Microcontroller-based NN model implementation for edge computing

BTP Final Presentation

Table of Contents

01

Introduction

Introduction

- Edge Computing
- Running neural networks on Raspberry Pi
- Quantization
- Running Object Recognition model
- Running Facial Recognition model
- Smart attendance system - Database and Server

02
Work Done

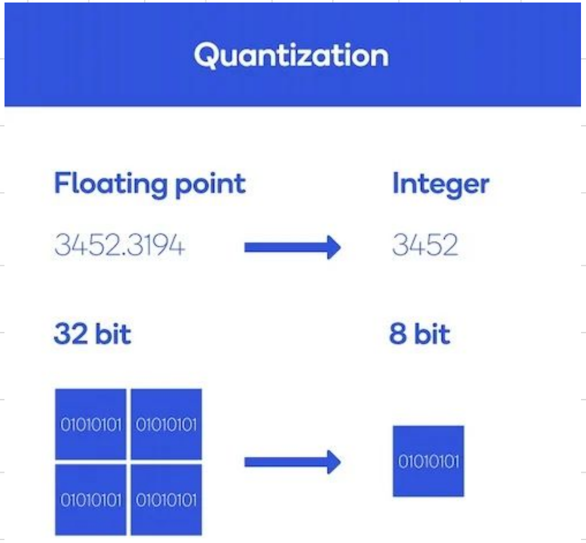
Work Done - Preparatory Work

- Raspberry Pi
- Tensorflow Lite (tflite)
- Quantization
- Driver code to run any tflite model on Raspberry Pi
- Takes frames of video as image input from the webcam and runs detection

Work Done - Set Up

Work Done - Quantization

- Convert weights and biases to integer format
- Less resource demanding
- Less calculations required, resulting into faster algorithms
- Post-training quantization



Work Done - Object Detection

- Used pretrained Coco Mobilenet v1
- Trained for 1000 classes over Coco dataset
- Supports multi-object detection taking image formed by frames of a video
- Frame is created around the recognized object showing the probability of the object being in the recognized class

Work Done - Object Detection

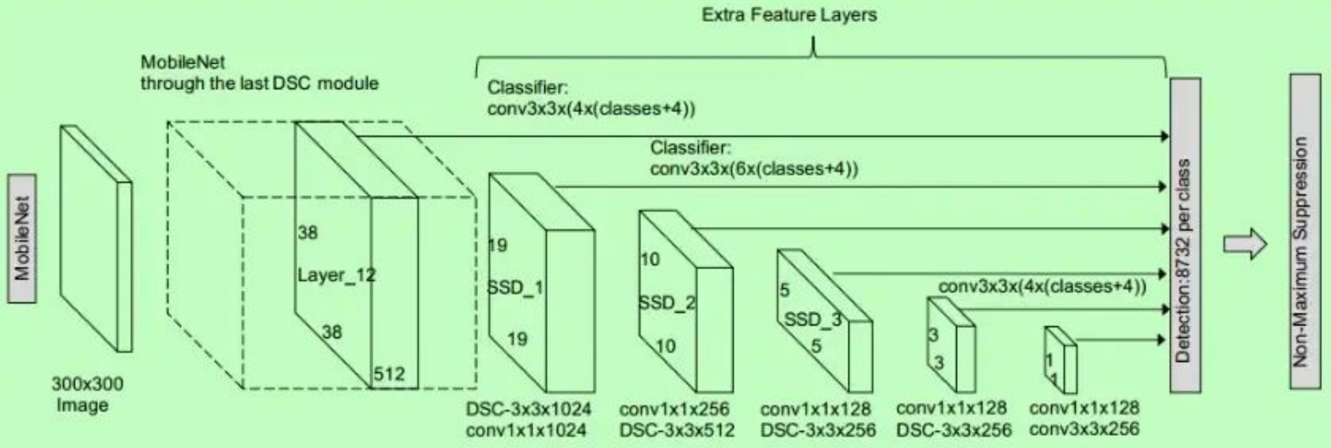
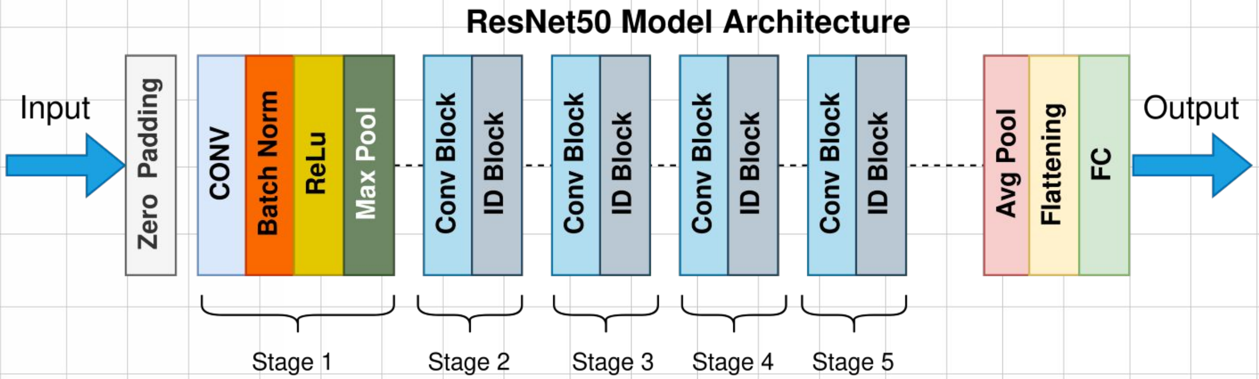


Figure 6. Mobile-Det: SSD-based detection with MobileNet as backbone, modified based on [14]

