**Abstract**

Emotion has been an integral part of human being which gives us the insight of various kinds of perceptions of human cognitive state. It helps us recognize how people feel about a particular situation, thing or object and this data is valuable. People all around the world make their decisions based on their emotions and logic and to know how these things drive them towards a particular goal helps us make better decisions. The following paper is being made for scholastic purposes, where we are trying to identify the relation between the emotions between and academics along with various other factors.

Various techniques are used by in order to know the emotion of a person, they can be done with the help of image recognition, speech recognition and even using the brain waves patterns to check upon the emotions, while all these techniques produce good results especially the brain waves, we have chosen image processing for our research for the reason that it is economical and availability of established datasets for the project would allow better authentication of projects results. The technique used for the image recognition is deep convolutional neural network (CNN), CNN is a sub field of deep learning which is mostly used for analysis of visual imagery. A CNN is a convolution tool that parts the different highlights of the picture for analysis and prediction.

There are various kinds of architectures used for the CNN models like the LeNet, LeNet is the first CNN architecture. The model has five convolution layers followed by two fully connected layers, LeNet could not train well due to the vanishing gradients problem. AlexNet is the deep learning architecture that popularized CNN. AlexNet network had a very similar architecture to LeNet, but was deeper, bigger, and featured Convolutional Layers stacked on top of each other. ZFnet is the CNN architecture that uses a combination of fully-connected layers and CNNs. the network has relatively fewer parameters than AlexNet, but still outperforms it on ILSVRC 2012 classification task by achieving top accuracy with only 1000 images per class. It was an improvement on AlexNet by tweaking the architecture hyperparameters, in particular by expanding the size of the middle convolutional layers and making the stride and filter size on the first layer smaller. VGG-Net is a 16-layer CNN with up to 95 million parameters and trained on over one billion images (1000 classes). It can take large input images of 224 x 224-pixel size for which it has 4096 convolutional features.  The VGG CNN model is computationally efficient and serves as a strong baseline for many applications in computer vision due to its applicability for numerous tasks including object detection. Its deep feature representations are used across multiple neural network architectures like YOLO, SSD, etc. Taking all of these architecture into account we were able to produce best results with VGG-16 architecture which provided with the highest accuracy for the datasets used.

These emotion recognition models are used along with other models like the head pose detection model or the eye tracking system in order to know the attentivity of the student. Like mentioned earlier that emotions helps us in understanding the cognitive state of mind , using this knowledge along with the other attentivity monitoring technique we can analyse the attentivity of a student in a class , these techniques helps us in knowing the attentivity of each individual student which helps in getting better insights for what measures to be taken to ensure more attentivity in a class so that there is better focus on the education of students rather than just plain teaching where there is no knowledge of the understanding of a topic which student maybe receiving. Inversely these techniques can be used in understanding the attentivity of a lecturer which also will aid us in providing various interesting insights to improve the teaching methods or other facilities in a class room.

Though we are extensively focused on the scholastic purposes of the model, this particular model can also be extensively be used in the fields of advertisements, screening tests , safety and protection etc. these models can be integrated with various kinds of IOT devices which can be remotely attached to any kind of a banner or place at an angle they can identify a person’s face clearly and also the kind the emotion they are expressing , which can be better titled as a feedback provider , which helps us understand how people feel about a thing .For example an advertisement of a chocolate bar at railway station can help us understand by customer response whether they feel crave towards the product by the emotion they must be expressing.

**TABLE OF CONTENTS**

List of Tables iv

List of Figures v

1. Introduction 1

1.1 Motivation 1

1.2 Problem Statement 2

1.3 Challenging Aspects 3

1.4 Summary of the key Contributions of the thesis

1.5 Outline of the thesis 5

2. Emotion Detection

2.1 Facial Expression Analysis

2.2 Image Pre-processing

2.3 Feature Extraction

2.4 Optical Flow Method

2.5 Feature Point Tracking

2.6 Darwin Facial Muscles Description

2.7 Facial Expression Code Description

2.8 Facial Landmark Detection

2.8.1 TensorFlow Attribute Importance

2.8.2 Estimating Face Direction from Landmarks

2.8.3 Understanding Facial Landmark Detector

2.9 CNN

2.9.1 VGG-16 Architecture

2.10 Proposed Emotion Recognition model

2.10.1 Dataset

2.10.2 Working model

3. Attentivity Model

3.1 Head pose Estimation

3.1.1 Roll, Pitch and Yaw for Head Pose

3.1.2 Pose Estimation Requirements

3.1.3 Working Algorithm of Head Pose

3.1.4 Estimated Face Pose Calculations

3.1.5 Expression Intensity Ranking

3.1.5.1 Expression intensity ranking problem

3.1.5.2 VFOA Recognition

3.2 Eye tracking system

3.2.1 Face Landmark detector

3.2.1.1 Estimating Face Direction from Landmarks

3.2.2 Working Model

3.3 Comparing both the models

4.Integrated Model

4.1 Headpose

4.1.1 Head Pose Estimation

4.1.2 Expression Intensity Ranking

4.2 Eye Tracking

4.2.1 Attention detection model

5. Innovative Teaching Strategies.

5.1:Role play

5.2 Predict Observe Explain(POE)

6. Results and Testing

6.1 Emotion Recognition

6.2 Head pose

6.2.1 Pitch, Roll, Yaw Output

6.3 Eye Tracking

6.4 Attention Estimation System

6.5 Testing

6.5.1 Unit Testing

6.5.2 Integration Testing

6.5.3 Comparison testing

6.5.4 Importance of Testing

7. Conclusion and future scope

**List of Tables:**

Table 2.6.1 Descriptions of facial muscles involved in the emotions

Table 4.1: Samples 1 and 2 test Watson and Tellegen's (1985) model; Sample 3 tests Russell's (1980) model. ' The polar angle deviates slightly from the hypothesized model

Table 6.2.1 Accuracies of different methods for head pose estimation and landmark locations in AFLW datasets

**List of Figures:**

Fig 2.1.1 Facial Expression Analysis

Fig 2.5.1 Feature Points Displacement

Fig. 2.6.2 Different expressions detected as per description

Fig 2.8.3.1 Relation between points on image

Fig 2.9.1: CNN Architecture

Fig 2.9.2: LeNet-5 architecture.

Fig 2.9.3:AlexNet Architecture.

Fig 2.9.1.1: VGG-16 Architecture for the ImageNet dataset

Fig 3.1.1.1 Euclidean loss Formula

Fig 3.1.1.2 Orientation of the head in terms of pitch, roll, and yaw movements describing the three degrees of freedom of a human head

Fig 3.1.2.1 To calculate the 3D pose of an object in an image you need the following information

Fig 3.1.2.2 Roll, Pitch, Yaw according to head pose

Fig 3.1.3.1 Transforming camera coordinates to world coordinates

Fig 3.1.3.2 Equation for Transforming coordinates

Fig 3.1.3.3 Expanded form of equation

Fig 3.1.4.2 Typical camera pinhole equation

Fig 3.3.4.1 Visualizing the 68 facial landmark coordinates

Fig 3.3.4.2 Head Pose estimation Output

Fig 3.1.5.1.1 RED-SVM Algorithm

Fig 3.2.1.1 : face\_landmark\_detector for multiples faces

Fig 3.2.2.1: Distance measurement using points from face landmark detector

Fig 4.2: Affective circumplex (Model adapted from Feldman Barrett and Russell [ 1998], published by the American Psychological Association)

Fig 4.3: Affective circumplex model used for our emotion recognition model.

Fig 4.1.1 Geometric Relationship between the attention target, position and head pose of learner

Fig 4.1.2 VFOA Target Formula

Fig 4.1.2.1 Description of deriving output

Fig 4.2.1: Face landmarks marked with number for detection of particular face feature

Fig 4.2.2: Integration of Emotion Detection and eye tracking system to make attentivity model

Fig 5.2.1: Students performance analysis by comparing against the role play technique of study.

Fig 5.2.2: various cognitive levels in the stages of POE.

Fig 5.2.3: Percentage of positive emotion change in various teaching strategies for students.

Fig 5.2.4: Percentage of positive emotion change in various teaching strategies for lecturer.

Fig 6.1.1: Accuracy for the 1st emotion recognition model.

Fig 6.1.2: Accuracy for the transfer learning model.

Fig 6.1.3: Result from the emotion recognition model where multiple faces are being detected and emotion being predicted.

Fig 6.1.4: Result from code run on Deep Learning Server

Fig 6.1.1.1: Expression detection of SENnet Teacher on FER dataset.

Fig 6.1.1.2: Expression detection of RAN on FER dataset

Fig 6.1.1.3: Expression detection of PSR on FER dataset

Fig 6.1.1.4: Expression detection of Modified VGG-16 on FER dataset

Fig 6.2.1 Head Pose Estimation Output

Fig 6.2.1.1: movement of face about the three orthogonal axes

Fig 6.3.1: Result from the eye tracking model looking at centre

Fig 6.3.2: Result from the eye tracking model looking at right

Fig 6.4.1: Result from mainrunner function integrating both the emotion recognition model and eye tracking model.

Fig 6.4.2: Data obtained and stored in the files.

Fig 6.4.3: results from the eye tracking and the emotion recognition model being shown for attention and engagement.

Fig 6.5.1: Unit testing on the emotion recognition model

Fig 6.5.2.1: Integration testing on the emotion recognition model and eye tracking model.

Fig 6.5.3.1: Comparing models and their accuracy

**1. INTRODUCTION**

**1.1 Motivation**

A critical factor that determines learning effectiveness in class is learning cognitive state, and the analysis of learners’ cognitive state has become a research hotspot and challenge in education [1]. With the boom of big data in education, it has become possible for educators to identify learners’ cognitive state in class in a quantitative manner from a large amount of time series data for calculating attention.

In addition to attention, emotion plays an important role in human learning. Emotion influences the ability to process information and interpret events. Previous research shows that positive emotions during the learning process such as interest and happiness can promote learners’ cognitive activities, whereas negative emotions such as depression and boredom can disrupt learners’ cognitive activities [2]. Most existing cognitive state analysis methods however focus on attention, while emotion is largely ignored. Hence, it is necessary to introduce emotion estimation for learning cognitive state analysis.

Number of methods for analysing emotion have been developed and they can be divided into two categories:

1) biological signal-based methods

2) facial expression-based methods.

However, it is difficult to apply these methods in the real-world classroom because they require highly sophisticated wearable devices to measure these physiological signals. Alternatively, facial expression is one of the most powerful social signals for human beings to convey emotion and intention, it can be used for emotion analysis. A study by a scientist named Mehrabian shows that 55% of emotional expression comes from facial expressions, 38% comes from voice, and 7% comes from words [3]. The study proves that this procedure is workable and produces valid results.

As it is difficult to apply the mentioned methods because of the requirement of highly sophisticated devices in real time environments, automatic facial expression analysis is a feasible solution for understanding emotion. The automatic facial expression analysis system consists of two main steps: Pre-processing and facial expression analysis (FEA). The objective of data pre-processing is to align and normalize the face with a series of operations including face detection, landmark location, and face alignment.

Recently, several multitask deep learning-based methods have been proposed for joint face detection and alignment, such as multi-task cascaded convolutional networks (MTCNN) [4] and hyperface [5]. By jointly learning multiple related tasks, multi-task learning can effectively address the overfitting problem caused by insufficient labelled samples. FEA consist of facial expression classification and expression intensity estimation.

Facial expression classification identifies the six basic facial expressions of anger, disgust, fear, happiness, sadness, and surprise. In addition to these we can also include neutral as an expression defining human emotions sometimes is neither positive nor negative. The number of images in the training set for the neutral expression is very less in most datasets. Recently, there is an increasing interest in a more fine-grained analysis, namely the facial expression intensity estimation, which is presented to rank facial expressions with different intensity levels.

However, it is highly time-consuming and expensive to label the expression intensity for the analysis of this data. Numerous works consider the expression intensity estimation as an ordinal regression problem that learns from a specific sequence beneath an unsupervised framework about the temporal order of pair-wise data.

To address this problem, our project focuses on the development of an emotion-sensitive learning cognitive state analysis framework in the intelligent classroom. To understand learners’ cognitive state in a non-invasive way, we attempt to obtain the learners’ attention and emotion to analyse the data for the attentivity of an individual.

**1.2 Problem Statement**

The main agenda of the attention estimation system is to take the features of both the emotion recognition system and the attentivity models to identify the attention along with the engagement of a person. Now knowing this in present thesis we try to discover the best of the emotion recognition models used along with the different kind of attention model used and the integration of both of the features to provide an estimation of the attentivity of an individual.

As mentioned above we divide this problem into 3 parts mainly the emotion recognition model, the attentivity model and at last the integration of both the models, while the emotion recognition discusses more about the type of the classification techniques used along with the use of a particular architecture along with the ways to enhance its accuracy, the attentivity model ponders around the discussion of which method focuses on better extraction of attentivity features which can be better explained in the later parts of thesis.

In order to increase the efficiency of the models being used, the model is also trained under the transfer learning techniques in order to enhance the accuracy of the models and lastly also to analyse how these data are being used to understand the attentivity of each individual and also how these features actually aid in the field.

**1.3 Challenging Aspects**

For the better understanding of the emotions, it is necessary to investigate the facial expression classification and expression intensity estimation simultaneously. Therefore, this needs urges to propose an approach to solve or overcome the pending problems:

**1. Integration of emotion estimation and attention into a unified framework for**

**the learning cognitive state analysis in the intelligent classroom environments.**

Though the models sure would work perfectly when working separately but the parallel execution of the processes may pose a problem, the problem can occur in ways such as slowing down the execution of a process considerably which would hinder the decision making process, the following would also be dependent on the specifications of a system and accordingly we would need to set a time between each snap of a image to process so that it works in a particular system, which also further raises a dilemma of reducing the amount of images which are processed which could mean a lesser accuracy of the output.

The other thing which can cause a problem is use of this model in an actual classroom which would mean identification of each face along with their emotion in the real time. This would require huge processing power which would also depend on the type of IOT device used for capturing this information and the processing being done on it.

**2. Estimation of the facial expression intensity in an unsupervised setting to address the problem of insufficient labelled samples with intensity levels for big data in education.**

The problem occurs in the dataset most of the times , that the type of data may be insufficient , usually in the image processing , we try to take a photo in the black and white format , for the very reason to decrease the amount of processing to be done along with increasing the accuracy of the model to recognize the pattern in the image more than the colour of the image itself , but the problem with the contrast and the lighting conditions which exists inn the databases may differ with the real life , this could lead to a decrease in the accuracy of the model, but even though this problem remains an effort using the transfer learning technique is carried out , but there is another problem which comes along with it is the use of wearables in the class which can be identified like pair of glasses which would probably reflect light making the model less accurate.

The similar type of problems may also occur in a situation of online lectures where the position of the camera to the face may hinder the working of the model, such as face being too close to camera, hence not allowing the model to recognize the face itself, as the model recognizes the emotion only when the face is detected.

3. **Use of systems in face lock systems is a threat to user’s privacy**.

The inclusion of security features in the system is the 1st requirement for this to work since the user’s valuable information can be easily be revealed to some hackers. There are ways to work around this, like use of the model only at the places which don’t posses much threat, but then again it would mean it is prone to vulnerabilities, then it would be meant the model can’t be used for most of the private purposes.

This can be checked by building a strong security feature, but still the model can store some sensitive information and existence of such information would make it worthwhile for a hacker even if strong security features exist, to try to extract the information.

**4. Increase in the better visual of the faces with the use of image processing can help us to better understand the emotion being expressed through a model**

The CNN model uses the patterns in the images to understand the emotion being expressed in the image, but for this to work the model need to clearly understand the pattern, now due to various conditions which could be varying across like the shade , lighting conditions etc. can be blamed , but we can also make it work it through with the help of image processing , by using the models which would help us extract the features in an image , to be given in the form of example can be said like a simple would make pattern like lips in ‘u-shape’ and the stretching of the eyes.

Now the image processing which could be used for the process can be something like the image segmentation or the use of highlighting the border etc., but use of these techniques can help the model to understand the emotion better.

**5. Using the affective circumplex model to integrate it for using it for engagement as the model is also prone to some errors thus using model which can more nearly determine the engagement.**

The model after using the emotion, uses the affective circumplex model in which one particular dimensional approach, termed the circumplex model of affect, proposes that all affective states arise from two fundamental neurophysiological systems, one related to valence (a pleasure–displeasure continuum) and the other to arousal, or alertness. The existence of this valence helps us determine the attentivity or alertness of an individual.

