

Executive Summary

With the attention spans of youth falling as low as 10 minutes, there is a pellucid need for a learning assistant that keeps the students informed about their inattentive behavior during the class hours and the topics that they missed out while they were not paying the required attention. The lecturers will also be benefitted if informed about the topics during which most of the student from the classroom were distracted.

Thus, I propose a model that not only does the aforementioned tasks but also a bunch of extra ones. In this model, we place a camera that acts as an external observer in the cohort-based classrooms. This camera records the classroom activity of the students and stores it in its memory. The lecturer then collects this data and processes it in his laptop using our model. The model then returns the details of the attendees of the class, students that came late to the class, students that left the class early, topics during which most of the students were inattentive and the number of students that were sleeping in the class. This data is then sent to an online database which can be accessed through an android application.

| | Page No. |
|---|---------------------|
| Acknowledgement | i |
| Executive Summary | ii |
| Table of Contents | iii |
| List of Figures | v |
| List of Tables | vii |
| Abbreviations | viii |
| Symbols and Notations | ix |
| 1 INTRODUCTION | 1 |
| 1.1 Objective | 1 |
| 1.2 Motivation | 1 |
| 1.3 Background | 3 |
| 2 PROJECT DESCRIPTION AND GOALS | 7 |
| 3 TECHNICAL SPECIFICATION | 8 |
| 4 DESIGN APPROACH AND DETAILS | 15 |
| 4.1 Design Approach | 15 |
| 4.2 Materials & Methods | 17 |
| 4.3 Constraints, Alternatives and Tradeoffs | 25 |
| 5 SCHEDULE, TASKS AND MILESTONES | 26 |
| 6 PROJECT DEMONSTRATION | 28 |
| 7 RESULT & DISCUSSION | 35 |

| | | |
|---|-------------------|----|
| 8 | SUMMARY | 36 |
| 9 | REFERENCES | 38 |
| | APPENDIX A | 40 |

List of Figures

| S.no | Figure Title | Page no |
|-------------|---------------------------|----------------|
| 1. | Log loss function | 9 |
| 2. | CNN architecture | 11 |
| 3. | Weights update formula | 11 |
| 4. | Yolo bounding box | 12 |
| 5. | Facial landmarks | 14 |
| 6. | Model Architecture | 16 |
| 7. | Firebase storage format | 17 |
| 8. | Face detection on images | 18 |
| 9. | Yawning detection | 20 |
| 10. | Affine Transform | 21 |
| 11. | Eye detection | 22 |
| 12. | Side face detection | 23 |
| 13. | Methodology | 24 |
| 14. | Attendance array | 28 |
| 15. | Emotions array | 29 |
| 16. | Anomalies and side faces | 29 |
| 17. | Android Landing Page | 30 |
| 18. | Android registration Page | 31 |
| 19. | Android Login Page | 31 |
| 20. | Post login Page | 32 |
| 21. | LCT topics | 32 |

| | | |
|-----|----------------------------|----|
| 22. | Class Mood | 33 |
| 23. | Attendees | 33 |
| 24. | Late comers, Early Leavers | 34 |
| 25. | Anomaly Presence | 34 |
| 26. | Sleep and Awake | 35 |

List of Tables

| Table No. | Title | Page No. |
|------------------|------------------------------|-----------------|
| 27. | Literature Survey | 3 |
| 28. | Functionality and dataset | 24 |
| 29. | Functionality and Classifier | 25 |
| 30. | Work Schedule | 26 |
| 31. | Testing, training accuracy | 36 |
| 32. | Noisy testing | 36 |

List of Abbreviations

| | |
|--------|--|
| ANN | Artificial Neural Network |
| CNN | Convolutional Neural Network |
| MT-CNN | Multi-Task cascading Convolutional Neural Network |
| HOG | Histogram of Gradients |
| CK+ | Cohn Kanade |
| LBP | Local Binary Patterns |
| LFW | Labelled Faces in Wild |
| CEW | Closed Eyes in Wild |
| CV | Computer Vision |
| IBUG | Intelligent Behavior Analysis Group |
| YOLO | You Only Look Once |
| IOT | Internet of Things |
| SVM | Support Vector Machines |
| FC | Fully Connected Layer |
| MP | Maximum Pooling |
| AP | Average Pooling |
| FPS | Frames Per Second |
| DSLR | Digital Single Lens Reflex Camera |

Symbols and Notations

| | |
|----------------------------|------------------------------------|
| Σ | Summation |
| \leq | Lesser than or Equal to |
| ∞ | Infinity |
| α (in triplet loss) | Threshold |
| α or η | Learning Rate |
| φ | Activation Function |
| ∂ | Partial Derivative |
| A | Anchor image |
| P | Positive Image |
| N | Negative Image |
| $d(A, P)$ | Euclidean distance between A and P |
| $\max()$ | Maximum of the elements |
| OpenCV | Open Computer Vision |
| NumPY | Numerical Python |
| ΔW | Change in weight |

1. INTRODUCTION

1.1. OBJECTIVE

Through this project I aim on developing a scalable working prototype of an automated student fatigue mood detection system that will help the students and teachers in a cohort based classroom arrangement. Through this the professor can monitor and analyze the students' response to his or her teaching style and can observe the various reactions of the students at the varied time intervals of the lecture. This will also be assisting the students in identifying the topics they missed while they were not attentive in the classroom, successfully removing the possibility of them disrupting the learning schedule of other students just to clarify their doubts. This model can also be used as a replacement for many environments that require a human observer to view and gauge the reactions of his/her audience to the presentations he/she is giving. This model was built to accommodate even the systems that lack the processing power required for continuous video streaming and processing. This was achieved by extracting the minimum no of frames required for the processing of the video instead of processing the entire frame set of it. I have also used the image subtraction method to identify the unique frames that require processing thus saving the system from re running on the similar image frames. The system was designed in a such a way that it has flexibility in its usage scenarios as it can be also used in board room meetings, unmoderated focus groups, effectiveness testing of the advertisement campaigns and also in the driver attention testing post any mishap. To account for the user friendly aspect of the project, the front-end was designed as an android application, whose interface would provide an easy access and navigation path to the end user.

1.2. MOTIVATION

It is widely observed that students often don't concentrate enough in the classrooms and regret not doing it during their exams. According to a study [6] the attention spans of healthy teenagers and young adults range between a maximum of 10 to 15 minutes in a classroom scenario. This however is known to be decreasing over the last few years due to multiple factors such as the distraction caused by the internet era.

This kind of behavior however fares bad on the student grades and performance report. Through this project I was to help reduce the damage done on the student performance by proposing a system that monitors their attention spans and behaviors in classrooms.

On the other hand, in a classroom environment, it will be really hard on the teachers to simultaneously monitor and teach a classroom of students. This model cuts down the job of monitoring and gauging for their reactions to see if they have understood the concept. Since I intended to apply the video processing techniques to a cohort classroom based scenario the next big obstacle to tackle would be the requirement of high performance GPUs and graphic drivers. To achieve the equivalent performance of the system on a pure CPU based device that lacks a graphics processor, I chose to process the frames extracted from the video instead of the video in its entirety. This can be done with little to no change in the obtained results as not much movement is generally observed in all the 30 frames extracted from a second of the captured video.

The authors of the Faces of engagement [7] paper had analyzed 44 studies and came to a conclusion that there are three different types of engagement, namely behavioral, emotional, and cognitive. Behavioral engagement describes a student's readiness in taking part in the learning process. This can be measured by seeing how regular he is when it comes to attending classes and completing the designated works or tasks on time. Emotional engagement deals with the student's emotional outlook on education in general. This can be quantified by measuring how attentive or positive he is while performing the allocated tasks. If a student is performing the tasks allocate on time but shows dislike or distaste while doing it has a good behavioral engagement but has a bad emotional engagement. The last classification of engagement is termed as Cognitive Engagement, this denotes the measure of improved attention, creative thinking and a good memorizing capabilities shown in a class. Through this project I am quantifying the cognitive and emotional engagement of the students of a classroom by identifying the learning associated fatigue and their emotional state portrayed through their faces.

1.3. BACKGROUND

Some attempts have been made for the capturing of electronic attendance in classrooms through computer vision, and some attempts were made towards the real-time fatigue detection in driving scenarios. Multiple different methods were devised to estimate the facial emotions shown by a person. In this project of mine I have cumulated, cascaded and modified some of the processes that were used to emulate the required functionality of the student behavior detection and analysis in a classroom scenario. To also add a new flare to the system I have made it more flexible to the users of devices that lack a graphic processing units.

