Capstone Project

Face Recognition & Drowsiness Detection

By - Dishant Toraskar

Content

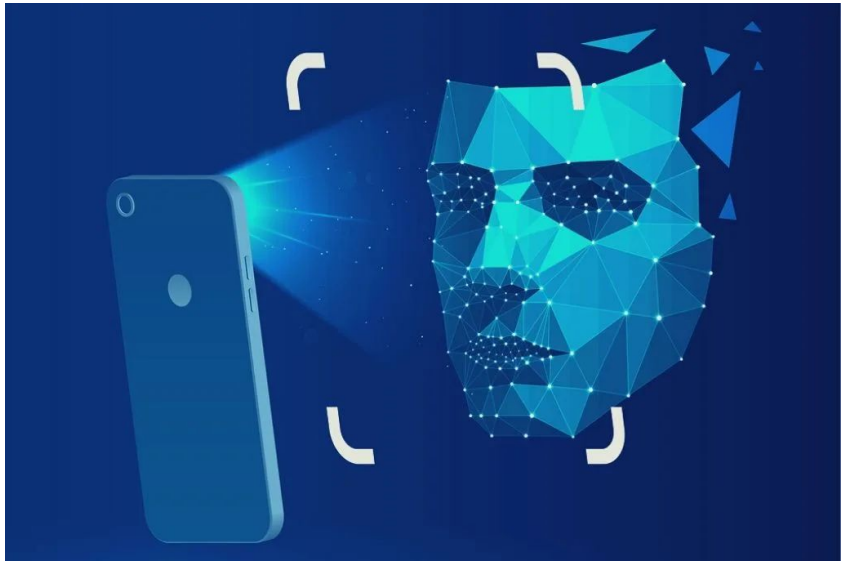
Introduction

In a physical classroom during a lecture teacher can see the faces and assess the emotion of the class and tune their lecture accordingly, whether lecture is going fast or slow. Teacher 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 analysed 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 analysed and tracked.

Problem Statement

1. Face Recognition -
   Build a face recognition system to identify the student name and if the legal student is attending the lecture and markdown the attendance and log session time of the student.

2. Drowsiness Detection -
   This will detect facial landmarks and extract the eye regions. The algorithm should be able to identify students who are not attentive and drowsing in the class.



Approach

Face Recognition

For face recognition, using python library face_recognition to identify the face.

Steps -

1. Identify where face is in the image.
2. Once identifying the face in a image, if the face is of the valid student present in a dataset.
3. If yes, enter the attendance of the student with log time. If not, show unknown student.

Drowsiness Detection

For drowsiness detection, we will use transfer learning for building a model.

Steps -

1. Preprocess the data.
2. Split the data into train and validation set.
3. Train the pre-trained model available. (We will use mobilenet).
4. Check the accuracy.
5. Test the model on real world data.

After developing both the models, we will integrate it and deploy it using streamlit.

Data Overview & Preprocessing

For **face recognition** we will use one image of a person to get the encoding of a face of that person and will store it. For testing purpose we will use the image of same person and one image of other person and will get the encodings. Then we will compare(difference) between the encoded matrix and the images which have less difference, the name of that person will show up.

For **drowsiness detection** we have 'MRL Eye Dataset'. We will train the model on approximately 1500 images of closed and open eyes.

Data Preprocessing -

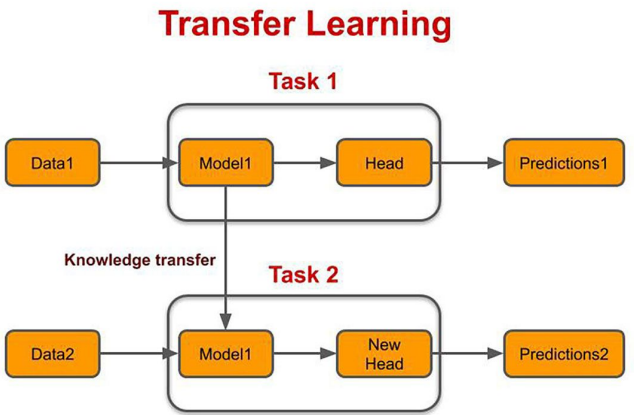
As we are using transfer learning and mobinet is the pre-trained network we are using, we will have to keep our input data same as the data on which mobilnet is trained.

