RCOEM
Shri Ramdeobaba College of Engineering and Management, Nagpur

Shri Ramdeobaba College of Engineering and Management, Nagpur
Department of Computer Science and Engineering
Session 2022-2023

A Seminar on:

FACIAL EXPRESSION RECOGNITION

PRESENTED BY:

Group No.: 7

Members:

Chirag Bamb (31)

Harsh Mata (35)

Ved Agrawal (61)

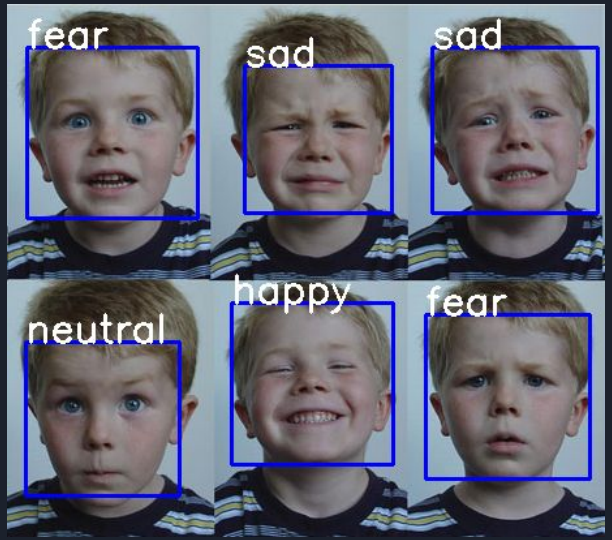
Harshal Dhunde (36)

Project Guide:

Dr. Ramchand Hablani

Objective

- Facial Expression Recognition using Deep Learning and Analysing the changes in accuracy on different Datasets and different layers



Literature Survey

Sr. No.	Title	Author	Journal/Source	Findings & Relevance	Research Gap
1	Deep Facial Expression Recognition:	Shan Li and Weihong Deng	IEEE	Using deep neural networks have increasingly been leveraged to learn discriminative representations for automatic FER	Due to lack of proper dataset unable to train model properly
2	Facial emotion recognition using deep learning: review and insights	Wafa Mellouka, Wahida Handouzia	Tlemcen university, BP 320, Chetouane Tlemcen 1300, Algeria	In this paper we have a detailed study of all model, their innovations and their problems	This analysis of 2019 so there is some ambiguity

Literature Survey

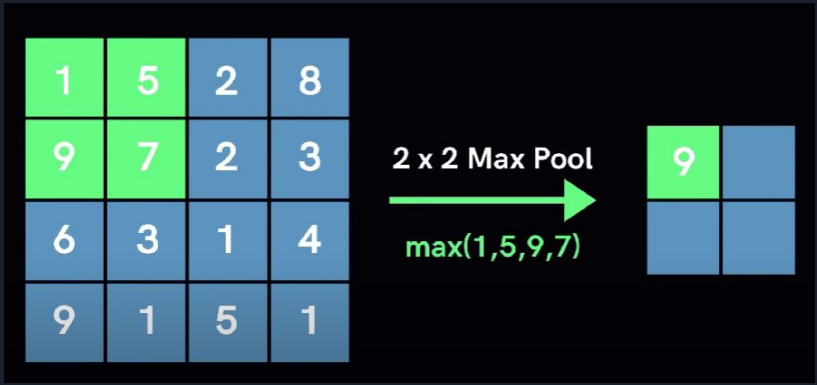
Sr. No.	Title	Author	Journal/Source	Findings & Relevance	Research Gap
3	Facial emotion recognition with minimal epochs and the significance of data diversity	Tanoy Debnath, Md. Mahfuz Reza, Anichur Rahman, Amin Beheshti, Shahab S. Band & Hamid Alinejad-Rokny	https://www.nature.com/articles/s41598-022-11173-0	Using 4- Layer ConvNet CNN Accuracy highly increased	Large and proper dataset. Illumination problems
4	Deep Facial Expression Recognition: Challenges, Applications, and Future Guidelines	Muhammad Sajjad, Fath U Min Ullah,, Mohib Ullah, Georgia Christodoulou , Faouzi Alaya Cheikh,, Mohammad Hijji , Khan Muhammad, Joel J.P.C. Rodriguesf	Alexandria Engineering Journal	Deep survey on latest issues and all available datasets and models	Now the shift of deep learning to CNN and GAN

Convolution Neural Network

- Allows us to extract visual effects in chunks
- Pooling - Reduce The Number of neurons Necessary in subsequent Layers
- Two types of polling max-polling and min-polling

$$(f * g)(i) = \sum_{j=1}^{m} g(j) \cdot f(i - j + m/2)$$

Dot Product!