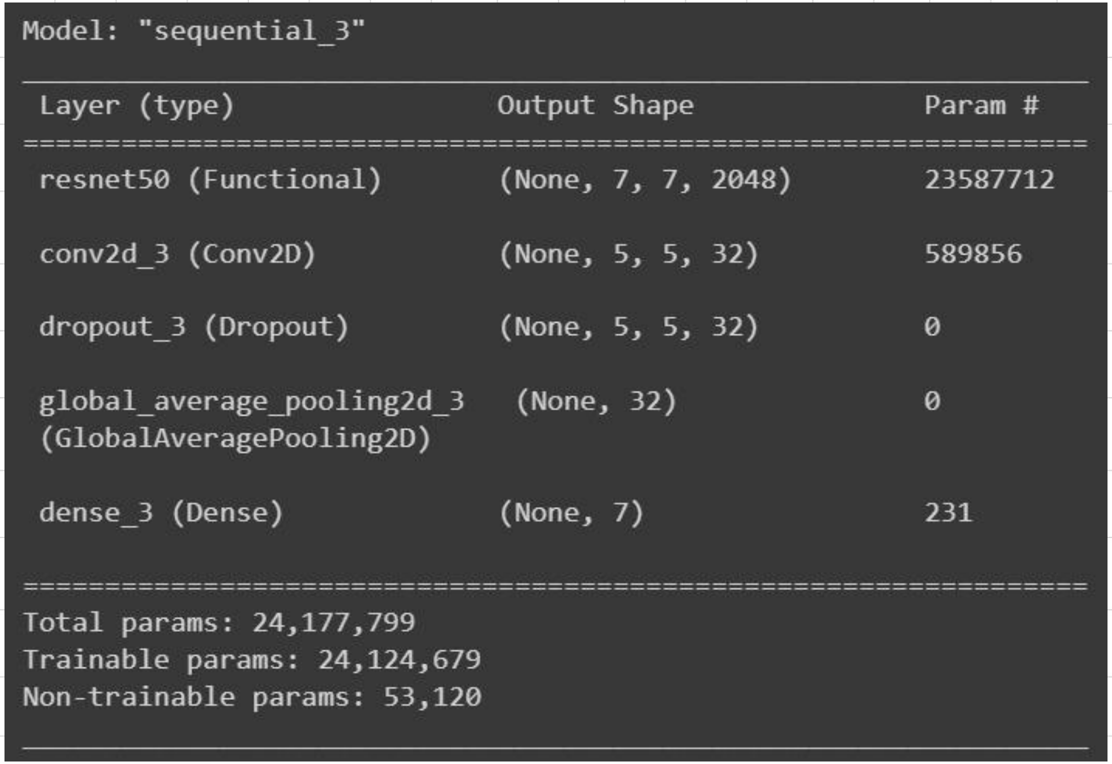
Work Done – Facial Recognition

- Performed Transfer Learning using ResNet 50 model
- ResNet 50 is a 50 layer deep convolution neural network (CNN)
- Added extra layers to ResNet 50's image recognition model and trained with 501 images in each class
- Batch size 64, Learning Rate 0.001, ADAM optimizer, Sparse categorical cross entropy loss, softmax activation in final layer
- Quantized model will run on Raspberry Pi

Work Done - Facial Recognition



Work Done - Facial Recognition



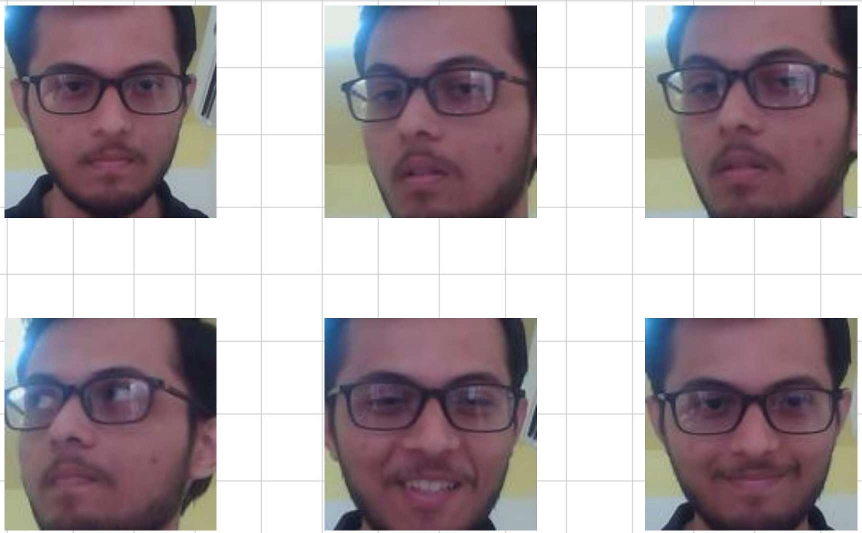
```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 7, 7, 2048)        23587712

 conv2d_3 (Conv2D)           (None, 5, 5, 32)          589856

 dropout_3 (Dropout)         (None, 5, 5, 32)          0

 global_average_pooling2d_3   (None, 32)               0
 (GlobalAveragePooling2D)

 dense_3 (Dense)             (None, 7)                 231

=================================================================
Total params: 24,177,799
Trainable params: 24,124,679
Non-trainable params: 53,120
_________________________________________________________________
```

Work Done - Data Collection

- Wrote a python script to detect face in a photograph by detecting forehead
- 501 images at different angles are collected when the above mentioned script is run
- Images are stored in a specific folder with label given from the input

Work Done - Data Collection

- This is how the collected images of the faces look



Work Done - Setting up tflite in raspberry pi

- We set up tflite library in raspberry pi following the instructions on the documentation on the official site of tensorflow
- We set up an environment which includes all the necessary libraries needed to run inference in the raspberry pi.
- We wrote python scripts to run inference