Though this model works well, it also has its various versions from which we follow Model adapted from Feldman Barrett and Russell [ 1998], published by the American Psychological Association, as it is regarded as the base model on which the others are judged on. The circumplex model has been developed by the department of neuro science where they have taken various experiments into account, but similar to a dataset this may defer in a different location but still it has been regarded as one of the best models. Also the existence of asymptomatic cases must also be put in place .

**1.4 Summary of the key Contributions of the thesis**

**1.5 Outline of the thesis**

The following section describes about the sequence of contents in the following thesis:

The second chapter of the thesis describes about the various types of image processing techniques along with the various types of classification techniques and the architecture of the emotion recognition model, this section of the thesis also focuses on the data points used in the identification of the image and also the numbered datapoints used for this information.

The later chapter describes about the use of the attentivity model which has been used in the form of head pose estimation and eye tracking system. Taking these two we try to draw parallels on the type of model which can better allow us to identify the attentivity of a person. This section also sheds light upon the mathematical calculation and the technology used behind the working of the model.

Upon explanation of both of the models, then we arrive at the integration of both models in the fourth chapter, where we discuss about the mathematical calculations used to determine the attentivity of a person in a class along with the engagement of the person. Since attention describes the external features while engagement refers to the insight to the cognitive state of mind.

Further into the thesis we arrive at fifth chapter where we discuss about experiment done on an actual classroom to identify the best teaching method, which not only focuses on the student but also on the emotions and attention expressed by the teacher. This chapter is used to draw insights with the help of an experiment where we completely understand the better technique which produces best of results.

The later parts of the thesis we arrive at the sixth chapter where we discuss about the results and the experiments done on various IOT devices along with the comparative study with the other methods used.

The final part, the seventh chapter of the model focuses on the future scope of the model and the various fields where it can be used other than the field of education and the ways in which the model can further be improved.

**2 Emotion Detection**

Facial emotion recognition is the process of detecting human emotions from facial expressions. The human brain recognizes emotions automatically, and software has now been developed that can recognize emotions as well. This technology is becoming more accurate all the time, and will eventually be able to read emotions as well as our brains do.  With deep learning we can detect emotions by learning what each facial expression means and applying that knowledge to the new information presented to it.

Emotional deep learning, or emotion AI, is a technology that is capable of reading, imitating, interpreting, and responding to human facial expressions and emotions. Recently, interest in facial expression intensity estimation has increased. There is a vast scope for researchers to do researches in the field related to facial expression analysis. However, the existing expression datasets with the AUs’ intensity labelling are limited and labelling these five levels is a specialized job.

To address the lack of standard expression intensity datasets, another effective method for expression intensity estimation is to construct a ranking model using the ordinal information of the expression sequence. Facial expression classification identifies the six basic facial expressions of anger, disgust, fear, happiness, sadness, and surprise. In addition to these there is another expression of interest which is neutral, it is neither a positive emotion nor a negative emotion. To explain the process a supervised learning process is formulated in which the emotional states involved in a speech signal is recognized based on its acoustic features by using a classification model.

A feature set is extracted from the training data to capture the important and discriminative attributes of speech signals. A training model is then constructed by feeding pairs of feature sets and the target values of emotion categories into the learning algorithm of support vector machines (SVMs). In the prediction process, the same features are extracted from unseen speech data, which are fed into the obtained training model to yield predicted labels for the target emotions.

**2.1 Facial Expression Analysis**

We need to pre-process the image and select the appropriate feature extraction and classification method for the target dataset. The conventional FER procedure [10] can be divided into three major steps: image pre-processing, feature extraction, and expression classification.

Symmetry 11 01189 g004 550

Fig 2.1.1 Facial Expression Analysis

#### 2.2 Image Pre-processing

This step is to eliminate irrelevant information of input images and enhance the detection ability of relevant information. Image pre-processing can directly affect the extraction of features and the performance of expression classification. For various reasons, pictures are often contaminated by some other signals. Some pictures may still have complex backgrounds, e.g., light intensity, occlusion, and other interference factors, even if they are basically free of noise. Moreover, many datasets are different in size, and some are composed of colour images, while some are composed of grayscale images. In addition, various shooting equipment can cause data diversity. These objective interference factors need to be pre-processed before recognition.

The process of image pre-processing is introduced as follows.

* **Face detection** has developed into an independent field. It is an essential pre-step in FER systems, with the purpose of localising and extracting the face region.
* **Normalisation** of the scale and grayscale is to normalise size and colour of input images, the purpose of which is to reduce calculation complexity under the premise of ensuring the key features of the face.

#### 2.3 Feature Extraction

Feature extraction is a process to extract useful data or information from the image, e.g., values, vectors, and symbols. These extracted “non-image” representations or descriptions are features of the image. Feature extraction may directly influence the performance of the algorithms, which is usually the bottleneck of the FER system. It is essential to take both applicability and feasibility into consideration when manually choosing an appropriate feature extraction method in conventional FER approaches.

#### 2.4 Optical Flow Method

Optical flow is the pattern of apparent motion caused by the relative motion. The features of the continuous moving face image sequence are extracted by using Horn–Schunck (HS) optical flow to combine the two-dimensional velocity field and the grayscale. The algorithm computes optical flow caused by facial expressions to identify the direction of motions.

#### 2.5 Feature Point Tracking

The main purpose of feature point tracking method is to synthesise the input emotional expressions according to the displacement of the feature points, as presented in Figure 2.5.1



Fig 2.5.1 Feature Points Displacement

It extracts over 20 points from the video stream as the feature points of the face model, then a variable 3D expression recognition model is constructed by tracking these feature points with particle filters. The algorithm improves the Scale Invariant Feature Transform (SIFT), making the feature points evenly distributed without aggregation.

**2.6 Darwin Facial Muscles Description**

The emotion is detected from the facial expression and the facial description is shown in the below. The table 2.6.1 shown below gives the facial description for each emotion and also description of the facial muscles involved in emotions as considered by Darwin universal.

**Emotion**  **Facial Description**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Eyes open

Fear Mouth open

Lips retracted

Eyebrows raised

\_\_\_\_\_\_

Eyes wide open

Anger Mouth compressed

Nostrils raised

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Mouth open

Disgust Lower lip down

Upper lip raised

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Eyes sparkle

Happiness Mouth drawn back at corners

Skin under eyes wrinkled

Upper lip raised

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Eyes open

Mouth open

Surprise Eyebrows raised

Lips protrude

Corner of mouth depressed

Sad Inner corner of eyebrows raised

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Slight Confusion

Neutral Unimpressed

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Table 2.6.1 Descriptions of facial muscles involved in the emotions

The table shown above describes the facial descriptions of different expressions fear, anger, disgust, happy, surprise, sad and neutral which are the basic expressions used in all the datasets for facial expression analysis or emotion detection projects. If the face is among any of the description given then the expression is detected as per given in the table. The expressions detected as per the description is as shown below in the image.

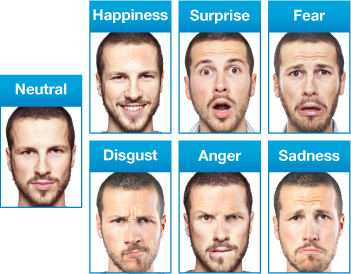
****

Fig. 2.6.2 Different expressions detected as per description

**2.7 Facial Expression Code Description**

The module first needs Keras to be imported. Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code. Image generator is imported from the pre-processing module of Keras and other packages such as layout, conv2D and maxpool2D are also imported from the layer’s module of Keras.

Other important modules such as NumPy, pandas and matplotlib are imported for plotting graphs. Then the dataset with 29,000 images is being read from the CSV file. The seven different expressions which we consider are given numbers to identify them. The images in the form of pixels are stored as arrays. we want to create an image pyramid, in order to detect faces of all different sizes. In other words, we want to create different copies of the same image in different sizes to search for different sized faces within the image.

The image is represented as the 24 x 24 kernel, resized back to the original image. We can calculate the width and height of the kernel. We are using the coordinates of the kernel in the original image to calculate the height and width. The width and height we get here are the width and height of the kernel when scaled back to its original size. Afterwards, we multiply the bounding box. Since the bounding boxes may not be square, we then reshape the bounding boxes to a square by elongating the shorter sides

Now, fitting the model by running it with number of epochs. Epoch refers to one cycle through the full training dataset. Standard number of epochs is thirty. The time taken by each epoch depends upon the CPU/GPU. On a normal basis when we run it on a CPU it takes 10-15 mins for each epoch. It runs faster on a GPU. For each epoch the accuracy of the model will be increasing from 35%-94%. The loss is decreased for each epoch running from around 2%- 0.5%.

Sometimes, an image may contain only a part of a face peeking in from the side of the frame. In that case, the network may return a bounding box that is partly out of the frame. For every bounding box, we create an array of the same size, and copy the pixel values (the image in the bounding box) to the new array. If the bounding box is out of bounds, we only copy the portion of the image in the bounding box to the new array and fill in everything else with a 0. This process of filling arrays with 0s is called padding.

After we pad the bounding box arrays, we resize them to 24 x 24 pixels, and normalize them to values between -1 and 1. Currently, the pixel values are between 0 to 255 (RGB values). A normalized confusion matrix is generated to know the performance of the model. A confusion matrix is a table that is often used to describe the performance of a classification model (or “classifier”) on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm. The weights are saved with a file name. Now that we have numerous 24 x 24 image arrays, we resize the boxes to 48 x 48 pixels and we reshape the bounding boxes to a square.

We need to load a cascade classifier which is xml file. The xml file which we used in our model is “haarcascade\_frontalface\_default.xml”. The haarcascade\_frontalface\_default.xml is a haar cascade designed by OpenCV to detect the frontal face. This haar cascade is available on GitHub. A Haar Cascade works by training the cascade on thousands of negative images with the positive image superimposed on it.

To use my webcam, I created a Video Capture object. Since I only have 1 camera, I passed in 0. cap.read() returns a Boolean (True/False) that states whether or not a frame is read in correctly. If an error occurs and a frame is not read it, it will return False and the while loop will be broken. Since sometimes a face may not be in the frame (and result will be empty), I added “if result!= []” for the program to continue running even when there are no faces in the frame. In addition, there may be more than one face in the frame. In that case, result will return back multiple sets of coordinates, one for each face. To close the window, all I have to do is press ‘q’. Once I do that, I release the video capture and close the window.

**2.8 Facial Landmark Detection**

**2.8.1 TensorFlow Attribute Importance**

The second part of the first module in our project is facial landmark detection. Facial landmark detection based on convolution neural network. The model is built with TensorFlow, the training code is provided so you can train your own model with your own datasets. The version 1.x of TensorFlow is required. Though the current version of TensorFlow is 2.x we need to install the 1.x version of TensorFlow. Because latest version of TensorFlow does not have the attributes such as Session (), ConfigProto (), loader.load(). These attributes are deprecated in the latest versions of TensorFlow. By default, if you do not mention the version while installing it on your terminal it will automatically install latest version. Sometimes it shows error while mentioning version we need to install nightly provided by TensorFlow.

So, we need older version of TensorFlow to be installed if not we need to re-install it. TensorFlow TF 1.x requires the developer write Python code that populates a “computational graph”. This graph basically exists as a sort of hidden data structure underneath the Python code - you can read the values of variables in the graph at runtime, but you need get the variable from the graph first. The latter approach isn’t particularly Python, as it forces (or at least strongly encourages) you to write code relying on a huge structure of named global variables. It can be quite nice to work with, but you need to know what you’re doing to keep things.

TF 2.0 does away with many of the problems of variable management by simply doing away with named variables. While this seems harsh, it will lead to much cleaner, better organised code. In TF 2.0, you’ll need to keep track of the Python variable (handle) of your network variables if you want to use them later. Another big change, which is related to the Pythonic approach mentioned above, is that tf.layers is depreciated in favour of tf.keras.layers. This again should lead to much more manageable code, and much more uniformity in terms of how people design their neural network codes.

There are a bunch of other changes, better tf.data.Dataset API, improvements to Tensor board and eager execution by default. Only the public APIs of TensorFlow are backwards compatible across minor and patch versions.

We need these attributes because a Session object encapsulates the environment in which Operation objects are executed, and Tensor objects are evaluated. A session may own resources, such as tf.Variable, tf.queue.QueueBase, and tf.compat.v1.ReaderBase. It is important to release these resources when they are no longer required. To do this, either invoke the tf.Session.close method on the session, or use the session as a context manager. We may use compat.v1.Session() as a substitute for Session() but to ensure we get no errors further it is better to install the older version.

The ConfigProto() is just for a protocol message with two child classes DeviceCountEntry and Experimental. It is used to configure the session. It can also take in parameters when running tasks by setting environmental variable. The ConfigProto() protocol buffer exposes various configuration options for a session. For example, to create a session that uses soft constraints for device placement, and log the resulting placement decisions this can be used. Returns a context manager that makes this object the default session. Alternatively, you can create a session that is automatically closed on exiting the context, including when an uncaught exception is raised. As we used substitute for Session() similarly we can use compat.v1.ConfigProto() instead of it. The loader attribute is also deprecated similar to Session() and ConfigProto() and unfortunately there is no substitute for it.

While installing TensorFlow we can use “python -m pip install tensorflow-gpu” command to install TensorFlow with GPU support. A GPU (Graphical Processing Unit) is a component of most modern computers that is designed to perform computations needed for 3D graphics. Their most common use is to perform these actions for video games, computing where polygons go to show the game to the user. With a lot of hand waving, a GPU is basically a large array of small processors, performing highly parallelised computation. You basically have a mini-supercomputer\* running right now! While each of the “CPUs” in a GPU is quite slow, there are a lot of them and they are specialised for numerical processing. This means a GPU can perform lots of simple numerical processing tasks at the same time. In a great stroke of luck, this is exactly what many machine learning algorithms need to do.

Most modern (last 10 years) computers have some form of GPU, even if it is built into your motherboard. Users of the systems need to consult their system’s documentation for installing it. While other graphics cards may be supportable it is recommended that you seek out a NVidia graphics card. To know the difference between these and advantage of GPU run your files with both CPU and GPU, you can do it by simply by writing CPU or GPU after the file name while executing your code. For example:

**“Python landmark\_detect.py CPU 1500”**

This will use the CPU with a matrix of size 1500 squared.

Then run the command:

**“Python landmark\_detect.py gpu 1500”**

The first thing you’ll notice when running GPU-enabled code is a large difference of time in showing a result, compared to a normal TensorFlow script.

In general, if the step of the process can be described such as “do this mathematical operation thousands of times”, then send it to the GPU. Examples include matrix multiplication and computing the inverse of a matrix. In fact, many basic matrix operations are prime candidates for GPUs. As an overly broad and simple rule, other operations should be performed on the CPU. There is also a cost to changing devices and using GPUs. GPUs don’t have direct access to the rest of your computer (except, of course for the display). Due to this, if you are running a command on a GPU, you need to copy all of the data to the GPU first, then do the operation, then copy the result back to your computer’s main memory. TensorFlow handles this under the hood, so the code is simple, but the work still needs to be performed.

Facial Landmark points has many applications like driver’s Drowsiness detection, yawn detection etc. Facial landmark points detection problem can be solved by:

Detecting the Face region and from the detected face region, find the landmark points. The detection on face region can be done by a three-stage algorithm which is MTCNN. The landmarks are calculated using Histogram of Oriented Gradients (HOG) feature combined with a linear classifier, an image pyramid and sliding window detection scheme. The MTCNN uses a 5-point facial landmark for facial landmark detection. The five points on face are two near the eyes, one on the nose tip and other two for the edges of the mouth. When talking about what distinguishes one face from another we normally talk about facial landmarks. For example, one person may have wide set eyes or a big nose. Identifying the location of these landmarks can help us compare faces.

Image transformations are pretty mind-bending. I found this resource to be incredibly helpful. To do affine transformations, you simply choose the operation you want, get the respective transformation matrix and perform a dot product on your original image. For instance, the scaling, the transformation matrix is:

cx 0 0

0 cy 0

0 0 1

With cx and cy being the percentage you want to scale the x and y parameters, respectively.

Rotation is a bit most complicated, but the same principal applied:

cosϴ sinϴ 0

-sinϴ cosϴ 0

0 0 1

A natural product of the transform is small gaps due to the transform. The easiest way to address that is to use the nearest algorithm to fill those black gaps in.