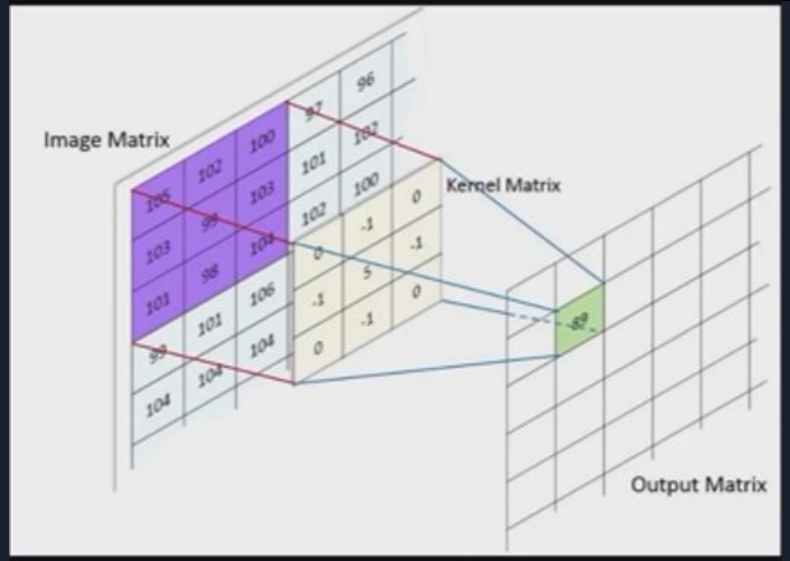
Convolve the Image

Pool the result

Repeat

Add few layers to help classify the image

Prediction in the output layer

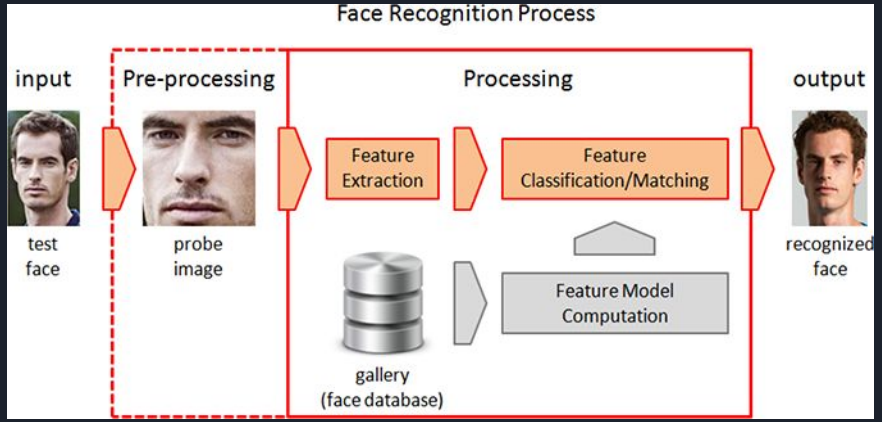


Datasets

Database	Facial Expressions	Sample Details	Type
FER-2013	Neutral, Sadness, Surprise, Happiness, Fear, Anger, Contempt, and Disgust	35887 Static Images	Spontaneous
RAF-DB	Angry, Disgust, Fear, Joy, Neutral, Sad, and Surprise	29672 facial image	Posed
EXP-W	Neutral, Sadness, Surprise, Happiness,Fear, Anger, And Disgust	91,793 Static Images	Posed

How Our Models Works

Testing → Framing → Face Detection → Feature Extraction → Natural, Surprise, Disgust, Fear, Angry, Sad, Happy



Training and validation Accuracy on FER dataset

```
Epoch 47/50
574/574 [==============================] - 43s 75ms/step - loss: 0.2127 - accuracy: 0.9254 - val_loss: 0.9216 - val_accuracy:
0.8496
Epoch 48/50
574/574 [==============================] - 40s 70ms/step - loss: 0.2203 - accuracy: 0.9220 - val_loss: 0.9954 - val_accuracy:
0.8250
Epoch 49/50
574/574 [==============================] - 40s 70ms/step - loss: 0.2043 - accuracy: 0.9273 - val_loss: 0.8506 - val_accuracy:
0.8641
Epoch 50/50
574/574 [==============================] - 43s 75ms/step - loss: 0.1852 - accuracy: 0.9317 - val_loss: 0.8848 - val_accuracy:
0.8599
```

Prediction On FER Dataset

Training and validation Accuracy on Transfer learning model Using MobileNetV2

```
Epoch 20/25
397/397 [==============================] - 663s 2s/step - loss: 0.2394 - accuracy: 0.9157
Epoch 21/25
397/397 [==============================] - 661s 2s/step - loss: 0.2124 - accuracy: 0.9282
Epoch 22/25
397/397 [==============================] - 664s 2s/step - loss: 0.2240 - accuracy: 0.9218
Epoch 23/25
397/397 [==============================] - 663s 2s/step - loss: 0.2001 - accuracy: 0.9301
Epoch 24/25
397/397 [==============================] - 665s 2s/step - loss: 0.1856 - accuracy: 0.9379
Epoch 25/25
397/397 [==============================] - 666s 2s/step - loss: 0.1735 - accuracy: 0.9380

Out[47]: <keras.callbacks.History at 0x2a1c8db9910>
```

Layer Used in Transfer learning Model

```
In [34]: 1 model = tf.keras.applications.MobileNetV2() ## Pre-trained model

In [35]: 1 model.summary()
```