Table 1: Table displaying literature survey

| REFERENCE | METHODS | COMPONENTS | CONCLUSION |
|-----------|--|--|---|
| [1] | IOT based Multi- modular motion | Accelerometer, Gyroscope, Camera | The authors used 3 hardware modules in tandem. It had a head motion module that consisted of an accelerometer and a gyroscope, a pen module that has an accelerometer and the visual focus module that has a camera. The motion of the head and the pen would be detected by their respective modules and the vision focus module placed on the student's head was used to detect if the QR code on the classroom presentation was visible. This was done to determine if the students were looking at the board. The major downside of this method is that the students have to wear these modules for it to work correctly. |
| [2] | You Only | CUDA, | In this paper the authors used Liris Human |

| | | | |
|-----|-------------------------------------|--|---|
| | Look Once YOLO R-CNN | Keras CNN library, GeForce GTX 1050 Ti | Activity dataset and trained their You Only Look Once object detection classifier on the various activities present in the dataset. They then compared the speed of the response and the results they have achieved with the another classifier called R-CNN. The YOLO seemed to work much faster when it came to detecting the region of interest from the image. Their architecture had 24 convolution layers and they used the CUDA to optimize and run their object detection on GPU. The downside to this would be that it can only be run on systems that possess nvidia GPU's as CUDA only works with nvidia. |
| [3] | Google Vision API For mobiles | Tensor Flow Lite backend, Front-camera of mobiles | The authors of this paper discussed various possible methods through which the attention state of the drivers can be detected. They elucidated on the usage of EEGs for the detection of elevated heart rates to detect the drop in attention. Their final product was made into an android application which made use of Google's Vision API which used TensorFlow Lite to detect the closed state of eyes of the driver in real time. This required a high bandwidth internet connection to relay the video being taken from the phone to the online server. They also proposed a scoring and measuring metric called the perclos which is the percentage of the time for which the eyes of the driver were closed. |

| | | | |
|-----|--|---|---|
| [4] | Multi Task Cascading Convolutional Neural Network | MT-CNN Python library, LFIW dataset | The authors of this paper first enlisted all the problems of the current attendance system and how it can easily fall prey to the proxy system devised by the students. They then compared and contrasted the accuracy of the current face detectors like vanilla CNN vs the MTCNN. They implemented an MTCNN based system to identify and recognize the faces in a classroom. Their model boasted of 98.7% accuracy on face recognition. However, they haven't accounted for the students coming and leaving the classrooms in between the lectures and the students coming to the classrooms and sleeping midway in. |
| [5] | Emotion Classifier Comparison of CNN vs facial landmarks vs Haar Cascades | Computer Vision library, Cohn Kanade Emotion labelled dataset, SVM library | This was a comparison paper that made a distinction between all the classifiers that were in usage for emotion prediction. Ranging from decision tree classifiers that collected their data from the hardware devices like the ECGs and EEGs to the CNN based fisher face which used the CK+ labeled emotion dataset. The hardware devices were removed from consideration for this project as it is not practical for the students to be moving around wearing them. In the software and camera based detectors, the facial landmark extraction and classification proved to be more accurate when compared to the CNN based method. |

| | | | |
|--|--|--|--|
| | | | <p>However, the CNN based method outperformed this when comparing the time taken for the prediction.</p> |
|--|--|--|--|

2. PROJECT DESCRIPTION AND GOALS

In the model proposed in this project a wide lensed camera will be replacing a human observer in a cohort based classroom of small strength. The camera reprises the emotion and feedback gauging role of a lecturer. This will facilitate the lecturer to completely concentrate on the lecture without having to turn around and see the responses of the students. Obtaining the video stream of the classroom will not only help the lecturer but will also help the students in identifying the time during which they lacked attention in the classroom. The process in its entirety will be automated and doesn't need any involvement from the external factors.

The video obtained from the classroom is sent for the processing, in which we first extract the frames from video. The number of frames that are to be extracted per a second of the video will be decision that can be made by the faculty. The greater the number of frames extracted the greater will be the processing time. The extracted frames are then subtracted from each other to only select the frames that are unique for processing. Doing this helps in reducing the wastage of processing time and power. The unique frames are then sent to the dlib face detector which uses CNN based method called MT-CNN and Histogram of Gradients to identify the faces in the image and return the co-ordinates of the bounding box of the faces in the image. Each face detected by this detector is cut out from the image frame and sent to recognize the person to whom the face belongs to, after identifying the person, then this face image is sent to classifiers to detect the presence of yawning, emotions, or the presence of side-way turned face. Then finally the face image is sent to an eye state detector that will detect if the eyes of the person in the frame are closed or open to identify sleeping patterns.

The main goals of this system are to provide the lecturer with the details of the topics during which most of the students are inattentive or asleep, the moods of the students in the classroom, names of the students that attended the class, the students that have come in late or have left early from the classrooms, sleep to awake ratio in the classrooms.

This design can also find out if any new guy that doesn't actually belong to this class has entered the class lecture and can also help the teachers identify the names of the students from their images. It can help the students identify the topics during which they were inattentive in the classroom and also provide them with their attendance details and the details of their mood during the classroom hours.

3. TECHNICAL SPECIFICATIONS

3.1. DIGITAL VIDEO AND IMAGE STORAGE FORMATS

Videos are nothing but combination of images that are depicting some sort of motion. Frame rate is the number of images that are captured in a second to cause the illusion of motion. General motion picture standard has a frame rate of 24 frames per second (fps), videos recorded in cellular phones can capture from 29 fps to 50 fps. Conventional DSLR cameras can capture from 30 to 60 fps, and the slow motion cameras can capture up to 120 fps. The images are storage as intensity matrices in digital format. If the image has only binary colors, namely, black and white they're called binary images, they are stored as matrices that have 1's and 0's in them. The 1's represent white and the 0's represent black. The images that have all the shades of gray are called the gray scale images, they are stored as a matrix that has its values between 0 and 255. 0's represents the darkest hue and the 255 represents the lightest hue. The gray scale images can be converted to binary by the usage of thresholding techniques. Both the gray scale and the binary images are called single channel images as they can be represented by a single image matrix, but the color images cannot be represented by a single matrix. The color images are generally represented by a combination of the intensities of the red, green and the blue channel matrices. Each of these channels have intensity values ranging between 0 to 255. Resolution is the size of this matrix, the image quality is said to be good if it has good resolution i.e. more intensity values representing a smaller physical area.

3.2. ARTIFICIAL AND CONVOLUTIONAL NEURAL NETWORKS

Artificial Neural Networks or the ANNs are the Machine Learning constructs of Supervised Learning. Supervised Learning is a subset Machine Learning that deals with the learning in the presence of a pre-labelled training dataset. The basic versions or the vanilla versions as they are called are generally made of an input layer, a few hidden layers and an output layer. The input layer has input neurons that take the input from the user, there will be connections from one layer to the next layer called the edges. These edges bear values called the weights. When transferring the data from one layer to the next layer, the weights are multiplied with the data and transferred. Once it reaches each node it is sent into a function called activation function, it is these functions that make the other linear neural networks model as non-linear. The activation functions are also used to squash the output values into small ranges like -1 to +1 or in the ranges of 0 to 1. Once the data comes out of the output nodes while training, a difference is calculated between the obtained data the actual data of the label or class. This is called the loss of the neural network. The loss is calculated using loss functions that range from simple Euclidian to more complicated binary cross-entropy or the log-loss functions. The figure 1. given below represents the log loss cross entropy function.

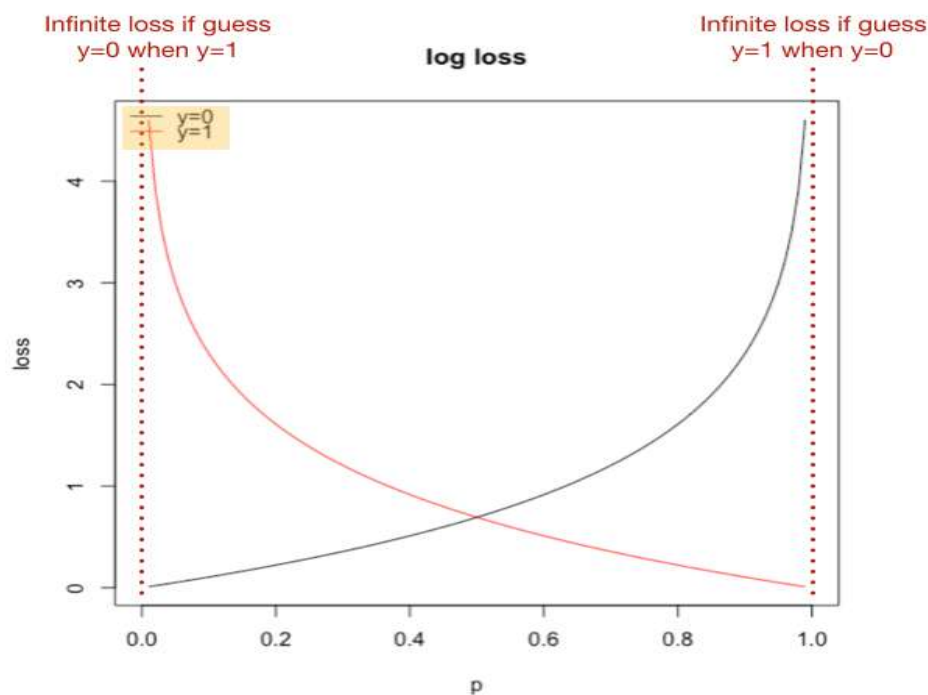


Figure1: Log loss cross entropy

This loss function is generally used while calculating the loss for binary classification problems. Once the loss of the prediction is obtained the back-propagation technique is used to calculate the amount by which the weights have to be changed, then this obtained value is multiplied by a constant called the learning rate which dictates the amount by which the weight values can change. The Convolutional Neural Networks or the ConvNets are a modification of the above mentioned Artificial Neural Network. They are generally employed when we are trying to predict any underlying patterns present in the data. This pattern recognizing feature makes them most useful for image based detections and recognitions.

The CNNs when processing the image data take the image in the form of a matrix, if we are processing on a binary or grayscale image we take a single channel matrix, and if we are processing a colored image we take 3 channel matrix. Then a series of image pattern recognition filters are applied on the taken image, this process is called convolution and hence the name Convolutional Neural Networks. The feature matrix obtained from this convolution operation is sent to the next layer called pooling layer. The pooling layer is generally used to reduce the size of the feature but still retain the important features. There are different types of pooling namely, max pooling, min pooling and average pooling. In max pooling, a matrix of chosen size is applied to feature matrix and the maximum element present in that pool size is taken and the rest are discarded. Similarly, in min pooling the minimum element is taken from the pooling matrix, and in the average pooling the average of all the elements in the matrix is taken in to consideration. The stride of a pooling filter is the number of steps by which the pooling filter moves across in an image while performing the pooling operation. There will be a series of convolution and pooling layers applied on the image, the higher their number the better the features extracted from the image. All the elements of the final feature matrix are copied into a single one dimensional array and sent to a fully connected classifier neural network to predict the class of the object in the image. The figure 2, shown below represents the architecture of a convolutional neural networks.

