for reduced processing requirements
The layers of a CNN consist of an input layer, an output layer and a hidden layer that includes multiple convolutional layers, pooling layers, fully connected layers and normalization layers. The removal of limitations and increase in efficiency for image processing results in a system that is far more effective, simpler to trains limited for image processing and natural language processing.

Basic Architecture



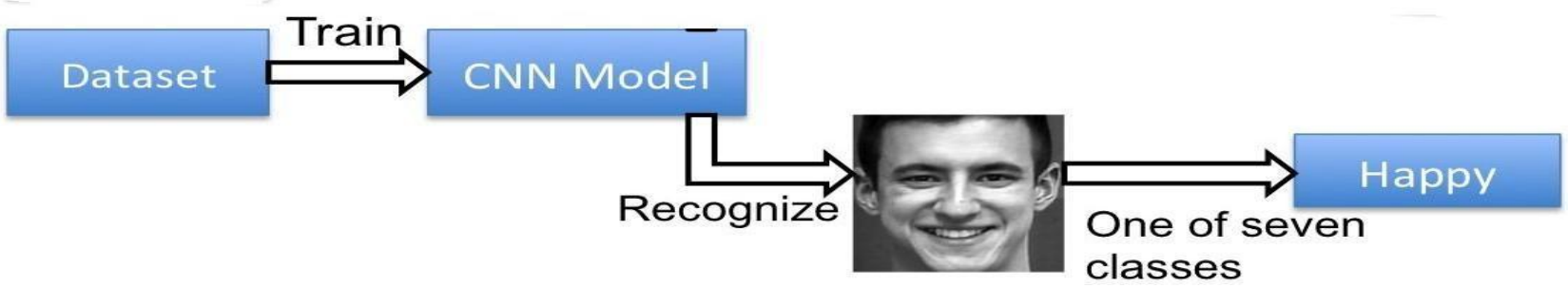
- A convolution tool that separates and identifies the various features of the image for analysis in a process called as Feature Extraction
- A fully connected layer that utilizes the output from the convolution process and predicts the class of the image based on the features extracted in previous stages.

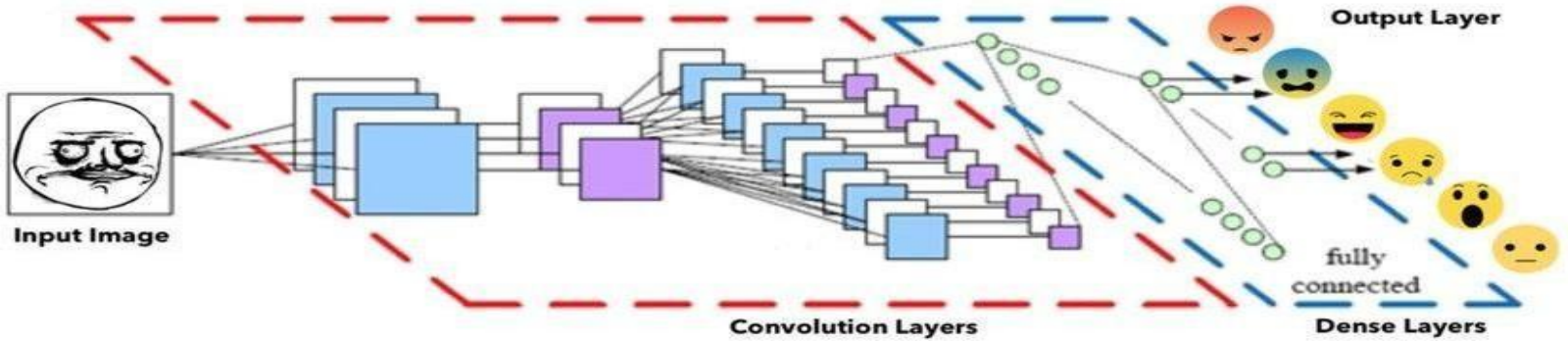
CNN Layers

There are three types of layers that make up the CNN which are the convolutional layers, pooling layers, and fully-connected (FC) layers. When these layers are stacked, a CNN architecture will be formed.

1. Convolutional Layer : This layer is the first layer that is used to extract the various features from the input images. In this layer, the mathematical operation of convolution is performed between the input image and a filter of a particular size.
2. In most cases, a Convolutional Layer is followed by a Pooling Layer. The primary aim of this layer is to decrease the size of the convolved feature map to reduce the computational costs. This is performed by decreasing the connections between layers and independently operates on each feature map.
3. The Fully Connected (FC) layer consists of the weights and biases along with the neurons and is used to connect the neurons between two different layers. These layers are usually placed before the output layer and form the last few layers of a CNN Architecture.

