Capstone Project-5

Face Emotion Recognition

Team Members

Humant Sattabhayya
Soumyadeep Pandit
Rajat Arya

Contents of the Presentation

- Introduction
- Problem Statement
- Data Summary
- Dependencies
- Model Creation
- Loss and Accuracy Plot
- Real-time Local Video Face Detection
- Deployment of Streamlit WebApp in Heroku and Streamlit
- Various prediction Images from the WebApp
- Challenges
- Conclusions

Introduction

- **What is Face Emotion Recognition ?**

  Facial Emotion Recognition (FER) is **the technology that analyses facial expressions from both static images and videos in order to reveal information on one's emotional state**.

- **Why Emotion Detection?**

  By using Facial Emotion Recognition, businesses can process images, and videos in real-time for monitoring video feeds or automating video analytics, thus saving costs and making life better for their users.

Problem Statement

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. 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 (ex Zoom) 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 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.

Data Summary

- We have 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 Face Expression Dataset.

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

Dataset Link : Dataset

Label	Emotion	Training Images	Test Images
0	Angry	3996	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. Numpy =1.21.2
2. Streamlit =0.87.0
3. Keras =2.4.3
4. Opencv-contrib-python-headless
5. Tensorflow-cpu
6. Streamlit_webrtc

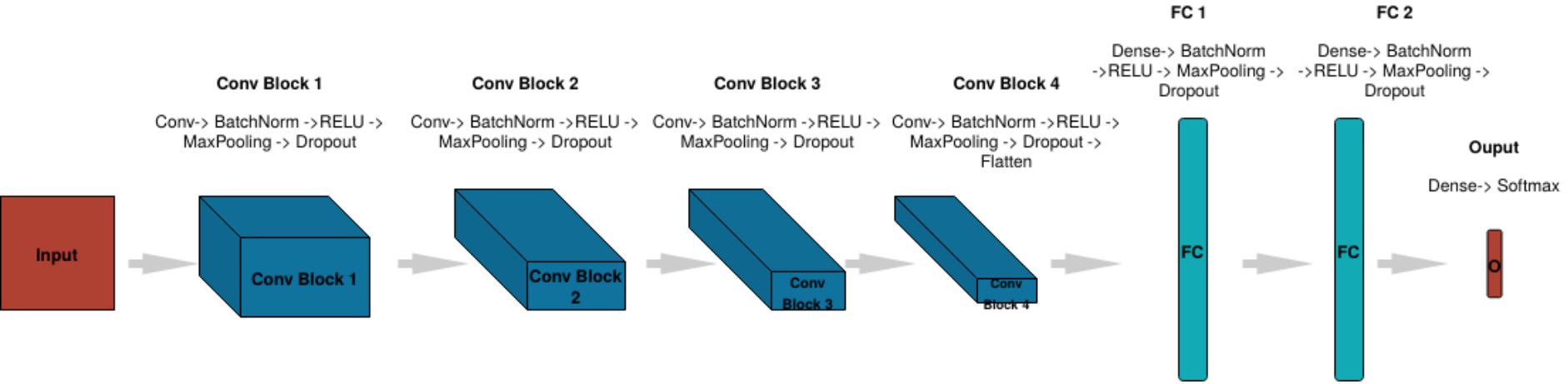
Model Creation

Model Creation Using CNN layers:

```
#1st CNN layer
model.add(Conv2D(64,(3,3),padding = 'same',input_shape = (48,48,1)))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Dropout(0.25))

#2nd CNN layer
model.add(Conv2D(128,(5,5),padding = 'same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Dropout (0.25))

#3rd CNN layer
model.add(Conv2D(512,(3,3),padding = 'same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Dropout (0.25))

#4th CNN layer
model.add(Conv2D(512,(3,3), padding='same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
```

```
#Fully connected 1st layer
model.add(Dense(256))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.25))


# Fully connected layer 2nd layer
model.add(Dense(512))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.25))

model.add(Dense(no_of_classes, activation='softmax'))
```