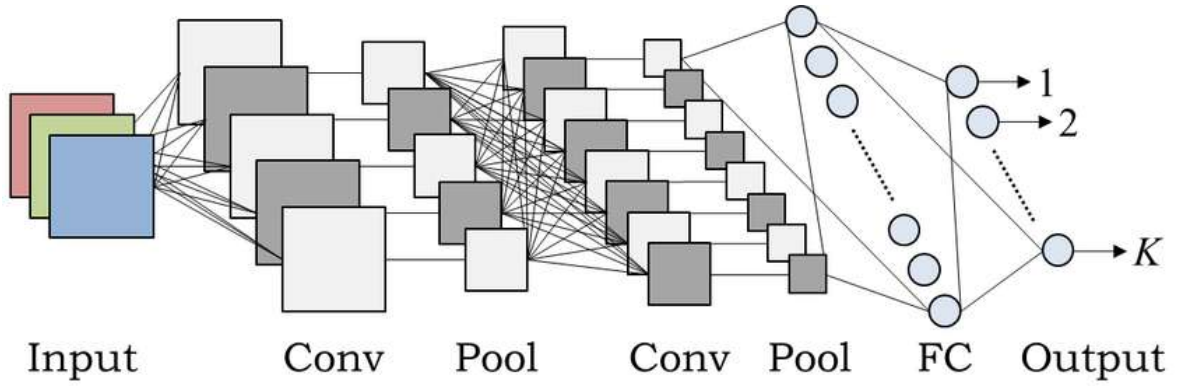


Figure 2: The architecture diagram of CNN

$$\Delta w_{ij} = \left(\eta * \frac{\partial E}{\partial w_{ij}} \right)$$

Figure 3: Weight updating formula of back-propagation

3.3. YOLO, MT-CNN AND HOG

You Only Look Once (YOLO) and the Multi-task cascading Convolutional Neural Network (MT-CNN) are variants of the CNN model that are specifically used for object detection and the prediction of a bounding box around the detected object. However good the CNNs were they were only able to tell the presence of an object in the image but were not able to pin-point on the location of the object detected, this gave rise to the YOLO and MT-CNN models. The object detection combined with the bounding box identification is called the object localization problem. Initially after taking the image the YOLO algorithm divides it into 19 by 19 grids, then with each grid obtained taken as a center, 5 random bounding boxes are created. The list of these co-ordinates that represent the bounding box is stored in a vector. Then our trained CNN model is slid across all the randomly generated bounding boxes to predict the probability (p) of the presence of an object in the box, bx, by (x and y co-ordinates of the centers of the bounding boxes), h, w (height and width of the box) and the class of the predicted object. Then a threshold of the probability is set and all the boxes that had

their prediction probability lesser than the set threshold are eliminated leaving us with a handful of boxes. Out of these boxes the ones that are overlapping are taken and the IoU values of each of these bounding boxes is calculated. The IoU (Intersection over Union) is a measure to identify the more accurate bounding box of all the predicted ones. The box with the greater IoU is taken and outputted as result. Then the obtained result vector containing the prediction probability and the bounding box co-ordinates is compared with the actual values from the training set and the loss is calculated. This loss value is then used by the back-propagation algorithm to train the neural network for modifying the weight values. MT-CNN is the latest modification on YOLO algorithm it functions similar to the YOLO till the creation of the bounding boxes, but the classifiers that are predicting the presence of the object in the bounding boxes are trained on the various scales of the same image, thus making it easier for this method to predict smaller objects in the bigger pictures and bigger objects in the smaller pictures.

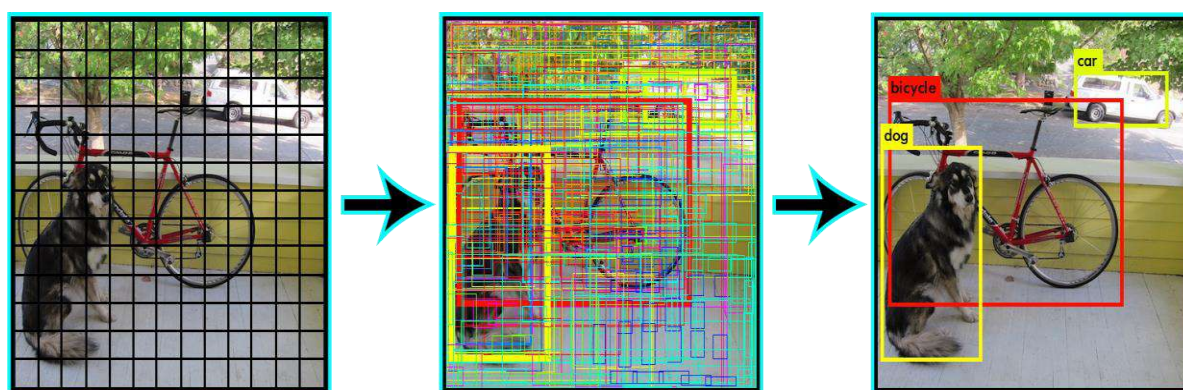


Figure 4: Shows the initial and final predictions of the bounding boxes

Histogram of Gradients (HOG) is a method used to reduce the number of features that are being sent to the classifiers. Generally, for standalone CNN based classifiers the entire pixel values of the image are sent for convolution, this however takes up a lot of time, as generally images are of huge sizes. HOG computes the horizontal and vertical gradients on the image data and uses pooling on the calculated values. Gradients have the highest values on the edges of the images as there is a sharp intensity jump of pixel around the edges. This makes the gradients good edge detectors. The horizontal gradients are calculated by applying $[-1 \ 0 \ 1]$ filter and vertical gradients are calculated by applying $[-1 \ 0 \ 1]^T$ on the pixel values.

3.4. FACE EMBEDDINGS, TRIPLET LOSS, ONE SHOT LEARNING

A Convolutional Neural Network that has over 24 hidden cascading layers of convolution and pooling is used in the modern day to identify the person from the face image. The Siamese twin network is one of such networks, these kinds of networks are trained on huge datasets of over 13,000 images called Labelled Faces in the Wild (LFIW). In these types of neural networks, the last fully connected neural network classifier layer from the CNN models are removed and they instead add multiple layers of hidden neurons of which have 129 output neurons. Out of these 129 output neurons the first output neuron produces the probability that the provided image is a face and the rest of the 128 neurons produce values called the facial embedding. These face embeddings will be used to uniquely represent a face. For training the neural network to predict similar face embedding for the same faces and different embeddings for different faces a new loss function called the triplet loss is used. The triplet loss works as follows, a random face image is taken (called the anchor image), then another image of the person from the anchor image is taken (called the positive image) and one more random image of a person that is not in the anchor is taken (called negative image). The face embeddings of each of these faces are calculated and stored in a 128 element array or vector. Then the Euclidian distance between the anchor and the positive is calculated (represented as $d(A, P)$), then the Euclidian distance between the anchor and negative is calculated (represented $d(A, N)$). These distances have to follow an inequality relation to be distinguishable from each other so that the distance between the faces of the same person is minimized and the one between the anchor and the different person is maximized. The in-equality is given by $d(A, P) + \alpha \leq d(A, N)$.

The aforementioned relation means that the distance between the positive and the anchor image should be at-least alpha less than the distance between the negative and the anchor. Bringing the right hand-side equation to the left it becomes $d(A, P) - d(A, N) + \alpha \leq 0$. Thus the final loss calculation function is given by:

$$\text{Loss} = \max ((d(A, P) - d(A, N) + \alpha), 0)$$

Once the corresponding loss is calculated for the triplet back propagation algorithm is used to reduce this obtained loss and train the neural network. This is done multiple

times and once the network has reached good accuracy the weights are saved into a files. The same network architecture is created again this time the weights from the file are loaded instead of initializing them randomly. The input image is sent for getting the embeddings and the obtained embeddings are compared to the previously saved embeddings of the people whose faces have to recognized. This process is called one shot learning.

3.5. FACIAL LANDMAKRS DETECTION

The facial landmark detectors are the Convolutional Neural Networks trained on ibug (intelligent behavior understanding group) annotated face image dataset. This dataset consists of images that are annotated with 68 facial landmarks that depict various features of a face like nose, eyes, mouth etc. This neural network has 136 output nodes each of which predict the x or y co-ordinate's position of the one of the face landmarks. After initial random initialization of weights, one forward pass is done for the image and the predicted positions of the landmarks are compared with the actual position in the image and the loss is calculated through the Euclidean distance and this loss is minimized through back propagation.