1. Convert the dataset into 4 dimension, as mobinet expects image must be in 4 dimensions.
2. Then we will normalize the data.

```
Model: "mobilenet_1.00_224"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 input_1 (InputLayer)         [(None, 224, 224, 3)]     0
```

Transfer Learning

Transfer learning is the reuse of pre-trained model on a new model. It can train deep neural network with very little data by tackling the problem of overfitting. In transfer learning, a machine exploits the knowledge gained from a previous task to improve generalization about other. We transfer the weights that a network has learned at “task A” to a new “task B”.



Mobilenet Architecture

Mobilenet is the lightweight pre-trained architecture for training new deep learning models.

First Few Layers

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
conv1 (Conv2D)	(None, 112, 112, 32)	864
conv1_bn (BatchNormalization)	(None, 112, 112, 32)	128
conv1_relu (ReLU)	(None, 112, 112, 32)	0
conv_dw_1 (DepthwiseConv2D)	(None, 112, 112, 32)	288
conv_dw_1_bn (BatchNormalization)	(None, 112, 112, 32)	128
conv_dw_1_relu (ReLU)	(None, 112, 112, 32)	0
conv_pw_1 (Conv2D)	(None, 112, 112, 64)	2048
conv_pw_1_bn (BatchNormalization)	(None, 112, 112, 64)	256
conv_pw_1_relu (ReLU)	(None, 112, 112, 64)	0
conv_pad_2 (ZeroPadding2D)	(None, 113, 113, 64)	0
conv_dw_2 (DepthwiseConv2D)	(None, 56, 56, 64)	576

Last Few Layers

dropout (Dropout)	(None, 1, 1, 1024)	0
conv_preds (Conv2D)	(None, 1, 1, 1000)	1025000
reshape_2 (Reshape)	(None, 1000)	0
predictions (Activation)	(None, 1000)	0

Total params: 4,253,864
Trainable params: 4,231,976
Non-trainable params: 21,888

dropout (Dropout)	(None, 1, 1, 1024)	0
flatten (Flatten)	(None, 1024)	0
dense (Dense)	(None, 1)	1025
activation (Activation)	(None, 1)	0

Total params: 3,229,889
Trainable params: 3,208,001
Non-trainable params: 21,888

Conv2D - Creates a convolutional kernel that is wind with input layers to produce the output.

0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

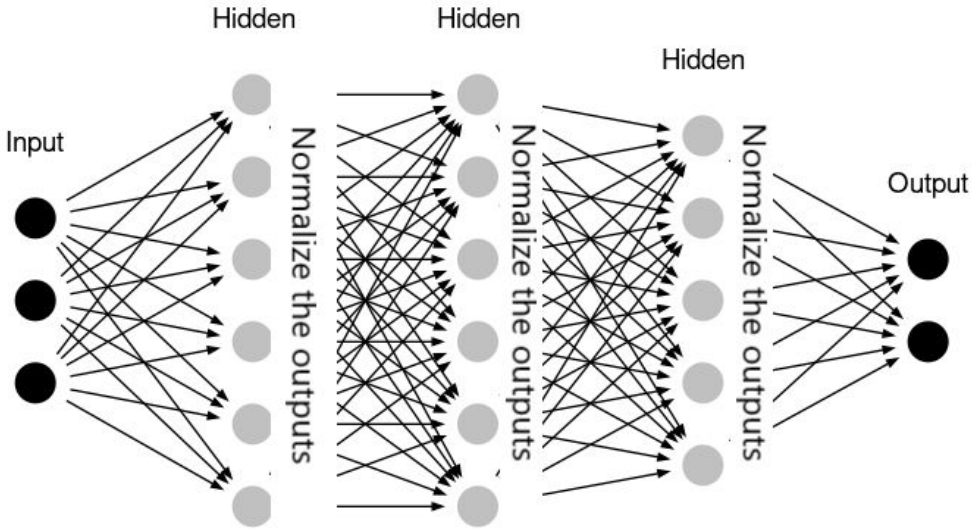
Kernel

0	-1	0
-1	5	-1
0	-1	0

114				

Normalization is a data pre-processing tool used to bring the numerical data to a common scale without distorting its shape.