Model Creation

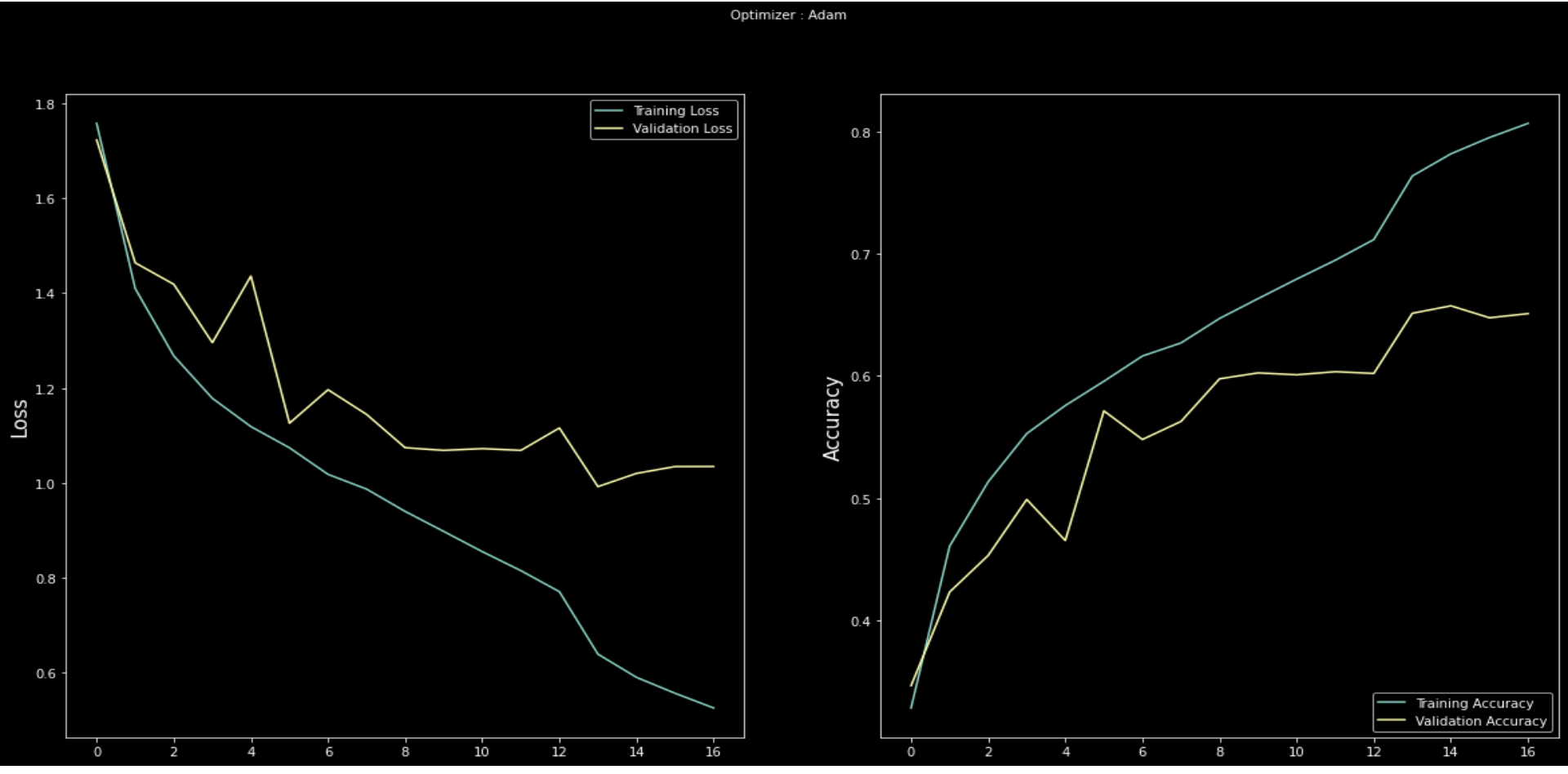


Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 48, 48, 64)	640
batch_normalization (BatchNo	(None, 48, 48, 64)	256
activation (Activation)	(None, 48, 48, 64)	0
max_pooling2d (MaxPooling2D)	(None, 24, 24, 64)	0
dropout (Dropout)	(None, 24, 24, 64)	0
conv2d_1 (Conv2D)	(None, 24, 24, 128)	204928
batch_normalization_1 (Batch	(None, 24, 24, 128)	512
activation_1 (Activation)	(None, 24, 24, 128)	0
max_pooling2d_1 (MaxPooling2	(None, 12, 12, 128)	0
dropout_1 (Dropout)	(None, 12, 12, 128)	0
conv2d_2 (Conv2D)	(None, 12, 12, 512)	590336
batch_normalization_2 (Batch	(None, 12, 12, 512)	2048
activation_2 (Activation)	(None, 12, 12, 512)	0
max_pooling2d_2 (MaxPooling2	(None, 6, 6, 512)	0
dropout_2 (Dropout)	(None, 6, 6, 512)	0
conv2d_3 (Conv2D)	(None, 6, 6, 512)	2359808
batch_normalization_3 (Batch	(None, 6, 6, 512)	2048

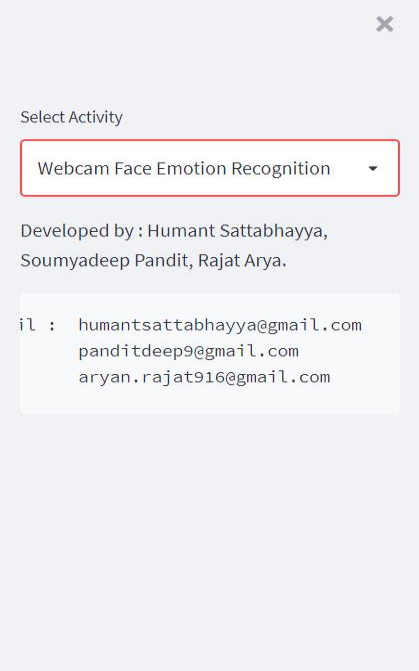
```
_________________________________________________________________
activation_3 (Activation)    (None, 6, 6, 512)         0
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 3, 3, 512)         0
_________________________________________________________________
dropout_3 (Dropout)          (None, 3, 3, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 4608)              0
_________________________________________________________________
dense (Dense)                (None, 256)               1179904
_________________________________________________________________
batch_normalization_4 (Batch (None, 256)               1024
_________________________________________________________________
activation_4 (Activation)    (None, 256)               0
_________________________________________________________________
dropout_4 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               131584
_________________________________________________________________
batch_normalization_5 (Batch (None, 512)               2048
_________________________________________________________________
activation_5 (Activation)    (None, 512)               0
_________________________________________________________________
dropout_5 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 7)                 3591
=================================================================
Total params: 4,478,727
Trainable params: 4,474,759
Non-trainable params: 3,968
```