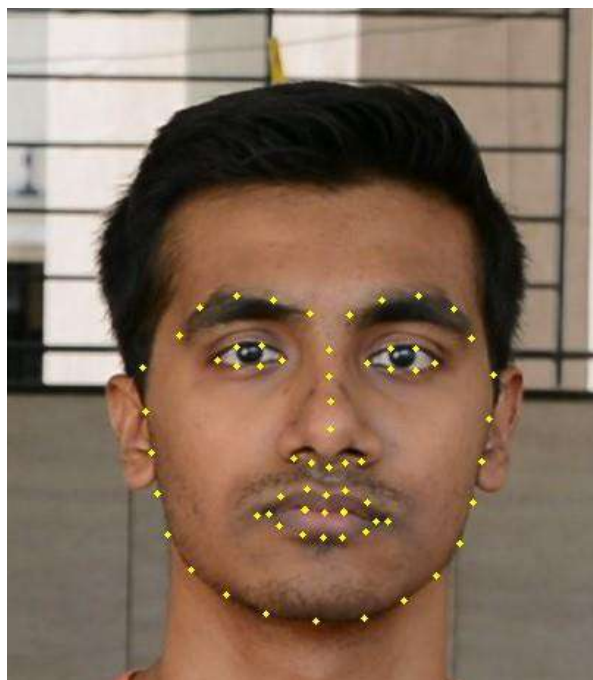


Figure 5: Figure depicting the 68-point facial landmark detection on face

4. DESIGN APPROACH AND DETAILS

4.1. DESIGN APPROACH

While designing this model for carrying out the designated functionality I have cascaded several classifiers in a specific order for optimal performance. Not only the classifiers but also the processes are ordered as follows for ensuring the smooth flow of processing. First I extract the frames from the video, one frame per second (as human movements generally do not change in fraction of seconds) and save them in a folder. Then I use the address of this folder to acquire all the extracted frames and run frame subtraction program for identifying the unique frames from the lot. Then I send the index number of the starting frame from which they are unique to a face detection process. The face detection algorithm used, finds all the faces in the image and returns their corresponding boundary boxes as a list. I then made a loop over all the boxes and extracted the faces present in those boxes and as each face is extracted I sent it for face recognition and yawning detection. Once the faces are recognized the attendance is provided for them for that second and if yawning is detected the emotion of the face for that second is labelled as yawn and the frame checking with emotion detection classifier and eye state detector are skipped (this is done to optimize the speed of the model and since we have detected that the person was yawning in that frame we can safely say that no emotion would be shown on his face in that frame and the eyes would be generally closed during this act). If yawning is not detected in the frame then the images are sent to emotion detection, if any of the emotions like anger, happy, disgust shocked are found we add them to the list, or else the classification will be as neutral. If the face is detected to be neutral, then it is sent for eye state detection classifier which detects if they eye during the frame was closed or open. This in effect finds out if the person was sleeping or attentive during the class. For selective frames for which the front faces were not detected will be sent to a side face detector for identifying if the person was present in the class but was looking away from the board. Through all the aforementioned steps we can identify the students that came late to the class or left early, students that slept in the class (close eyes or head bent on the bench), the emotions they portrayed during the lecture hours, attendance of the students and also a presence of any outsider in the classroom.

The attendance data we obtain after processing each frame is stored in a list in the form of binary numbers. Zero represents absence and 1 represents presence. In a similar way the emotions identified in each frame are stored in a list of integers with indices from 0 to 5. Once the process is over these lists or arrays are parsed to obtain the required responses. For the attendance criteria the students will be given a grace period of 2 seconds for every 3 second interval, this is to overcome any discrepancies like bending head down for picking up a pen or closing the eye lids while natural reflexes like blinking. Once the processed array is obtained, the students present in the 80% of the class timing are provided with the attendance for the classroom. This is done to accommodate any misrecognitions or wrongful classifications. For the ease of usage, the user interface of this product has been made into an android application. The android works as a front-end authenticator for students and teachers. The entire data processing will be done on the teacher's laptop or can be outsourced to a server. Once the data processing is done the results will be uploaded to a free online real-time database called firebase. Firebase is an online free database service company backed up by google. They also provide cloud computing servers, with iOS, android and laptop integration. Firebase uses NoSQL database format to store the data it receives. We add an event listener to our firebase instance online to make it work in real-time. The above mentioned NoSQL database stores the incoming data in a JSON parse tree format with root nodes and key value pairs.

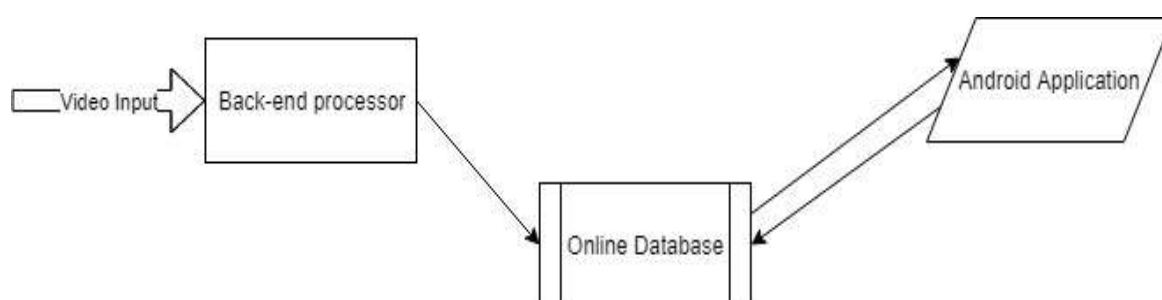


Figure 6: The architecture of the online database.

As shown in the diagram the video input is given to the back-end processor which is generally the laptop of the teaching staff and the processed results are sent to the online database called firebase. Then for the accessing of the data the teachers and students use their android application accounts.

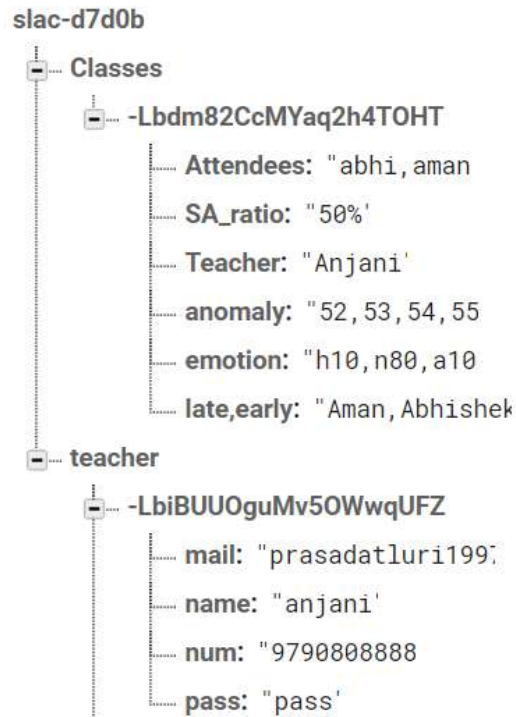


Figure 7: image depicting the data storage structure in firebase

4.2. MATERIALS AND METHODS

For the designing of the required system I have made use of multiple classifiers in cascade, each of which were trained on a few datasets. For the models that required higher amount of training from huge datasets I have made use of pretrained models and for the other I have used CNN models that were made and trained by me. Haar cascade based classifiers were used in moderation whenever necessary to keep the processing time low.

4.2.1. FRAME EXTRACTION AND IMAGE SUBTRACTION

For facilitating the reading of the video file, I am using a python library called openCV (open computer vision). The opencv library has a function called “VideoCapture” that collects the video data and converts it into an object, which has a function called get that allows us to get the details like the number of frames in the video and the frame rate of the video. By dividing the number of frames in the video with the frame rate we obtain the duration of the video in seconds.

We can then set the object to return the instance of a particular frame number by using a function called set, we then use this reference of that frame to call a read function on this frame number to get the image data from the frame. We then take a counter and initialize it to 0 to point at the 0th frame and call a loop and keep reading each frames one by one with the increment of the loop as the frame rate. This gives us the required images with a one frame per second frequency. The extracted frames are stored in a folder, on which we will be applying image subtraction to identify the frames that have unique content in them and remove the repetitive frames. To achieve this, we read the consecutive images from the list of images in the folder and we do a matrix subtraction on all the 3 (red, green and blue) channels of the color images. Once the subtraction is done we count the number of non-zero elements in each of these three matrix channels. If the count is equal to zero, then it means that both the images are same and have the same pixel content. Doing this helps us weed out the redundant frames that don't need to be processed on saving us some processing time.

4.2.2. FACE DETECTION

The finalized image frames are then sent to a dlib based face recognizer. Dlib is a cross-platform library written in C++ that has wrapper in python. It was written in C++ to ensure the speed of processing. For face detecting dlib has a front face detector class that uses the MT-CNN and HOG methods to identify faces from images. It was trained on 7198 face images from the dlib 5 point annotated face dataset. Once all the faces in the image are detected it gives out the list of the bounding boxes of the faces in the image, we then iterate over each of the box and send the acquired data to the subsequent classifier.

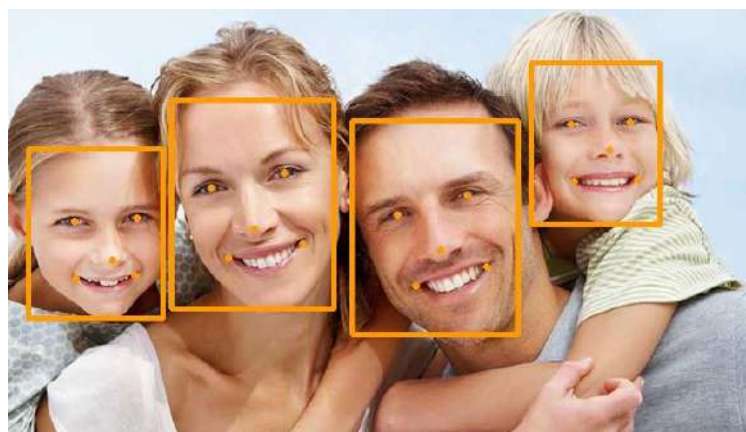


Figure 8: Detected faces from a testing image

After an initial usage of Haar and LBP cascades for face detection from images, I have shifted to using YOLO based detection using a library called faced. Through Haar and LBP cascades made their detections in a small amount of time they were not very accurate and were unable to detect faces present in the image at an angle. The YOLO based detector was slow but produced a more accurate prediction, but the problem of angled face detection persisted and that was when I shifted to the dlib based detector. The dlib based detector was faster and more accurate when compared to YOLO based one. As the image dataset was huge (over 7,000 face images) it was hard to download and train them on a normal laptop device, that's why I have used pretrained models for face detection and recognition.