We essentially transformed someone’s face into a numerical representation of points (landmarks). The logical next step to facial recognition would be to compare these landmarks and calculate some kind of landmark distance. But these are just the landmarks that make sense to humans. They may not be that distinguishing, or there may exist some other landmarks that are more distinguishing.

So, while this technique is useful in capturing different parts of a person’s face, they may not be best suited to tell us definitively whether two photos contain the same person. Modern machine learning techniques do a similar process of converting a face into a numerical representation. Saying you have a large value in point 61 of your 128-point facial vector representation may not mean anything to us.

After we have a numerical representation of the face, we can now use Euclidian distance to measure how different the vector is from some previous vector. There is no magical distance that we could use to tell us that two people are the same. We can only measure distances and test their accuracy. We must initialize the cascade classifier from the OpenCV-provided models. The facemark detector will work around the detected faces, beginning at the bounding boxes. We use the standard trained classifier on frontal faces.

**2.8.2 Estimating Face Direction from Landmarks**

OpenCV contains one of the first robust face detectors freely available to the public. In fact, OpenCV, in its early days, was primarily known and used for its fast face detection feature, implementing the canonical Viola-Jones boosted cascade classifier algorithm, and providing a pre-trained model. We may want to tweak the two parameters that govern the face detection:

1) pyramid scale factor 2) number of neighbours.

The pyramid scale factor is used to create a pyramid of images within which the detector will try to find faces. This is how multi-scale detection is achieved since the bare detector has a fixed aperture.

In each step of the image pyramid, the image is downscaled by this factor, so a small factor (closer to 1.0) will result in many images, longer runtime, but more accurate results. We also have control of the lower threshold for a number of neighbours. This comes into play when the cascade classifier has multiple positive face classifications in close proximity.

Here, we instruct the overall classification to only return a face bound if it has at least three neighbouring positive face classifications. A lower number (an integer, close to 1) will return more detections, but will also introduce false positives.

**2.8.3 Understanding Facial Landmark Detector**

Having obtained the facial landmarks, we can attempt to find the direction of the face. The 2D face landmark points essentially conform to the shape of the head. So, given a 3D model of a generic human head, we can find approximate corresponding 3D points for a number of facial landmarks.

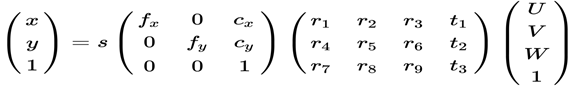


Fig 2.8.3.1 Relation between points on image

From these 2D–3D correspondences, we can calculate 3D pose (rotation and translation) of the head, with respect to the camera, by way of the Point-n-Perspective (PnP) algorithm. The camera that took the preceding picture has a rigid transformation, meaning it has moved a certain distance from the object, as well as rotated somewhat, with respect to it. If we had enough 2D and 3D corresponding points, we can write a system of linear equations, where each point can contribute two equations, to solve for all of these coefficients. We can then write the relationship between points on the image (near the camera) and the object as in the above given figure

This is an equation where U, V, W are object’s 3D position and x, y are points in the image. This equation also includes a projection, governed by the camera intrinsic parameters (focal length f and centre point c), that transforms the 3D points to 2D image points, up to scale s. Say we are given the intrinsic parameters by calibrating the camera, or we approximate them, we are left to find 12 coefficients for the rotation and translation. If we had enough 2D and 3D corresponding points, we can write a system of linear equations, where each point can contribute two equations, to solve for all of these coefficients. In fact, it was shown that we don’t need six points, since the rotation has less than nine degrees of freedom, we can make do with just four points. OpenCV provides an implementation to find the rotation and translation with its cv::solvePnP functions of the calib3d module. After obtaining the rotation and translation, we project four points from the object coordinate space to the preceding image: tip of the nose, x axis direction, y axis direction, and z axis direction, and draw the arrows in the image.

Facial landmark points capture rigid and non-rigid deformation of faces in a very compact description and are therefore valuable for many different face analysis tasks. For face recognition or different categorisation tasks such as gender, age, ethnicity or expressions, a rough pose nor-malisation is needed in order to apply other algorithms. For 3D face tracking or reconstruction facial landmarks are often used as initialisation for more sophisticated approaches. As input, landmark detectors use the image itself and a face box provided by a previous detector. In the field these consecutive detection steps are often looked at separately. Temporal stability of landmarks is an issue becoming increasingly important with more applications shifting from single images to videos. There are only a few datasets with ground truth landmarks for videos. This is with facial landmark detection.

**2.9 CNN**

CNNs are a type of deep learning algorithm that are used to process data with a grid-like topology. CNNs are a type of deep learning algorithm that is used to process data that has a spatial or temporal relationship. CNNs are similar to other neural networks, but they have an added layer of complexity due to the fact that they use a **series of convolutional layers**. Convolutional layers are an essential component of Convolutional Neural Networks (CNNs). The picture below represents a typical CNN architecture

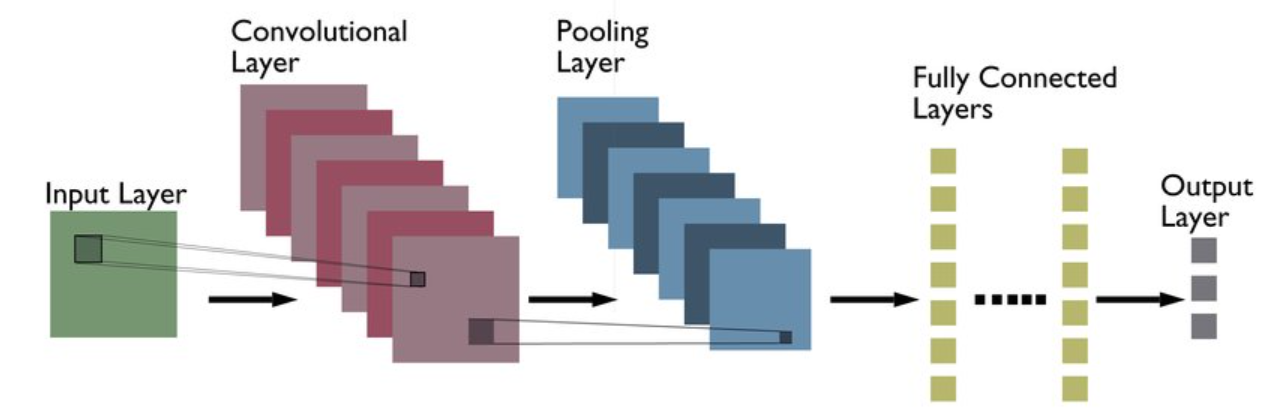


Fig 2.9.1: CNN Architecture

The following are definitions of different layers shown in the above architecture:

* **Convolutional layer**: Convolutional layers are made up of a set of filters (also called kernels) that are applied to an input image. The output of the convolutional layer is a feature map, which is a representation of the input image with the filters applied. Convolutional layers can be stacked to create more complex models, which can learn more intricate features from images.
* **Pooling layer**: Pooling layers are a type of convolutional layer used in deep learning. Pooling layers reduce the spatial size of the input, making it easier to process and requiring less memory. Pooling also helps to reduce the number of parameters and makes training faster. There are two main types of pooling: max pooling and average pooling. Max pooling takes the maximum value from each feature map, while average pooling takes the average value. Pooling layers are typically used after convolutional layers in order to reduce the size of the input before it is fed into a fully connected layer.
* **Fully connected layer**: Fully-connected layers are one of the most basic types of layers in a convolutional neural network (CNN). As the name suggests, each neuron in a fully-connected layer is Fully connected- to every other neuron in the previous layer. Fully connected layers are typically used towards the end of a CNN- when the goal is to take the features learned by the previous layers and use them to make predictions. For example, if we were using a CNN to classify images of animals, the final Fully connected layer might take the features learned by the previous layers and use them to classify an image as containing a dog, cat, bird, etc.

CNNs are often used for image recognition and classification tasks. For example, CNNs can be used to identify objects in an image or to classify an image as being a cat or a dog. CNNs can also be used for more complex tasks, such as generating descriptions of an image or 1 identifying the points of interest in an image. CNNs can also be used for time-series data, such as audio data or text data. CNNs are a powerful tool for deep learning, and they have been used to achieve state-of-the-art results in many different applications.

The following is a list of different types of CNN architectures:

**LeNet**: LeNet is the first CNN architecture. It was developed in 1998 by Yann LeCun, Corinna Cortes, and Christopher Burges for handwritten digit recognition problems. LeNet was one of the first successful CNNs and is often considered the “Hello World” of deep learning. It is one of the earliest and most widely-used CNN architectures and has been successfully applied to tasks such as handwritten digit recognition. The LeNet architecture consists of multiple convolutional and pooling layers, followed by a fully-connected layer. The model has five convolution layers followed by two fully connected layers. LeNet was the beginning of CNNs in deep learning for computer vision problems. However, LeNet could not train well due to the vanishing gradients problem. To solve this issue, a shortcut connection layer known as max-pooling is used between convolutional layers to reduce the spatial size of images which helps prevent overfitting and allows CNNs to train more effectively. The diagram below represents LeNet-5 architecture.

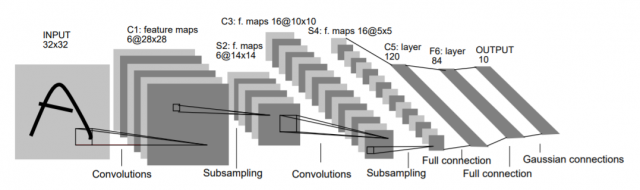
[](https://vitalflux.com/wp-content/uploads/2021/11/LeNet-Architecture.png)

Fig 2.9.2: LeNet-5 architecture.

The LeNet CNN is a simple yet powerful model that has been used for various tasks such as handwritten digit recognition, traffic sign recognition, and face detection. Although LeNet was developed more than 20 years ago, its architecture is still relevant today and continues to be used.

**AlexNet**: AlexNet is the deep learning architecture that popularized CNN. It was developed by Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. AlexNet network had a very similar architecture to LeNet, but was deeper, bigger, and featured Convolutional Layers stacked on top of each other. AlexNet was the first large-scale CNN and was used to win the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012. The AlexNet architecture was designed to be used with large-scale image datasets and it achieved state-of-the-art results at the time of its publication. AlexNet is composed of 5 convolutional layers with a combination of max-pooling layers, 3 fully connected layers, and 2 dropout layers. The activation function used in all layers is Relu. The activation function used in the output layer is SoftMax. The total number of parameters in this architecture is around 60 million.

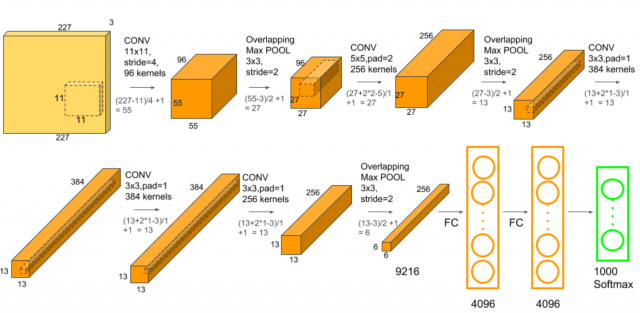
[](https://vitalflux.com/wp-content/uploads/2021/11/AlexNet-Architecture.png)

Fig 2.9.3:AlexNet Architecture.

**ZF Net**: ZFnet is the CNN architecture that uses a combination of fully-connected layers and CNNs. ZF Net was developed by Matthew Zeiler and Rob Fergus. It was the ILSVRC 2013 winner. The network has relatively fewer parameters than AlexNet, but still outperforms it on ILSVRC 2012 classification task by achieving top accuracy with only 1000 images per class. It was an improvement on AlexNet by tweaking the architecture hyperparameters, in particular by expanding the size of the middle convolutional layers and making the stride and filter size on the first layer smaller. It is based on the Zeiler and Fergus model, which was trained on the ImageNet dataset. ZF Net CNN architecture consists of a total of seven layers: Convolutional layer, max-pooling layer (downscaling), concatenation layer, convolutional layer with linear activation function, and stride one, dropout for regularization purposes applied before the fully connected output. This CNN model is computationally more efficient than AlexNet by introducing an approximate inference stage through deconvolutional layers in the middle of CNNs.

**2.9.1 VGG-16 Architecture**

VGG16 is a convolution neural net (CNN) architecture which was used to win ILSVR(ImageNet) competition in 2014. It is considered to be one of the excellent vision model architectures till date. Most unique thing about VGG16 is that instead of having a large number of hyper-parameters they focused on having convolution layers of 3x3 filter with a stride 1 and always used same padding and Maxpooling layer of 2x2 filter of stride 2. It follows this arrangement of convolution and max pool layers consistently throughout the whole architecture. In the end it has 2 FC (fully connected layers) followed by a SoftMax for output. The 16 in VGG16 refers to it has 16 layers that have weights.[7] This network is a pretty large network and it has about 138 million (approx.) parameters. The VGG-16 model can be seen in the figure 2.9.1.1 below:

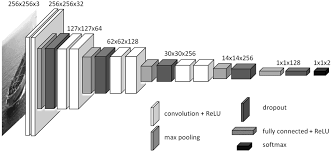


Fig 2.9.1.1: VGG-16 Architecture for the ImageNet dataset

The model shown above takes the input in the form of 256 x 256-pixel images which are in the rgb format, now these images are spaced in a sequential model order. The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. It was one of the famous models submitted to ILSVRC-2014. It makes the improvement over Alex Net by replacing large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another [8].

Though the model works great on the images but it faces its fair share of criticism on the training part where it takes huge amount of time in order to train a model.

**2.10 Proposed Emotion Recognition model:**

The first part discussing about the dataset used for building the model.

**2.10.1 Dataset**

FER is an open-source dataset which is first, created for an ongoing project by Pierre-Luc Carrier and Aaron Courville, then shared publicly for a Kaggle competition, shortly before ICML 2013[13]. This dataset consists of 35.887 grayscale, 48x48 sized face images with various emotions -7 emotions, all labelled-.

Emotion labels in the dataset:

**0:** -4593 images- Angry  
**1:** -547 images- Disgust  
**2:** -5121 images- Fear  
**3:** -8989 images- Happy  
**4:** -6077 images- Sad  
**5:** -4002 images- Surprise  
**6:** -6198 images- Neutral

During the competition, 28.709 images and 3.589 images were shared with the participants as training and public test sets respectively and the remaining 3.589 images were kept as private test set to find the winner of the competition. The dataset was set to accessible to everyone after completing the competition. The faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. The training set consists of 28,911 samples and the public test set consists of 7,066 samples. The percentage of train – test split is training sample – 80% and testing sample – 20%.

**2.10.2 Working model**

The model which we are building cannot be made like a VGG-16 model as we have a dataset of pixel size 48 x 48; we need to make a model such that it can incorporate these as input in a grayscale format. That’s why we need to mention the conv2d layer to take the input in the format 48 x 48 as a grayscale image while taking the kernel size as mentioned as 2 x 2 and stride 1 x 1. The Conv2d layer creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs. If use bias is True, a bias vector is created and added to the outputs. Finally, if activation is not None, it is applied to the outputs as well. When using this layer as the first layer in a model, provide the keyword argument input shape (tuple of integers or None, does not include the sample axis), e.g., input shape= (48, 48, 1) for 48 x 48 grayscale pictures in data format="channels last". You can use None when a dimension has variable size.

The model still takes ample amount of time to build as it takes around 1 hour to complete one epoch so in order to overcome this problem, we use batch normalization technique which help us in decreasing the training time by almost 10 minutes which greatly reduces the training time without paying much in return for the accuracy of the model.

After building the architecture the model needs to be complied after which we need to train the model for which we use optimizers where optimizers are algorithms or methods used to change the attributes of your neural network such as weights and learning rate in order to reduce the losses. Optimization algorithms or strategies are responsible for reducing the losses and to provide the most accurate results possible.

**3. ATTENTIVITY MODEL**

As seen in the last chapter, the emotion recognition model is being used to get the details of the emotion of the individual in a situation. The current chapter is going to focus on the attentivity of an individual, now the attentivity can be easily defined as the action of paying close attention to something. There exists variety of ways have been used to know the attentivity of an individual while considering a case of a normal classroom but to get that information to be processed computationally will require use of algorithms and different strategies to identify the attentiveness of an individual.

We are trying to use the attentivity model to know the attentiveness of an individual using two methods:

* Head pose estimation
* Eye Tracking system

Both of these methods are proven methods and we should be able to generate a good attentivity model with the help of both of these techniques.