CNN layers


Dataset
Train
CNN Model
Recognize
Happy
One of seven classes


Input Image
Output Layer
fully connected
Convolution Layers
Dense Layers

CNN Model building

Parameters

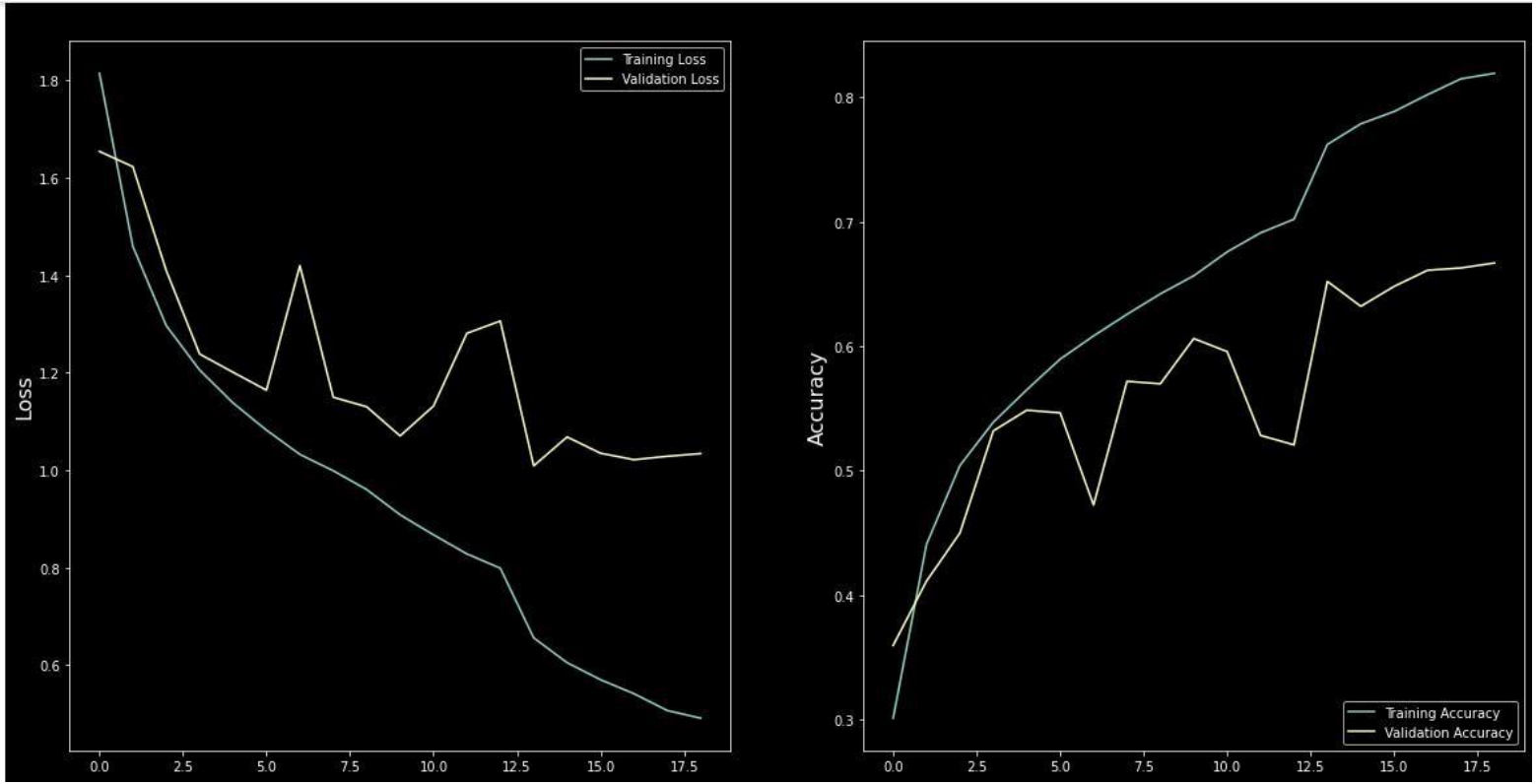
- **Activation Function - ReLu, Softmax**
- **Epoch - 40**
- **Optimizer - Adam**
- **Batch size -32**

