Project No: 160

BEHAVIOR DETECTION FOR AI TUTOR

Project by : Samyuktha
Mentors : Ritendu Bhattacharya , Aishwarya

Project Leadership

Sharat Manikonda
Director at Innodatatics and
Sponsor
linkedin.com/in/sharat-chandra

Team Members

Name: Samyuktha
www.linkedin.com/in/samyuktha-p-506137225

Contents

- Business objective
- Business Constraints
- Project Architecture
- Data collection and details
- Exploratory Data Analysis
- Visualization
- Modeling
- Evaluation
- Deployment

Project Overview and Scope

The project involves leveraging deep learning techniques, particularly transfer learning using pre-trained models , capable of accurately detecting and interpreting facial expressions. The AI tutor will analyze students' facial cues in real-time to gauge their emotional responses during learning sessions or interviews, enabling it to adapt its teaching style and content delivery accordingly. The AI tutor aims to create a more effective learning environment, ultimately improving students' educational outcomes and assessing student/employee behavior during interviews to improve their chances of securing a job.

Project Overview and Scope

Business understanding – understanding the business problem and objective and constraint

Data understanding

- Data collection : primary data- custom facial images data collected . Total 42k images . 6k in each class
- Secondary data – images of actors and FER-2013 dataset used
- Data type : unstructured data.
- Emotions class labels : sad , angry , smile , happy , neutral , eye contact, surprise

Data preprocessing :

- Roboflow: label facial images according to each emotions
- Python : face detection using dlib→ using cv2 to crop and resize image to (224,224,3)→ classes are balanced by using Augmentor library to balance the data by image augmentation .

Model building :

- Use ImageDataGenerator from tensorflow to perform real time augmentation of the input images while training the model .
- The model employed is pretrained models like densenet169 , Xception, inception etc

Model evaluation :

We used accuracy as evaluation metrics

- The model with highest accuracy was chosen
- The model and weights are saved for deployment

Model deployment :

- The model is deployed using streamlit
- The user gives an input image ans the top three emotions along with their probability is returned as output

Monitoring and maintenance

- Continuously monitor the performance metrics of the facial recognition model, including accuracy, precision, recall, and F1-score. Set up automated monitoring systems to alert when performance metrics deviate significantly from predefined thresholds.

Business Problem

Poor posture, lack of maintaining eye contact and poor body language during interviews can give an undesirable impression on the candidate, reducing the chance of them getting job.

Business Objective

Objective

- Maximize the percentage of getting the job
- Maximize employment rate
- Maximize Job Interview Success Rate