**3.1 Head pose Estimation**

**3.1.1** **Roll, Pitch and Yaw for Head Pose**

Head pose estimation is the final part of the first module in our project. Although the head pose can be estimated directly by the located landmarks, the multi-task learning method reasonably leverages the relations among multiple tasks, usually leading to the improvement of individual performance.

Similar to the bounding box regression and landmark location, head pose estimation is treated as a regression problem and trained with the Euclidean loss with formula as shown in below figure.



Fig 3.1.1.1 Euclidean loss Formula

where  is the estimated value of the roll, pitch and yaw.

Pitch, yaw and roll are the three dimensions of movement when an object moves through a medium. There are in fact six degrees of freedom of a rigid body moving in three-dimensional space. As the movement along each of the three axes is independent of each other and independent of the rotation about any of these axes, the motion has six degrees of freedom. To understand the roll, pitch and yaw see the below figure. In computer vision the pose of an object refers to its relative orientation and position with respect to a camera. You can change the pose by either moving the object with respect to the camera, or the camera with respect to the object.

The pose estimation problem described in this tutorial is often referred to as Perspective-n-Point problem or PNP in computer vision jargon. As we shall see in the following sections in more detail, in this problem the goal is to find the pose of an object when we have a calibrated camera, and we know the locations of n 3D points on the object and the corresponding 2D projections in the image. To understand the roll, pitch and yaw angles in the perspective of the human face see the below figure.

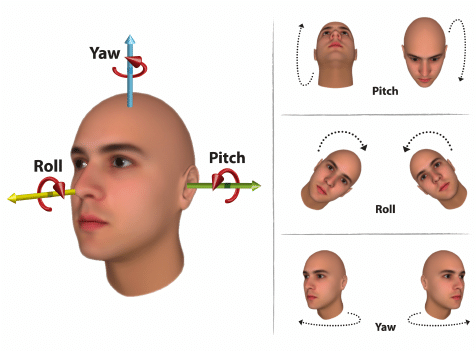


Fig 3.1.1.2 Orientation of the head in terms of pitch, roll, and yaw movements describing the three degrees of freedom of a human head

**3.1.2 Pose Estimation Requirements**

We need the 3D points for head pose estimation. To do that first we need the 2D (x, y) locations of a few points in the image. In the case of a face, we choose the corners of the eyes, the tip of the nose, corners of the mouth etc. DLIB’s facial landmark detector provides us with many points to choose from. In our project, we will use the tip of the nose, the left corner of the left eye, the right corner of the right eye, the left corner of the mouth, and the right corner of the mouth. You also need the 3D location of the 2D feature points. You might be thinking that you need a 3D model of the person in the photo to get the 3D locations. Ideally yes, but in practice, you don’t. A generic 3D model will suffice. Well, you really don’t need a full 3D model. You just need the 3D locations of a few points in some arbitrary reference frame. For example, consider the below image to see the points in some arbitrary frame.

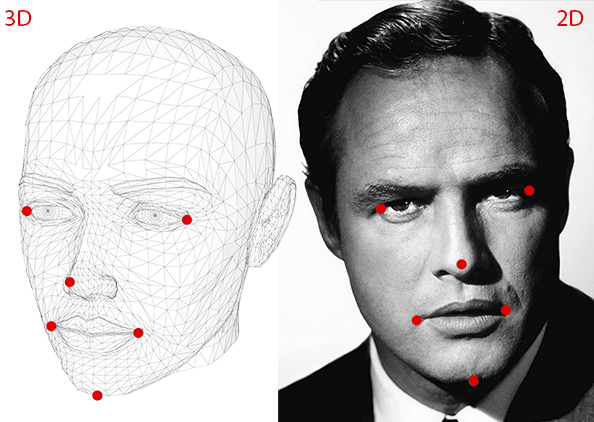


Fig 3.1.2.1 To calculate the 3D pose of an object in an image you need the following information

The roll, pitch and yaw angles for the above picture can be calculated as mentioned before, by knowing the 2D points. Then the 3D locations of a few points in some arbitrary reference frame to be obtained. We are going to use the following 3D points. Here chin is extra facial landmark for which roll, pitch and yaw are calculated.

1) Tip of the nose: (0.0, 0.0, 0.0)

2) Left corner of the left eye: (-225.0f, 170.0f, -135.0)

3) Right corner of the right eye: (225.0, 170.0, -135.0)

4) Left corner of the mouth: (-150.0, -150.0, -125.0)

5) Right corner of the mouth: (150.0, -150.0, -125.0)

6) Chin: (0.0, -330.0, -65.0)

Note that the above points are in some arbitrary reference frame / coordinate system. This is called the World Coordinates. The camera is assumed to be calibrated. In other words, you need to know the focal length of the camera, the optical centre in the image and the radial distortion parameters. So, you need to calibrate your camera. Of course it is not at all that easy to calibrate the camera and find out focal length and optical centre of the camera, this is a lot of work. But we can simply use approximations not with too large errors but with a small difference.

We are already in approximation land by not using an accurate 3D model. We can approximate the optical centre by the centre of the image, approximate the focal length by the width of the image in pixels and assume that radial distortion does not exist.

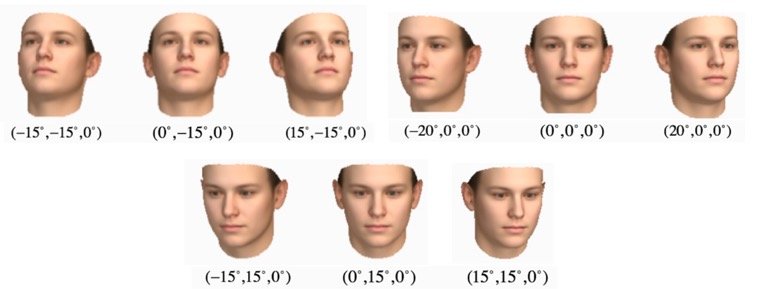


Fig 3.1.2.2 Roll, Pitch, Yaw according to head pose

The above image is just the few examples of the angles of yaw, pitch and roll of a human face according to head pose of the human. The angles shown in the above picture are approximate angles of the human head pose. The fraction part after the point is ignored.

**3.1.3 Working Algorithm of Head Pose**

There are three coordinate systems in play here. The 3D coordinates of the various facial features shown above are in world coordinates. If we knew the rotation and translation (i.e. pose), we could transform the 3D points in world coordinates to 3D points in camera coordinates. The 3D points in camera coordinates can be projected onto the image plane (i.e. image coordinate system) using the intrinsic parameters of the camera (focal length, optical centre, etc.).

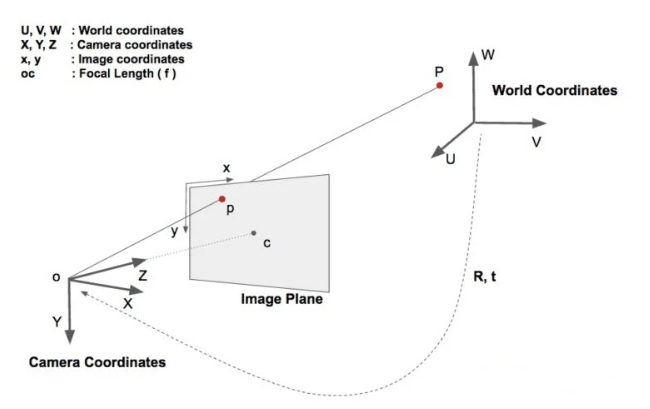


Fig 3.1.3.1 Transforming camera coordinates to world coordinates

In the figure above, O is the centre of camera and plane shown in the figure is the image plane. We are interested in finding out what equations govern the projection of the 3D point P onto the image plane. Assume we know the location (U, V, W) of a 3D point P in world coordinates. If we know the rotation R (3\*3 matrix) and translation t (3\*1 vector) of the world coordinates with respect to the camera coordinates, we can calculate the location (X, Y, Z) of the point P in the camera coordinate system using the following equation.

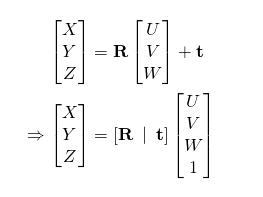


Fig 3.1.3.2 Equation for Transforming coordinates

In expanded form, the above equation looks like this

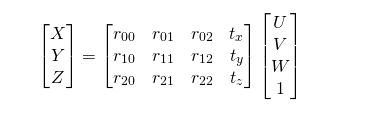


Fig 3.1.3.3 Expanded form of equation

The above is a linear system of equations where the rij and (tx, ty, tz) are unknowns and you can trivially solve for the unknowns. This is the process of obtaining the coordinates.

There are a variety of facial landmark detectors, but all methods essentially try to localize and label the following facial regions:

Mouth, Right eyebrow, left eyebrow, Right eye, left eye, Nose, Jaw.

The method starts by using:

1) A training set of labelled facial landmarks on an image. These images are manually labelled, specifying specific (x, y)-coordinates of regions surrounding each facial structure.

2) Priors, of more specifically, the probability on distance between pairs of input pixels.

**3.1.4 Estimated Face Pose Calculations**

The pre-trained facial landmark detector inside the DLIB library is used to estimate the location of 68 (x, y)-coordinates that map to facial structures on the face. Once we got the 68 facial landmarks, a mutual PnP algorithm is adopted to calculate the pose

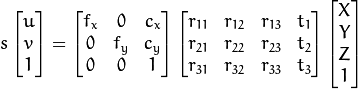


Fig 3.1.4.2 Typical camera pinhole equation

‘s’ represents Z coordinate of point in camera coordinate system.

Right, K[R|t] is projection matrix, which maps 3d coordinates in some object/world/global coordinate system into image 2d coordinates as in equation above. It is not so easy, because you often don't know point coordinates in **camera coordinate system**, but know 2D coordinates in **image coordinate system**. Transformation between **camera coordinates system** and **image coordinate system** loses one dimension, and there is also scale factor which makes our equation not-exactly linear. That's why it is not so easy to compute. Different algorithms use different approaches to add additional information needed for solution. For example, DLT (direct linear transform) method uses features of projection matrix. Beside analytic solutions there are also many methods which use nonlinear optimization - for example Levenberg-Marquardt used in OpenCV. Basically, a shape predictor can be generated from a set of images, annotations and training options. A single annotation consists of the face region, and the labelled points that we want to localize the face region can be easily obtained by any face detection algorithm (like OpenCV HaarCascade, Dlib HOG Detector, CNN detectors, …), instead the points have to be manually labelled or detected by already-available landmark detectors and models (e.g., ERT with SP68). Lastly, the training options are a set of parameters that defines the characteristics of the trained model.

The indexes of the 68 coordinates can be visualized on the image below:

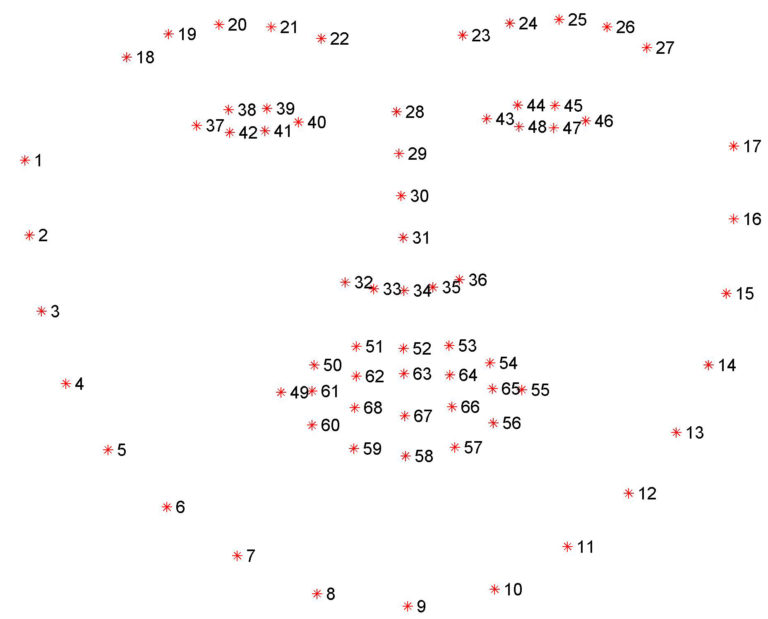


Fig 3.3.4.1 Visualizing the 68 facial landmark coordinates

Once all optimisation process is done, finally the head pose is estimated as shown below. This is the head pose estimation on a pre-existing GIF from the system library.

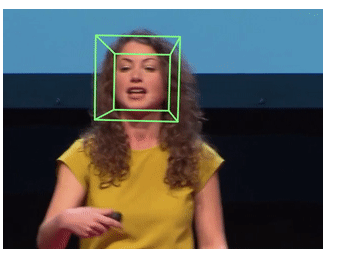


Fig 3.3.4.2 Head Pose estimation Output

**3.1.5 Expression Intensity Ranking**

The second module of our proposed system is the expression intensity ranking model. An expression intensity ranking CNN (EIR-CNN) is proposed to recognize the facial expression and evaluate its intensity from sequences simultaneously. The EIR-CNN is proposed to train the ranking model to estimate expression intensity using the ordinal information of facial expression sequences. The proposed method only needs a set of facial expression sequences evolved from onset to apex without labelling the intensity value for each frame. The EIR-CNN is applied to classify the interested facial expression by increasing inter-class dissimilarity with the intensity measure and reducing intra-class dissimilarity with the deep difference features. Moreover, a facial expression with any intensity can be classified by the intensity comparison with the reference of neutral expression. First, the expression intensity ranking problem and

organization of the training data for facial expression recognition are introduced. Then, the EIR-CNN is proposed to construct the expression intensity ranking model. Finally, the application of the proposed method to recognize facial expression is discussed.

**3.1.5.1 Expression intensity ranking problem**

Ranking was widely used in information retrieval, where many algorithms have been proposed in the past. Multi-class classification approaches overlook the ordinal consistency of the predicted intensities. Besides, regression approaches suffer from the difficulty of gathering the ground-truth values for the continuous intensity labels. Therefore, learning-to-rank (or ordinal regression) is a better choice for discrete intensity-level estimation. Among the learning-to-rank approaches, the thresholds model is a simple and intuitive method for employing the ordinal information, which can produce consistent and non-ambiguous ranking results. A ranking model that applies the large margin principle with the thresholds model is proposed. In this approach, there are a set of parallel hyperplanes that divide the data monotonically. By considering the cost-sensitive property to improve the performance, the parallel hyperplanes approach is extended and reduces ordinal ranking to binary classifications.

Given an expression sequence X = {xi |i = 1, . . . , |X|} evolved from onset to apex, where xi is the i -th frame and |X| is the length of the sequence, intensity labels associated with X is denoted as Y = {yi |i ∈ 1, . . . , |X|}. Facial expression intensity estimation can be regarded as an ordinal regression problem. However, the facial expression intensity of each image is difficult to label with absolute intensity values due to the lack of standard rules. Therefore, it is urgent to develop an unsupervised method to learn the ranking model. In an unsupervised settings, suppose that the intensity labels increase monotonously according to the temporal order of pair-wise data as yi ≤ y j , ∀(i , j ) ∈ {(i , j ) |1 ≤ i < j ≤ |X|}.

Hence, the ordinal regression problem can be transformed to a binary classification that distinguishes higher rank and lower rank between the pair-wise data. In order to further recognize the different expressions, it is necessary to define the intensity ranking relationship among different expressions of the same subject. Given an interested expression sequence Ei , the rank of the interested expression is higher than that of the other expressions Eo. To this end, facial expression data is organized as shown in Fig. 4, i.e., the intensity of interested expression sequence Ei decreases from the apex to the onset and then connects to the onset of the other expression Eo, and the intensity of expression Eo decreases from the onset to the apex.

Thus, the ranking score of the organized data X satisfies the constraint as R(XEi ,apex ) > R(XEi ,onset ) > R(XEo,onset ) > R(XEo,apex ). In this work, we employ the RED-SVM developed in our intensity-level estimation framework. An important characteristic of RED-SVM is rank-monotonic, i.e., g(x, 1) ≥ g(x, 2) ≥ … ≥ g(x, K-1 )for every x. Note that the rank-monotonic function g can provide consistent ranking results.

The consistent property is important to design a ranking rule r that there are no conflicts among the binary classifiers [g(x, k)> 0], 1 ≤ k ≤ K-1. In summary, RED-SVM constructs parallel hyperplanes to divide the data in a kernel space specified by the implicit mapping Φ. In our work, the radial basis function (RBF) kernels are used. The parameters of C and RBF kernel are selected using five-fold cross validation in the training dataset, and the default value of λ is 1 in RED-SVM.