Also we use some common techniques for each layer

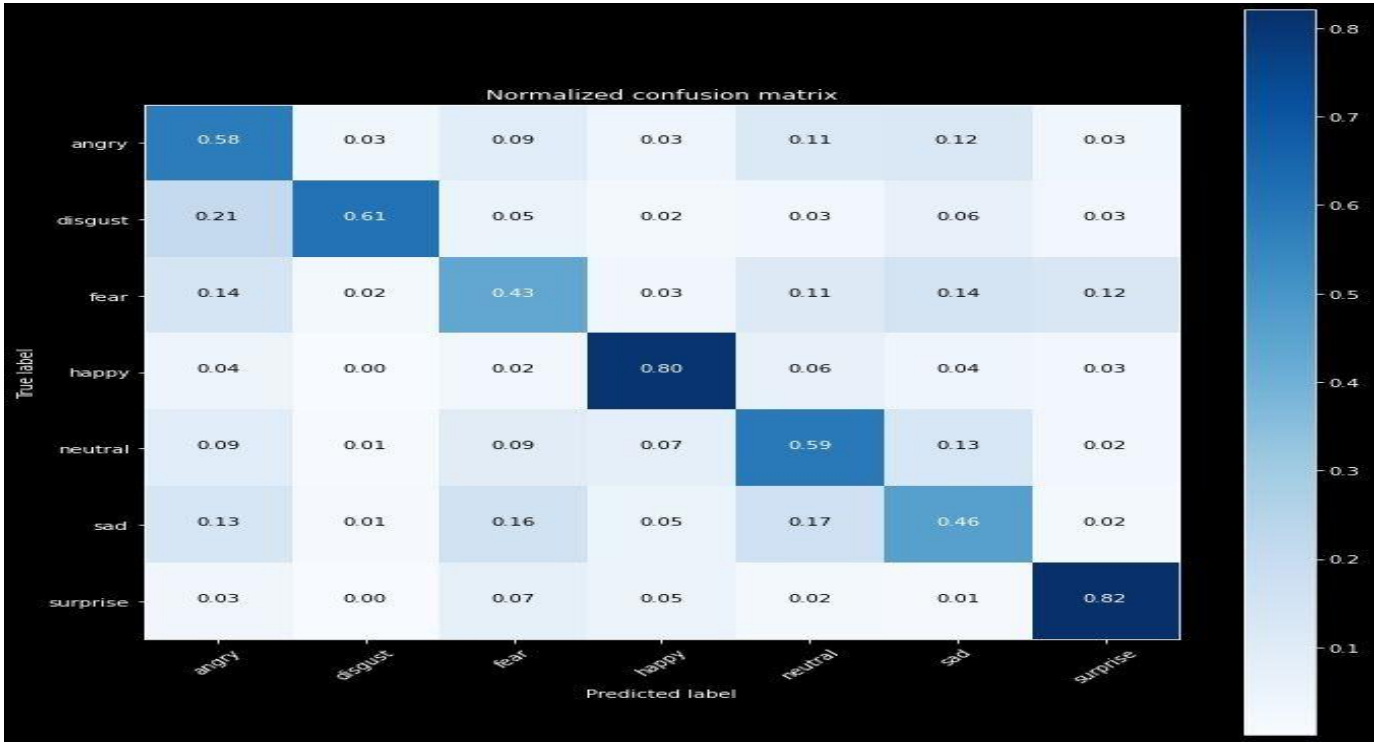
- **Batch normalization**
- **Dropout**

We choose softmax as our last activation function as it is commonly used for multi-label classification.

Loss & Accuracy Plot



Confusion Matrix



- **Our model is very good for predicting happy and surprised faces. However it predicts quite poorly feared faces maybe because it confuses them with sad faces.**

Real-Time Local Video Face Emotion Detection

We've created two patterns for detecting and predicting single faces and as well as multiple faces using OpenCV videocapture in local.

For Webapp , OpenCV can't be used. Thus, using Streamlit-Webrtc for front-end application.

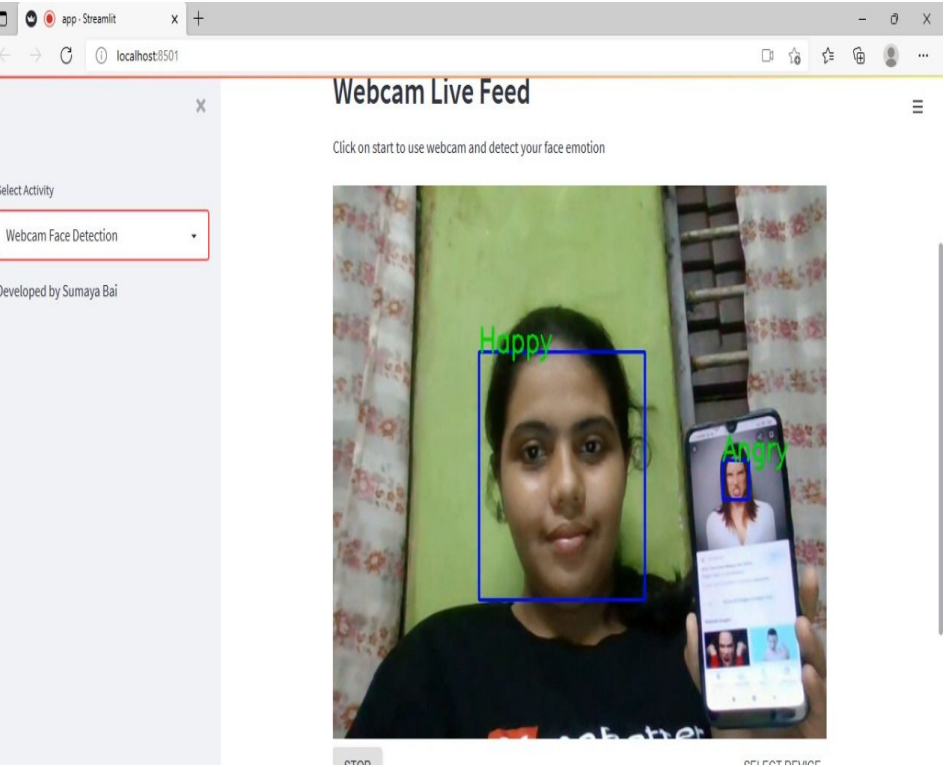
Deployment of Streamlit WebApp in Heroku and Streamlit