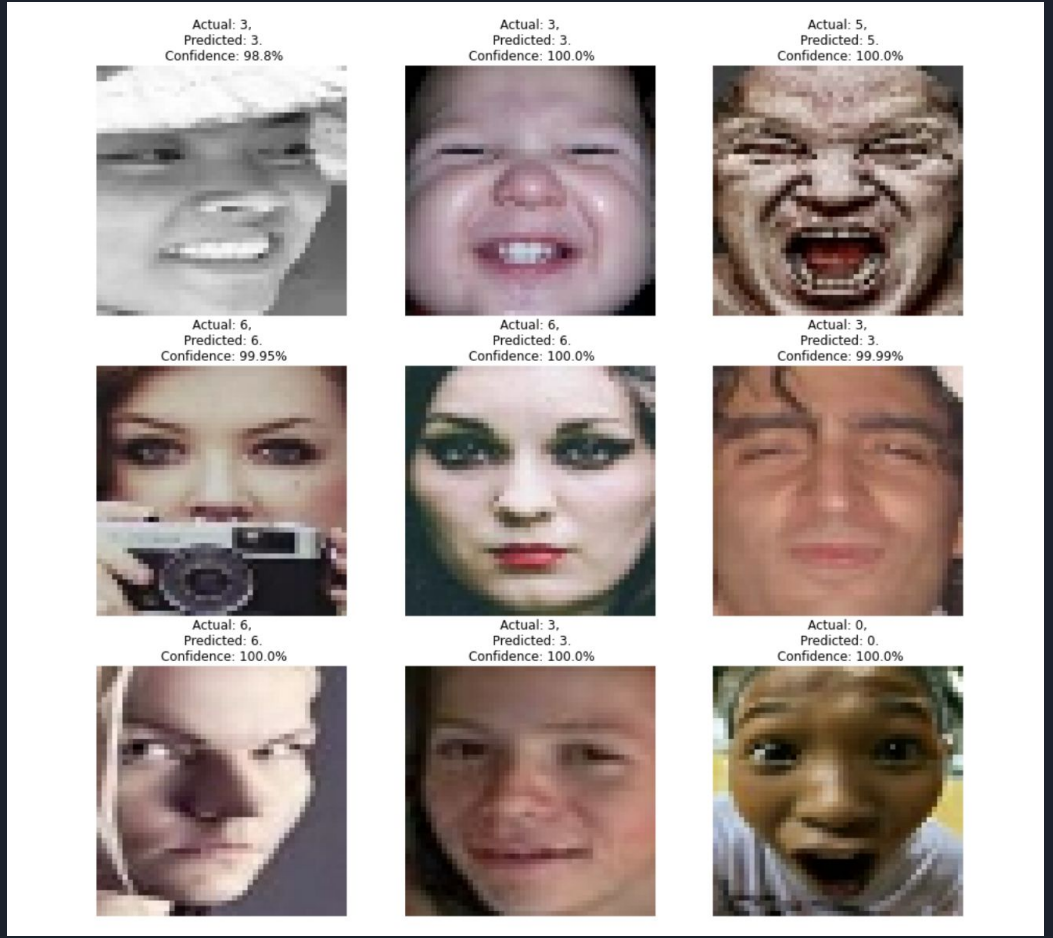
```
Model: "mobilenetv2_1.00_224"
_________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #     Connected to
=================================================================================================
input_1 (InputLayer)         [(None, 224, 224, 3       0           []
                             )]

Conv1 (Conv2D)               (None, 112, 112, 32       864         ['input_1[0][0]']
                             )

bn_Conv1 (BatchNormalization)  (None, 112, 112, 32     128         ['Conv1[0][0]']
                             )

Conv1_relu (ReLU)            (None, 112, 112, 32       0           ['bn_Conv1[0][0]']
                             )

expanded_conv_depthwise (Depth  (None, 112, 112, 32    288         ['Conv1_relu[0][0]']
wiseConv2D)                  )
```

Training and validation Accuracy RAF dataset 48X48.

In [31]:
```
history = model.fit(
    train_ds,
    batch_size=BATCH_SIZE,
    validation_data=val_ds,
    verbose=1,
    epochs=50,
)
```

```
0.9245
Epoch 45/50
384/384 [==============================] - 20s 52ms/step - loss: 0.0596 - accuracy: 0.9798 - val_loss: 0.4974 - val_accuracy:
0.9258
Epoch 46/50
384/384 [==============================] - 32s 83ms/step - loss: 0.0797 - accuracy: 0.9729 - val_loss: 0.5086 - val_accuracy:
0.9134
Epoch 47/50
384/384 [==============================] - 24s 61ms/step - loss: 0.0742 - accuracy: 0.9738 - val_loss: 0.5154 - val_accuracy:
0.9108
Epoch 48/50
384/384 [==============================] - 20s 51ms/step - loss: 0.0452 - accuracy: 0.9841 - val_loss: 0.5604 - val_accuracy:
0.9193
Epoch 49/50
384/384 [==============================] - 33s 85ms/step - loss: 0.0864 - accuracy: 0.9680 - val_loss: 0.5441 - val_accuracy:
0.9043
Epoch 50/50
384/384 [==============================] - 22s 57ms/step - loss: 0.0813 - accuracy: 0.9722 - val_loss: 0.5400 - val_accuracy:
0.9147
```

Prediction ON RAF dataset 48X48.



Training and validation Accuracy RAF + FER dataset 48X48.

```
y: 0.8677
Epoch 45/50
656/656 [==============================] - 52s 79ms/step - loss: 0.2367 - accuracy: 0.9124 - val_loss: 0.6687 - val_accurac
y: 0.8534
Epoch 46/50
656/656 [==============================] - 47s 72ms/step - loss: 0.2430 - accuracy: 0.9101 - val_loss: 0.6984 - val_accurac
y: 0.8490
Epoch 47/50
656/656 [==============================] - 47s 72ms/step - loss: 0.2259 - accuracy: 0.9182 - val_loss: 0.6225 - val_accurac
y: 0.8628
Epoch 48/50
656/656 [==============================] - 49s 74ms/step - loss: 0.2175 - accuracy: 0.9223 - val_loss: 0.6784 - val_accurac
y: 0.8532
Epoch 49/50
656/656 [==============================] - 49s 75ms/step - loss: 0.2247 - accuracy: 0.9187 - val_loss: 0.6389 - val_accurac
y: 0.8699
Epoch 50/50
656/656 [==============================] - 52s 79ms/step - loss: 0.1896 - accuracy: 0.9314 - val_loss: 0.7323 - val_accurac
y: 0.8571
```

Prediction ON RAF + FER dataset 48X48.

Layers in model on RAF 100 X 100 .