4.2.3. FACE RECOGNITION AND YAWNING DETECTION

The cropped face images from the previous classifier are then sent to the deep networks modeled on the version 1 of the nn4 small architecture. This contains 14 hidden convolution layers followed by the pooling layers, which are followed by 7 inception layers. Inception layers are the ones that do convolution on the images but with image filters of different sizes, this helps the neural network to also learn the ideal filter size to apply on the images to extract the most useful features. This network is provided to us by the open face library of python, this detector was trained on a huge data of 13,000 images from the dataset Labelled Faces in Wild (LFW). This dataset contained over 13,000 images of popular faces with over 5,000 classes for prediction. Due to the sheer amount of the training data required I have instead used a method called one shot learning. In one shot learning we send our images through a pretrained model and get their corresponding embedding vectors that contain 128 elements. We then store these 128 facial embeddings of each person we want to recognize and then compare them with the embedding obtained from the current image. The facial embeddings that have the least Euclidean distance from the current ones is taken to be the class of the detected face. To obtain the correct embeddings for each face the network was trained with a triplet loss function as explained earlier in the technical specifications.

From the obtained face the dlib detector receives the facial landmarks and compares them with the landmarks of an aligned model face, then the landmarks of the face detected by us are translated and rotated into the model face using affine transformation. This will make the slanted faces straight and bring in all the facial features to their common places. Doing this helps in obtaining similar embeddings to the same faces that are turned in different angles. For detecting yawning from the images we collect and store a separate set of embeddings for the images of the detected person yawning and compare our currently obtained ones with them. During the initial stages of the project I have designed a CNN based facial recognition of my own on a self-curated dataset. The model performed really well on the training and testing data but failed to perform during the real time execution and had to be sacked. The reason for the failure of this initial model can be attributed to the lack of quality and quantity in the dataset designed.

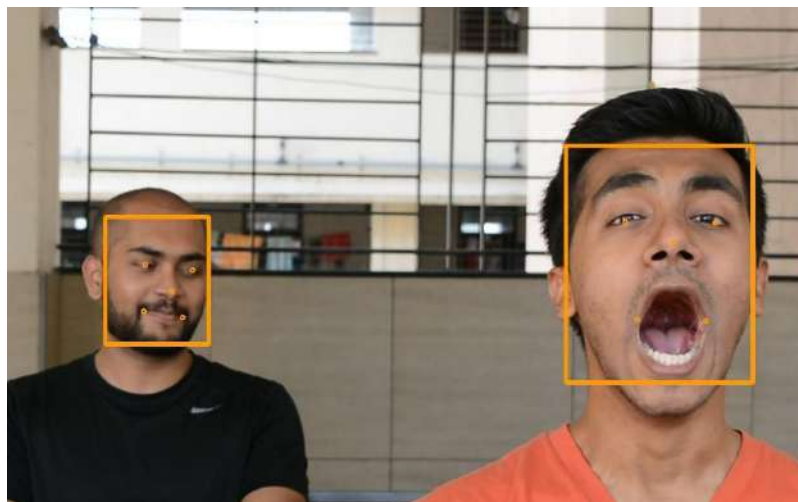


Figure 9: Facial detection of a yawning and a normal person

The above figure depicts the faces of the students present in it getting detected. Post which they will be sent for face recognition. During this recognition process the first face detected (the face of the person smiling at the back) will be sent to the recognizer and its corresponding facial embeddings will be obtained. These obtained embeddings will be compared with the pre-stored ones and the label of the closest embeddings will be given to the face.

With the second face the obtained embeddings will be the closest to the yawning version of the person's face than his natural face thus the recognizer classifies it as yawning image. The corresponding array positions of the frame will be filled with 1 if the person is present or 0 if he is absent.

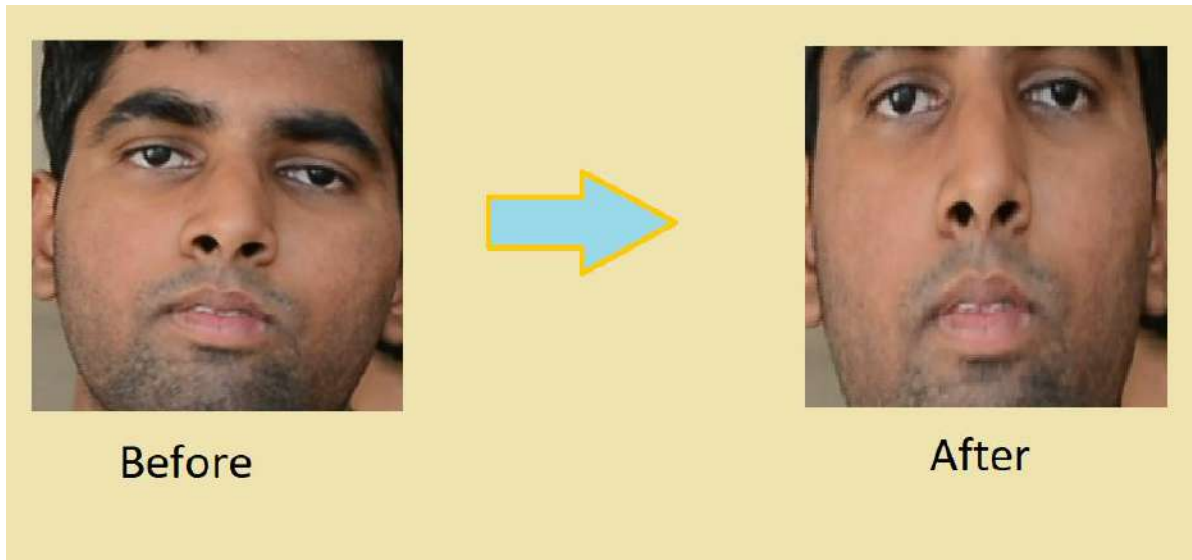


Figure 10: Figure depicts the faces before and after the affine transformation

4.2.4. EMOTION AND EYE STATE DETECTION

Once the faces are recognized and yawning detection is done the image is forwarded to the emotion detector. The emotion detector is used only when the yawning is detected to be negative, because if the yawning is detected in a frame then there will be no emotion that is being portrayed by the person in the frame, thus skipping it saves us time and wastage of processing power. A fisher face classifier is used for predicting the emotions displayed on a face. This classifier was trained on a CK+ (Cohn Kanade) emotion dataset that consisted of images for 8 emotions namely, neutral, angry, disgust, contempt, fear, happy, sadness and surprise. The number of images in these sets was very varied due to poor dataset curation standards. Some emotions like neutral have over 500 images and some like contempt only have 18. This variance actually caused for the poor initial training and testing accuracies of the model.

Then the emotions that had lesser number of images were removed and the model is trained again with fewer emotions namely, anger, disgust, happy, neutral, surprise. The model comparatively showed better performance of 84.94% of training and 85.55% of testing accuracy. The images before sending to the classifiers were resized to 350 by 350 pixels and converted to grayscale. For eye state detection I first had to extract the eyes positions and images from the frames. This was achieved with the help of multiple Haar cascades of eyes, namely, left eye, right eye, eyes through glasses etc.

A Convolutional Neural Network model of 4 hidden convolutional layers with filter size of 3x3 and pooling size of 2x2. It was trained on the closed eyes in the wild (CEIW) dataset that had 2,530 images of opened eyes and 2,384 images of closed eyes. The activation function used for the intermediate neurons was Relu and for the final neuron was sigmoid. Each of these images is of the size 24 by 24 pixels, and the training was done with back-propagation and Adam's optimizer. The loss function used was binary cross entropy like every common binary classification problem. The images were converted to grayscale to reduce the amount of data that needs to be processed. The model was trained for 50 epochs with a batch size of 30. The data from the dataset was divided in 8:2 ratios for training and testing. The model finally achieved 98% training accuracy and 96% testing accuracy with a loss of 0.36. This model on the contrary to the emotion detection performed very well on the real-time data by detecting 0 false positives. The eyes detected with the help of Haar cascades are cropped and sent to the CNN classifier for detecting their state, i.e. if they are open or closed.

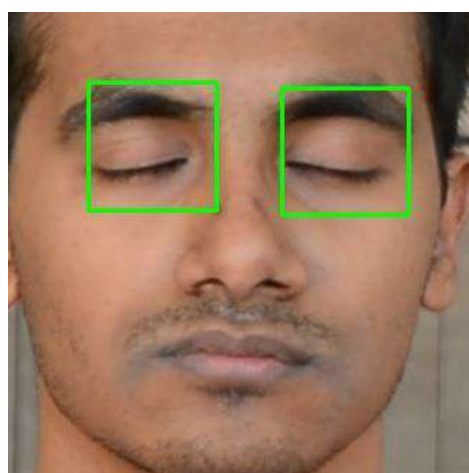


Figure 11: eye detection from face

4.2.5. SIDE FACE DETECTION AND ANDROID INTEGRATION

This model is also capable of detecting the side-way turned faced from the images of the faces extracted from the classroom video. To achieve this functionality, I have made use of profile face Haar cascade and LBP cascades. The converted grayscale image is given to the cascade classifier and the classifier returns the bounding box positions of the side faces detected. This classification will generally be checked on the frames that recorded absence of faces, this is done to ensure that even if a person is turned to his side is found and given attendance but is later informed through the login that he was turned to a side. The side face detection is treated as a sign of distraction in the classroom. The detected side faces are then sent to the face recognition embeddings generator that generates the facial embeddings for the side faces, which are then matched with the embeddings of the previously obtained side faces and the closest ones will be assigned with the class or the identity of the person. The front-end for this application was made in android for ensuring easy usability by the users. The layouts in android were made using XML markup language and the functionalities are coded in Java. Two different classes are made for each student to write their member functions and store their custom member variables like name, phone number, email etc.