Constraint

Minimize the mental stress of not getting the job

CRISP-ML(Q) Methodology

There are six stages of CRISP-ML(Q) Methodology

1.Business and data understanding

2.Data preparation

3.model building

4.Model evaluation

5.Model deployment

6.Monitoring and maintenance

Technical Stacks

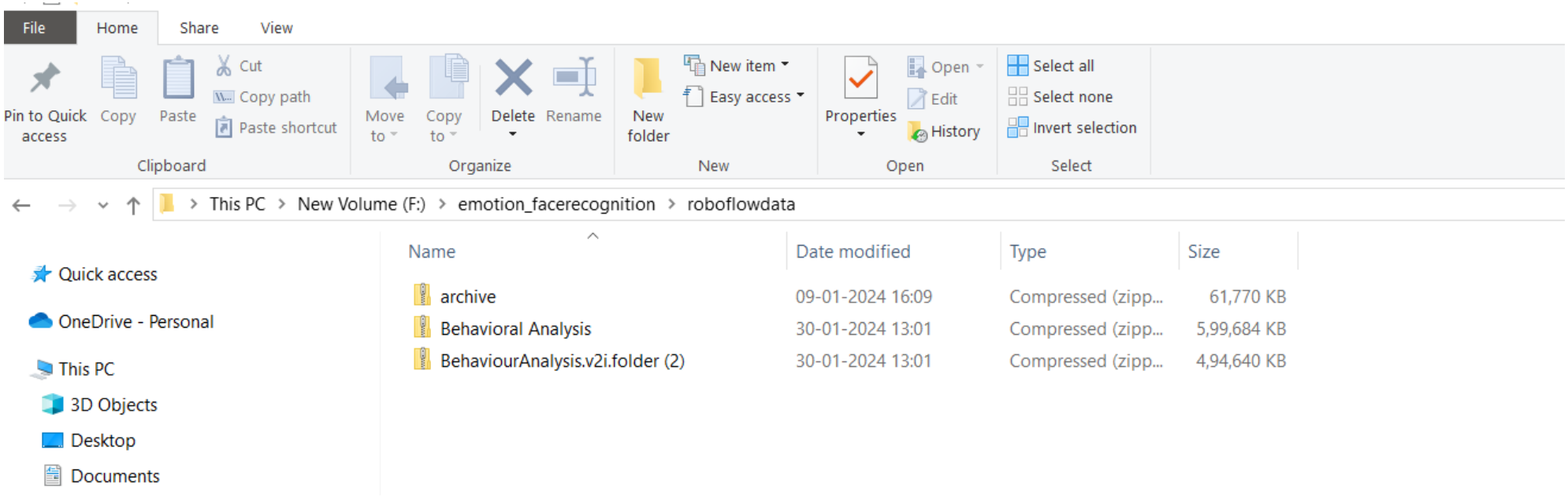
+

Data Collection and Understanding

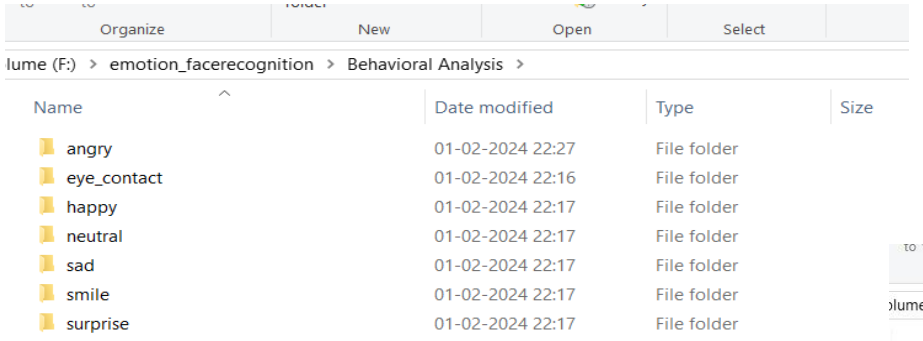
- The data collected is unstructured image dataset
- Primary data : custom facial images were used
- Secondary data: FER-2013 dataset , Actors facial expression
- The dataset was balanced using the augmentation process
- Total images : 42,000 facial expression images . 6000 in each class

Data Information

Total of 42,000 facial images were included in the project .
The datasets included were : FER-2013 and custom dataset were labelled using Roboflow