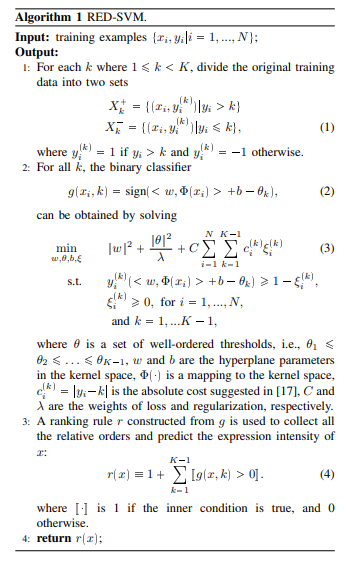


Fig 3.1.5.1.1 RED-SVM Algorithm

**3.1.5.2 VFOA Recognition**

VFOA recognition of the learner means Visual Focus of attention recognition of the learner. In order to evaluate learners’ VFOA recognition, we firstly evaluated the accuracies of learners’ head poses and landmark location in the AFLW dataset [6]. An accuracy within ±15◦ error tolerance was adopted as the metric of performance measurement of head pose, and the normalized average location error was adopted as the metric of performance measurement of single nose tip location and all the landmark locations.

We compared the proposed method with the original MTCNN and Hyperface. Compared with the original MTCNN, the proposed method achieved a better performance in landmark location which shows the added task of head pose estimation boosts the performance of landmark location. The average accuracy within ±15◦ error tolerance of head pose estimation using the extended MTCNN was 96.03%, which is 2.30% higher than that of Hyperface. The average error of all the landmark locations was 6.13 pixels, a comparable result to that of Hyperface, which adopts a deeper network.

The learning cognitive state is classified into three levels, i.e., low, moderate and high, which includes the learner’s attention and emotion. When the VFOA is estimated as T2, it shows that the learners’ attention is unfocused and the level of learning cognitive state is classified as ‘low level’; when the VFOA is estimated as T1 and the emotion is negative, the learning cognitive state is classified as ‘moderate level’; when the VFOA is T1 and the emotion is positive, the learning cognitive state is classified as ‘high level’.

To evaluate the learning cognitive state, every student’s learning records were stored on the intelligent classroom management system, then an expert in the area of education helped in interpreting a learning cognition state for the classification in ‘High’, ‘Moderate’, or ‘Low’ the levels based on the obtained attention and emotion of students. Additionally, in order to get the actual feed from the student, we send 100 questionnaires to students and teachers to access the true class learning cognitive state. The measurement of learning cognition state enables personalized intervention to promote learning outputs.

**3.2 Eye tracking system**

The eye tracking system is using the help of face landmark detector in order to make a model for the eye tracking system, so in order to know more about the eye tracking system we need to know more about the face landmark detector which can be used from the Dlib library in python.

**3.2.1 Face Landmark detector**

The second part of the first module in our project is facial landmark detection. Facial landmark detection based on convolution neural network. The model is built with TensorFlow, the training code is provided so you can train your own model with your own datasets. The version 1.x of TensorFlow is required. Though the current version of TensorFlow is 2.x we need to install the 1.x version of TensorFlow. Because latest version of TensorFlow does not have the attributes such as Session(), ConfigProto(), loader.load(). These attributes are deprecated in the latest versions of TensorFlow. By default, if you do not mention the version while installing it on your terminal it will automatically install latest version. Sometimes it shows error while mentioning version we need to install nightly provided by TensorFlow [14].

So, we need older version of TensorFlow to be installed if not we need to re-install it. TensorFlow TF 1.x requires the developer write Python code that populates a “computational graph”. This graph basically exists as a sort of hidden data structure underneath the Python code - you can read the values of variables in the graph at runtime, but you need get the variable from the graph first. The latter approach isn’t particularly Python, as it forces (or at least strongly encourages) you to write code relying on a huge structure of named global variables. It can be quite nice to work with, but you need to know what you’re doing to keep things.

TF 2.0 does away with many of the problems of variable management by simply doing away with named variables. While this seems harsh, it will lead to much cleaner, better organised code. In TF 2.0, you’ll need to keep track of the Python variable (handle) of your network variables if you want to use them later. Another big change, which is related to the Pythonic approach mentioned above, is that tf.layers is depreciated in favour of tf.keras.layers. This again should lead to much more manageable code, and much more uniformity in terms of how people design their neural network codes.

There are a bunch of other changes, better tf.data.Dataset API, improvements to Tensor board and eager execution by default. Only the public APIs of TensorFlow are backwards compatible across minor and patch versions.

We need these attributes because a Session object encapsulates the environment in which Operation objects are executed, and Tensor objects are evaluated. A session may own resources, such as tf.Variable, tf.queue.QueueBase, and tf.compat.v1.ReaderBase. It is important to release these resources when they are no longer required. To do this, either invoke the tf.Session.close method on the session, or use the session as a context manager. We may use compat.v1.Session() as a substitute for Session() but to ensure we get no errors further it is better to install the older version.

The ConfigProto() is just for a protocol message with two child classes DeviceCountEntry and Experimental. It is used to configure the session. It can also take in parameters when running tasks by setting environmental variable. The ConfigProto() protocol buffer exposes various configuration options for a session. For example, to create a session that uses soft constraints for device placement, and log the resulting placement decisions this can be used. Returns a context manager that makes this object the default session. Alternatively, you can create a session that is automatically closed on exiting the context, including when an uncaught exception is raised. As we used substitute for Session() similarly we can use compat.v1.ConfigProto() instead of it. The loader attribute is also deprecated similar to Session() and ConfigProto() and unfortunately there is no substitute for it.



Fig 3.2.1.1 : face\_landmark\_detector for multiples faces

While installing TensorFlow we can use “python -m pip install tensorflow-gpu” command to install TensorFlow with GPU support. A GPU (Graphical Processing Unit) is a component of most modern computers that is designed to perform computations needed for 3D graphics. Their most common use is to perform these actions for video games, computing where polygons go to show the game to the user. With a lot of hand waving, a GPU is basically a large array of small processors, performing highly parallelised computation. You basically have a mini-supercomputer\* running right now! While each of the “CPUs” in a GPU is quite slow, there are a lot of them and they are specialised for numerical processing. This means a GPU can perform lots of simple numerical processing tasks at the same time. In a great stroke of luck, this is exactly what many machine learning algorithms need to do.

**3.2.1.1 Estimating Face Direction from Landmarks**

OpenCV contains one of the first robust face detectors freely available to the public. In fact, OpenCV, in its early days, was primarily known and used for its fast face detection feature, implementing the canonical Viola-Jones boosted cascade classifier algorithm, and providing a pre-trained model. We may want to tweak the two parameters that govern the face detection:

1) pyramid scale factor 2) number of neighbours.

The pyramid scale factor is used to create a pyramid of images within which the detector will try to find faces as shown in the figure 2.2.1. This is how multi-scale detection is achieved since the bare detector has a fixed aperture.

In each step of the image pyramid, the image is downscaled by this factor, so a small factor (closer to 1.0) will result in many images, longer runtime, but more accurate results. We also have control of the lower threshold for a number of neighbours. This comes into play when the cascade classifier has multiple positive face classifications in close proximity.

Here, we instruct the overall classification to only return a face bound if it has at least three neighbouring positive face classifications. A lower number (an integer, close to 1) will return more detections, but will also introduce false positives.

**3.2.2 Working Model**

The eye tracking system takes the points from the face landmark detector as mentioned above which would be point 36-39 and 42-45 for the eyes which are basically the coordinates for different points of the eye. These points are used to calculate the position of the eye which would help us to analyse various features like knowing the position if the iris and simultaneously the approx. location of the pupil of the eye which would help us in getting the point of gaze of a person. The points mentioned are used to calculate the distance between them to know the average size of the eye as shown in the figure 3.2.2.1 below

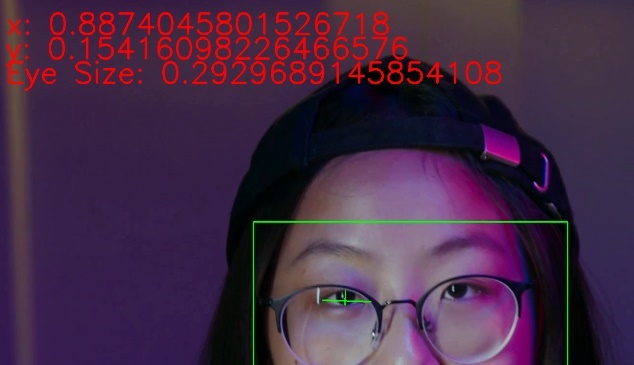


Fig 3.2.2.1: Distance measurement using points from face landmark detector

Now similarly using these points we will be able to identify the distance of the iris from these points and once we encounter that we later would also be able to get the distance between two far points of an iris which would provide with the size of iris ultimately using these points we calculate the centroid and as pupil usually falls in the centre of the iris we will be able to idnetify the position of the iris.

Once located we calculate the horizontal ratio and the vertical ratio and use those details to be passed on as input to the next model that would be the attention detection model.

**3.3 Comparing both the models**

The techniques used in calculating the head pose and eye tracking are similar since both of them require the use of facial landmark detector. While head pose focuses on the relative positions of the head by calculating the vectors as shown above , the eye tracker works in a much more simpler manner since it only requires to identify the eyes of an individual and would require lesser amount of focus and computation to be done for attentivity since lesser amount of data is being utilized, the technique would also be more accurate since a person maybe having perfect head pose, but may not actually be looking at the screen which would make it difficult for the model to generate a correct evaluation of attention.

The head pose estimation still works when compared in a holistic way, when compared to eye tracking which sure is convenient as it would as similar to head pose be calculated using the data from a normal camera there are various setup conditions which can hinder the eye tracking like the reflection of light or use of any sort of wearable would hinder the model and it would be difficult for it to identify the eye.

Though both models have their own advantages and disadvantages, they show pretty good results when used for attentivity and would work perfectly with a normal camera.

**4. INTEGRATED MODEL (ATTENTION ESTIMATION MODEL):**

The emotions generated are assigned with the values in order to identify the engagement of the student in the say any given lecture. This was made possible with the help of the affective circumplex model which was built by Feldman Barrett and Russell who used the previous done experiment with the name AEQ. Construction of the AEQ was based on the theoretical considerations outlined earlier and on a series of preliminary empirical studies. These studies included exploratory investigations analysing the occurrence and structures of various achievement emotions [9]. and four quantitative studies focusing on scale development [10]. The studies were guided by theory and were used to inform further development of theory-based emotion taxonomies which, in turn, were employed to construct the ﬁnal AEQ scales.

Thus, the strategy used involved theory-evidence loops integrating

both rational and empirical perspectives.

This experiment helped in the making of the effective circumplex which would help us to get the correlation between the various emotions and engagement. There are various models made with the help of these results by the people who have shown their way of circumplex which would constitute taking the various objects at various angles preferably taking account as 8 different types as shown in figure 4.2.3:

* + - * + high activation
        + low activation
        + positive
        + negative
        + activated positive
        + activated negative
        + deactivated positive
        + deactivated negative

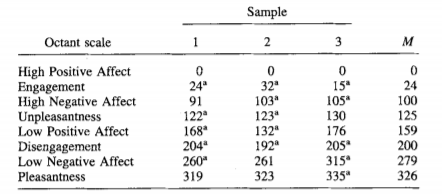


Table 4.1: Samples 1 and 2 test Watson and Tellegen's (1985) model; Sample 3 tests Russell's (1980) model. ' The polar angle deviates slightly from the hypothesized model

These 8 types help us in deciding the engagement of the student, where high activation is similar to saying engaged and low activation is similar to saying not engaged. The positive and the negative stating the emotion with positive being happy type emotions and negative being sad type emotions. The activated positive and activated negative can be then taken to stating the engagement with emotion similarly the deactivated mentions the opposite of activated. These points mention about the various emotions. Taking the prepared model built into account we built a new circumplex in order to fit it according to the seven emotions that we have available from the FER dataset. The angles for these points have been mentioned which help us in determining the value for the engagement of the student.

There have been various methods used in order to decide the effectiveness of the circumplexes and the angle they constitute according to them but the closest used were of Watson and Tellegen's and Feldman Barrett and Russell where they calculated the polar angles after trying to plot them accordingly in a 2D coordinate as given in table 4.2.1.

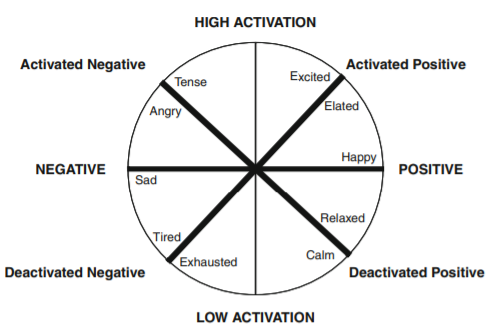
.

Fig 4.2: Affective circumplex (Model adapted from Feldman Barrett and Russell [ 1998], published by the American Psychological Association)

Now, taking the inspiration from the table 4.2.3 we make another circumplex model specific to our emotions from the data set we have. Both the circumplex mentioned say about the same with but use different descriptions like the activated positive in the Feldman’s model actually would mean engaged and deactivated negative would mean not engaged, similarly the part where they mention activated negative means unpleasantness and deactivated positive would mean pleasantness. Taking these things into account and also the table mentioned above we can take engagement as the main reference stating it to be 1, we can take the emotions at the following angles for the figure 4.2.4:

**0:** Angry - 125  
**1:** Disgust - 200  
**2:** Fear - 125  
**3:** Happy - 329  
**4:** Sad - 159  
**5:** Surprise - 24  
**6:** Neutral – 279

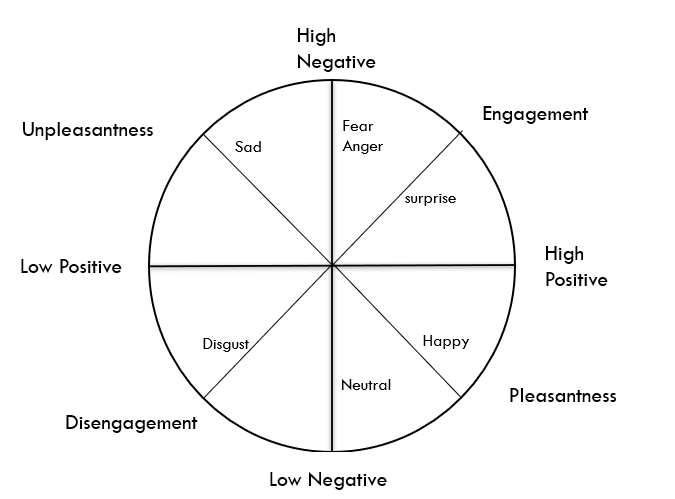


Fig 4.3: Affective circumplex model used for our emotion recognition model.

**4.1 Headpose**

Imagine three lines running through a point and intersecting at right angles.

Rotation around the front-to-back axis is called **roll**.

Rotation around the side-to-side axis is called **pitch**.

Rotation around the vertical axis is called **yaw**.

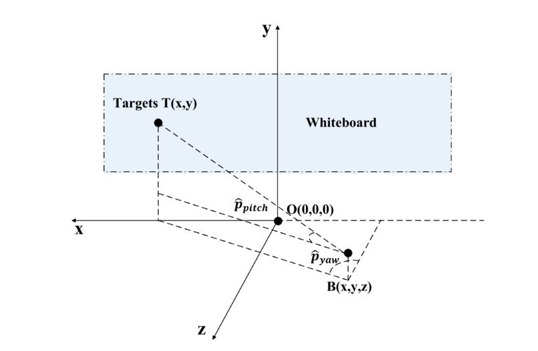


Fig 4.1.1 Geometric Relationship between the attention target, position and head pose of learner

All these parameters are taken into consideration and are used for calculating the headpose, by taking Target T as the blackboard and the student as the center point B. The VFOA target T(x,y) can be computed as:

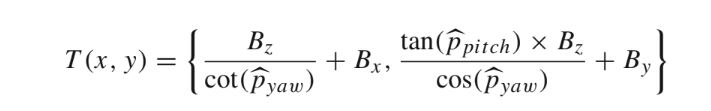


Fig 4.1.2 VFOA Target Formula

Where proll,ppitch and pyaw is the estimated values of roll, pitch and yaw.