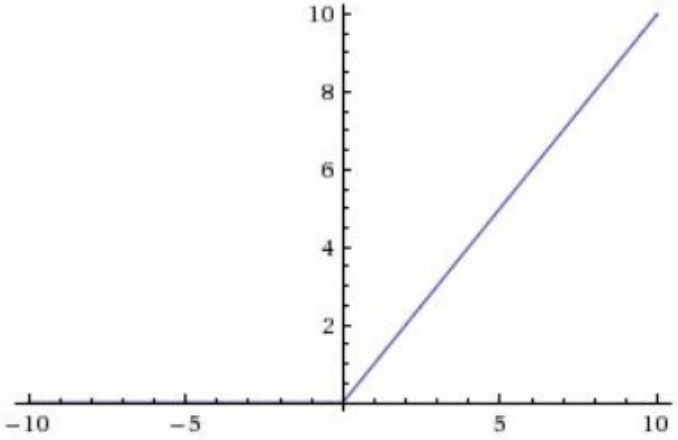
Batch normalization is a technique for training very deep neural networks that standardizes the inputs to a layer for each mini-batch. This has the effect of stabilizing the learning process and dramatically reducing the number of training epochs required to train deep networks.



ReLU - Rectified Linear Unit

The ReLU activation functions can be used for hidden layers in neural networks. Returns 0 if receives negative value and for any positive value x, return that actual value x.

