International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 07 | July 2021 www.irjet.net p-ISSN: 2395-0072

ACUITY: A Student Alertness Monitoring System

Rushabh Dharmesh Kheni[1], Vikram Ramkrishna Kini[2], Yash Bafna[3], Sunil Khachane[4]

[1]*Student, Dept. of Computer Science, MCT's Rajiv Gandhi Institute of Technology, Mumbai, Maharashtra, India*
[2]*Student, Dept. of Computer Science, MCT's Rajiv Gandhi Institute of Technology, Mumbai, Maharashtra, India*
[3]*Student, Dept. of Computer Science, MCT's Rajiv Gandhi Institute of Technology, Mumbai, Maharashtra, India*
[4]*Professor, Head of Dept. of Computer Science, MCT's Rajiv Gandhi Institute of Technology, Mumbai, Maharashtra, India*

Abstract - *The COVID-19 pandemic has resulted in schools shut all across the world. Globally, over 1.2 billion children are out of the classroom. As a result, education has changed dramatically, with the distinctive rise of e-learning, whereby teaching is undertaken remotely and on digital platforms. Apart from various advantages of e-learning including efficiency, affordability, accessibility there are some major disadvantages. Due to the limited view of a student teachers find it difficult to check whether every student is attentive.*

***Key Words*: Caffe Model, CRST Tracker, Optical Flow, Face-Recognition, Image Organizer**

1.INTRODUCTION

Automated learning analytics is becoming an important topic in the educational community, which needs effective systems to monitor the learning process and provides, feedback to the teacher. Recent advances in visual sensors and computer vision methods enabled automated monitoring of behavior and affective states of learners at different levels from the university level 1 to the preschool level 2. Student affective states such as interested, tired, and confused are automatically determined from facial expressions, and attention state is computed from different visual cues such as face gaze, head motion, and body postures .

The basic idea of our work is to utilize capabilities of webcams to unobtrusively collect behavioral data of multiple students while attending traditional lectures in the classroom or virtually. We propose a methodology to compute features from the webcam data corresponding to visually observable behaviors and to apply machine learning methods to build models to predict attentive state of the individual students.

The issue was defining student attention in the way to correspond to observations made by the teacher or other hu- man observer. We analyze attention scores provided by human observers and match them with the observable behaviors, activities, gestures, etc. of the students. Those results allow us to define a meaning of observable attention levels in terms of student behavior.

We aimed to develop an automated feedback and monitoring tool for teachers, by measuring the overt student behaviors.

It also allows the students to observe their attention levels for future improvements. The system helps to (i) Classify the student behaviors into five affective states (ii) Automate the measurement of student engagement by deriving transitions from optical flow of each student (iii) Provide the analysis result to the teacher to plan better. This greatly helps the teachers to analyze their teaching methods, the mode of delivery of content, interest of students on the topic and also to follow the behavioral transitions for a better curriculum design.

2. METHODOLOGY

2.1 Face Detection

For performing face detection we are using a well-known and reliable python library named "OpenCV".

This module supports a highly improved "deep neural networks" (dnn) module, including the Caffe model.

When using OpenCV's deep neural network module with Caffe model, you'll need two sets of files

- The .prototxt file(s) which define the model architecture(i.e., the layers themselves).
- The .caffemodel file which contains the weights for theactual layers.

Both files are required when using models trained using Caffe for deep learning to accurately detect faces.

2.2 Face Tracking

Tracking can be defined as the problem of a face in the image plane as it moves around a scene. The aim of a face tracker is to generate the trajectory of a face over time by locating its position in every frame of the video. Object tracking is the important key for face tracking.

There are a lot of object tracking algorithms. Speed and steady performance are the basic requirements in our case.

A human face must be steadily tracked during it being within the picture. At the same time, tracking algorithms must operate fast enough to provide a particular frame rate that would allow for a certain tracking quality.

For the algorithm being developed four tracking techniques were considered:

- KCF Tracker (Kernelized Correlation Filters).
- CSRT Tracker (Discriminative Correlation Filter with Channel and Spatial Reliability).
- TLD Tracker(Tracking–Learning–Detection).
- MIL Tracker (Multiple instance learning).

The following requirements were applied to the algorithm:

- Real-time operation on a PC with a low-capacity processor
- Robust tracking for different viewing angles of the face, ideally up to 90 degrees inclusive.

TLD tracker is not applicable, as it gives many false negatives. KCF tracker is not the optimum choice because it is demanding high processor speed, while using other techniques lacks computation effectiveness.