After obtaining the model trained by the CNN, the locations of landmarks (two eye corners) are used for face alignment; that is, the reference line connecting the two eye centres is transformed to a standard location with a zero-degree orientation and a one-unit line length, and all the faces in the sequences are normalized by transforming and scaling together with the reference line. The aligned face sequences are employed to analyse the learners’ emotion, as introduced in the next section. On the other hand, the location of nose tip and head pose are used to estimate the visual focus of attention (VFOA). The VFOA is recognized by computing the learner’s attention target T(x,y) under the estimated head pose (ppitch, pyaw) and its position B(x,y,z).

**4.1.1 Head Pose Estimation**

In computer vision the pose of an object refers to its relative orientation and position with respect to a camera. You can change the pose by either moving the object with respect to the camera, or the camera with respect to the object. The pose estimation problem described in this tutorial is often referred to as Perspective-n-Point problem or PNP in computer vision jargon. A 3D rigid object has only two kinds of motions with respect to a camera.

1. **Translation**: Moving the camera from its current 3D location (X, Y, Z) to a new 3D location (X', Y', Z') is called translation. As you can see translation has 3 degrees of freedom — you can move in the X, Y or Z direction. Translation is represented by a vector \mathbf{t} which is equal to ( X' - X, Y' - Y, Z' - Z ).

**2. Rotation**: You can also rotate the camera about the X, Y and Z axes. A rotation, therefore, also has three degrees of freedom. There are many ways of representing rotation. You can represent it using [Euler angles](https://en.wikipedia.org/wiki/Euler_angles) ( roll, pitch and yaw ), a 3\times3 [rotation matrix](https://en.wikipedia.org/wiki/Rotation_matrix), or a [direction of rotation (i.e. axis ) and angle](https://en.wikipedia.org/wiki/Axis%E2%80%93angle_representation). To calculate the 3D pose of an object in an image you need the following information

**1. 2D coordinates of a few points**: You need the 2D (x,y) locations of a few points in the image. In the case of a face, you could choose the corners of the eyes, the tip of the nose, corners of the mouth etc. Dlib’s facial landmark detector provides us with many points to choose from. MTCNN will use the tip of the nose, the left corner of the left eye, the right corner of the right eye, the left corner of the mouth, and the right corner of the mouth.

**2. 3D locations of the same points**: You also need the 3D location of the 2D feature points [9]. We need a 3D model of the person in the photo to get the 3D locations. Ideally yes, but in practice, you do not. A generic 3D model will suffice. Where do we get a 3D model of a head from? Well, we really do not need a full 3D model. We just need the 3D locations of a few points in some arbitrary reference frame.

**4.1.2 Expression Intensity Ranking**

Given an expression sequence X = {xi|i = 1,...,|X|} where xi is the i-th frame and |X| is the length of the sequence, intensity labels associated with X is denoted as Y = {yi|i ∈ 1,..., |X|}. Facial expression intensity estimation can be regarded as an ordinal regression problem. However, the facial expression intensity of each image is difficult to label with absolute intensity values due to the lack of standard rules. Therefore, we develop an unsupervised method to learn the ranking model.

In an unsupervised setting, suppose that the intensity labels increase monotonously according to the temporal order of pair-wise data as yi ≤ y j, ∀(i, j) ∈ {(i, j)|1 ≤ i < j ≤ |X|}. Hence, the ordinal regression problem can be transformed to a binary classification that distinguishes higher rank and lower rank between the pair-wise data.

The video recorder captures the real-time videos of the students in a classroom while a lecture is going on. This recording is processed through the designed CNN algorithm with the set parameters and the results are generated. The results generated are the facial landmark locations, which in turn generate the facial expression and the head pose estimate. The former is used for face alignment and the latter is used for attention estimation.

These results are then processed to the second module which uses CNN (used as a metric) for calculating the result, which is a combination of the expression and the head-pose estimate. The result produced is a live feed and can be monitored in real-time for continuous evaluation.

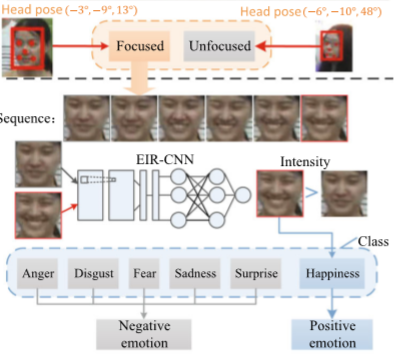


Fig 4.1.2.1 Description of deriving output

In the scenario above, two subjects are being studied. The first students’ head pose coordinates determine her to be focussed during the lecture. The second students’ coordinates show that she is unfocussed.

So, we further analyse the student who is focussed in the classroom. A sequence of her emotions is compared and processed through the EIR-CNN, which also takes the students’ emotions into consideration and the final concentration level of the student is estimated.

**4.2 Eye Tracking**

The later part of the model is tracking the eye movement of the person in order to determine the attentivity of the student. The eye tracking of the student is made possible with the help of the [face landmarks detection](https://pysource.com/2019/03/12/face-landmarks-detection-opencv-with-python/) approach which provides us with 68 points on the face or better said as the landmarks on the face with each point provided with an index or number, we detect the two eyes in the face with the help of following indices as shown in Fig 4.3.1[10]:

* Left eye indices: (36, 37, 38, 39, 40, 41)
* Right eye indices: (42, 43, 44, 45, 46, 47)

With 36 being the leftmost and 39 being rightmost part of the left eye and 42 being the leftmost and 45 being the rightmost part of the right eye.

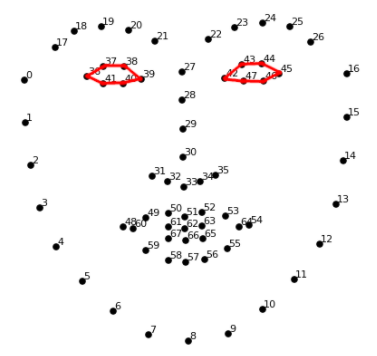


Fig 4.2.1: Face landmarks marked with number for detection of particular face feature

These indices provided by the face landmark detector provides us with the data of the coordinates of these points which help us detect the eye from the whole face. The later part is to determine the pupil of the person which is made possible by marking eye into two parts mainly saying the white part and the dark part which would help us to get the pixels of the iris from the eye. We extract the pupil by ignoring the white part and after knowing their coordinates we calculate the centroid of the eye. This is done both for the left eye and the right eye and thus this centroid allows us to know the position of the pupil at any time.

In order to know the attentiveness of the student we use the help of the coordinates of the pupil we can determine so by calculating the horizontal and the vertical ratio of the eye with the horizontal ratio being calculated with the help of sum of the coordinates of the eye with the coordinates of the ratio and similarly with the vertical ratio. The horizontal ratio generates a number between 0 to 1 with 0 looking at extreme right and 1 looking at extreme left. Similarly, the vertical ratio shows the value between 0 to ~1.2 with 0 looking extreme down and ~1.2 stating looking extreme up. These values have been studied for knowing the attentivity of the student and we arrived at the conclusion that the values ranging under the 0.5 to 0.75 can be said to be attentive in the horizontal ratio and the 0.6 to 0.8 range can be said to be attentive for the vertical ratio as per the figure 4.3.2.

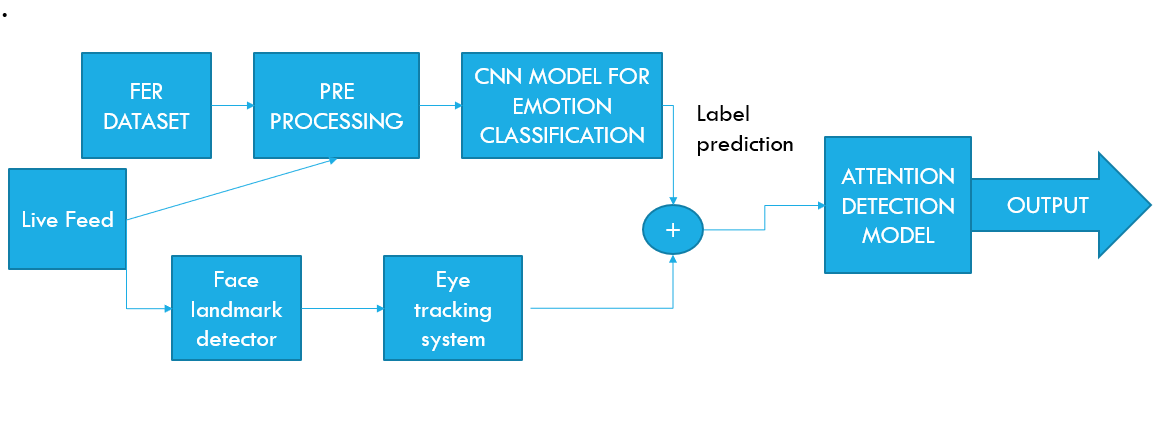


Fig 4.2.2: Integration of Emotion Detection and eye tracking system to make attentivity model.

**4.2.1 Attention detection model**:

This model built combines the output obtained from the emotion recognition model and the eye tracking model. The code when run stores all the values obtained from the emotion recognition model and also the eye tracking model and stores in an excel file. The data stored in the excel files are stored taking data as 1 frame per second, basically saying that that the data takes 60 frames per minute and both horizontal and vertical ratio for eye tracking and emotion detected are stored in this file.

The emotion recognition model stores the data stating the emotion of the person and these emotions as mentioned above are classified into 7 categories. These categories are calculated for their engagement according to the above circumplex which describes about the correlation between the emotions and engagement of a person. The values obtained after knowing the emotion are then processed by calculating the average of the emotions which would provide us with the metric of percentage of engagement of a person which would be calculated at every 5-minute checkpoint.

The eye tracking similar to the emotion detection would also take the note of the horizontal ratio and the vertical ratio and calculating the average of the ratios obtained is taken to decide the attentivity of student in the class such that the model also decides to take a note the count of times the student is not looking towards the screen as 20% limit of the time to which we will be able to determine the students absence in the that particular time frame and those frames are given -1 value stating that student was not attentive in this timeframe. The time frame where the attention is not detected is also taken account for engagement calculation and the engagement of student is not given.

**5. Innovative Teaching Strategies.**

“In education, student engagement refers to the degree of attention, curiosity, interest, optimism, and passion that students show when they are learning or being taught, which extends to the level of motivation they have to learn and progress in their education”. When students are engaged with the lesson being taught, they learn more and retain more. Students who are engaged in the work tend to persist more and find joy in completing the work.

Student engagement represents two critical features of collegiate quality: how much time and effort students put into their studies and other educationally motivated activities; and how the educational institution deploys its resources and organises the curriculum and other learning opportunities in order for students to participate in activities that are linked to student learning.

Student engagement also refers to the degree of attention, curiosity, involvement, optimism, and passion that students display while being taught, which improves how much they learn and retain, as well as their persistence and enjoyment in completing work.

If student engagement is proven to be the more effective teaching strategy, then why is it adopted so poorly? One potential reason highlighted by a Nebraska study is a lack of training for faculty in the skills needed to utilise the innovative strategies that improve student engagement, such as smaller class sizes, open classroom layouts, and other strategies that minimise the reliance on conventional lecturing.

There are various kinds of innovative teaching strategies used, but we use this opportunity to compare the methods like the **Role-play** , Predict-Observe –Explain and normal ppt mode of teaching to determine the best way teaching strategy .

**5.1 Role play**

**Role-play** is a technique that allows students to explore realistic situations by interacting with other people in a managed way in order to develop experience and trial different strategies in a supported environment. Depending on the intention of the activity, participants might be playing a role similar to their own (or their likely one in the future) or could play the opposite part of the conversation or interaction. Both options provide the possibility of significant learning, with the former allowing experience to be gained and the latter encouraging the student to develop an understanding of the situation from the ‘opposite’ point of view.

Participants are given particular roles to play in a conversation or other interaction, such as an email exchange, typical of their discipline. They may be given specific instructions on how to act or what to say, as an aggressive client or patient in denial, for example, or required to act and react in their own way depending on the requirements of the exercise. The participants will then act out the scenario and afterwards there will be reflection and discussion about the interactions, such as alternative ways of dealing with the situation. The scenario can then be acted out again with changes based on the outcome of the reflection and discussion.

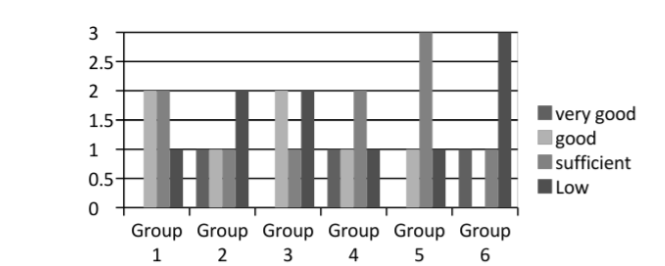
**5.2 Predict Observe Explain(POE):**

The POE strategy was developed by White and Gunstone (1992) to uncover individual students’ predictions, and their reasons for making these, about a specific event.

POE learning models can include ways that can be taken by a teacher to assist students in improving the understanding of the concept and their psychomotor. POE learning model engages students in predicting a phenomenon, observations through demonstrations or experiments, and finally explain the results of the demonstration as well as their hypothesis. By doing this way, acquired knowledge will be preserved in students’ memory and increase students' science processing skills. To make an active teaching-learning process, students need to be able to clearly express themselves in written form and verbal form; teachers need to introduce a new teaching strategy like the Predict-Observe-Explain (POE) that can be used in association with demonstrations and hands-on activities that can help to enhance classroom practice by identifying the learner’s conception. POE is also suited to be applied in physics subjects that can mostly be observed in experiments, and help to solve misunderstanding According to the given statements, Predict-Observe-Explain (POE) should be able to be applied as one of the solutions to solve the problem at school regarding the topic of vibration and wave.

The first model to be experimented upon is the role play model. The model consisted of around 19,200 images obtained through the video of lectures. **Role-play** is a technique that allows students to explore realistic situations by interacting with other people in a managed way in order to develop experience and trial different strategies in a supported environment [11]. Role-play is a very flexible teaching approach because it requires no special tools, technology or environments, for example student could work through a role-play exercise just as effectively in a lecture hall as in a seminar room. However, technology can provide significant advantages, and even new possibilities, for using the approach as a learning activity.

There have been various experiments done on this technique, one we can take as a comparative study is by L Hidayati1 and P Pardjono in their paper The implementation of role-play in education of pre-service where they try to understand the way a student is behaving to role play and without role play. They divided the students in 6 groups to determine its effects as seen in figure 5.2.1.

The end result of the student’s reaction to role-playing confirmed that the gaining knowledge of model: made college students less difficult in knowledge gaining knowledge through role play than gaining knowledge of via way of means of themselves (100%); being capable of domesticate the mind-set and to educate the student’s responsibility (90%); turned into clean to put into effect in college level (90%); cultivated the mind-set and educated student’s awareness (100%); being capable of domesticate the mind-set and to educate the student’s independence (75%); educated the scholars as a potential teacher (100%).

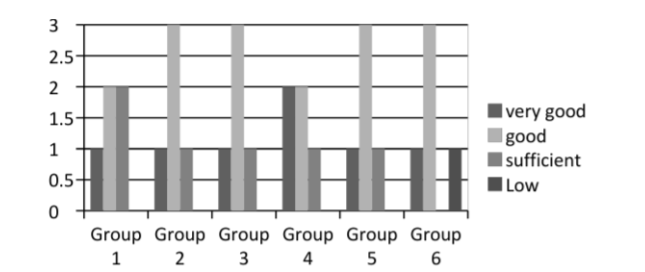


Fig 5.2.1: Students performance analysis by comparing against the role play technique of study.

Individual results of the experiment conducted show the change of student’s engagement to differ by a great margin of 81% when the experiment has been done from our end. This acts as a clear indication of superiority of a teaching method applied in enhancement of students understating of concepts. The use of concept not only allowed us to identify the role of teacher in making learning more prudent but also the use of groups to allow students to explore the topic to point out the mistakes in it their own way. The way of experiment is sure different yet both of the experiments determine the role play to be a better way of teaching strategy. As seen in the figure 6, the percentage change in positive emotions of the students is greater in this section as to amount about 45.4% of the total suggesting more involvement of student in a class and the next model, we have used is for predict observe explain.

In Predict-Observe –Explain strategy students are required to predict the outcome of an event or experiment. The experiment is then performed and observations made by students are probed.

The use of experiment of POE has also been done earlier to make a study for it lets compare it with The Effect of Predict-Observe-Explain (POE) Strategy on Students’ Conceptual Mastery and Critical Thinking in Learning Vibration and Wave by Dandy Furqani1\*, Selly Feranie 2, Nanang Winarno1 who used the method by using the concepts of wave and vibration to understand student’s understanding of the concept [12].