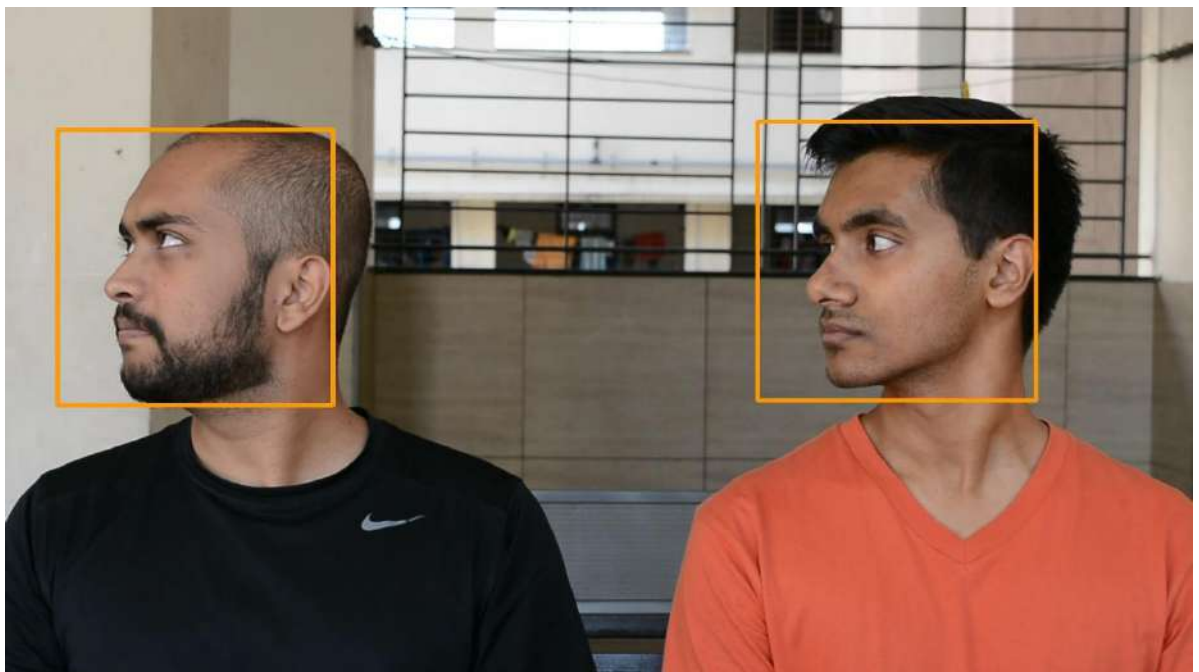


Figure 12: Figure displaying side faces in an image.

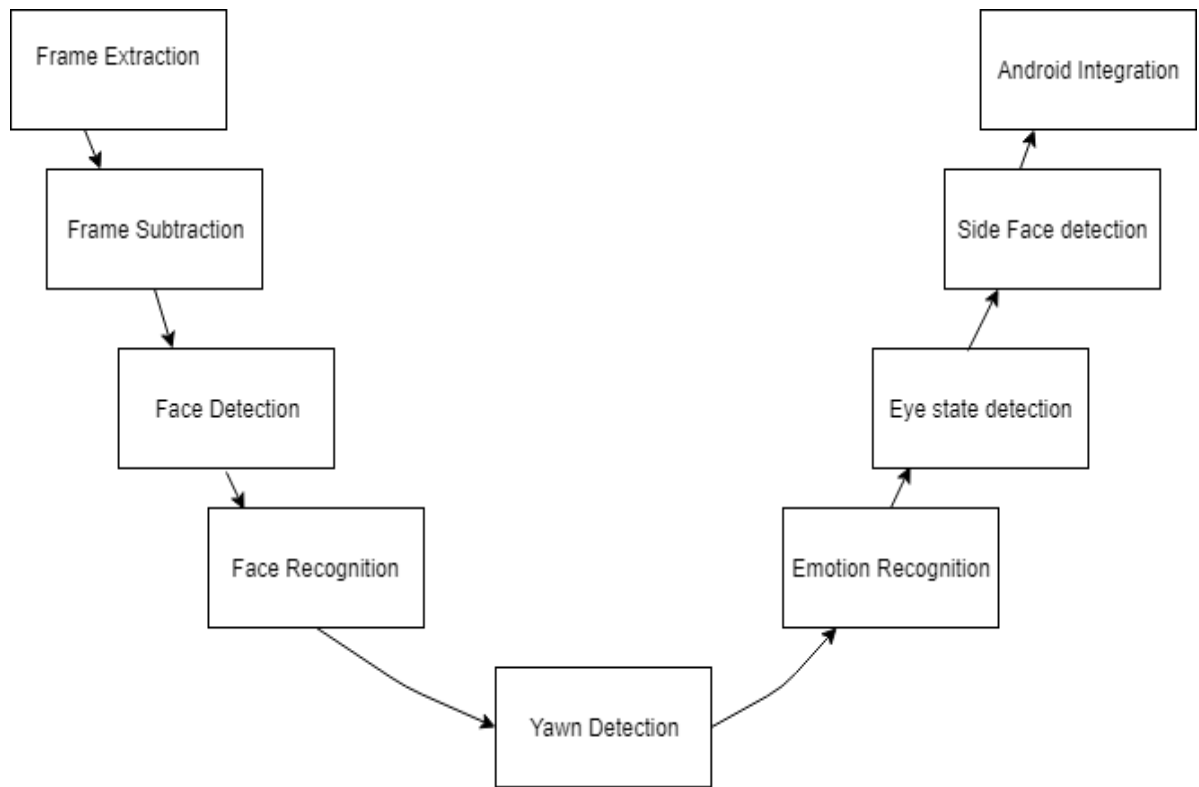


Figure 13: Diagram representing the steps in the methodology

Table 2: functionality and dataset table

| S.no | Functionality | Dataset / XML Cascade |
|------|---------------------|---|
| 1. | Face Detection | Dlib 5-point face dataset (7198 images) |
| 2. | Face Recognition | LFIW (Labelled Faces in Wild) (13,000 images) |
| 3. | Emotion Detection | CK+ (Cohn Kanade) (919 images) |
| 4. | Eye detection | Left-eye Haar Cascade |
| 5. | Eye state detection | CEW (Closed Eyes in wild) (4910 images) |
| 6. | Side Face detection | Profile face Haar Cascade |

For the first and the second models pre-trained classifiers were used due to the lack of extensive processing power.

Table 3: functionality and classifiers

| S.no | Functionality | Classifier/ Model |
|------|---------------------|--|
| 1. | Face Detection | Dlib's front face detector (uses MT-CNN, HOG) |
| 2. | Face Recognition | OpenFace nn4.v1 small model (uses face embeddings) |
| 3. | Emotion Detection | Fisherface recognizer |
| 4. | Eye Extraction | Haar cascade, facial landmarks |
| 5. | Eye State Detection | CNN |
| 6. | Side Face Detection | Haar and LBP cascades |

4.3. CONSTRAINTS, ALTERNATIVES AND TRADEOFFS

The extra advantage of being able to run this code on a pure CPU based system comes with a trade-off in speed of the processing. Currently the system charting at 9 minutes 34 seconds, to process a 181 second video, with two faces and all the classifiers intact. The second major constraint would be not being able to extract the necessary eye region data from a person that is very far away from the camera. This can be solely attributed to the technical difficulties of the system. In certain DSLR cameras the capture of the video while the subject in it is in motion is causing a motion blur in the video, which is at 2 instances of the 181 frames caused wrongful, face recognition. This however, wasn't the case when a mobile based video recording was used. The lack of appreciable accuracy of the emotion detection classifier can be attributed to the lack big enough dataset of faces showing emotions with almost equal number of images for each category. This has caused the classifier to skew a little towards the neutral classification (as there are more number of neutral faced images than the others) of most of the images. As an alternative for this emotion detection classifier I could have used the facial landmark based method which requires a much smaller dataset and gets a better accuracy over this, but the major issues with that method is that it might overfit easily and it takes a lot of more time when compared to the fisher-face classifier.

The same goes for the eye region extraction, even though I could have used facial landmarks based method I chose to go with “Haar cascades” as they are faster than the facial landmarks based method.

5. SCHEDULE, TASKS AND MILESTONES

The major tasks that I have performed during the span of my entire project are presented below in the form of a table.

Table 4: Work Schedule table

| S.no | Month | Schedule |
|------|---|---|
| 1. | December 2 nd to December 23 rd | Problem formulation, Background study, getting introduced to Machine Learning, Convolutional Neural Networks and MT-CNNs and YOLO. Identifying the goals of the project, performing a literature survey on the existing models. |
| 2. | January 3 rd to January 25 th | Learning to use Opencv2, Consideration of various face detectors, Making “Haar cascade face detector”. Learning basic android and Flask, developing android landing, login and register pages and linking them to the Flask localhost running in Python. Identifying the test cases of the project. |
| 3. | January 26 th to February 26 th | Designing Frame extraction and Frame subtraction from cv2 in python. Using YOLO and MT-CNN based face detectors. Making a self-curated dataset for facial recognition, and making a CNN based face recognition model. Changing the face detector to dlib face detector, changing the face recognizer to OpenFace base nn4 small face recognizer. Making post-login activities for the android application. Migrating from localhost to Google Firebase. Developing eye rotation technique for face alignment in images. Gathering datasets for eye state detection and emotion detection. |
| 4. | February 27 th | Building a CNN model for eye state detection, training and |

| | | |
|-----------|---|---|
| | to March 31 st | testing the built eye state CNN model, designing eye region extraction with eye Haar cascades, replacing the face alignment using eye rotation with affine transform, replacing the eye extraction using haar with eye extraction using facial landmarks. Making the emotion detection model, cleaning the emotion dataset, training and testing the emotion detection model. Making the side face detection and recognition, making the yawning detection and recognition, making the side face detection test cases. Making forgot password android page with twilio SMS service integration, making the android layouts and activities for helpline and send bugs email intents. |
| 5. | April 1 st to April 20 th | Firestore integration with Python and Android application, performing noisy tests on all the classifiers, generating unified test cases for the classifiers, creating the generated unified test cases, rolling back to eye extraction with haar cascade, integrating and cascading all the designed classifiers. Making the required documentation of the work. |