Thus, the CSRT Tracker, being fast and accurate enough, best fit the problem. Face detection using the CAFFE model and its efficiency when applied to facial tracking in the pictureis considered in this article.

2.3 Facial Landmark Tracking

Facial feature points such as eyebrows, eyes, nose and mouth are prominent landmarks surrounding facial components. They depict critical information about facial expression and head movement. Hence accurate and perfect detection of facial features is important in several applications including face tracking.

The pre-trained facial landmark detector inside the dlib library is used to estimate the location of coordinates that mapto facial structures on the face.

Using these facial landmarks we are applying computer vision techniques to calculate the eye aspect ratio i.e. the distance between the upper eyelid and the lower eyelid and using this calculation, the student's drowsiness status can be estimated.

The distance between the upper lip and the lower lip is also calculated using the facial landmarks. This calculation can reveal if the student is yawning. The yawning and drowsiness status derives the students attentive state.

3. ALGORITHMS USED

3.1 Convolutional Neural Networks

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analysing visual imagery [2][3]. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics. They have applications in image and video recognition, recommender systems, image classification, medical image analysis, natural language processing, brain-computer interfaces, and financial time series

CNNs [2][3] are regularized versions of multilayer perceptrons. Multilayer perceptrons [6] usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "fully-connectedness" of these networks makes them prone to overfitting data. Typical ways of regularization include adding some form of magnitude measurement of weights to the loss function. CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns. Therefore, on the scale of connectedness and complexity, CNNs are on the lower extreme.

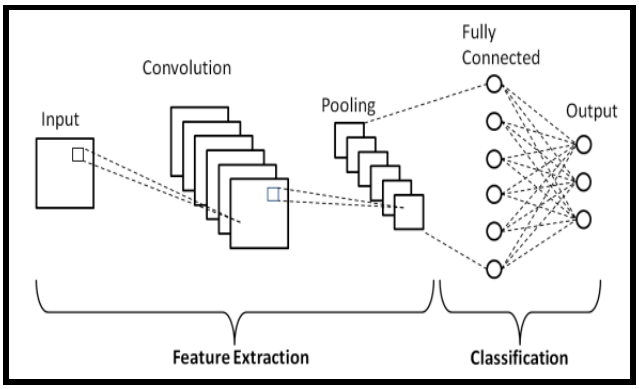

Fig -1: Working of Convolutional Neural Network

CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns the filters that in traditional algorithms were hand-engineered. This independence from prior knowledge and human effort in feature design is a major advantage.

3.2 CAFFE Model

Caffe is a deep learning framework developed by the Berkeley Vision and Learning Center (BVLC). It is written in C++ and has Python and Matlab bindings. Caffe is a deep

learning framework developed by the Berkeley Vision and Learning Center (BVLC). It is written in C++ and has Python and Matlab bindings.

There are 4 steps in training a CNN using Caffe:

Step 1 - Data preparation: In this step, we clean the images and store them in a format that can be used by Caffe. We will write a Python script that will handle both image pre-processing and storage.

Step 2 - Model definition: In this step, we choose a CNN architecture and we define its parameters in a configuration file with extension .prototxt.

Step 3 - Solver definition: The solver is responsible for model optimization. We define the solver parameters in a configuration file with extension .prototxt.

Step 4 - Model training: We train the model by executing one Caffe command from the terminal. After training the model, we will get the trained model in a file with extension .caffemodel.

After the training phase, we will use the .caffemodel trained model to make predictions of new unseen data.

3.3 CSRT

CSRT tracker is C implementation of the CSR-DCF (Channel and Spatial Reliability of Discriminative Correlation Filter) tracking algorithm in OpenCV library.

OpenCV has a Deep Neural Network (DNN) module that can support pre-trained deep learning object classifier which implements forward pass (inferencing) with deep networks. The object detection method Faster RCNN also built on with the help of some dnn layers where OpenCV dnn module can support easily. Detection model of OpenCV represents a high-level API for object detection networks, and also allows to set parameters for preprocessing input images. Furthermore, it creates a net form with trained weights and config, sets preprocessing input, runs forward pass and return result detections, also supports SSD (Single Shot Detector), Faster RCNN, and YOLO (You Look Only Once) topologies. Also the classification part is the same with the detection model, it also uses trained weights and config files, sets preprocessing input, runs forward pass and returns top - 1 prediction. Proposed method structure is given in Figure below:

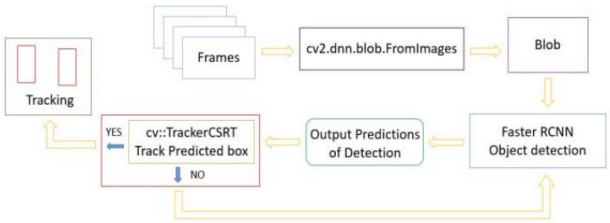

Fig -2: Structure of the tracking method

Implemented object tracking process contains two parts: first is already explained above, training object classifier and generating object detection model from training outcome file. Second part is the implementation of the tracking algorithm. In this figure we can see integration of the object classifier and tracker as tracking systems. From the coming frame algorithm takes blobs and gives it to the object detection model to predict location of object and classify as an object class. Output predictions of the detection passes to the tracking algorithm to track predicted box of object class and so on, any other circumstances of object prediction changes tracker will invoke the object classifier and restarts the process from the beginning. Even in subsequent frames a person will move away from the frame our detection model will identify the object with any position of it across a series of frames.

4. DESIGN DETAILS

4.1 Architecture

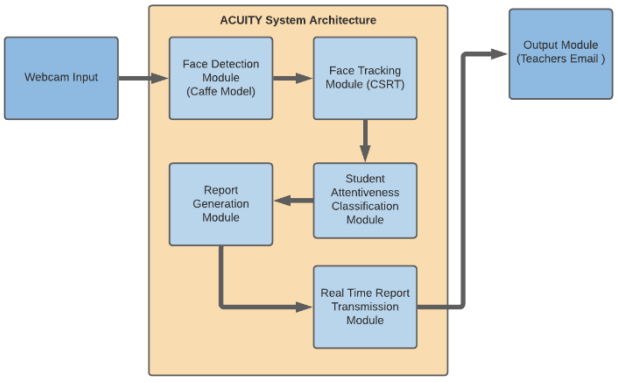

Fig -3: The Proposed System/System Architecture

1. Webcam Input :- Live frames from the webcam input is taken and transferred to the Face Detection Module

2. Face Detection Module :- Face detection module is applied on each frame and face is detected, the bounding box is formed and then is passed on to the Face Tracking Module.

3. Face Tracking Module :- The bounding box received by the Face Detection Module is tracked in the succeeding frames.

4. Student Attentiveness Classification Module :- The displacement of the bounding box is calculated, the distance between the eyelids is calculated and the distance between the upper and lower lip is calculated in order to classify the student's attentive state.

5. Report Generation Module :- The data generated by the Student Attentiveness Classification Module is summarized and converted into an excel file and a bar graph is also generated.

6. Real Time Report Transmission Module :- The excel report along with the bar graph png is sent to the respective teacher through a SMTP server.

4.2 Flow Diagram

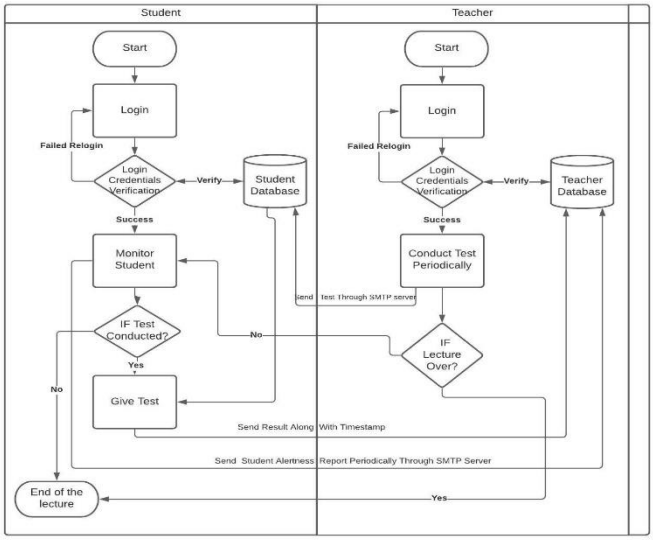

Fig -4: Flow Diagram

In the Flow diagram of Acuity we have two swimlanes for the student and teacher respectively. Every student and teacher has to login with correct credentials. This will be checked using the student and teacher database respectively.

Once the login is successfully completed the program will start monitoring the student continuously and will send periodic attentiveness reports to the teacher. Thus allowing the teacher to track the attention status of the student.