POE method shows enhancement in college students’ conceptual mastery, indicated by normalized N-gain value 0.29 as for vital questioning skills, POE method appears appropriate to enhance vital questioning ability [13]. Using POE method, the end result confirmed that college students advantage growth in vital questioning from stage 1,30 (challenged thinker) to 2,07 (starting thinker) POE is appropriate to put in force the know-how into college students. The results being compared shows the effectiveness of the technique while the experiment from the perspective of the other researcher may not show as better result when compared to our experiment, it can also mean the involvement of other factors which could effect the understanding of a student in this method, yet there is no denying the fact of its effectiveness too be better than other method of teaching. As shown in figure 5.2.2 are the cognitive levels according to the various methods in predict (c3), observe (c2) and explain (c1).

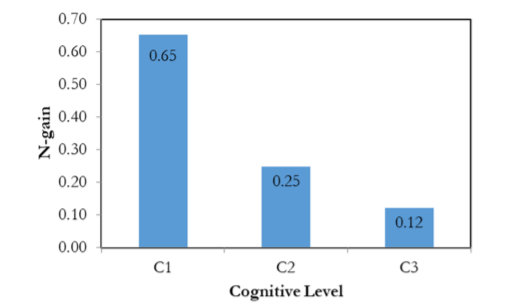


Fig 5.2.2: various cognitive levels in the stages of POE.

When predictions and observations are inconsistent with each other the students’ explanations are explored and as the experimental results suggested the change in positive emotions of the students is around 43.8% and lastly the normal teaching method using the ppt had the attention estimating a 10.7% of the total. Now we can do conclude that innovating teaching method have shown more positive result than the normal teaching methods as seen in the figure 5.2.3.

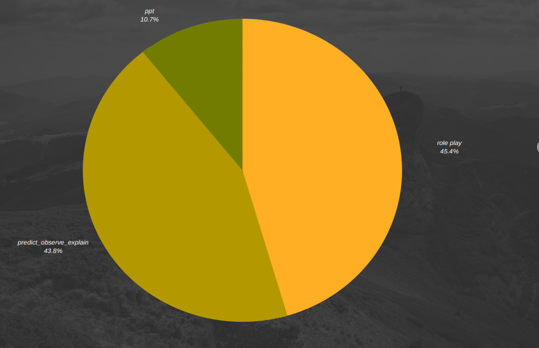


Fig 5.2.3: Percentage of positive emotion change in various teaching strategies for students.

Though the innovative teaching method sure make a difference there is yet to observe the attention of a lecturer during the process and doing so we have obtained the following results. The role plays though producing much promising results also invokes a lecturer to pay more attention towards the whole session which was calculated to around 52.5% of the total. The ppt follows the lead with 25.4% of the total engagement, not producing results on par with role-play or predict-observe -explain. Lastly, the predict-observe-explain uses least engagement by the lecturer and yet producing more results as seen in the figure 5.2.4.

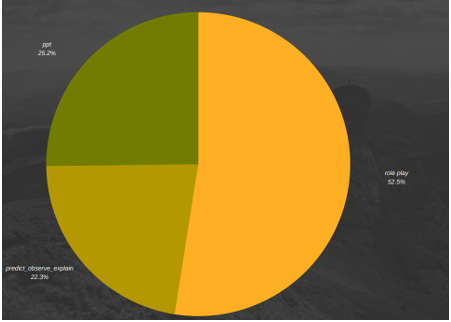


Fig 5.2.4: Percentage of positive emotion change in various teaching strategies for lecturer.

These two results helps us to understand the difference between the innovative teaching stratergies and also the normal methods and also their correlation with the amount of attentiveness paid by the student and also the lecturer in these sessions.Though the role play seems to act better in gaining attention, yet has to also be provided with much attention by the lecturer, whereas infact the involment of lecturer is least in the predict observe explain was still able to draw out nearly as much attentiveness as the role-play and the normal ppt method has been proved to be least efficient of the all methods.

**6. RESULTS & TESTING**

**6.1 Emotion Recognition**

The result from the emotion recognition model shows multiple faces being detected along with the emotions also being detected with the model. The model that we have prepared as said before is able to identify emotions with an accuracy of 97%. The earlier accuracy was 94% from the model but later we applied transfer learning on the existing model and thus were able to reach an accuracy of 97% as shown in the code in figure 6.1.1 and figure 6.1.2 below.

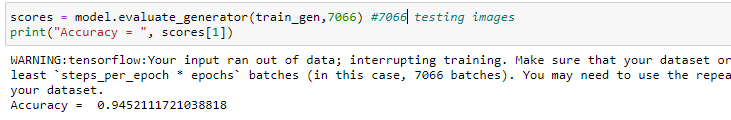


Fig 6.1.1: Accuracy for the 1st emotion recognition model.

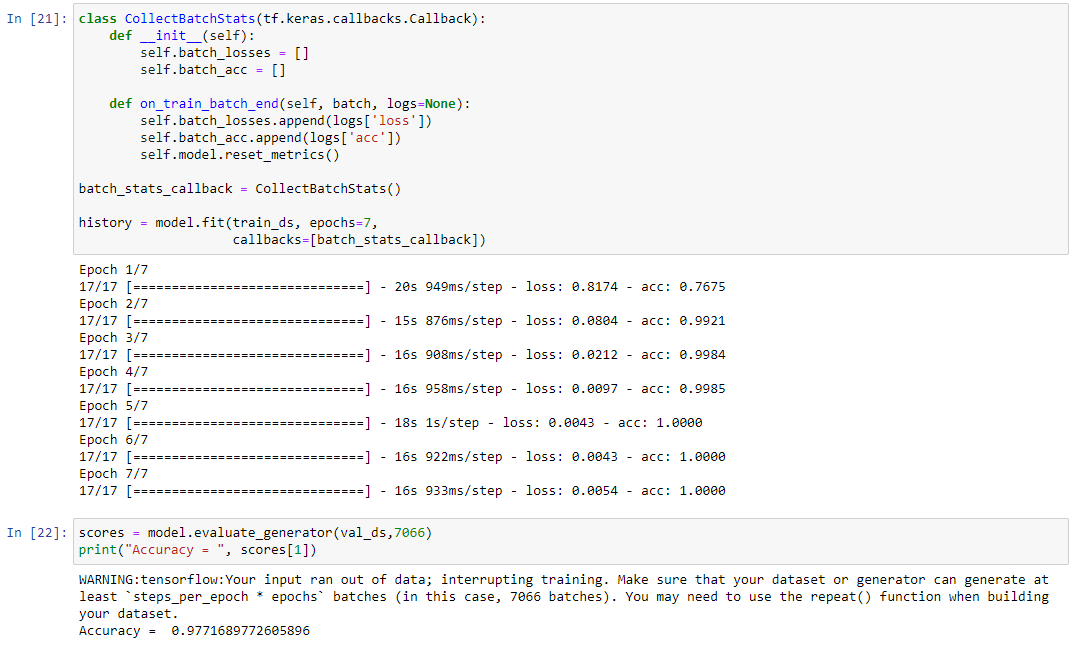


Fig 6.1.2: Accuracy for the transfer learning model.

The model is made with the help of the architecture similar to VGG 16 which took the input as 48 x 48 pixels images that’s why we need to convert even the images we are taking from the videos and convert them into grayscale images along with resizing them to the size of 48 x 48 whose result is shown in figure 6.1.3. The result of running the emotion recognition code on the AI Server is given in the figure 6.1.4.

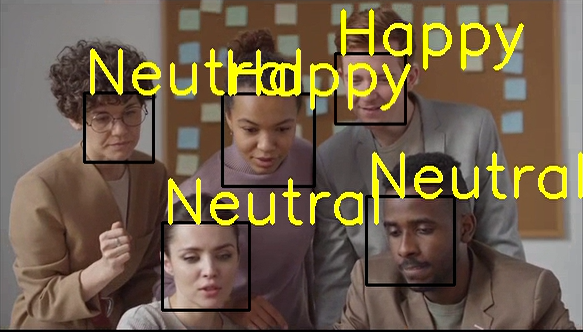


Fig 6.1.3: Result from the emotion recognition model where multiple faces are being detected and emotion being predicted.

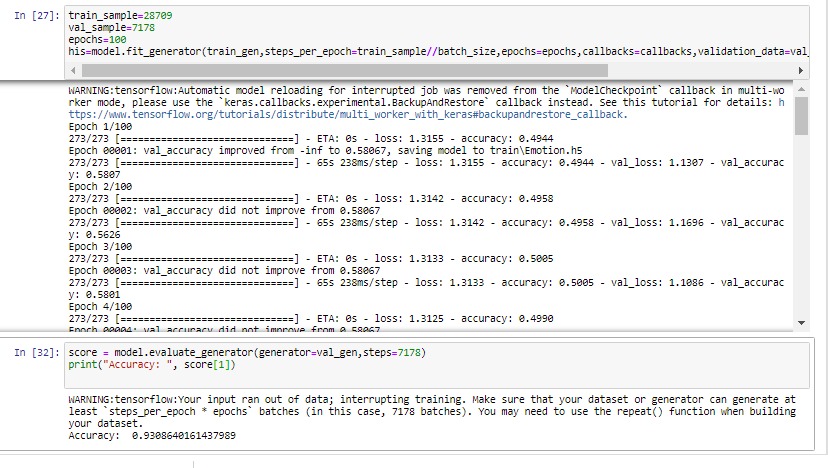


Fig 6.1.4: Result from code run on Deep Learning Server

**6.1.1 Comparative study of emotion recognition model:**

The experiments on the FEC dataset were also done before FEC dataset is a dataset similar to CK+ dataset with minor changes, which would be addition of more data.

The proposed model is compared with the earlier experiments done like SENet Teacher, RAN, PSR.

The first being SENet Teacher ,they consider the task of learning embeddings for speech classification without access to any form of labelled audio. They had a simple hypothesis: that the emotional content of speech correlates with the facial expression of the speaker. By exploiting this relationship, they show that annotations of expression can be transferred from the visual domain (faces) to the speech domain (voices) through cross-modal distillation. Though the model was good, it’s efficiency decreased in the testing field.[14]

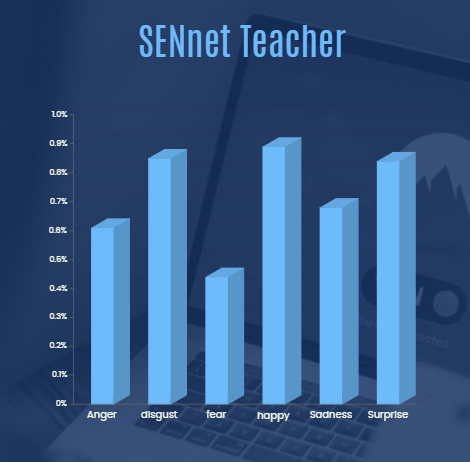


Fig 6.1.1.1: Expression detection of SENnet Teacher on FER dataset.

The next method used is the RAN they stimulate the research of FER under real-world occlusions and variant poses and build several in-the-wild facial expression datasets with manual annotations for the community. Then propose a novel Region Attention Network (RAN), to adaptively capture the importance of facial regions for occlusion and pose variant FER. The RAN aggregates and embeds varied number of regions features produced by a backbone convolutional neural network into a compact fixed-length representation.

# 

Fig 6.1.1.2: Expression detection of RAN on FER dataset

# This paper focuses on automatic FER on a single in-the-wild (ITW) image. ITW images suffer real problems of pose, direction, and input resolution. In this study, we propose a pyramid with

# super-resolution (PSR) network architecture to solve the ITW FER task. They also used prior distribution label smoothing (PDLS) loss function that applies the additional prior knowledge.

# 

Fig 6.1.1.3: Expression detection of PSR on FER dataset

The earlier models seem to work poor in some of the emotions. The proposed models use the convolutional neural network to as modified VGG-16 model in-order to predict the above-mentioned classes. The classes mentioned were predicted with an accuracy of 93%.



Fig 6.1.1.4: Expression detection of Modified VGG-16 on FER dataset

# The model after reaching a point seem to be overfitting the model so optimizers such as early stopping, reduceonLR have to be used in order to prevent the model from overfitting and making the model worse.

**6.2 Head pose**

The proposed learning cognitive state analysis framework has been thoroughly evaluated with different quality images from publicly available databases and our CCNU classroom dataset. The CCNU classroom dataset was collected from a front camera in an intelligent classroom and contains unconstrained face images with poses ranging from −90◦ to +90◦ and spontaneous expressions.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Method | Ppitch (%) | Proll(%) | Pyaw(%) | Average error of nose  tip location | Average error of all the landmark locations |
| Hyperface | 97.4 | 94.8 | 89.0 | **-** | 4.26 |
| Original MTCNN | **-** | - | **-** | 1.57 | 7.19 |
| Extended MTCNN | 98.0 | 95.5 | 94.6 | 1.50 | 6.13 |

Table 6.2.1 Accuracies of different methods for head pose estimation and landmark locations in AFLW datasets

The accuracy of pitch, roll and yaw of extended MTCNN is more than that of hyperface. The average error of all the landmark locations for extended MTCNN is less than original MTCNN and more than hyperface.

A screen shot of a person

Description automatically generated

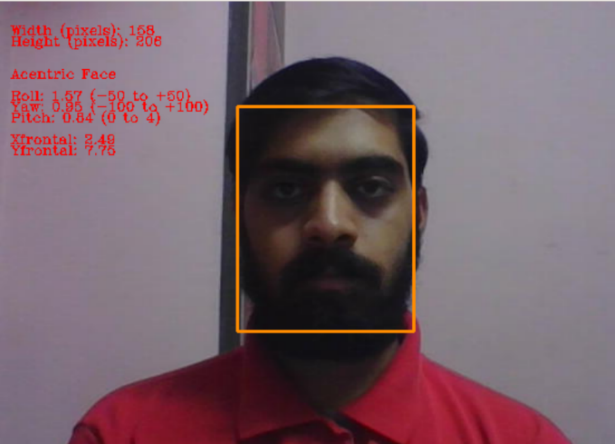
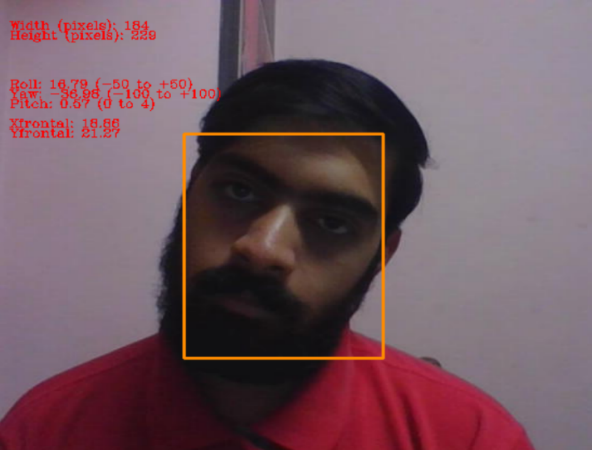
A screen shot of a person

Description automatically generated

Fig 6.2.1 Head Pose Estimation Output

**6.2.1 Pitch, Roll, Yaw Output**

The change in the angles of pitch, roll and yaw according to the movement of face about the three orthogonal axes is shown in the below figures

 ****

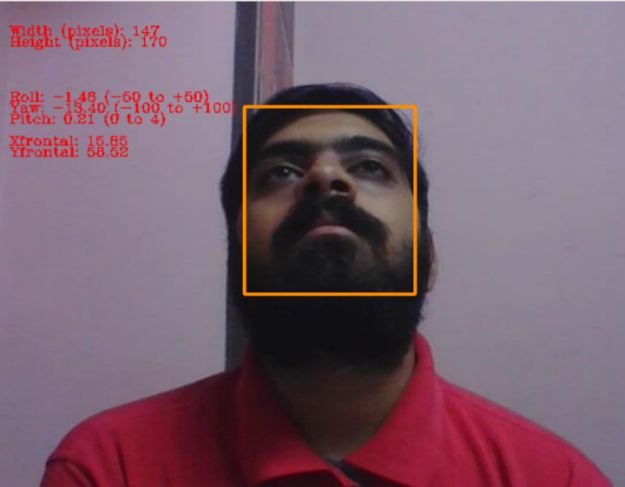
** **

Fig 6.2.1.1: movement of face about the three orthogonal axes

**6.3 Eye Tracking**

The eye tracking code as described allow us to determine the point of gaze of a person as seen in the figure below, we are able to identify that the person is looking at centre which is made possible with the help of horizontal ratio and vertical ratio. The photo also shows the coordinates of each eye as shown in figure 6.3.1 and figure 6.3.2. As mentioned, the ratio for horizontal ratio and vertical ratio where horizontal ratio should have a range between 0 to 1 and the vertical ratio should have a ratio between 0 to ~1.2.