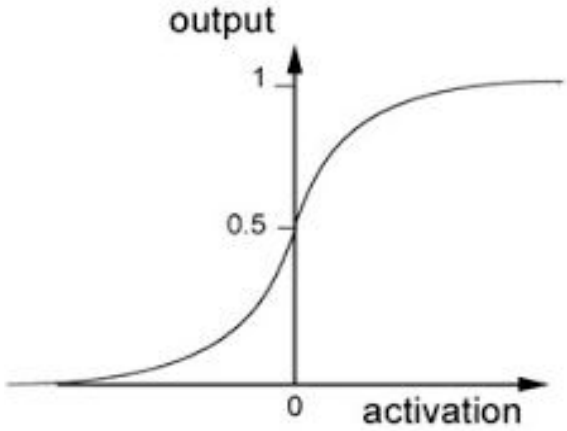
$f(x) = \max(0, x)$



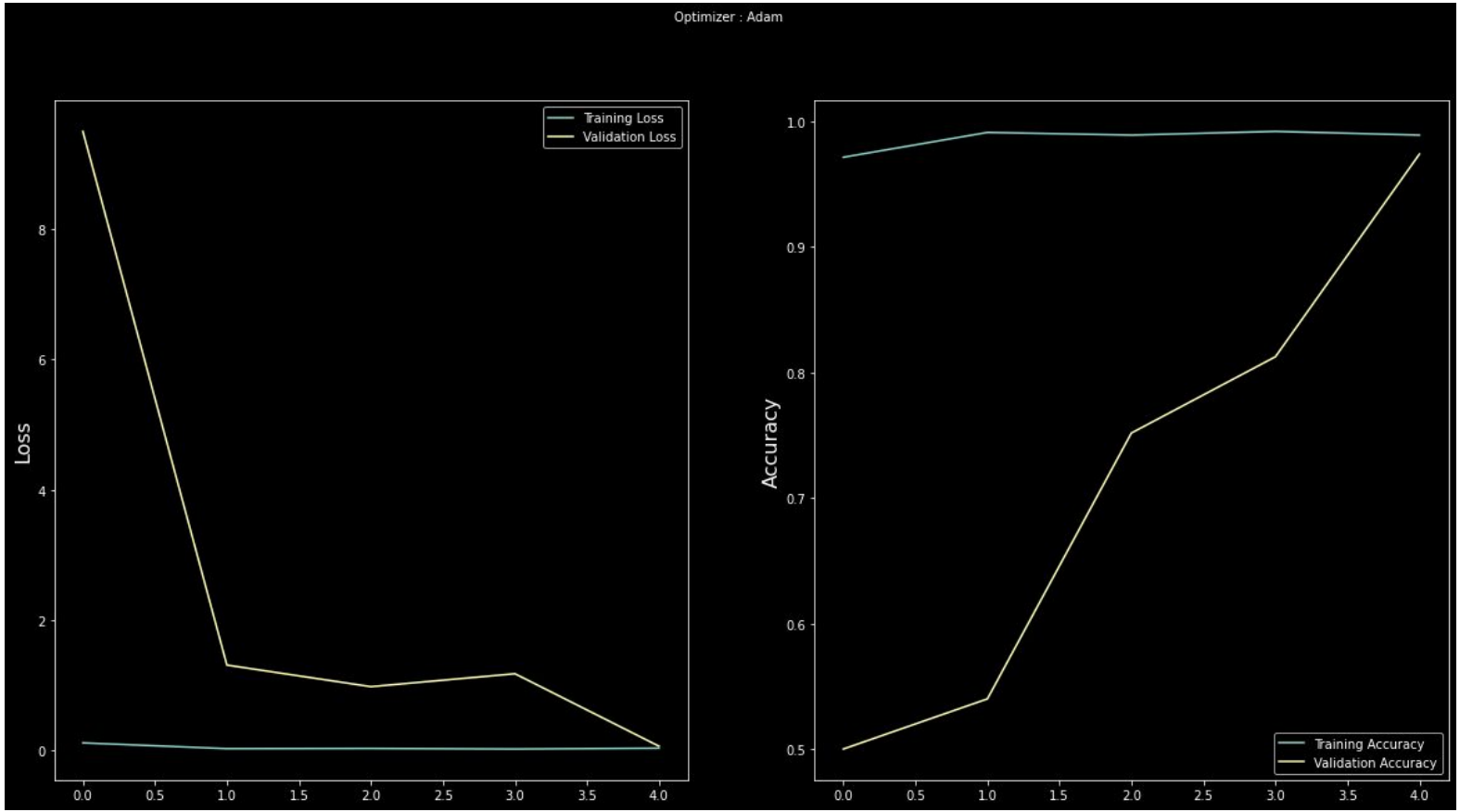
Sigmoid Activation Function

Mostly used for output layer.
Squash/Returns the value between 0 to 1 by taking any input value. Used for binary classification problems.