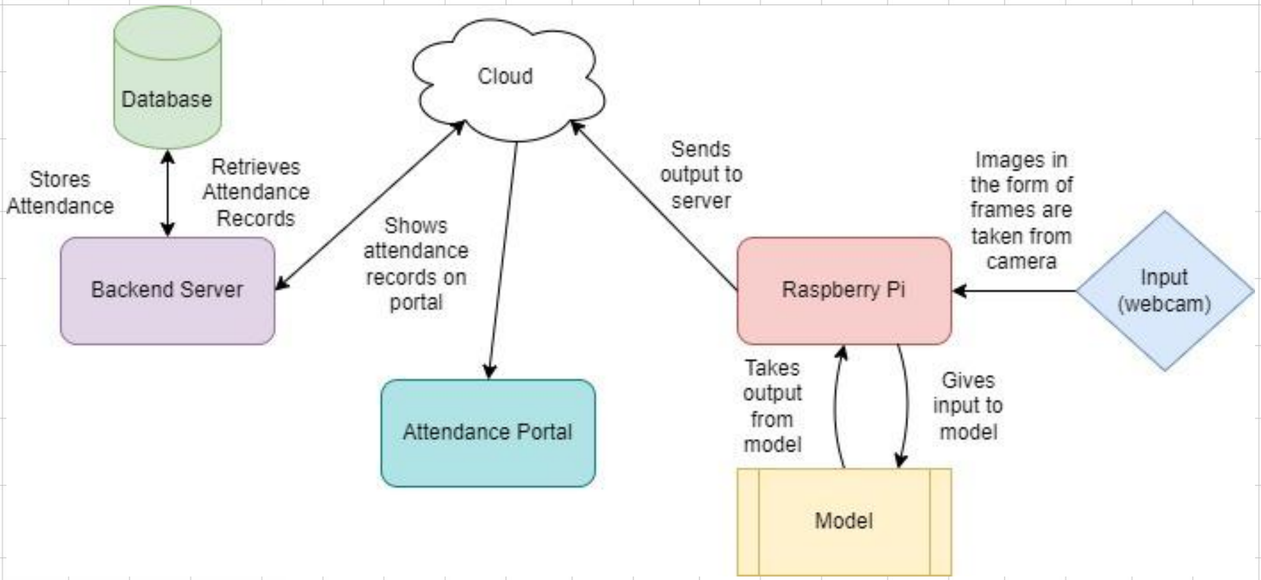
Work Done - Database and Server

- A server code was written in NodeJS to connect the Raspberry Pi to a database
- The list of names of students in a frame will be sent to the database along with the date and time when the recognition happened
- A MongoDB based database was created to store the above mentioned information

Work Done - Website

- A website was created to view the attendance records
- The website was hosted with Heroku app
- Can be modified later based on the requirements including security and authentication features
- Link - https://btp-attendance-server.herokuapp.com/show-attendance-records