Teacher also has an option to conduct the test. The test will be in an MCQ format and will be sent to the students through SMTP servers. The students will answer the test and test results will be sent to the teacher along with the attentiveness report. The monitor program will stop once the lecture is ended or the student disconnects from the meet.

5. PROCEDURE

1. Faces of students are detected using opencv's deep neural network with the help of CAFFE model and the bounding box around the the face is created

2. This bounding box ((X1,Y1) and (X2,Y2)) is then given to the CSRT (The Channel and Spatial Reliability Tracker) object tracking algorithm, which locates the student's face in each frame.

3. In every succeeding frame, the displacement of the bounding box is calculated, and then using simple calculations the student's attentive state is calculated.

4. With the help of facial landmark predictor, facial features like eyes and mouth are detected

5. The distance between the eyelids (EAR distance) is calculated in order to calculate the student's drowsiness state.

6. The distance between the upper lips and lower lips is calculated in order to calculate the student's yawning state.

7. The test will be conducted in the form of MCQ test if the teacher wishes.

8. The test's results along with the student's comprehensive attentive states (including a bar graph) will be sent to the respective teacher through SMTP server.

6. PSUEDO CODE

```
while lecture is going on :
frame = vs.read() #get current frame through webcam
tracker = cv2.TrackerCSRT_create() #initialise csrt tracker
if face not detected in frame :
detect face using caffe model
get bounding box (X1,Y1),(X2,Y2)
pass bounding box to the tracker
set face = detected
else if face detected in frame :
using facial landmark predictor draw contours over eyes and mouth
        calculate distance between eyelids (EAR distance)
        if EAR < threshold value :
                drowsing = yes
        else :
                drowsing = no
        calculate distance between upper lip and lower lip (LIP distance)
        if LIP > threshold value :
                yawning = yes
        else :
                yawning = no
```

new bounding box = tracker.update(frame)
#update the bounding box by getting the new bounding box from the csrt tracker in
the next frame
calculate X and Y displacement of the new bounding box
if displacement > threshold value :
attentive state = Distracted
else :
attentive state = Paying attention
if test is available :
give test
else :
skip
add attentive states and test score to excel file
create summary of excel file
create bar graph of the summary
send the excel file and bar graph to teacher through smtp server
end