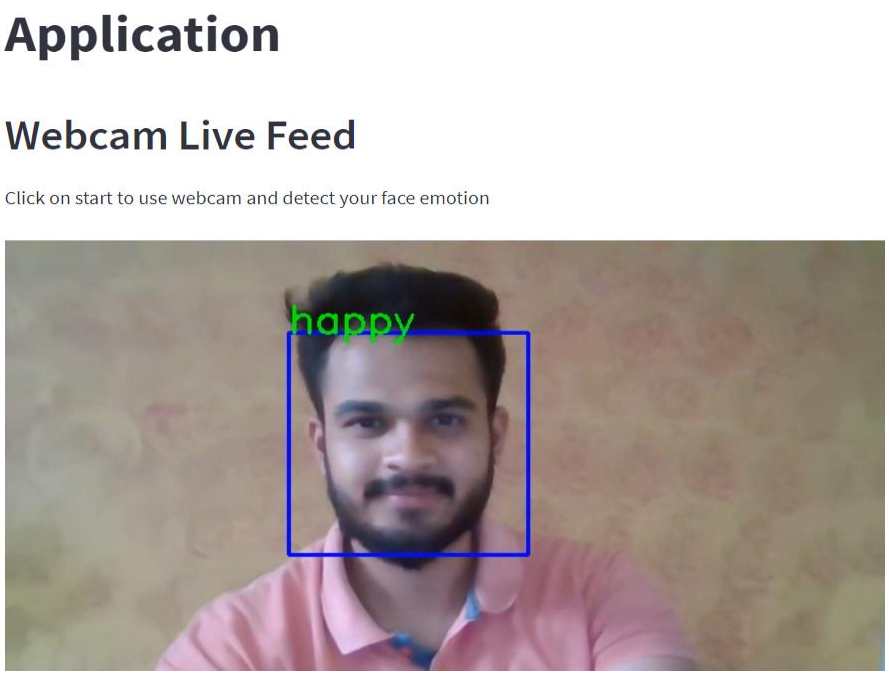
```
Epoch 1/50
225/225 [==============================] - 32s 136ms/step - loss: 1.7579 - accuracy: 0.3280 - val_loss: 1.7229 - val_accuracy: 0.3463
Epoch 2/50
225/225 [==============================] - 30s 134ms/step - loss: 1.4099 - accuracy: 0.4605 - val_loss: 1.4641 - val_accuracy: 0.4230
Epoch 3/50
225/225 [==============================] - 30s 131ms/step - loss: 1.2680 - accuracy: 0.5133 - val_loss: 1.4188 - val_accuracy: 0.4530
Epoch 4/50
225/225 [==============================] - 35s 154ms/step - loss: 1.1786 - accuracy: 0.5528 - val_loss: 1.2958 - val_accuracy: 0.4989
Epoch 5/50
225/225 [==============================] - 30s 134ms/step - loss: 1.1190 - accuracy: 0.5758 - val_loss: 1.4361 - val_accuracy: 0.4652
Epoch 6/50
225/225 [==============================] - 30s 131ms/step - loss: 1.0743 - accuracy: 0.5957 - val_loss: 1.1260 - val_accuracy: 0.5714
Epoch 7/50
225/225 [==============================] - 30s 135ms/step - loss: 1.0183 - accuracy: 0.6164 - val_loss: 1.1967 - val_accuracy: 0.5480
Epoch 8/50
225/225 [==============================] - 30s 133ms/step - loss: 0.9873 - accuracy: 0.6271 - val_loss: 1.1446 - val_accuracy: 0.5628
Epoch 9/50
225/225 [==============================] - 30s 132ms/step - loss: 0.9402 - accuracy: 0.6472 - val_loss: 1.0745 - val_accuracy: 0.5977
Epoch 10/50
225/225 [==============================] - 30s 133ms/step - loss: 0.8981 - accuracy: 0.6634 - val_loss: 1.0688 - val_accuracy: 0.6026
Epoch 11/50
225/225 [==============================] - 30s 132ms/step - loss: 0.8555 - accuracy: 0.6794 - val_loss: 1.0726 - val_accuracy: 0.6010
Epoch 12/50
225/225 [==============================] - 30s 133ms/step - loss: 0.8154 - accuracy: 0.6949 - val_loss: 1.0688 - val_accuracy: 0.6036
Epoch 13/50
225/225 [==============================] - 29s 130ms/step - loss: 0.7711 - accuracy: 0.7118 - val_loss: 1.1160 - val_accuracy: 0.6021

Epoch 00013: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
Epoch 14/50
225/225 [==============================] - 30s 135ms/step - loss: 0.6395 - accuracy: 0.7639 - val_loss: 0.9925 - val_accuracy: 0.6514
Epoch 15/50
225/225 [==============================] - 30s 133ms/step - loss: 0.5905 - accuracy: 0.7821 - val_loss: 1.0203 - val_accuracy: 0.6575
Epoch 16/50
225/225 [==============================] - 29s 130ms/step - loss: 0.5570 - accuracy: 0.7954 - val_loss: 1.0347 - val_accuracy: 0.6477
Epoch 17/50
225/225 [==============================] - 30s 132ms/step - loss: 0.5260 - accuracy: 0.8070 - val_loss: 1.0348 - val_accuracy: 0.6511
Restoring model weights from the end of the best epoch.

Epoch 00017: ReduceLROnPlateau reducing learning rate to 4.0000001899898055e-05.
Epoch 00017: early stopping
```