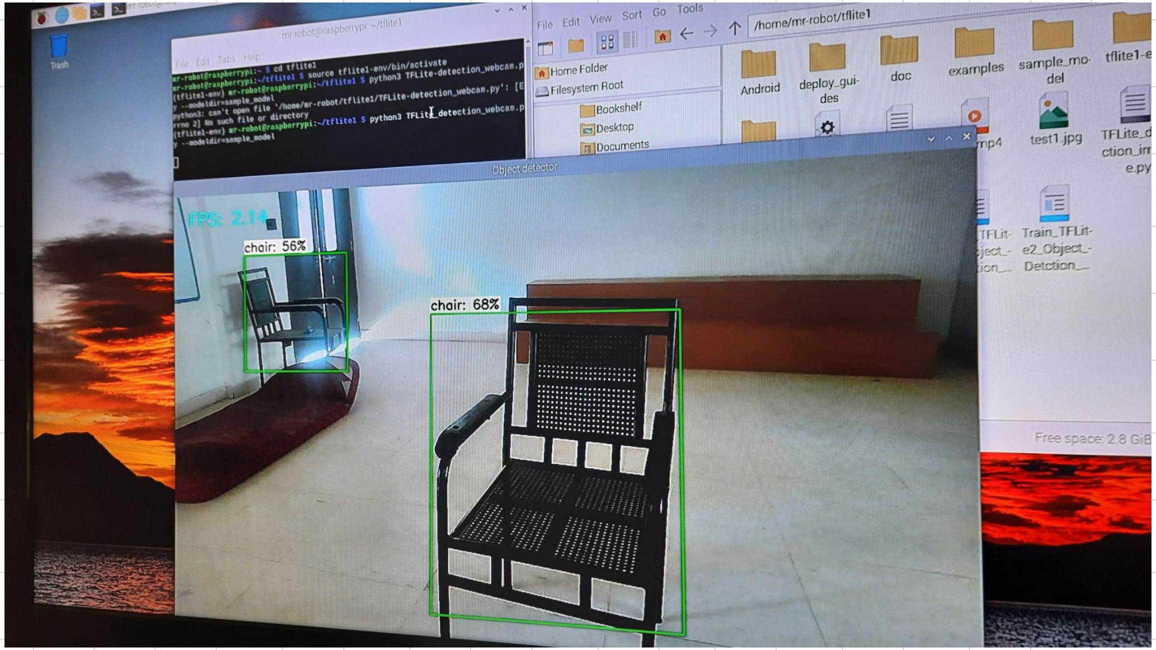
Work Done - Workflow



03
Results

Results - Object Detection

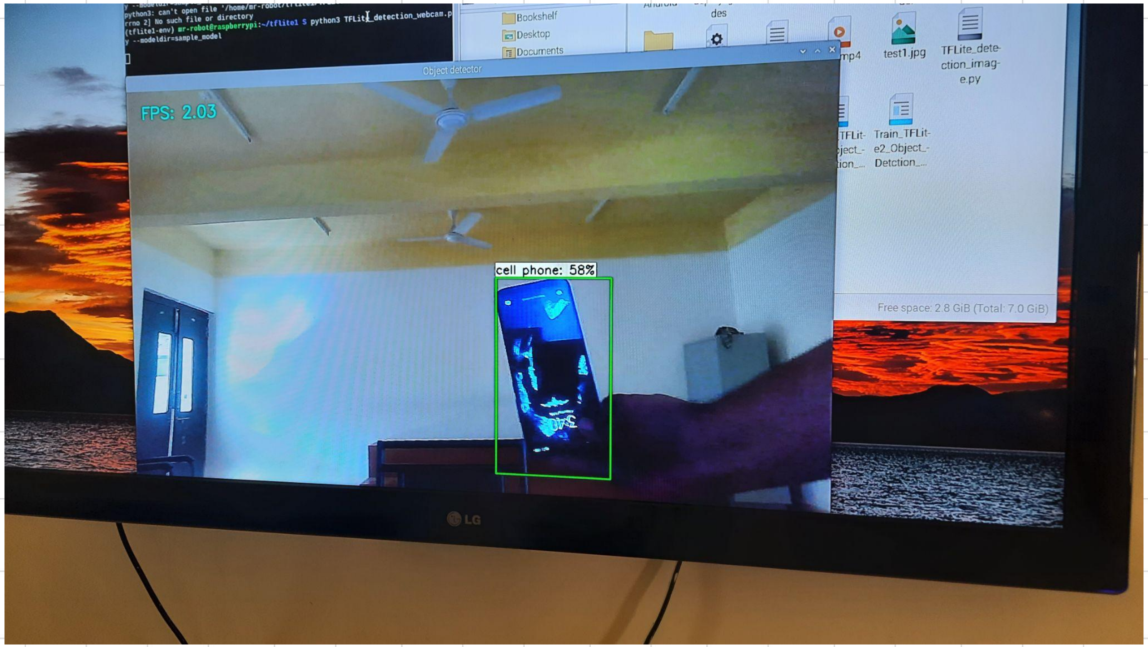
- Sample Image



```
mr-robot@raspberrypi:~ $ cd tflite1
mr-robot@raspberrypi:~/tflite1 $ source tflite1-env/bin/activate
(tflite1-env) mr-robot@raspberrypi:~/tflite1 $ python3 TFLite-detection_webcam.p
y --modeldir=sample_model
python3: can't open file '/home/mr-robot/tflite1/TFLite-detection_webcam.py': [E
rrno 2] No such file or directory
(tflite1-env) mr-robot@raspberrypi:~/tflite1 $ python3 TFLite_detection_webcam.p