In [61]:
```
input_shape = (BATCH_SIZE, IMAGE_SIZE, IMAGE_SIZE, CHANNELS)
n_classes = 7

model = models.Sequential([
    resize_and_rescale,
    layers.Conv2D(32, kernel_size = (3,3), activation='relu', input_shape=input_shape),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64,  kernel_size = (3,3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64,  kernel_size = (3,3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(64, activation='relu'),
    layers.Dense(n_classes, activation='softmax'),
])
model.build(input_shape=input_shape)
```

In [62]:
```
model.summary()
```

Layers

- Convolutional 2D refers to a type of neural network layer that performs a 2D convolution operation. It is commonly used for spatial convolution over images[1]. The `Conv2D` class in the Keras API is an example of a 2D convolution layer.
- MaxPooling2D is a type of pooling layer that performs a max pooling operation for 2D spatial data. It downsamples the input along its spatial dimensions (height and width) by taking the maximum value over an input window (of size defined by `pool_size`) for each channel of the input.
- Flatten is a term that can refer to several different things. In the context of arrays, it refers to the process of converting a multi-dimensional array into a one-dimensional array
- A dense layer is a type of layer in a neural network where each neuron is connected to every neuron in the previous layer. It is also known as a fully connected layer. In Keras, you can create a dense layer using the `Dense` class from the `keras.layers` module.

Training Accuracy and Validation accuracy on RAF 100 X 100 .

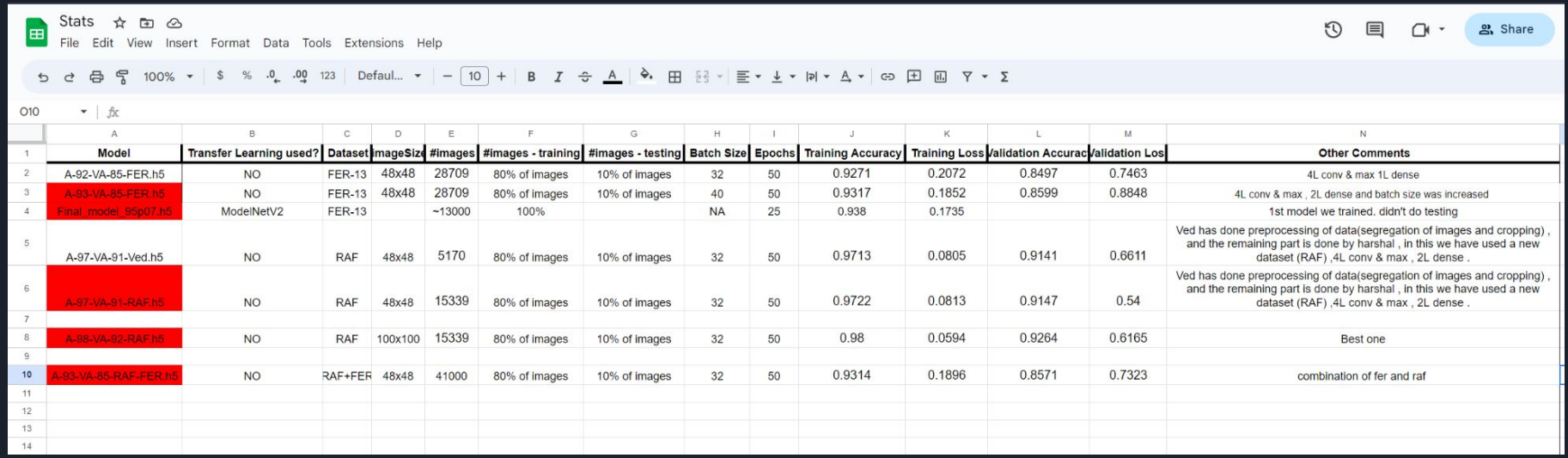
```
384/384 [==============================] - 117s 304ms/step - loss: 0.0753 - accuracy: 0.9761 - val_loss: 0.5445 - val_accuracy:
0.9258
Epoch 46/50
384/384 [==============================] - 96s 249ms/step - loss: 0.0501 - accuracy: 0.9835 - val_loss: 0.6539 - val_accuracy:
0.9134
Epoch 47/50
384/384 [==============================] - 93s 242ms/step - loss: 0.0634 - accuracy: 0.9791 - val_loss: 0.5876 - val_accuracy:
0.9212
Epoch 48/50
384/384 [==============================] - 93s 242ms/step - loss: 0.0611 - accuracy: 0.9796 - val_loss: 0.6276 - val_accuracy:
0.9121
Epoch 49/50
384/384 [==============================] - 93s 242ms/step - loss: 0.0650 - accuracy: 0.9782 - val_loss: 0.6073 - val_accuracy:
0.9284
Epoch 50/50
384/384 [==============================] - 95s 246ms/step - loss: 0.0594 - accuracy: 0.9799 - val_loss: 0.6165 - val_accuracy:
0.9264
```

Testing Accuracy on RAF 100 X 100 .

```
1/1 [==============================] - 0s 71ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 74ms/step
1/1 [==============================] - 0s 73ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 71ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 72ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 62ms/step
Total testing images => 1536
Testing Accuracy of our model is => 0.935546875

<Figure size 1080x1080 with 0 Axes>
```

Prediction On RAF 100 X 100 .

Stats we maintained...