Loss & Accuracy Plot



Various prediction Images from the WebApp





Select Activity

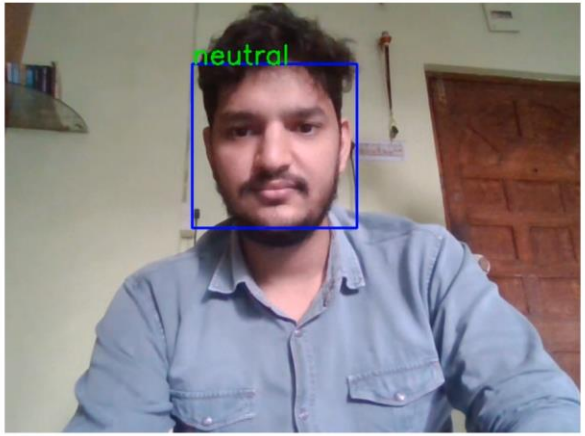
Webcam Face Emotion Recognition

Developed by : Humant Sattabhayya, Soumyadeep Pandit, Rajat Arya.

il : humantsattabhayya@gmail.com
panditdeep9@gmail.com
aryan.rajat916@gmail.com

Webcam Live Feed

Click on start to use webcam and detect your face emotion



STOP

SELECT DEVICE

Select Activity

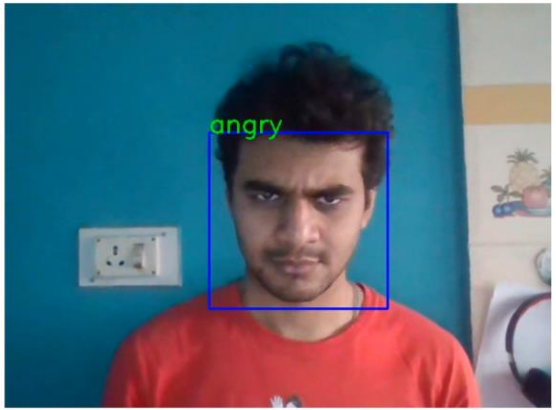
Webcam Face Emotion Recognition

Developed by : Humant Sattabhayya, Soumyadeep Pandit, Rajat Arya.

il : humantsattabhayya@gmail.com
panditdeep9@gmail.com
aryan.rajat916@gmail.com

Webcam Live Feed

Click on start to use webcam and detect your face emotion



STOP

SELECT DEVICE

Select Activity

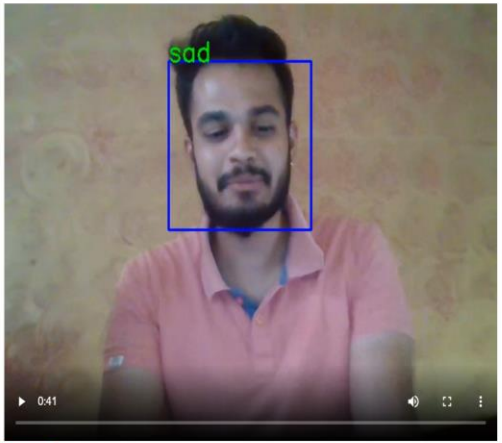
Webcam Face Emotion Recognition

Developed by : Humant Sattabhayya,
Soumyadeep Pandit, Rajat Arya.