y --modeldir=sample_model
```

Results – Object Detection

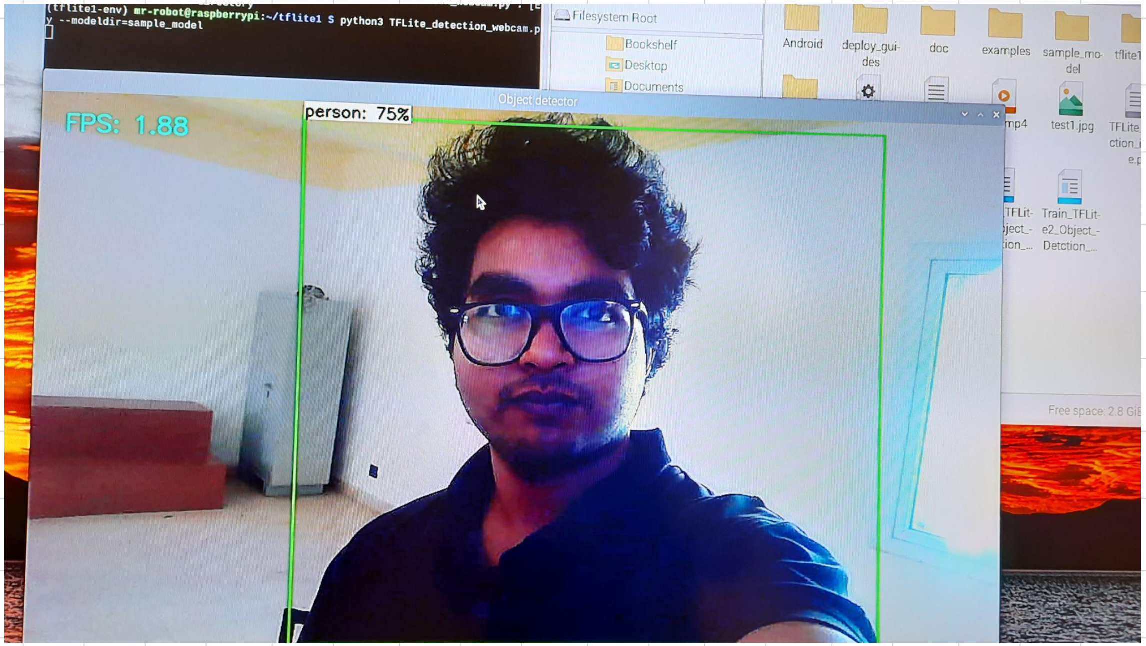
- Sample Image



```
python3: can't open file '/home/mr-robot/tflite1/
rrno 2] No such file or directory
(tflite1-env) mr-robot@raspberrypi:~/tflite1 $ python3 TFLite_detection_webcam.p
y --modeldir=sample_model
```

Results - Object Detection

- Sample Image



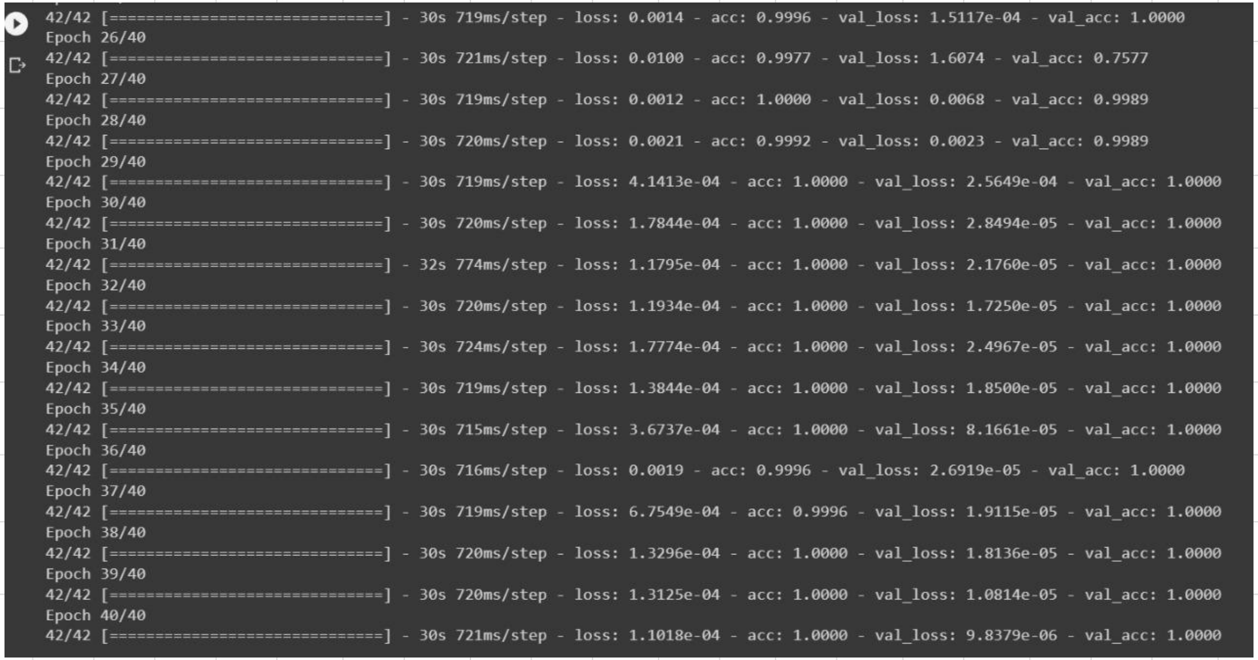
```
(tflite1-env) mr-robot@raspberrypi:~/tflite1 $ python3 TFLite_detection_webcam.p
y --modeldir=sample_model
```