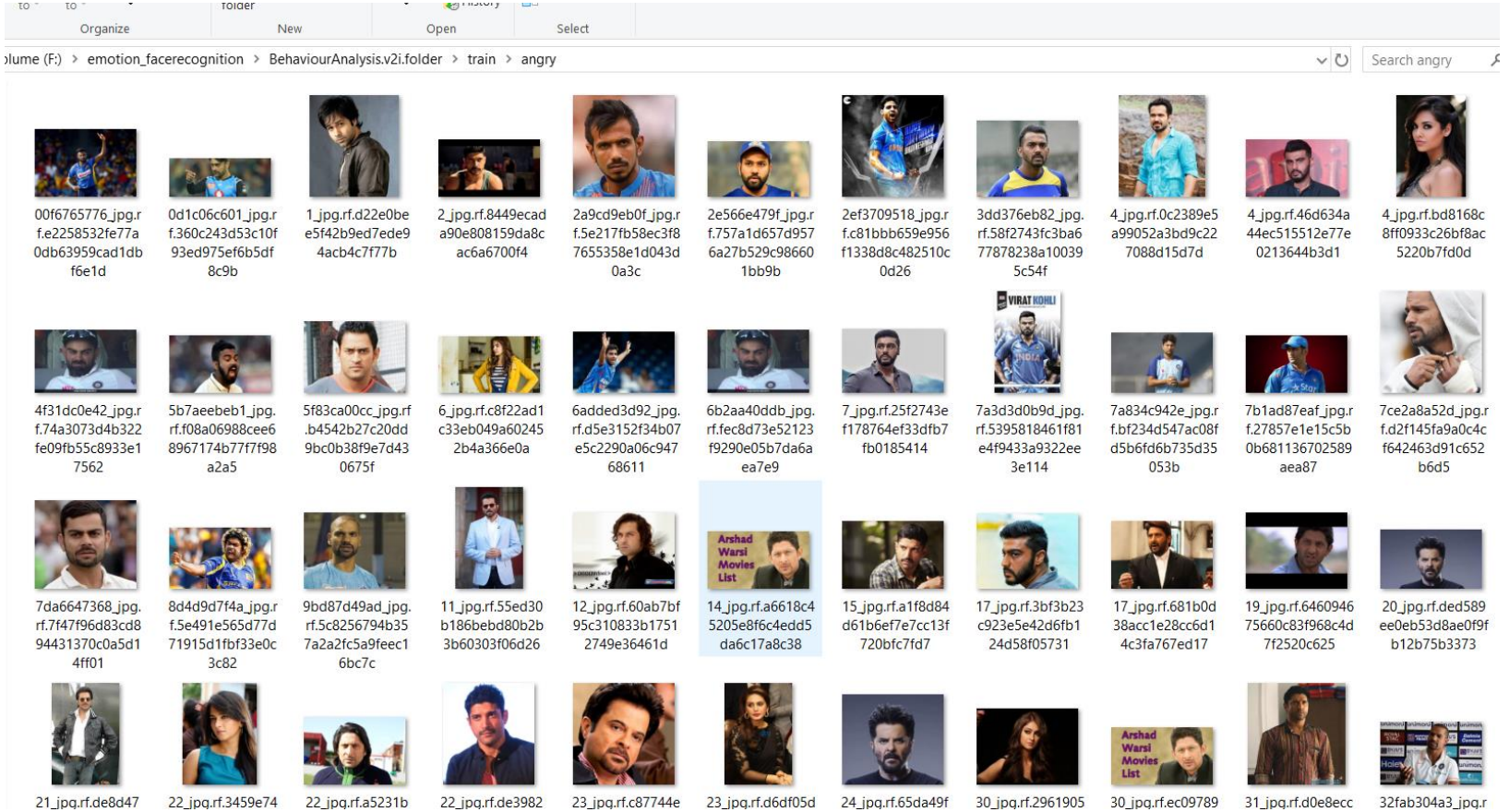


Data Dictionary



The class labels are the folder names containing those expression.
The labels include 7 emotions : eye_contact , surprise , sad , happy , smile , neutral , angry .

The images were cropped to face and resized to (224 , 224 , 3)