Stats

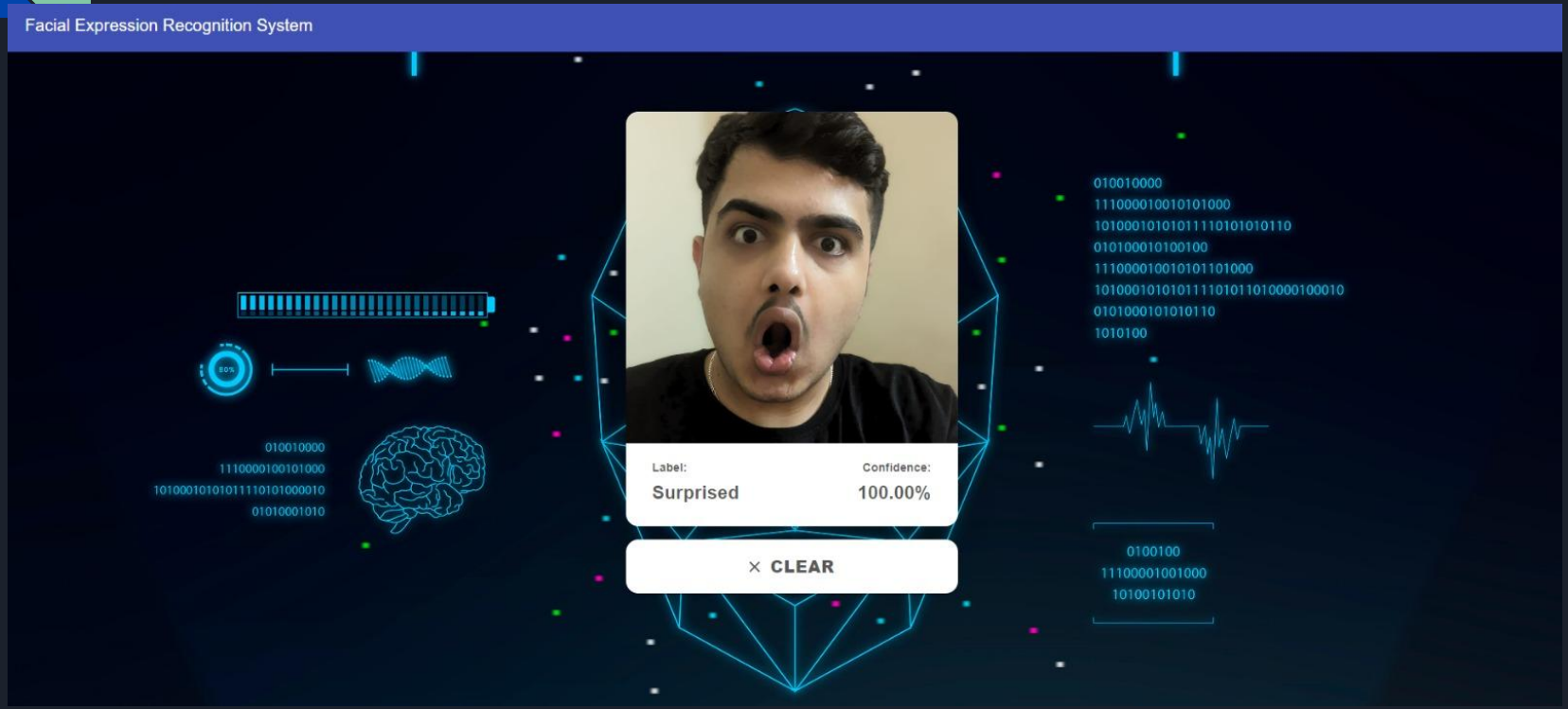
File Edit View Insert Format Data Tools Extensions Help