Results - Object Detection

- Sample Videos
- Link - https://drive.google.com/file/d/1HKo-F2PTybEAVFqi61Mrsa-5Mgg-qlyG/view
- Link - https://drive.google.com/file/d/1J-LZx3qjub82PPkyHA2g4ZqnZjLoYoha/view

Results – Training model

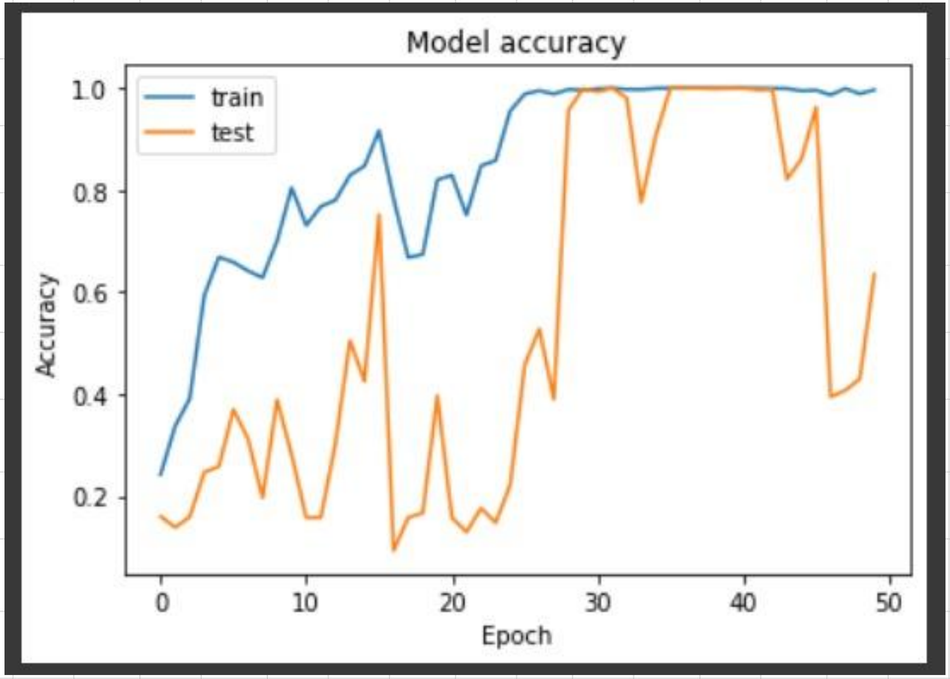
- Accuracy while training of facial recognition model



```
42/42 [==============================] - 30s 719ms/step - loss: 0.0014 - acc: 0.9996 - val_loss: 1.5117e-04 - val_acc: 1.0000
Epoch 26/40
42/42 [==============================] - 30s 721ms/step - loss: 0.0100 - acc: 0.9977 - val_loss: 1.6074 - val_acc: 0.7577
Epoch 27/40
42/42 [==============================] - 30s 719ms/step - loss: 0.0012 - acc: 1.0000 - val_loss: 0.0068 - val_acc: 0.9989
Epoch 28/40
42/42 [==============================] - 30s 720ms/step - loss: 0.0021 - acc: 0.9992 - val_loss: 0.0023 - val_acc: 0.9989
Epoch 29/40
42/42 [==============================] - 30s 719ms/step - loss: 4.1413e-04 - acc: 1.0000 - val_loss: 2.5649e-04 - val_acc: 1.0000
Epoch 30/40
42/42 [==============================] - 30s 720ms/step - loss: 1.7844e-04 - acc: 1.0000 - val_loss: 2.8494e-05 - val_acc: 1.0000
Epoch 31/40
42/42 [==============================] - 32s 774ms/step - loss: 1.1795e-04 - acc: 1.0000 - val_loss: 2.1760e-05 - val_acc: 1.0000
Epoch 32/40
42/42 [==============================] - 30s 720ms/step - loss: 1.1934e-04 - acc: 1.0000 - val_loss: 1.7250e-05 - val_acc: 1.0000
Epoch 33/40
42/42 [==============================] - 30s 724ms/step - loss: 1.7774e-04 - acc: 1.0000 - val_loss: 2.4967e-05 - val_acc: 1.0000
Epoch 34/40
42/42 [==============================] - 30s 719ms/step - loss: 1.3844e-04 - acc: 1.0000 - val_loss: 1.8500e-05 - val_acc: 1.0000
Epoch 35/40
42/42 [==============================] - 30s 715ms/step - loss: 3.6737e-04 - acc: 1.0000 - val_loss: 8.1661e-05 - val_acc: 1.0000
Epoch 36/40
42/42 [==============================] - 30s 716ms/step - loss: 0.0019 - acc: 0.9996 - val_loss: 2.6919e-05 - val_acc: 1.0000
Epoch 37/40
42/42 [==============================] - 30s 719ms/step - loss: 6.7549e-04 - acc: 0.9996 - val_loss: 1.9115e-05 - val_acc: 1.0000
Epoch 38/40
42/42 [==============================] - 30s 720ms/step - loss: 1.3296e-04 - acc: 1.0000 - val_loss: 1.8136e-05 - val_acc: 1.0000
Epoch 39/40
42/42 [==============================] - 30s 720ms/step - loss: 1.3125e-04 - acc: 1.0000 - val_loss: 1.0814e-05 - val_acc: 1.0000
Epoch 40/40
42/42 [==============================] - 30s 721ms/step - loss: 1.1018e-04 - acc: 1.0000 - val_loss: 9.8379e-06 - val_acc: 1.0000
```