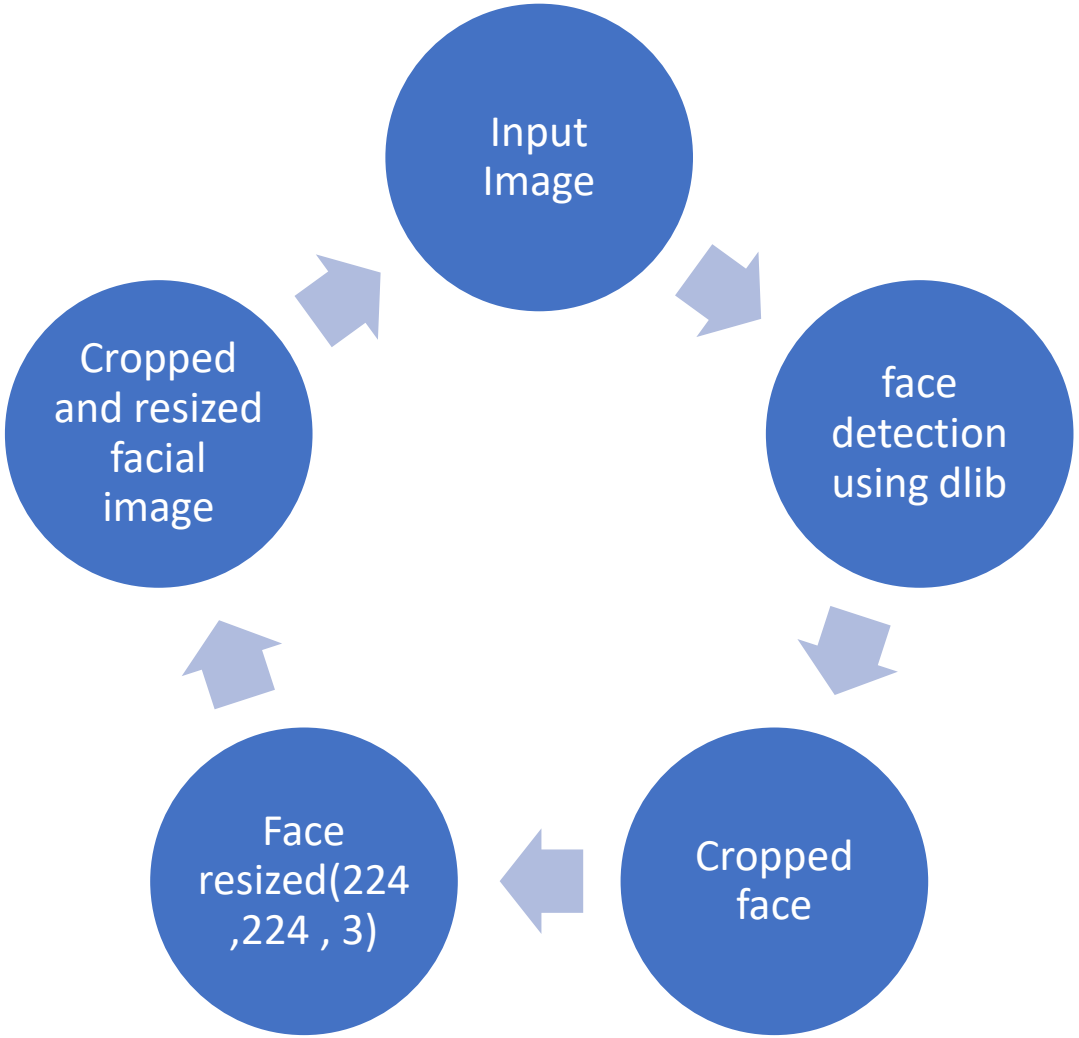
System Requirements

- Google Colab T4 GPU was utilized in building model
- Good cloud storage availability to store huge size image datasets
- minimum 4GB RAM
- Python version utilized was 3.9

Exploratory Data Analysis [EDA]

- There were 7 emotions and behaviors included as our class labels.
- angry , sad , smile , surprise , eye_contact , happy , neutral are the emotions
- the dataset was balanced and included 6000 images in each folder as class label