6. PROJECT DEMONSTRATION

The project's working is divided into three sections, processing, uploading to online database and finally accessing the uploaded information. There will be no display associated with the processing part, and once the processing on the test cases taken is done we get 3 arrays per face in the classroom describing the behavior of each student, one is attendance array, in which the presence of the student is marked as "1" and his absence is marked as "0". The second array is for emotions in which unlike the attendance is not a binary array. The values in the emotion array range from 0 to 5, 0 to 4 denoting the emotions "neutral, anger, disgust, happy, surprise" and the 5th index refers to yawning. The integer values -1 will be appended to the array if the person is found to be absent from the frames. The third array is that of the side faces, it is an integer array that displays the frame number (or here the seconds during which the students had their faces turned). The fourth and the last one is the anomaly array, there will be one anomaly array generated for one video irrespective of the number of students present in the classroom. The anomaly array is used to check if there is any outsider present in the class, and if one is present it copies all the seconds in which he appears. So the anomaly array is an integer array that has the timing of the anomaly array presence in the class. The arrays are then processed with the help of script that identifies the positions and number of consecutive 0's in attendance to see at what times and for how long the student has slept or was inattentive in the classroom. The emotions array is processed to get the average of each emotions displayed by the student during the classroom hour.

[illegible]

Figure 14: Figure displaying the attendance array of one of the students

```

C:\Users\Dell\venv\Scripts\python.exe E:/projectImplementation/projectFiles/nn.py
[ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 0 0 0 0
0 0 5 5 0 0 0 0 0 0 0 0 0 0 0 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3
0 0 0 3 3 0 0 3 3 0 3 3 3 3 0 0 0 0 2 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1]

Process finished with exit code 0

```

Figure 15: Figure displaying the emotions array

The figure 15 displays the emotion array in which the -1's imply that the person was absent during that frame, the 0's neutral, 2's anger, 3's disgust and the 5's are yawning detection.

```

C:\Users\Dell\venv\Scripts\python.exe E:/projectImplementation/projectFiles/nn.py
Anomalies Present at:
[152 154 155 156 157 158 159 160]

Side faces present at:
[52, 53, 54, 55, 56, 57, 58, 59, 60]

Process finished with exit code 0

```

Figure 16: Values in the Anomaly and side face arrays

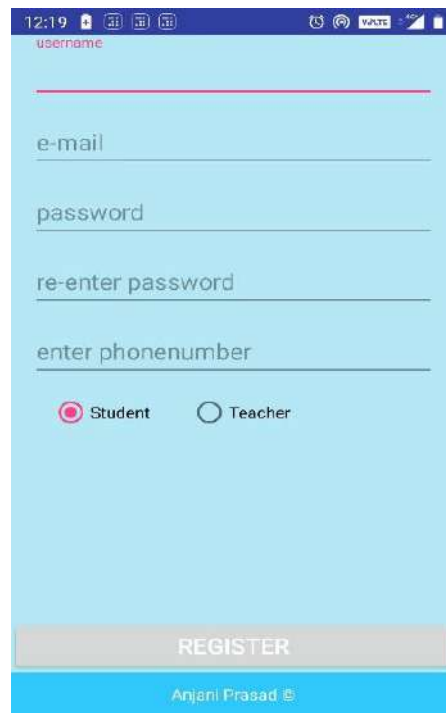
The figure 16 displays the values in the anomaly and the side face arrays. The values in these arrays represent the frames at which these actions were being performed. Since we have extracted 1 frame per second, the frame numbers correlate with the seconds in the duration. Making the values the seconds at which these actions were performed.

These values from the arrays after the array processing are sent to online database called the firebase. The data will be stored in this online NoSQL database in the form of a JSON key value pair elements. This data can then be accessed or read form the front end android application by the respective teachers and the students.

The teachers and the students register in the application with their email ID, phone number and name details and set a password. Once logging in both the teachers and students can see the details of the attendance, moods in the classroom, and the times during which sleep was identified in the classrooms. The teachers can see the details of the students that came late to the classroom, students that left the classroom early, if any non-student is present in the classroom and can find the name of any student from his image.

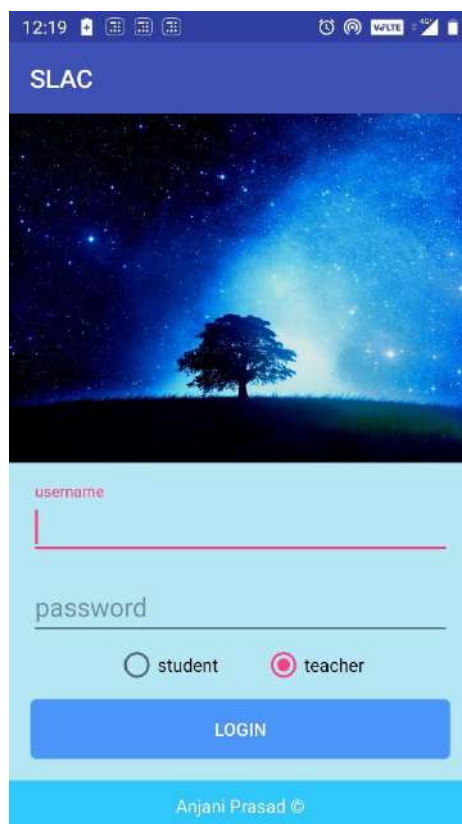


Figure 17: landing page of the Android application



A mobile application interface for user registration. The screen has a light blue background. At the top, there is a status bar with the time 12:19 and various icons. Below the status bar, the word "username" is displayed in red text above a text input field. This is followed by input fields for "e-mail", "password", "re-enter password", and "enter phonenumber". Below these fields are two radio buttons: "Student" (selected) and "Teacher". At the bottom, there is a grey "REGISTER" button and a footer with the text "Anjani Prasad ©".

Figure 18: user registration page



A mobile application interface for user login. The screen has a light blue background. At the top, there is a status bar with the time 12:19 and various icons. Below the status bar, the word "SLAC" is displayed in white text on a dark blue header. Below the header is a large image of a tree silhouette against a starry night sky. Below the image, the word "username" is displayed in red text above a text input field. This is followed by a "password" input field. Below these fields are two radio buttons: "student" and "teacher" (selected). At the bottom, there is a blue "LOGIN" button and a footer with the text "Anjani Prasad ©".