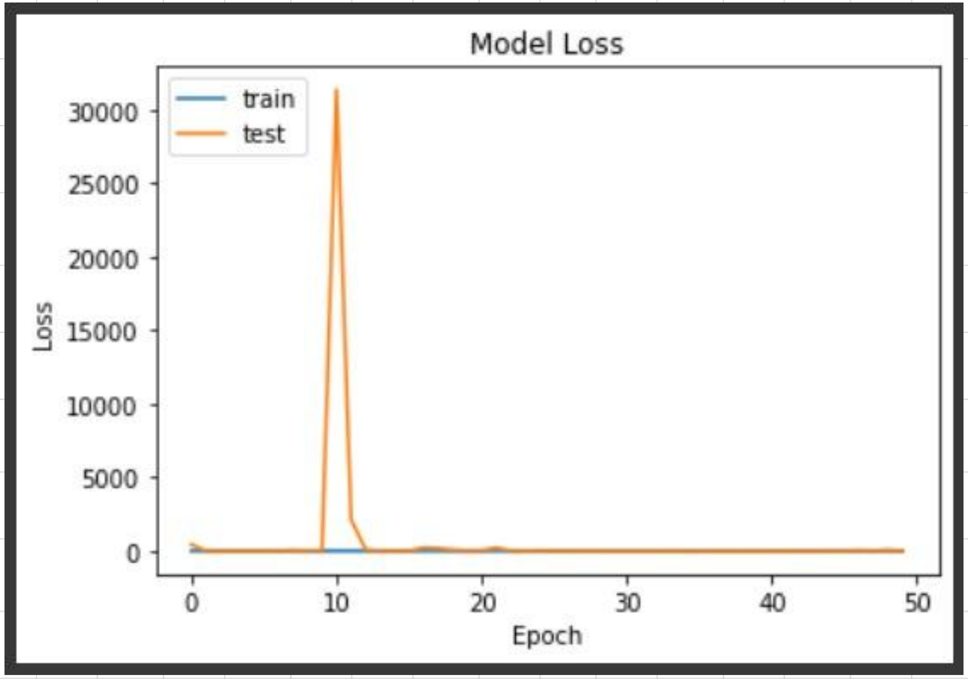
Results - Training model

- Model accuracy over epochs for training and testing set



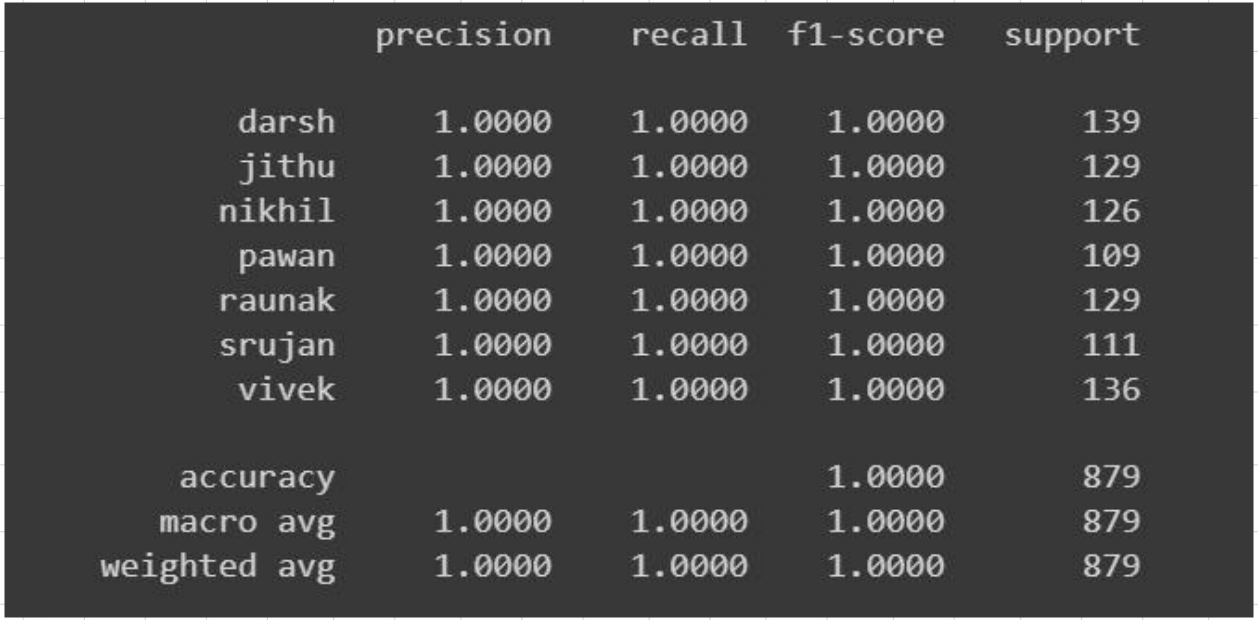
Results - Training model

- Model loss over epochs for training and testing set



Results - Training model

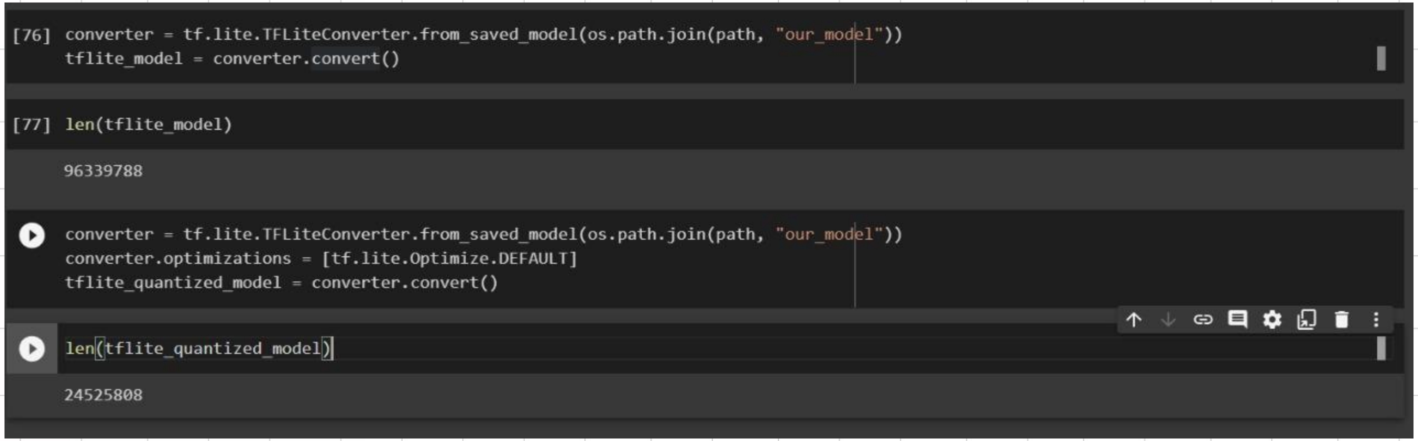
- Class wise performance



```
              precision    recall  f1-score   support

       darsh     1.0000    1.0000    1.0000       139
       jithu     1.0000    1.0000    1.0000       129
      nikhil     1.0000    1.0000    1.0000       126
       pawan     1.0000    1.0000    1.0000       109
      raunak     1.0000    1.0000    1.0000       129
      srujan     1.0000    1.0000    1.0000       111
       vivek     1.0000    1.0000    1.0000       136

    accuracy                         1.0000       879
   macro avg     1.0000    1.0000    1.0000       879
weighted avg     1.0000    1.0000    1.0000       879
```

Results - Training model

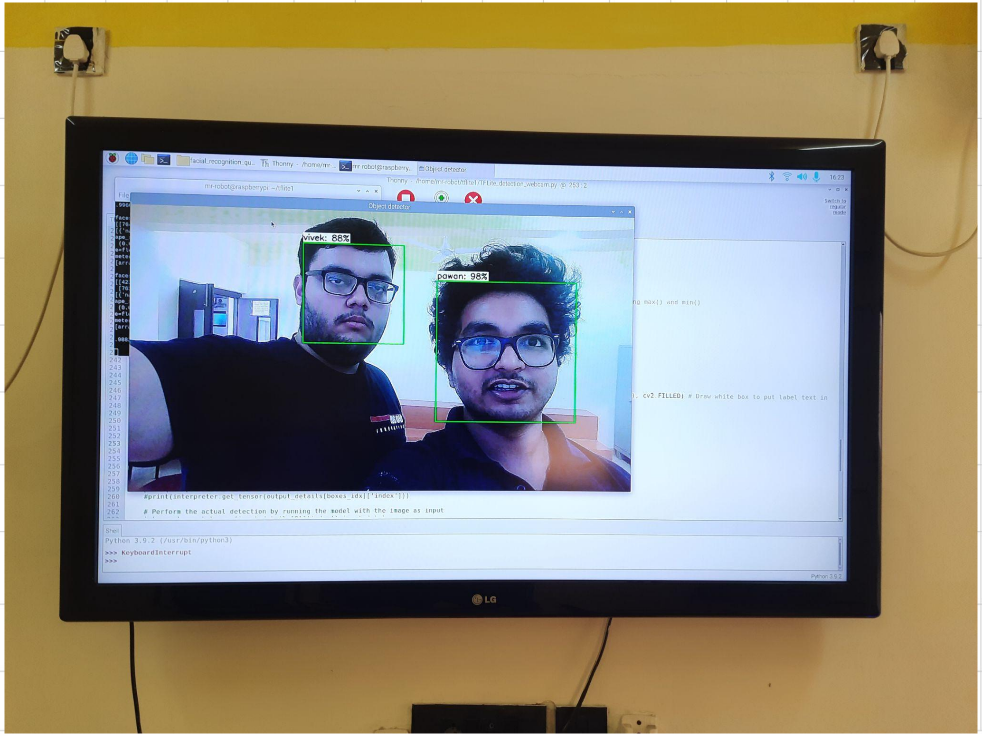
- Size comparison of quantized and not quantized models



```
[76] converter = tf.lite.TFLiteConverter.from_saved_model(os.path.join(path, "our_model"))
     tflite_model = converter.convert()

[77] len(tflite_model)

     96339788

     converter = tf.lite.TFLiteConverter.from_saved_model(os.path.join(path, "our_model"))
     converter.optimizations = [tf.lite.Optimize.DEFAULT]
     tflite_quantized_model = converter.convert()

     len(tflite_quantized_model)

     24525808
```