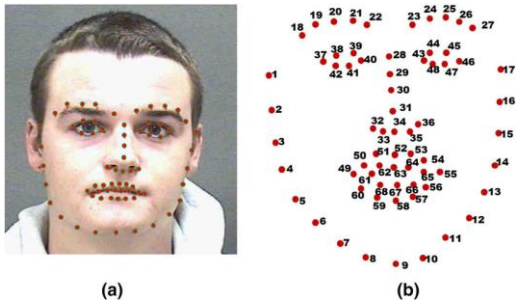
Data Preprocessing



Data Preprocessing

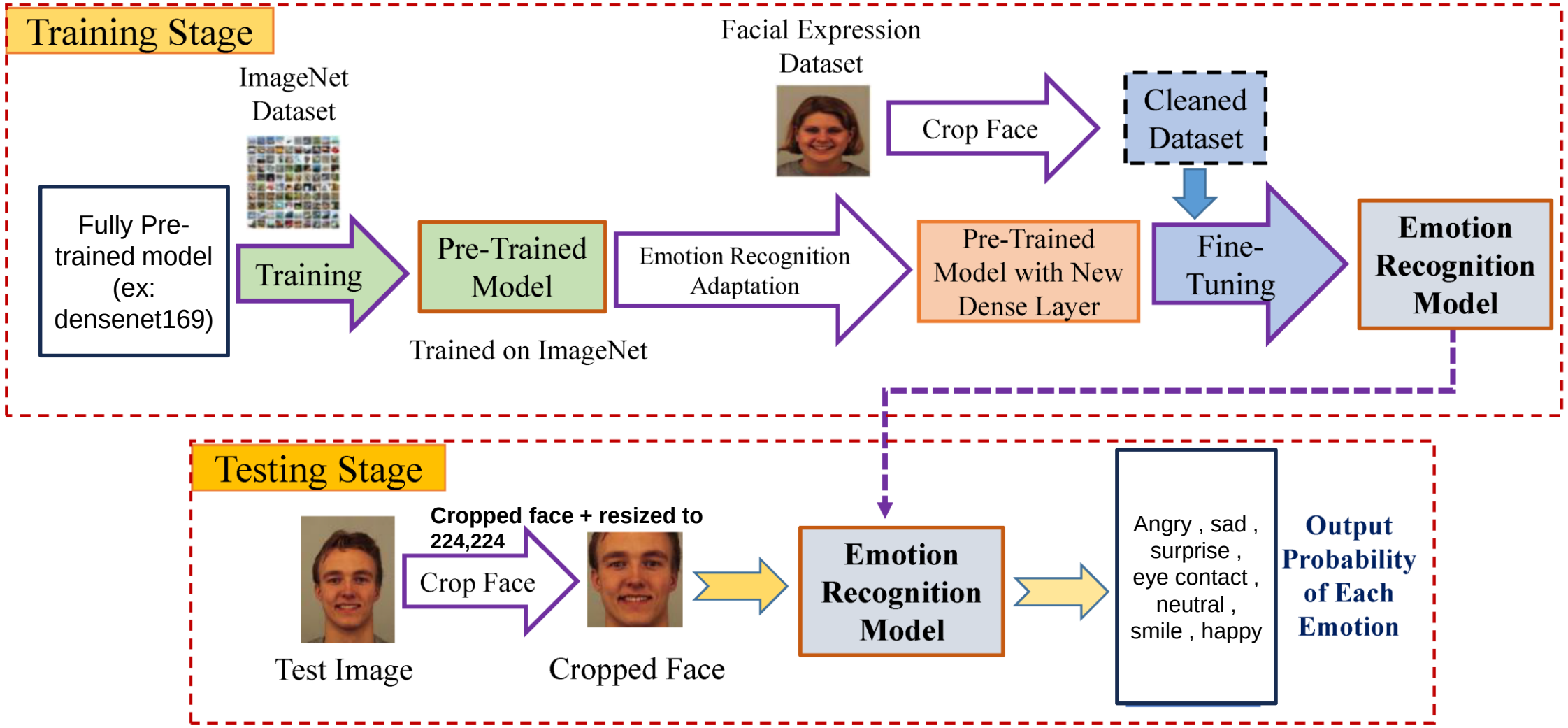
Dlib is a popular open-source library written in C++ that provides tools and algorithms for machine learning, computer vision, and image processing tasks. One of the key functionalities of Dlib is its face detection capabilities, which are commonly used in various applications such as facial recognition, emotion detection, and face tracking.