Fig 6.3.1: Result from the eye tracking model looking at centre



Fig 6.3.2: Result from the eye tracking model looking at right

**6.4 Attention Estimation System**

The attention estimation system uses both the eye tracking system and the emotion recognition system to identify the attention and engagement of the student. As mentioned above these values both are identified and maintained in a log file with 1 frame per second with values of horizontal ratio, vertical ratio and the emotion being expressed as shown in figure 6.4.1 and figure 6.4.2.



Fig 6.4.1: Result from mainrunner function integrating both the emotion recognition model and eye tracking model.

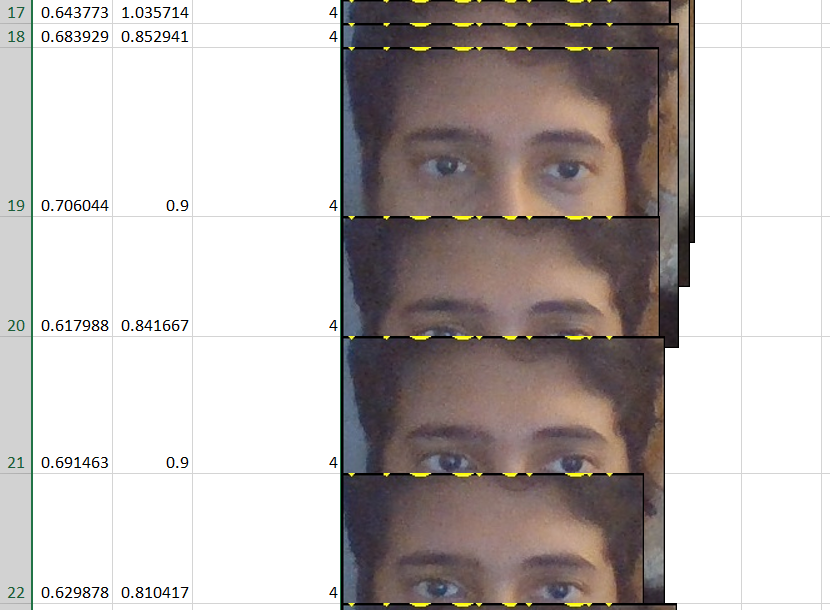


Fig 6.4.2: Data obtained and stored in the files.

The above figure shows the data that is being generated in the mainRunner.py filw where 1st column is horizontal ratio , 2nd column is vertical ratio, 3rd column is the emotion and 4th column are the images of these data.

* 5 min interval data for 40 minutes

The data which is made with the attention estimation as have now been converted into an excel file. This excel file is processed to get the final result where we have the average of both the ratios which decide the attentivity of student and after solving things through the circumplex we will be able to identify the engagement of the student as shown in figure 6.4.3.

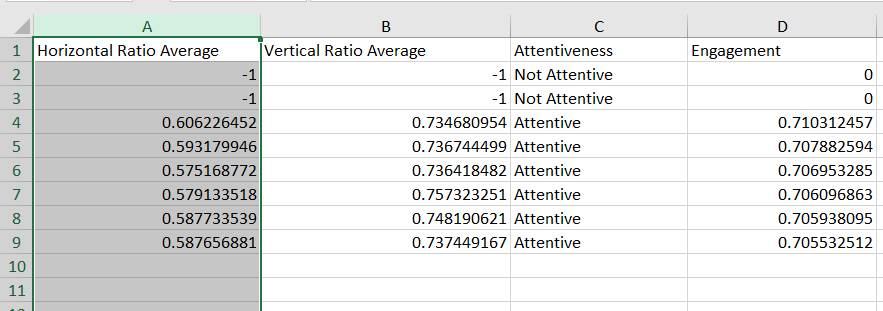


Fig 6.4.3: results from the eye tracking and the emotion recognition model being shown for attention and engagement.

**6.5 TESTING**

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. In simple words, testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements. Testing has always been part of Software Carpentry. We admit that testing can't possibly uncover all the mistakes in a piece of software, but testing has always been useful

**6.5.1 Unit Testing:**

Unit testing is the first level of testing in software testing where individual components of a software are tested. Unit Testing is also called Module Testing or Component Testing. It is done during the development of an application to check whether the individual unit or module of the application is working properly. Considers each component, as a single system and evaluates it. Scope of Unit testing is limited to a particular unit under test. It comes under White Box Testing as shown in the code in figure 6.5.1 below.



Fig 6.5.1: Unit testing on the emotion recognition model

In the figure above we are unit testing the emotion recognition model to check its functionality and if it is working properly is when we move on to the next part of the code.

**6.5.2 Integration Testing**

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing in general. Integration testing helped us uncover the problems we had while trying to integrate the independent components as a whole. The individual modules are first tested in isolation. Once the modules are unit tested, they are integrated one by one, till all the modules are integrated, to check the combinational behaviour, and validate whether the requirements are implemented correctly or not as shown in the code in figure 6.5.2.1 below.

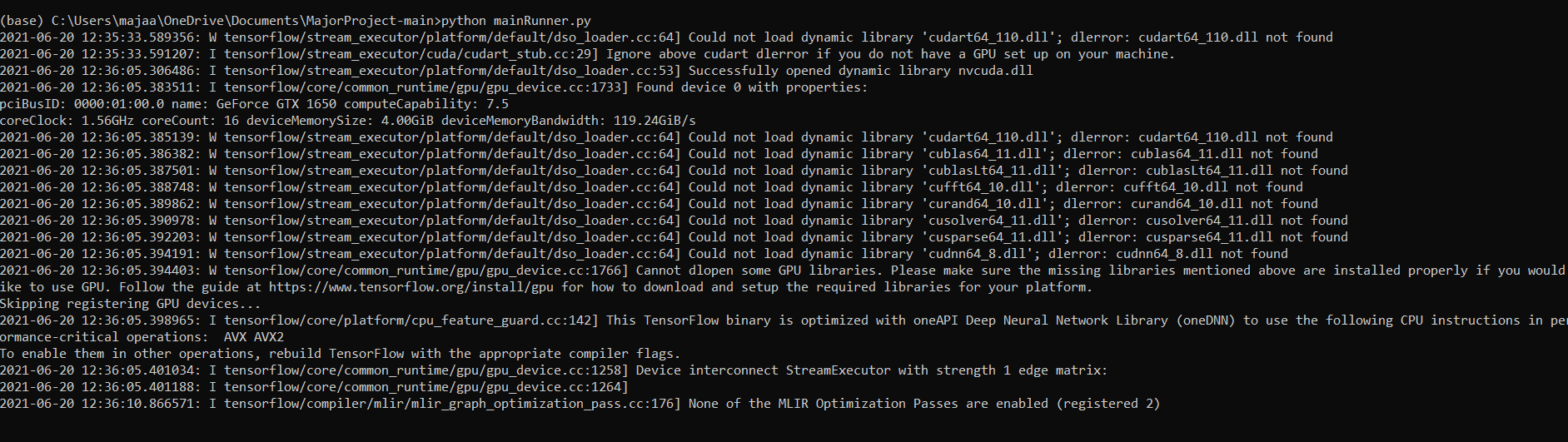
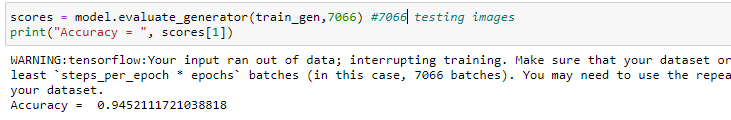


Fig 6.5.2.1: Integration testing on the emotion recognition model and eye tracking model.

Integration testing done with the both eye tracking and emotion recognition system.

**6.5.3 Comparison testing**

It is a type of testing in the product's strength and weaknesses is compared with its previous versions or other similar products. In our project the comparison is between the datasets we used for emotion detection. The emotion detection module is tested for FER2013 dataset with the model first made with the normal process and the second one with the transfer learning and it sure shows the improvement of accuracy as shown in the code in figure 6.5.3.1 below.



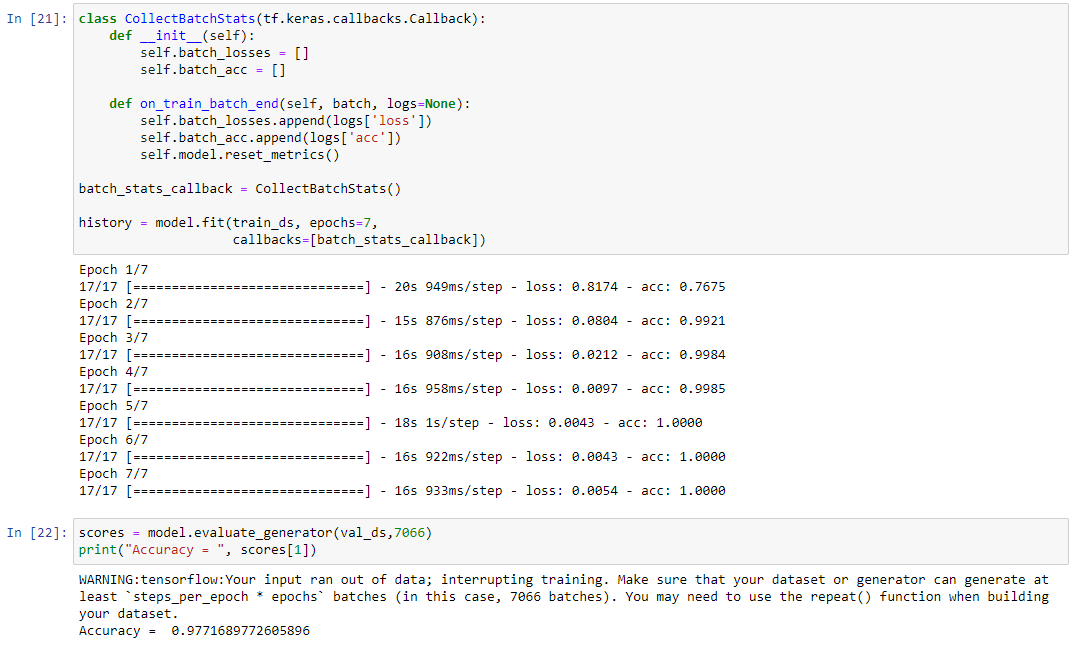


Fig 6.5.3.1: Comparing models and their accuracy

**6.5.4 Importance of testing:**

The main advantage of Unit testing are it simplifies the debugging process. When unit testing is implemented only the later changes made in the code need to be debugged. The test cases are based on the specifications of the software component under test. Whenever there is a floating-point computation that is then quantized to produce an output image. Testing each unit is important because all units must perform in an efficient manner. Software testing is now a key component of software product development because it improves consistency and performance. If all bugs are removed, the software will be more accurate and reliable.

**7. Conclusion and future scope**

Learning cognitive state analysis in a non-invasive way is an onerous task, and the associated tasks of attention and emotion estimation from time series data are two significant research challenges. To address this problem, this paper proposes an emotion-sensitive learning cognitive state analysis system applied in intelligent classrooms. The proposed system has two modules. The first module implements the CNN for face detection, landmark location, and head pose estimation. The located landmarks are used to align the face for facial expression analysis. The head pose is used to recognize the learners’ VFOA. The performance of head pose estimation can be enhanced with the subtask of face detection and landmark location. The second module uses an EIR-CNN to recognize facial expression and evaluate intensity simultaneously. The facial expression is analysed to estimate the learners’ positive/negative emotion when focused on the whiteboard. The proposed EIR-CNN outperformed the existing methods in two public datasets. The experimental results show that the method can estimate a learner’s attention and emotion with correctness rates of 79.5% and 88.6%, respectively.

This can be used also with the eye tracking system mentioned in order to estimate the attentivity which can be used not only in educational sector but also for marketing, analysis etc . The modification of the VGG – 16 model helps us in improving the accuracy. Though FER dataset contains images the dataset can be improved to have better quality images with accurately labelled classes so that models can be made more accurate further any use of speech recognition along with the existing model can help us in getting even greater accuracy.

The emotions generated though produce the results according to the affective circumplex model but it’s also important to check for errors if they are being generated in the model and maintain a note of the model to make the circumplex more accurate to provide more accurate correlations between the emotion and engagement. This model can be further developed to get the attention and engagement of not only the students but also other people like for advertisements in malls or roads to identify what catches the eye of a customer basically saying it can be used for situations where we need to determine any person’s attentiveness and engagement.

The model successfully was able to prove the superiority of the innovative teaching stratergies in increasing the engagement of a student in a class. The innovative teaching stratergies like the role play and predict observe explain proved to be more engaging in a class and thus produced better results where as on other hand the conventional teching stratergy shows less promising results.Though this stands true , there is also other angle for the things to be seen , where we analyse the engagement of the teacher , which would help us understand the amount of invovlement and focus that the teacher needs to exert in the whole session.Even though where we were able to see practically less difference in the role play and ppt though we can see the difference in the amount of output being given by both of them when compared side to side.Thus taking everything into account , we can finally conclude without any doubt that the innovative teaching stratergies far exceed in enagaging a student in a class rather than the conventional teaching methods.

The proposed model can serve other purposes also other than in intelligent classrooms like, it can be used where facial recognition is used for security lock, as the emotion is associated along with the face, it cannot be opened easily. The security is increased and also it reduces the threat to user’s privacy. It does not require any highly sophisticated devices so it can be used easily.

The future work can also be done on using various kinds of IOT devices as mentioned above to use the model in remote areas too or we can just use the IOT devices at compact places as mentioned for advertisement, security etc . We in this particular thesis have focused on only the academic ways in which it could be used , it can act as an excellent feedback model and can also be used in corporates to understand the employees mental state to some extent and provide any aid to them if necessary.

Existing systems use invasive technique which harm the subject. Experimental results demonstrated that our methods consistently outperform the other methods. It achieves real time performance for 640x480 VGA images with 20x20 minimum face size, we will further improve performance even for lesser face size . The proposed model uses non-invasive technique so it can be used without harming the subject. In future work, we will attempt to fuse the multimodal features to understand learners’ cognitive state. We will further improve the accuracy, and address the efficiency of the proposed method for other datasets.

**REFERENCES:**

[1]: Siau K, ShengH,NahFH(2006) Use of a classroom response system to enhance classroom interactivity. IEEE Trans Educ 49(3):398–403

[2]: Ashby FG, Isen AM, TurkenAU(1999)A neuropsychological theory of positive affect and its influence on cognition. Psychol Rev 106(3):529–550

[3]: Mehrabian A, Wiener M (1967) Decoding of inconsistent communications. J Pers Soc Psychol 6(1):109–114

[4]: . Zhang K, Zhang Z, Li Z, Qiao Y (2016) Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Process Lett 23(10):1499–1503

[5]: Ranjan R, Patel VM, Chellappa R (2016) Hyperface: a deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. IEEE Trans Pattern Anal Mach Intell 41:121–135

[6]: LiuY,Wang L, LiW(2017) Emotion analysis based on facial expression recognition in virtual learning environment. Int J Comput Commun Eng 6:49–56

[7]: Russakovsky, O., Deng, J., Su, H., et al. “ImageNet Large Scale Visual Recognition Challenge.” International Journal of Computer Vision (IJCV). Vol 115, Issue 3, 2015, pp. 211–252

[8]: Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[9]: Pekrun, 1992c; Pekrun et al., 2002; Spangler, Pekrun, Kramer, &

Hofmann, 2002

[10]: Pekrun, Goetz, Perry, Kramer, & Hochstadt, 2004;

Titz, 2001

# [11]: Selly Feranie, Nanang Winarno, Dandy Furqani, December 2018, The Effect of Predict-Observe-Explain (POE) Strategy on Students’ Conceptual Mastery and Critical Thinking in Learning Vibration and Wave

[12]: 4th International Seminar of Mathematics, Science and Computer Science Education 14 October 2017, Bandung, Indonesia

[13]: Bonwell, C. C., & Eison, J. A. (1991). Active learning: Creating excitement in the classroom. Washington, DC: The George Washington University.

[14]: Emotion Recognition in Speech using Cross-Modal Transfer in the Wild arXiv:1808.05561v1 [cs.CV] 16 Aug 2018