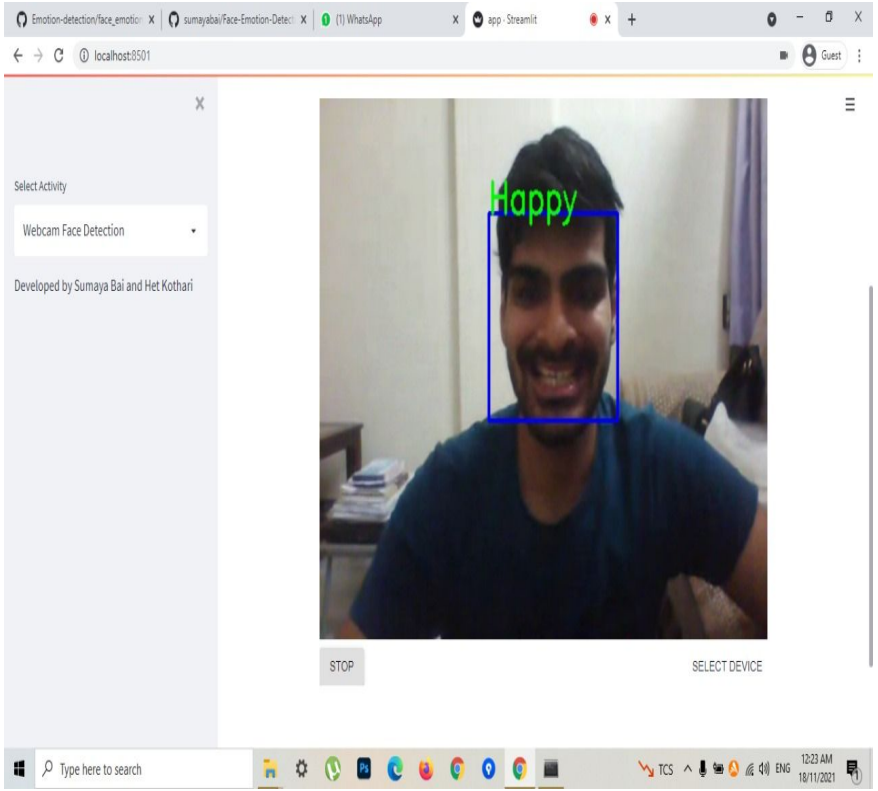
Deployment Softwares : Deployment softwares are a category of software developed by third-party vendors that mimic the features and functionalities of Online Staffing firms, often offering the software as a white-labeled “app.”
We’ve used Heroku and streamlit for the purpose of deployment.
Heroku : Heroku is a cloud platform as a service (PaaS) supporting several programming languages. One of the firstc cloud platforms, Heroku has been in development since June 2007, when it supported only the Ruby programming language, but now supports Java, Node.js, Scala, Clojure, Python, PHP, and Go

Streamlit : Streamlit sharing lets you deploy, manage, and share your apps – all for free! If you have a Streamlit app hosted publicly on GitHub, you are now one click away from sharing it with the world.

Heroku : https://face-emotion1.herokuapp.com/
streamlit : https://share.streamlit.io/sumayabai/face-emotion-detection/main/app.py

Output


app · Streamlit
localhost:8501
Webcam Live Feed
Click on start to use webcam and detect your face emotion
Select Activity
Webcam Face Detection
Developed by Sumaya Bai
Happy
Angry


localhost:8501
Select Activity
Webcam Face Detection
Developed by Sumaya Bai and Het Kothari
Happy
STOP
SELECT DEVICE
Type here to search

Conclusion

1. Our model is giving an accuracy of 81% and is robust in that it works well even in a dim light environment.

2. The application is able to detect face location and predict the right expression while checking it on a local webcam.

3. The front-end of the model was made using streamlit for webapp and running well on local webapp link.

4. We've deployed the Streamlit Web App on Heroku and Streamlit share that runs on a web server.

THANK YOU

Q & A