The face detection algorithm identifies regions of the image that likely contain human faces based on learned patterns and features.



OpenCV (cv2) is a widely-used library in Python for computer vision tasks, including resizing and cropping images. The images were then normalized

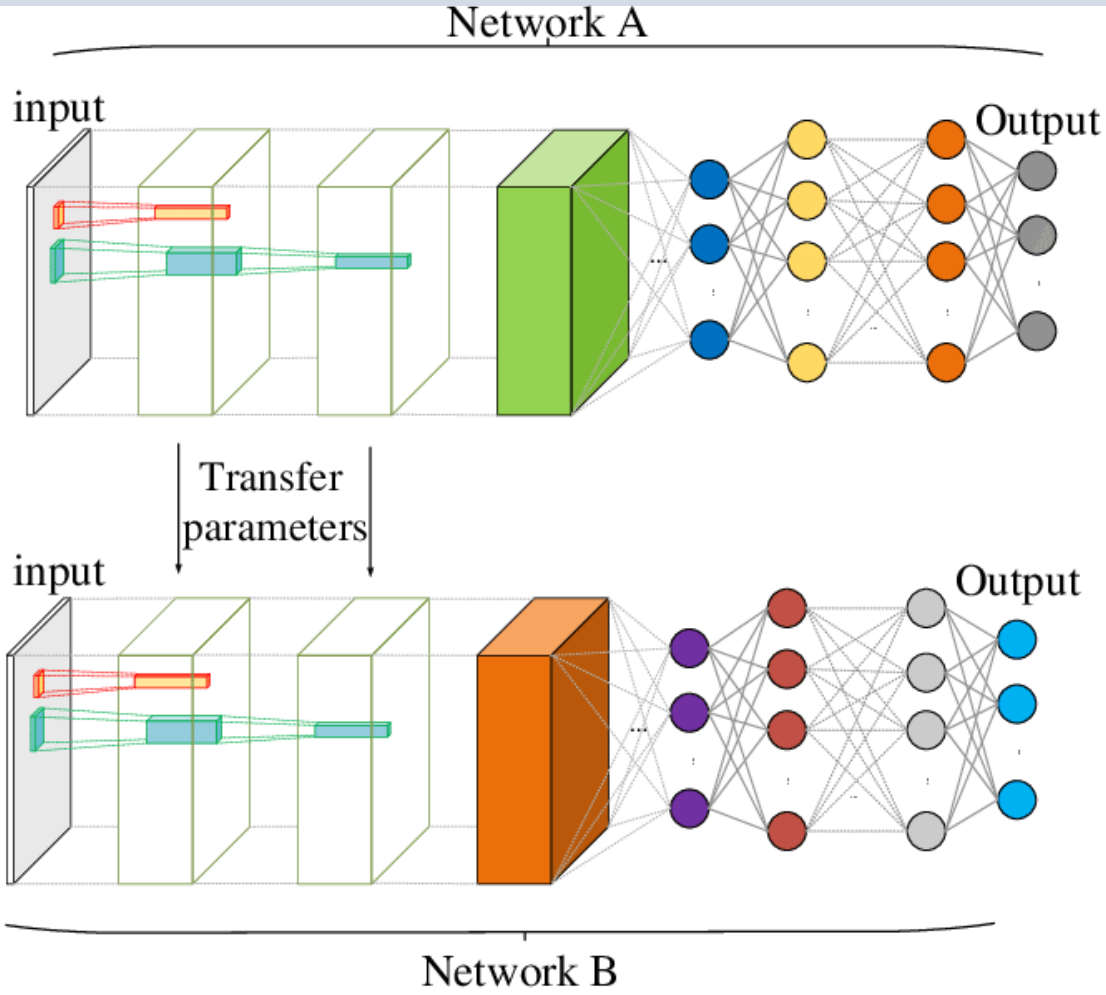
Model Building



Transfer learning

Instead of training a new model from scratch, transfer learning leverages knowledge learned from a source domain to improve learning in a target domain

Here training from their network means we won't do back propagation and initialize random weight or change the hyper-parameter, we will just give our data to the network and it will train using it's weights and profound hyper-parameters like kernel, pooling, optimizer, learning rate etc. Here what changes we have to make in pre-created network is, we will not include the last layer of the model that is the output layer, instead we will include our own custom FCL(fully connected layer) to train our dataset and at the end of our FCL we will add one output layer then we will combine our FCL with the pre-trained network. Finally we will train it.



Here you can see it graphically that we will take the network of the pre-trained network A and at the end we will add out customized layer of neural network(neurons) and train it.

Our method includes :
Split the dataset into train , validation , test (70:20:10) → pass them to Imagedatagenerator class→ Generate batches of tensor image data with real-time data augmentation→ freeze the initial layer of pretrained networks and change the output layer from 1000 to 7 neurons as there are 7 emotions→ train the model with augmented images → test it on test data → calculate the accuracy

Performed transfer learning on 5 models → densenet169 , mobilenetv2 , inceptionv3 , xception , nasnetlarge
Maximum accuracy was achieved by densenet169

Model Accuracy Comparison

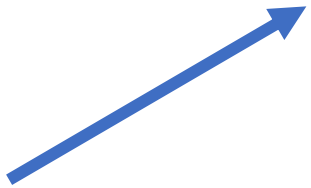
Train , val , test size	model	Epochs	Train	Val	test
29400 ,8400 , 4200	densenet169	18	93	93	92
29400 ,8400 , 4200	mobilenetv2	18	92.96	91.06	90.06
29400 ,8400 , 4200	Inceptionv3	15	82	84	85
29400 ,8400 , 4200	xception	18	91	90.48	90.86
29400 ,8400 , 4200	nasnetlarge	15	87.51	88.95	89.3

Best Model –

The best model is densenet 169 as it gave an accuracy of 93%

Model architecture :

Layers	Output Size	DenseNet 169
Convolution	112×112	7×7 conv, stride 2
Pooling	56×56	3×3 max pool, stride 2
Dense Block (1)	56×56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56×56	1×1 conv
	28×28	2×2 average pool, stride 2
Dense Block (2)	28×28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28×28	1×1 conv
	14×14	2×2 average pool, stride 2
Dense Block (3)	14×14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$
Transition Layer (3)	14×14	1×1 conv
	7×7	2×2 average pool, stride 2
Dense Block (4)	7×7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$
Classification Layer	1×1	7×7 global average pool
	1000	1000D fully-connected, softmax



Last layer instead of 1000 neurons as ImageNet have 1000 classes we will replace it with our 7 classes (7 Neurons)

Model Deployment - Strategy

Model was deployed using streamlit .
The weights and model which was saved was loaded and the uploaded images underwent preprocessing to give us the output .
The output was the top 3 emotions that are predicted and its probability values

Screen shot of output

Facial Emotion Recognition with DenseNet

Upload an image containing a face to detect emotions.