Share

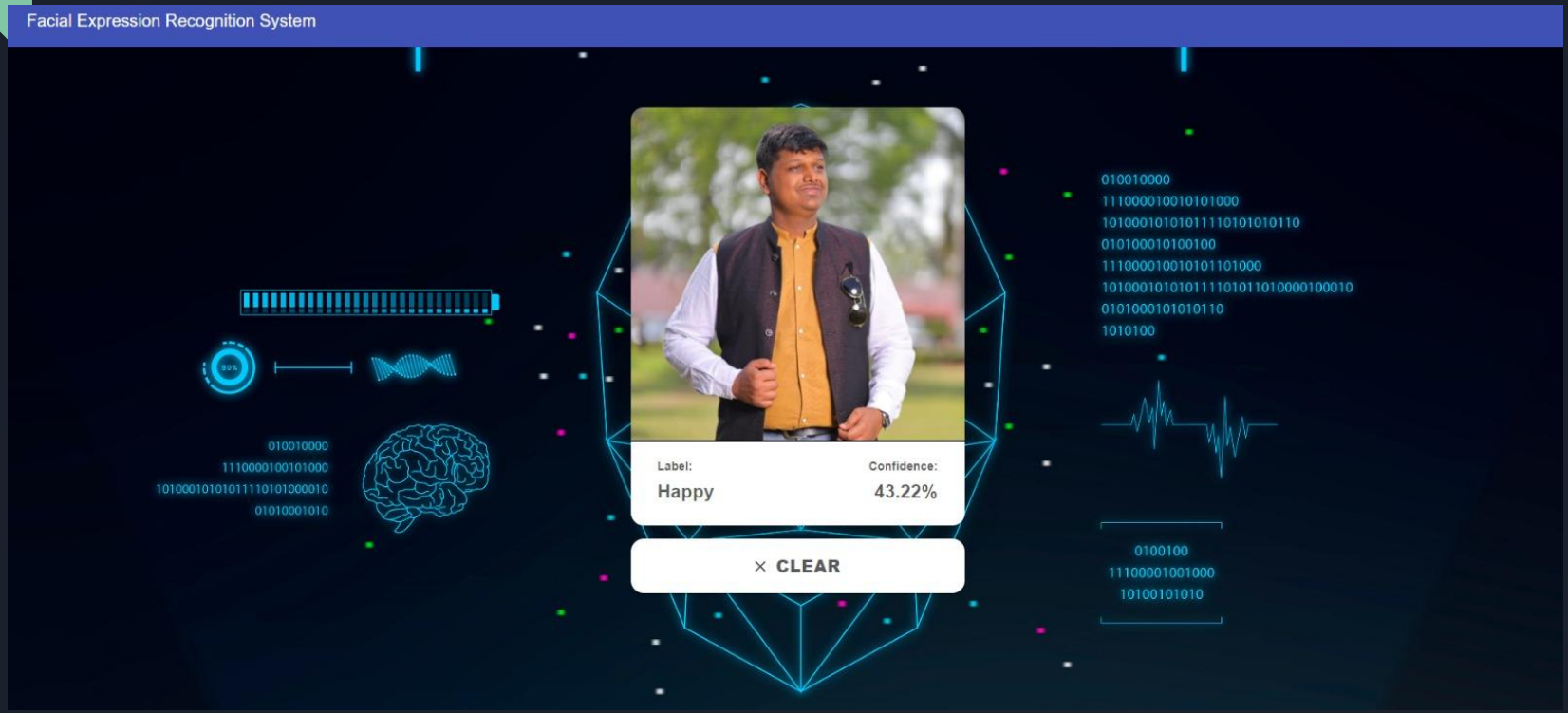
O10

	Model	Transfer Learning used?	Dataset	ImageSize	#images	#images - training	#images - testing	Batch Size	Epochs	Training Accuracy	Training Loss	Validation Accurac	Validation Los	Other Comments
2	A-92-VA-85-FER.h5	NO	FER-13	48x48	28709	80% of images	10% of images	32	50	0.9271	0.2072	0.8497	0.7463	4L conv & max 1L dense
3	A-93-VA-85-FER.h5	NO	FER-13	48x48	28709	80% of images	10% of images	40	50	0.9317	0.1852	0.8599	0.8848	4L conv & max , 2L dense and batch size was increased
4	Final_model_95p07.h5	ModelNetV2	FER-13		~13000	100%		NA	25	0.938	0.1735			1st model we trained. didn't do testing
5	A-97-VA-91-Ved.h5	NO	RAF	48x48	5170	80% of images	10% of images	32	50	0.9713	0.0805	0.9141	0.6611	Ved has done preprocessing of data(segregation of images and cropping) , and the remaining part is done by harshal , in this we have used a new dataset (RAF) ,4L conv & max , 2L dense .
6	A-97-VA-91-RAF.h5	NO	RAF	48x48	15339	80% of images	10% of images	32	50	0.9722	0.0813	0.9147	0.54	Ved has done preprocessing of data(segregation of images and cropping) , and the remaining part is done by harshal , in this we have used a new dataset (RAF) ,4L conv & max , 2L dense .
7														
8	A-98-VA-92-RAF.h5	NO	RAF	100x100	15339	80% of images	10% of images	32	50	0.98	0.0594	0.9264	0.6165	Best one
9														
10	A-93-VA-85-RAF-FER.h5	NO	RAF+FER	48x48	41000	80% of images	10% of images	32	50	0.9314	0.1896	0.8571	0.7323	combination of fer and raf
11														
12														
13														
14														

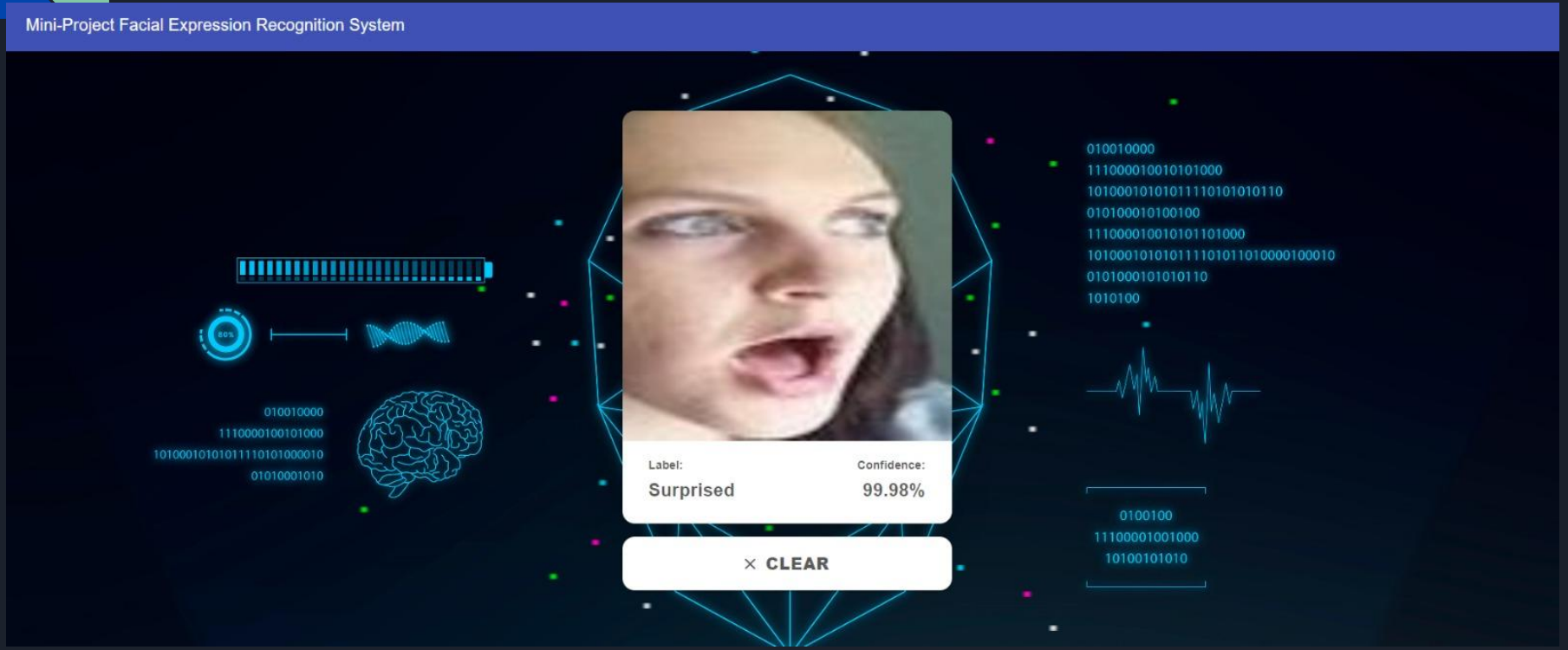
Sample Outputs:



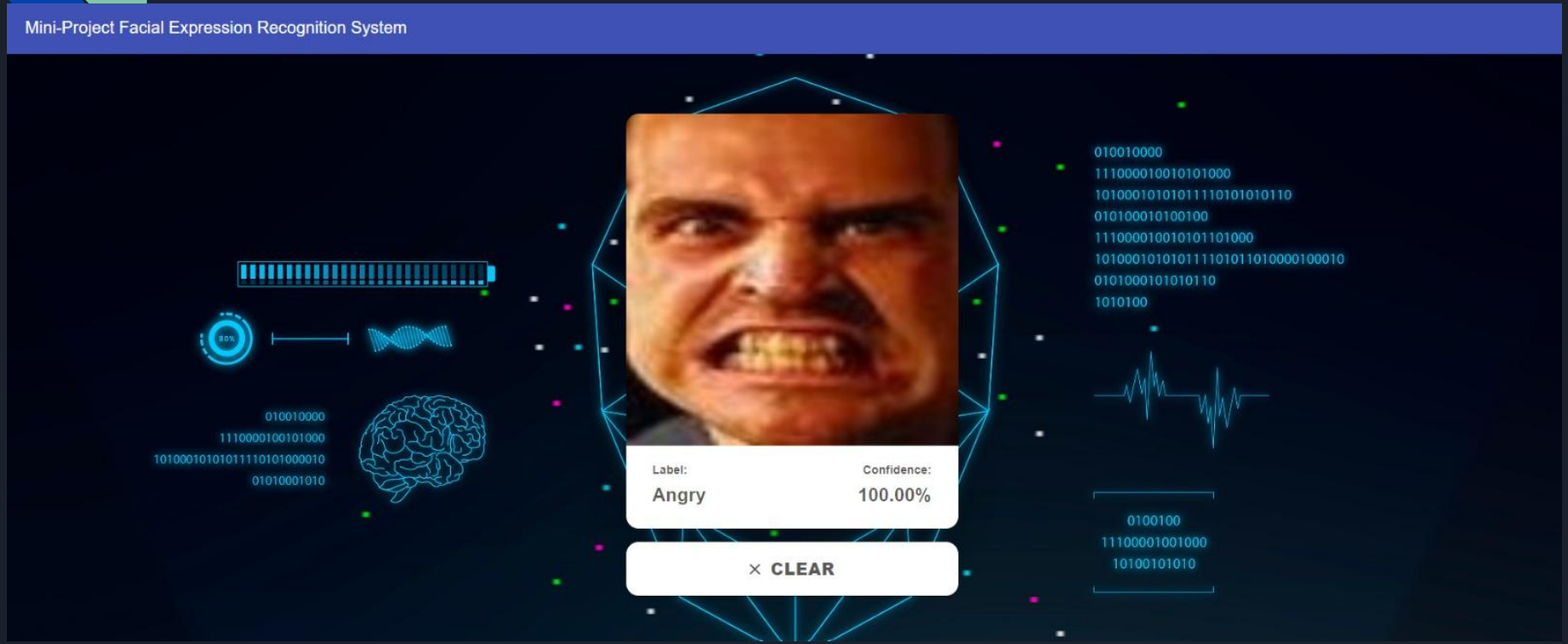
Sample Outputs:



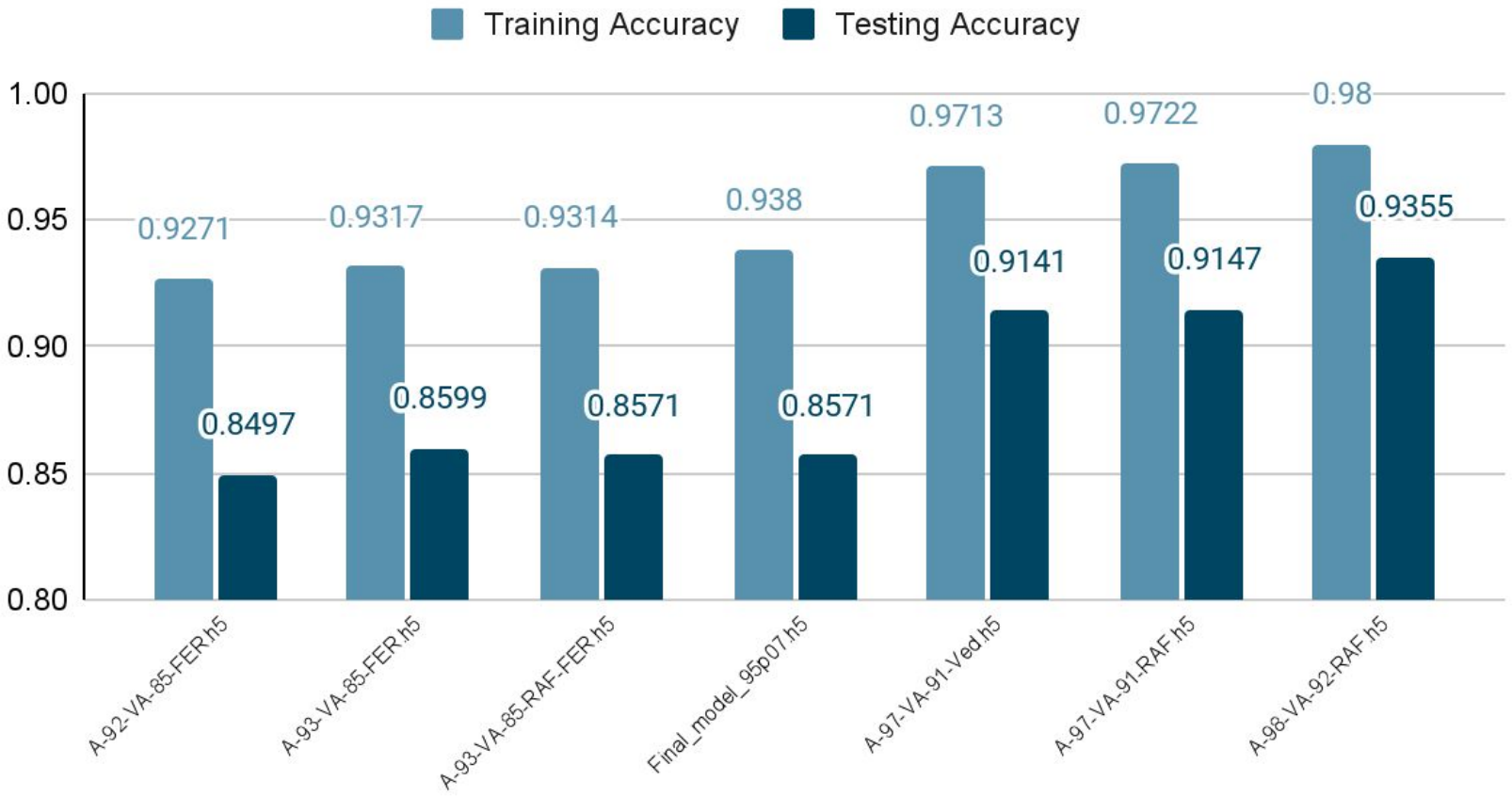
Sample Outputs:



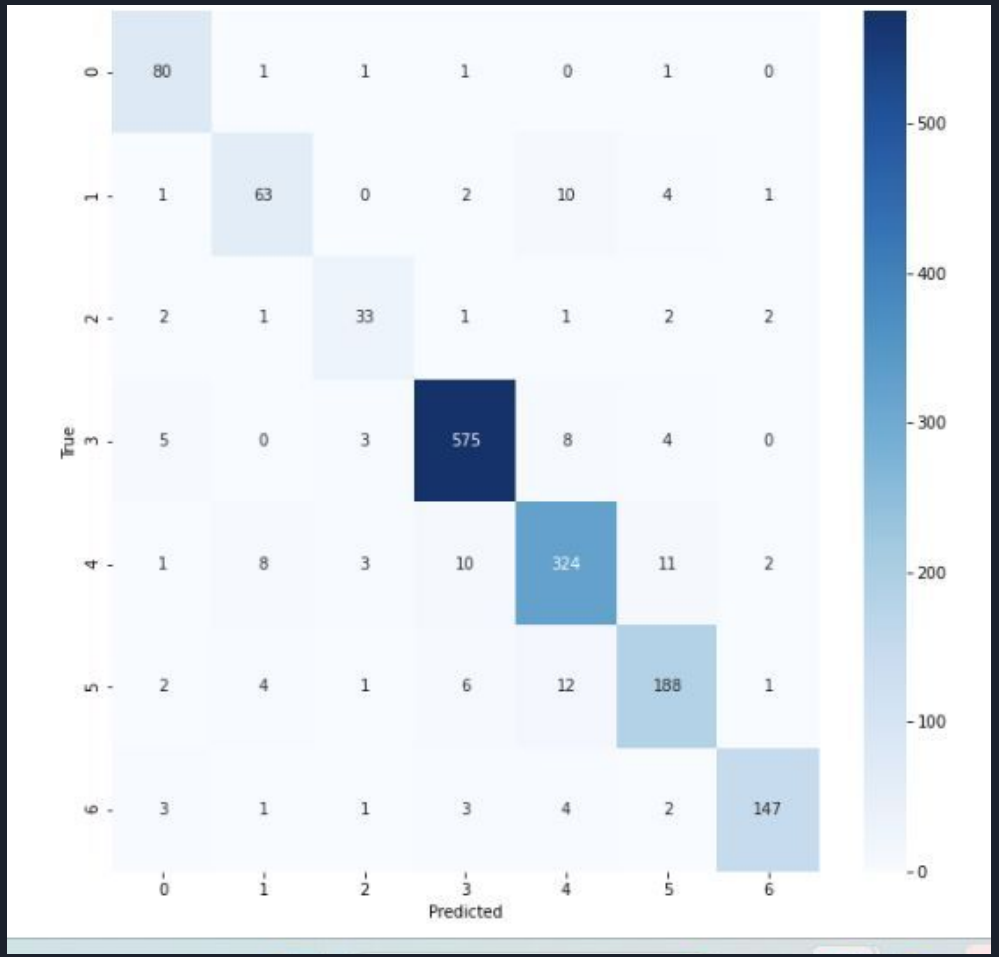
Sample Outputs:



Points scored



Confusion Matrix



References

Datasets :

1. FER-2013 [21] :- http://www.consortium.ri.cmu.edu/ckagree/
2. RaFD [41] :- http://www.socsci.ru.nl:8180/RaFD2/RaFD/
3. RAF-DB [44], [45] :- http://www.whdeng.cn/RAF/model1.html/
4. ExpW [47] :- http://mmlab.ie.cuhk.edu.hk/projects/socialrelation/index.html/

Research Papers :

1. Facial emotion recognition using deep learning: review and insights
2. Facial Expression Recognition Based on Deep Learning Convolution Neural Network: A Review
3. Emotional face expression recognition in problematic Internet use and excessive smartphone use: task-based fMRI study
4. PTZ-Camera-Based Facial Expression Analysis using Faster R-CNN for Student Engagement Recognition
5. Four-layer ConvNet to facial emotion recognition with minimal epochs and the significance of data diversity
6. A Comprehensive Survey on Deep Facial Expression Recognition: Challenges, Applications, and Future Guidelines
7. A deep prototypical learning with local attention network for dynamic micro-expression recognition
8. The Relationship Between Facial Expression and Cognitive Function in Patients With Depression
9. Fine-Grained Facial Expression Analysis Using Dimensional Emotion Model

Thank you!