il : humantsattabhayya@gmail.com
panditdeep9@gmail.com
aryan.rajat916@gmail.com

Webcam Live Feed

Click on start to use webcam and detect your face emotion



STOP

SELECT DEVICE

Select Activity

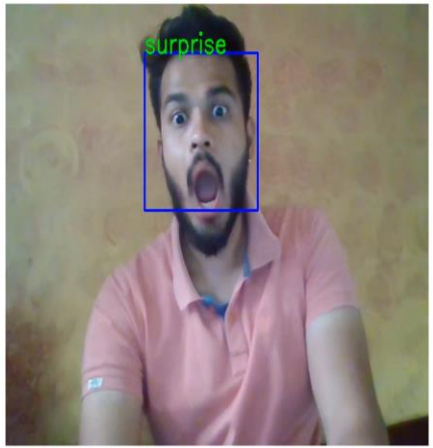
Webcam Face Emotion Recognition

Developed by : Humant Sattabhayya,
Soumyadeep Pandit, Rajat Arya.

il : humantsattabhayya@gmail.com
panditdeep9@gmail.com
aryan.rajat916@gmail.com

Webcam Live Feed

Click on start to use webcam and detect your face emotion



STOP

SELECT DEVICE

Multiple Face Emotion Detection

Select Activity

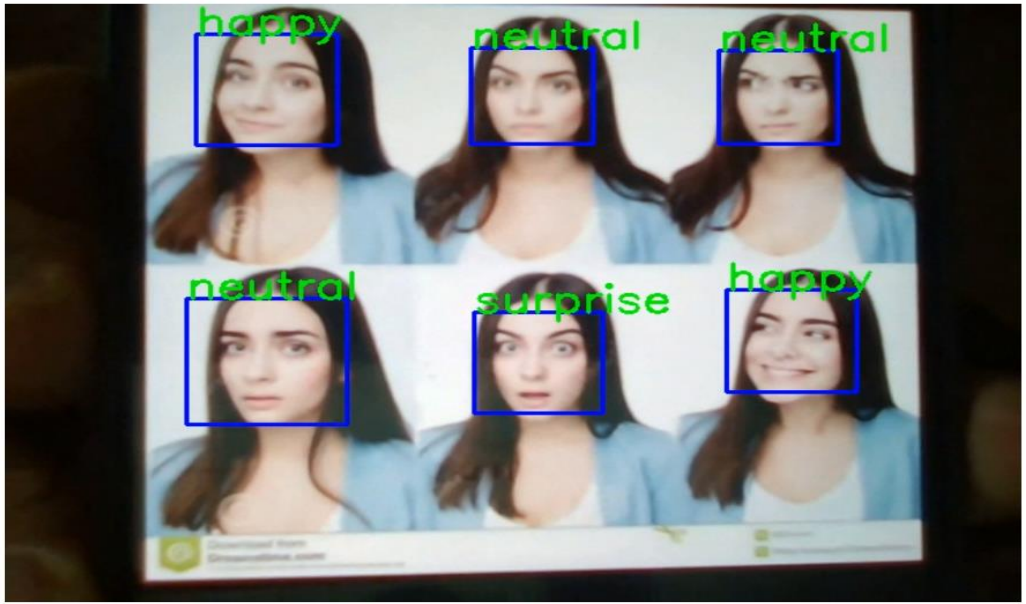
Webcam Face Emotion Recognition

Developed by : Humant Sattabhayya,
Soumyadeep Pandit, Rajat Arya.

Email : humantsattabhayya@
panditdeep9@gmail.
aryan.rajat916@gma

Webcam Live Feed

Click on start to use webcam and detect your face emotion



STOP

SELECT DEVICE

Challenges

- Face Expression was very huge due to which it took too much time to train the model to reaching desirable accuracy.

- Deep learning project requires lots of libraries also it requires the GPU that’s why we are unable to run our code on google colab and jupyter notebook.

- The dataset is too large so tuning of hyper parameters too much time.

Conclusion

- Finally We build the WebApp using streamlit and deployed on Heroku.
- The model which was created using CNN that gave training accuracy of 80% and validation accuracy of 65%.
- Model also work for multiple face detection.

THANK YOU