Results - Facial Recognition

- Sample Image

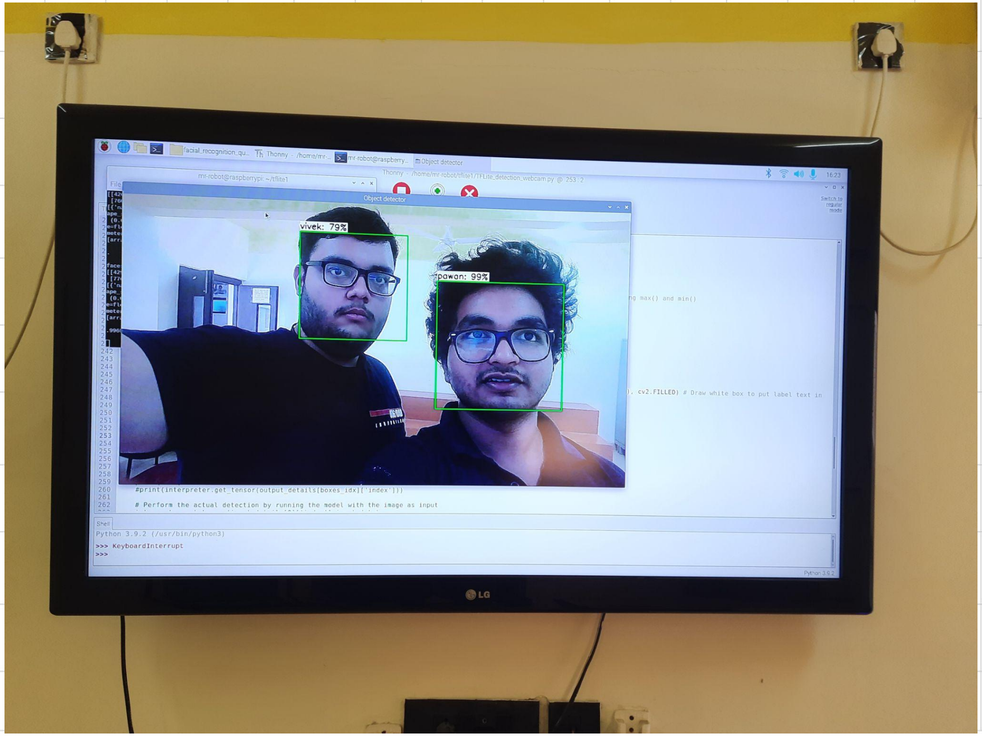


```
#print(interpreter.get_tensor(output_details[boxes_idx]['index']))

# Perform the actual detection by running the model with the image as input
```

```
Python 3.9.2 (/usr/bin/python3)
>>> KeyboardInterrupt
>>>
```

Results - Facial Recognition

- Sample Image



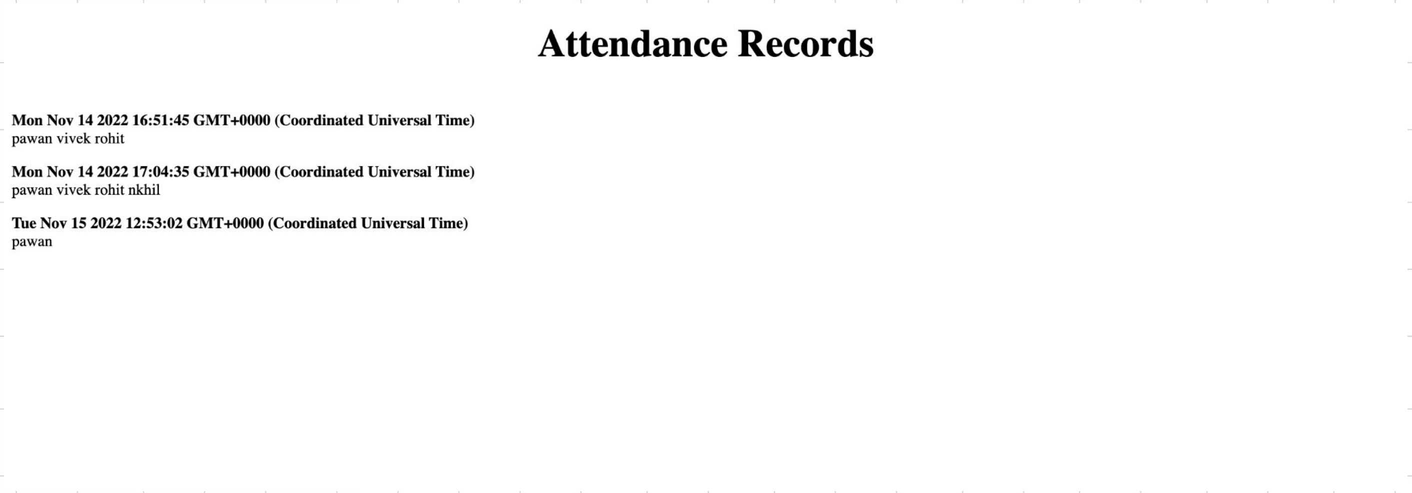
```
#print(interpreter.get_tensor(output_details[boxes_idx]['index']))

# Perform the actual detection by running the model with the image as input
```

```
Python 3.9.2 (/usr/bin/python3)
>>> KeyboardInterrupt
>>>
```

Results - Facial Recognition

- Sample Video
- Link - https://drive.google.com/file/d/1GA2opoFzfRVyHoPNSVNvHJYZgD4l1OEa/view

Results - Server and Database


Attendance Records

Mon Nov 14 2022 16:51:45 GMT+0000 (Coordinated Universal Time)
pawan vivek rohit

Mon Nov 14 2022 17:04:35 GMT+0000 (Coordinated Universal Time)
pawan vivek rohit nkhil

Tue Nov 15 2022 12:53:02 GMT+0000 (Coordinated Universal Time)
pawan

Results - GitHub

- All the codes and the models were uploaded to a GitHub repository
- Link - https://github.com/PawanSuryavanshi95/BTP-Edge-Computing
- All the results can be reproduced by following the instructions mentioned in the readme file

04
Conclusion

Conclusion

- Object Recognition and Facial Recognition models were successfully run on the Raspberry Pi kit
- Results are uploaded with the final report
- Code can be accessed from the mentioned GitHub repository
- There can be numerous applications of edge computing
- Only smart attendance system is explored in this project

Future Work

- The robustness of the model needs to be tested when trained over large data (data for only 7 students was recorded in this case)
- The attendance record website can be improved based on requirements (including the features of security and authentication)

Questions?

References

- GitHub repository - https://github.com/PawanSuryavanshi95/BTP-Edge-Computing
- Coco Mobilenet - https://www.tensorflow.org/lite/examples/object_detection/overview
- Resnet 50 - https://viso.ai/deep-learning/resnet-residual-neural-network/
- Attendance records website - https://btp-attendance-server.herokuapp.com/show-attendance-records

Courtesy

Team Members:
Pareek Vivek Manishkumar (B19EE061)
Pawandeep Suryavanshi (B19EE063)

Supervisor:
Dr. Binod Kumar

Thanks!