Figure 19: User login page



Figure 20: Post login page

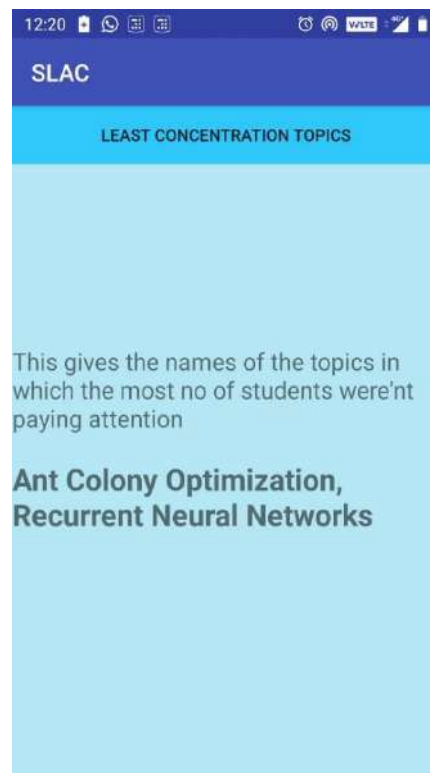


Figure 21: LCT topics page

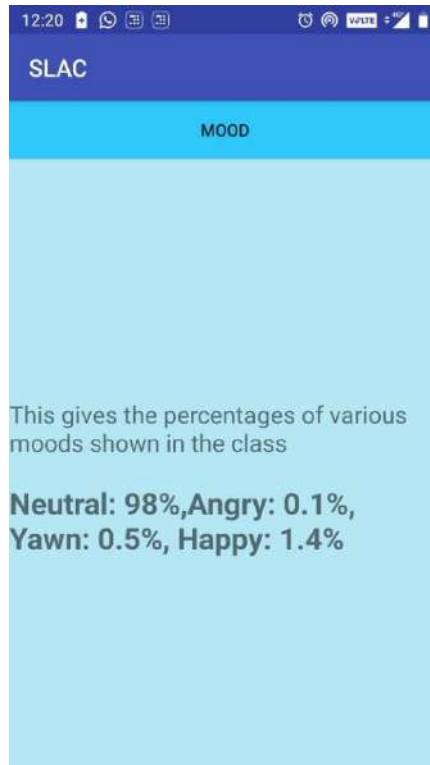


Figure 22: classroom mood page

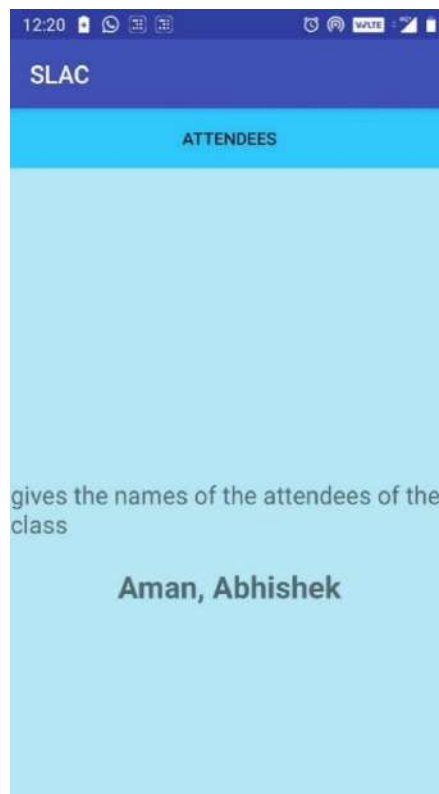


Figure 23: classroom attendees page

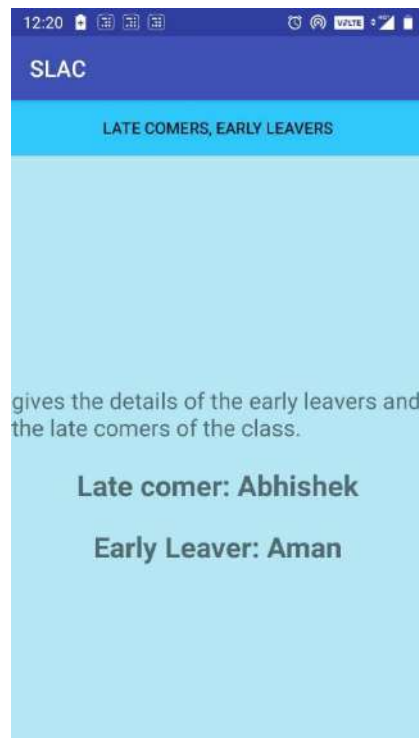


Figure 24: Late comers, Early Leavers page

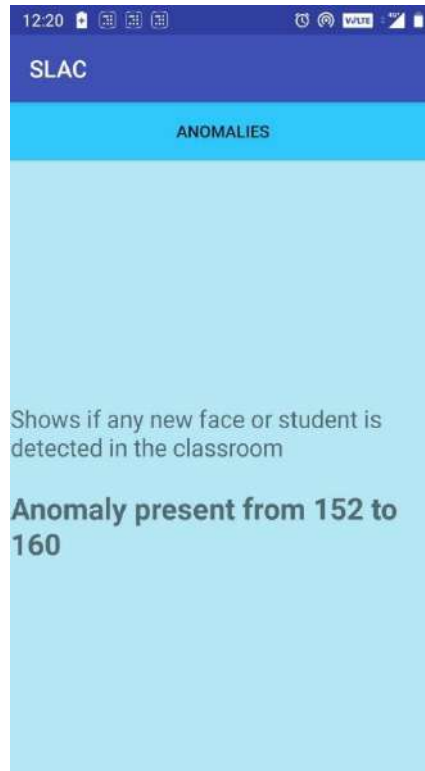


Figure 25: Anomalies page

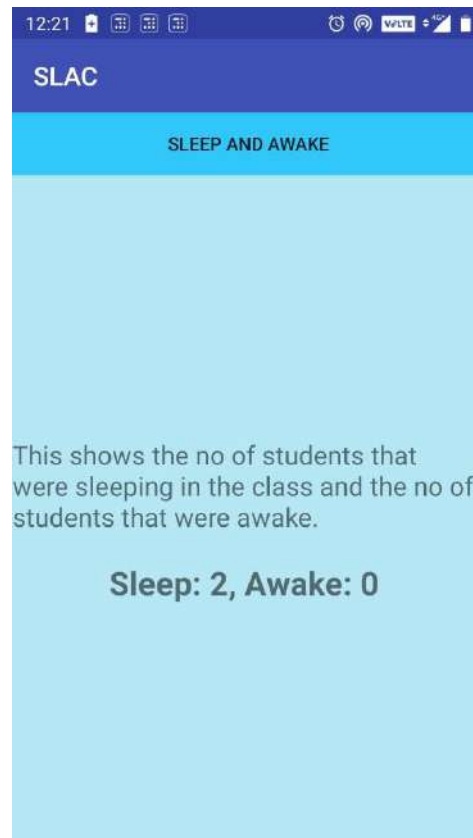


Figure 26: sleep and awake page

7. RESULTS AND DISCUSSION

After building the classifiers and making the test-cases I have performed 3 accuracy checks, one was while I was training the data, one after the classifiers are trained and the final one was on the data from the collected demonstration test cases. The first check was when the classifiers were getting trained and the testing set was made from the 20% of the dataset. All the classifiers gave a good performance on that data from the datasets. The second check was from a set of extremely noisy face images collected from my mobile phone. These images were mostly shot during the night and thus lacked the extra edge that the training set had. The final check was done on the test sets created for the demonstration and almost all the classifiers worked exceptionally well. The under performance of the emotion detection classifier can be attributed to the lack of a proper and equally distributed dataset.

Table 5: Dataset training and testing accuracies of classifier

| S.no | Classifier | Testing Accuracy | Training Accuracy |
|------|-----------------------|-----------------------|-------------------|
| 1. | Eye state detector | 96% (loss: 0.3696) | 98% |
| 2. | Emotion Detection | 85.55% | 84.94% |
| 3. | Face Recognizer (old) | 98.12% (loss: 0.0629) | 99.33% |

Table 6: Noisy data testing results

| S.no | Classifier/Model | Accuracy | Size of test set |
|------|-----------------------------------|----------|---------------------------|
| 1. | Dlib Face Detector | 89.74% | 78 (70:C 8:W) |
| 2. | OpenFace Recognizer | 82.35% | 51 (42:C 9:W) (8 classes) |
| 3. | Side Face detector (Haar, LBP) | 92.857% | 28 (26:C 2:W) |
| 4. | Side Face Recognition (nn4.small) | 72% | 25 (18:C 7:W) |

In the final demonstration dataset based checking all the faces were detected correctly, 4 face were recognized incorrectly, for 1 face eye extraction did not work, for 3 eyes the eye states were classified wrongly, 20 emotions were classified wrongly, all the side faces were detected correctly but 1 side face was recognized wrongly and all the yawning frames were detected and recognized wrongly.

8. SUMMARY

The Required model was designed using cascading of various classifiers and was found to be working for a classroom environment. Several tests including noisy data classification tests were done and the corresponding accuracies were found. The model works well for a small cohort based classrooms but will not work in its full potential in a bigger classroom where the students are a longer distance from the camera.

This is because the minimum resolution of the eye images required for the eye status detection is 24 by 24 pixels and if the eye region detected is smaller than the required the smaller image will be padded decreasing its chances of getting correctly classified.

The model is designed for GPU less pure CPU based systems, it can run smoothly on a laptop with 4 GB RAM, but the downside with running it on a CPU would be the slow processing speed that comes as a trade-off for the low processing power requirement.

The suggestible implementation for this model would be for the universities to get a standalone server for the processing of the classroom videos for a faster rate of processing. This model can be however used in various other scenarios like board room meeting, pitch meetings and also moderator less focus groups.

References

- [1] Xin Zhang, Cheng-Wei Wu: Analyzing Students' Attention in Class Using Wearable Devices. 2017 IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks
- [2] Shubham Shindea, Ashwin Kotharia: YOLO based Human Action Recognition and Localization. International Conference on Robotics and Smart Manufacturing (RoSMa2018)
- [3] Fraklin Silvia, Eddie E. galarza: Real Time Driver Drowsiness Detection Based on Driver's Face Image Behavior Using a System of Human Computer Interaction Implemented in a Smartphone Advances in Intelligent Systems and Computing. January 2018
- [4] Rong Fu, Dan Wang, Dongxing Li: University Classroom Attendance Based on Deep Learning. 2017 10th International Conference on Intelligent Computation Technology and Automation.
- [5] Byoung Chul Ko: A Brief Review of Facial Emotion Recognition Based on Visual Information. Sensors 2018, s18020401
- [6] Lei Zhao, Zengcai Wang "Human fatigue expression recognition through image-based dynamic multi-information and bimodal deep learning", Journal of Electronic Imaging, 2016.
- [7] Jacob Whitehill "The Faces of Engagement", IEEE transactions on Affective Computing, April 2014
- [8] Li-Jia Li, Mohamma Saberian "Multi-view Face Detection Using Deep Convolutional Neural Networks", GroundAI Journal, Feb 09, 2015
- [9] J. Deng, S.Zafeiriou , Y.zhou"Joint Multi-view Face Alignment in the Wild", GroundAI Journal, Aug 20,2017
- [10] Zhen-hua Feng, Josef Kittler, Patrik Huber "Face Detection, Bounding Box Aggregation and Pose Estimation for Robust Facial Landmark Localisation in the Wild", GroundAI Journal, May 05, 2017
- [11] Jacob Foytik, Richard Tompkins, "Multi-Pose Face Recognition And Tracking System", Science Direct Journal, 2017
- [12] <https://www.physiology.org/doi/full/10.1152/advan.00109.2016>
- [13] <https://www.pyimagesearch.com/2018/05/14/a-gentle-guide-to-deep-learning-object-detection/>

- [14] <http://www.site.uottawa.ca/~shervin/yawning/>
- [15] <https://answers.opencv.org/question/101291/improving-face-recognition-accuracy/>
- [16] <https://medium.com/neurohive-computer-vision/state-of-the-art-facial-expression-recognition-model-introducing-of-covariances-9718c3cca996>
- [17] <https://www.apprendimentoautomatico.it/apprendimentoautomatico-wpblog/en/emotions-detection-via-facial-expressions-with-python-opencv/>
- [18] <https://towardsdatascience.com/faced-cpu-real-time-face-detection-using-deep-learning-1488681c1602>
- [19] <https://www.learnopencv.com/training-better-haar-lbp-cascade-eye-detector-opencv/>
- [20] https://www.researchgate.net/figure/Opencl-frontal-and-profile-face-detector-results_fig1_228460706