$f(x) = 1 / 1 + e^{-x}$



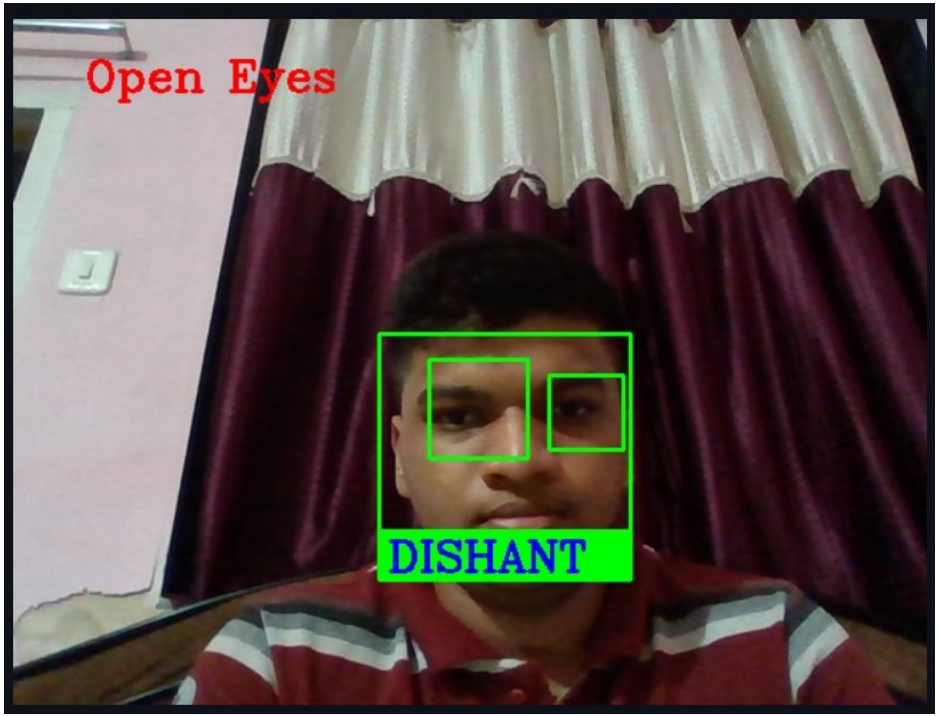
Model Testing & Accuracy

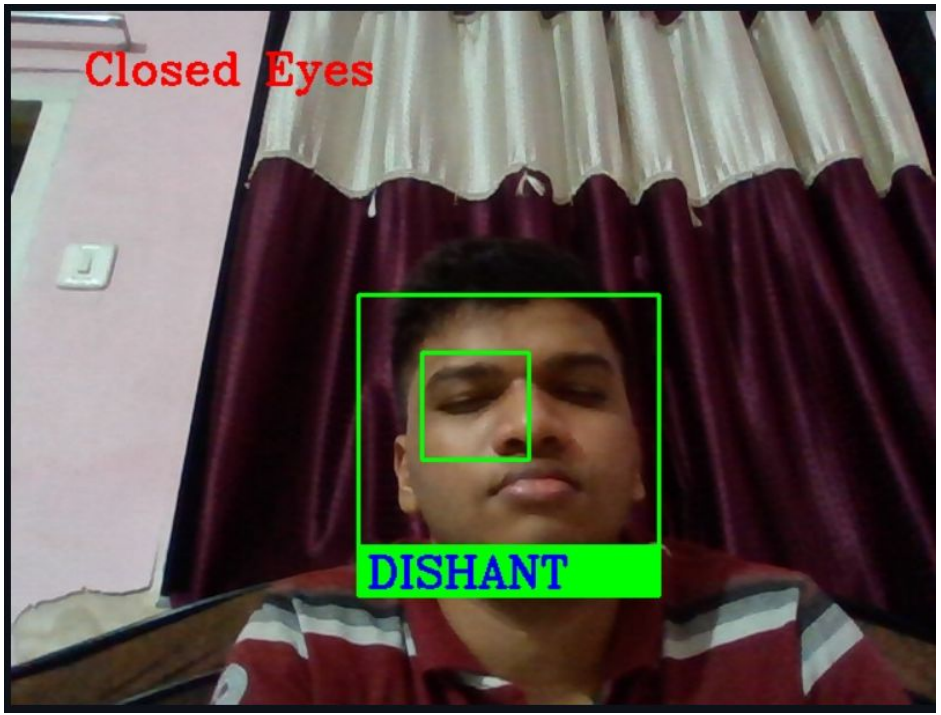


Got training accuracy of 98.91% & validation accuracy of 97.40%.

Integrating Face Recognition & Drowsiness Detection Model + Streamlit Deployment

For the purpose of problem statement, we have integrated face recognition with drowsiness detection model. This integrated model will recognize the face of a person and will output his/her name & also will detect the eyes landmarks of a person and will output whether person is sleepy or not. That is, whether eyes of a person is open or closed.





Future Scope & Application

1. Can improve the accuracy of the model by training more and getting more data. We can try to achieve bayes optimal error.
2. Can make more interactive website, where we can accept the image and name of newly enrolled student and add it to the dataset.
3. This model can also be used for driver drowsiness detection to alert the driver if he/she sleeps while driving.
4. Can research more about the model we used and also try different pre-trained model that can be used for transfer learning.
5. Can integrate with IoT devices for more efficiency.

Conclusion

1. Developed a model to detect whether student is drowsing in the class or not and also integrated it with face recognition based attendance system.
2. Used face-recognition library for task of recognizing face.
3. Used transfer learning for training drowsiness detection model.
4. Mobilenet architecture is used for transfer learning.
5. Got training accuracy of 98.91% & validation accuracy of 97.40%.
6. Integrated face-recognition with drowsiness detection and deployed the app using streamlit.
7. Suggested what can be the future scope and more real world application of the model.

THANK YOU