Choose an image

Drag and drop file here

Limit 200MB per file • JPG, JPEG, PNG

Browse files

AM21HAS.JPG 14.1KB

Uploaded Image

Top 3 Predicted Emotions:

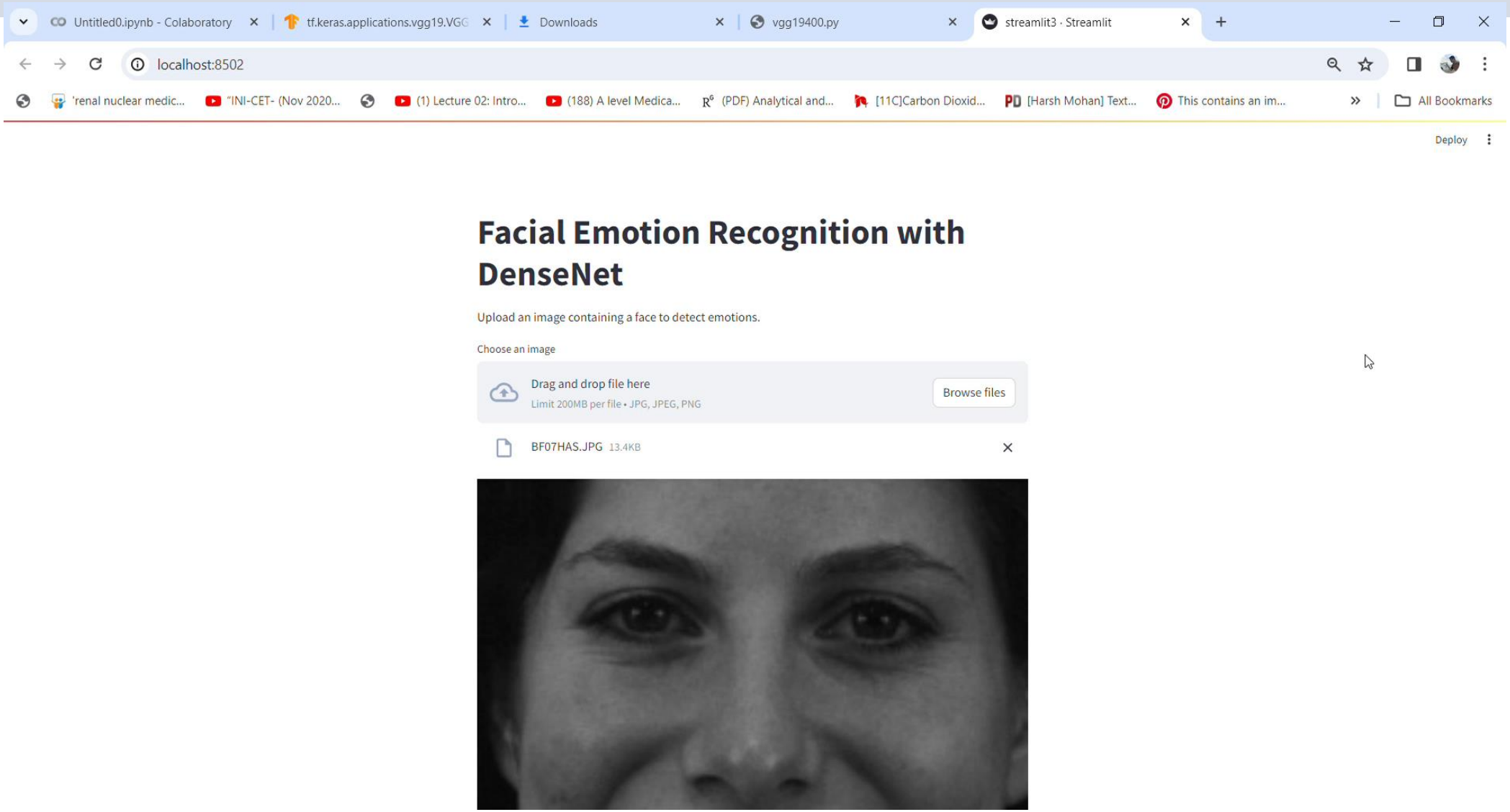
happy: 0.73

smile: 0.25

eye_contact: 0.02

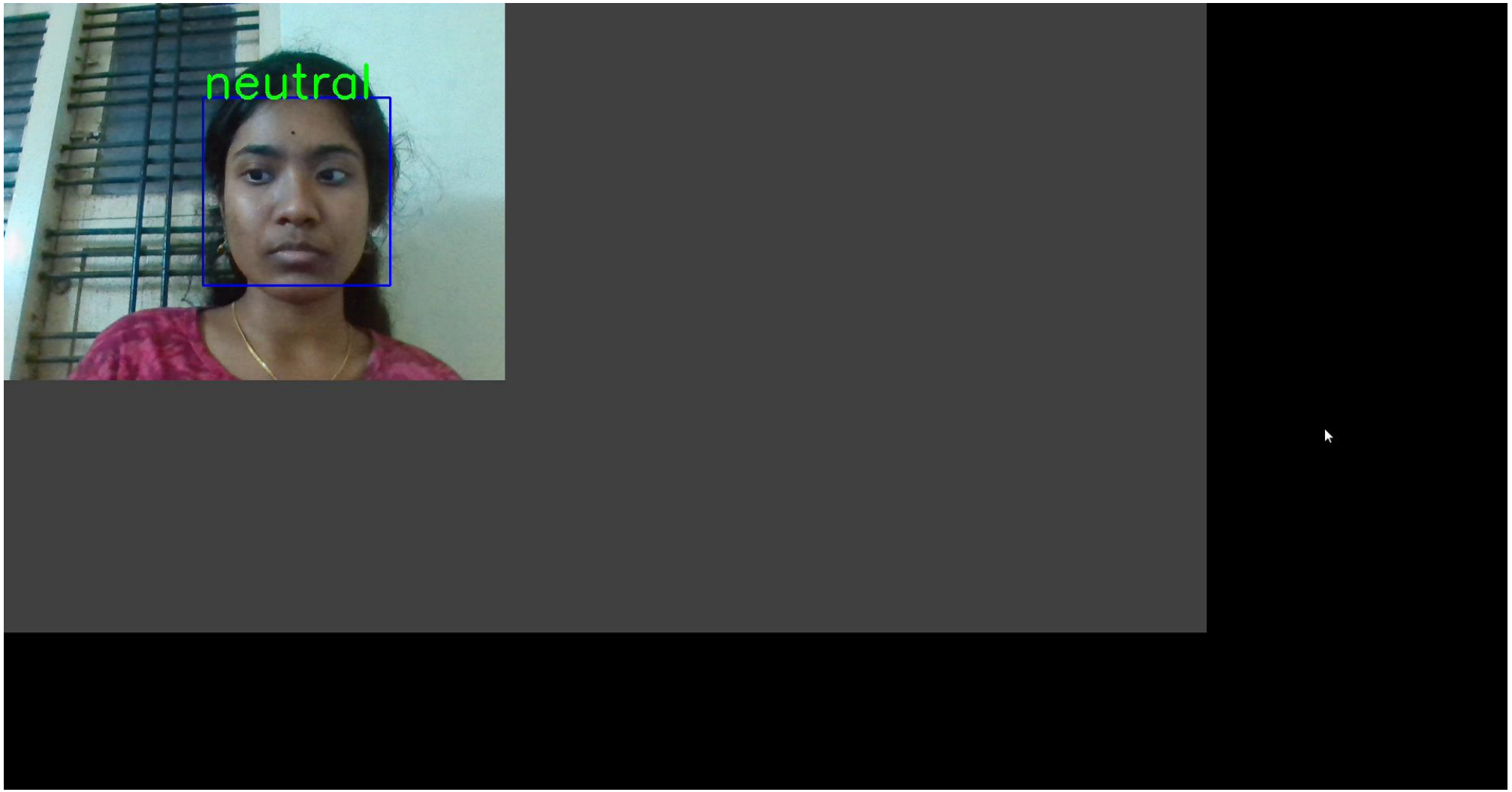
INNODATATICS
Innovation • Data • Analytics

Video of output



INNODATATICS
Innovation • Data • Analytics

Video of output



Challenges

- Poor quality of images
- Poorly illuminated pictures
- Lack of amount of data
- Intra class variation (smile vs happiness)

Future Scopes

- Enhance the AI tutor's ability to recognize and respond to students' emotional states, enabling more empathetic and supportive interactions.
- Implement real-time feedback mechanisms to provide immediate feedback on students' performance and progress
- Integrate the AI tutor with existing learning management systems (LMS) and educational platforms to streamline the deployment and adoption process.
- Integrate additional modalities such as speech recognition, and understanding emotions also through speech combined with videos/images

Queries ?


Thanks