7. RESULTS

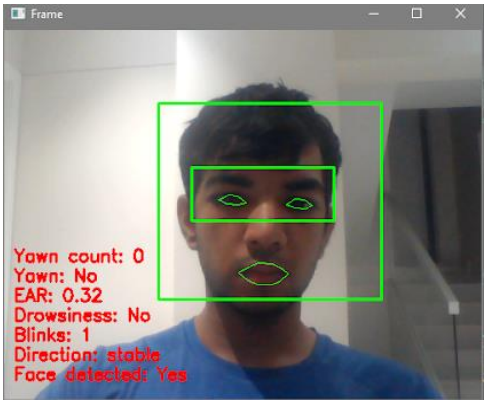

Fig -5: Student Paying Attention

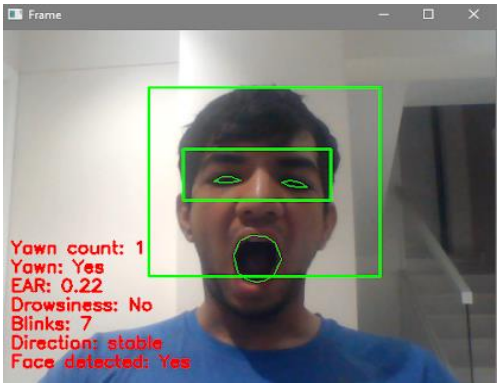

Fig -6: Student Yawning

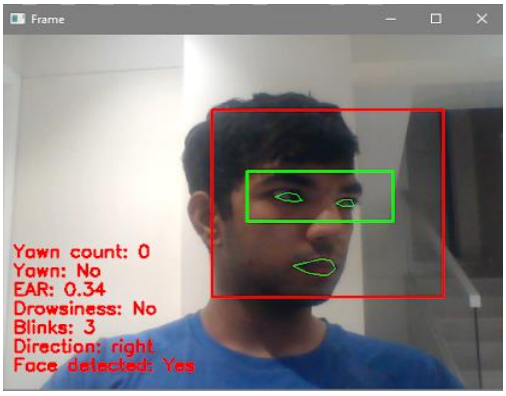

Fig -7: Student Distracted

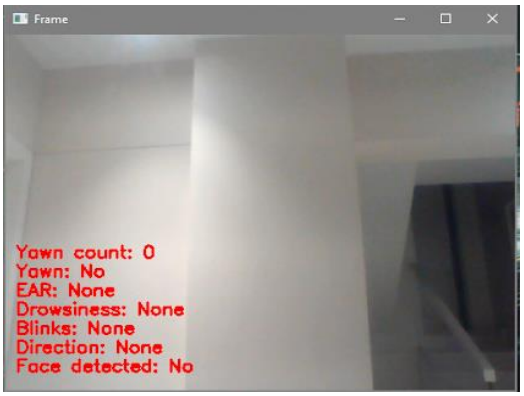

Fig -8: Student Out of View

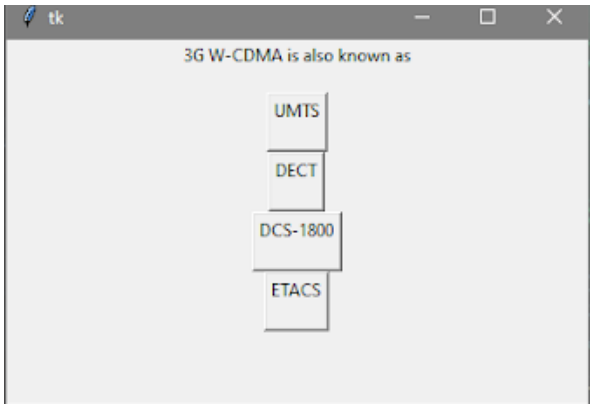

Fig -9: The Test received by the student In MCQ Form

	A	B	C	D	E	F	G	H	I
1		Time	Roll No	Attention	Transcribing	Yawning	Drowsing	Face	Test
2	0	22:02:09-22:02:12	A749	Paying Attention	Not Transcribing	Not Yawning	Not Drowsing	Detected	
3	1	22:02:12-22:02:17	A749	Paying Attention	Not Transcribing	Yawning	Not Drowsing	Detected	
4	2	22:02:17-22:02:18	A749	Paying Attention	Not Transcribing	Not Yawning	Not Drowsing	Detected	
5	3	22:02:18-22:02:19	A749	Not Paying Attention	Not Transcribing	Not Yawning	Not Drowsing	Detected	
6	4	22:02:19-22:02:24	A749	Paying Attention	Transcribing	Not Yawning	Not Drowsing	Detected	
7	5	22:02:24-22:02:25	A749	Not Paying Attention	Not Transcribing	Not Yawning	Not Drowsing	Detected	
8	6	22:02:25-22:02:28	A749	Paying Attention	Transcribing	Not Yawning	Not Drowsing	Detected	
9	7	22:02:28-22:02:29	A749	Not Paying Attention	Not Transcribing	Not Yawning	Not Drowsing	Detected	
10	8	22:02:29-22:02:33	A749	Paying Attention	Not Transcribing	Not Yawning	Not Drowsing	Detected	
11	9	22:02:33-22:02:39	A749	Paying Attention	Not Transcribing	Yawning	Not Drowsing	Detected	
12	10	22:02:39-22:02:39	A749	Paying Attention	Not Transcribing	Not Yawning	Not Drowsing	Detected	

Fig -10: Comprehensive report of student's attentive states

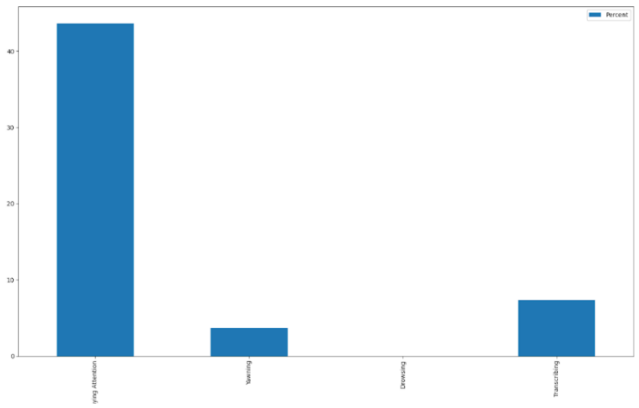

Fig -11: Graphical Representation of Student Attentiveness Report

7. CONCLUSIONS

With an accuracy of more than 90 % we are developing a prototype of ACUITY (Student Alertness Monitoring System) which detects and tracks student's faces and calculates student's attentiveness based on the eye movement, facial displacement, mouth movement and provide a detailed analysis to the respective teacher, and therefore allowing the teacher to analyse the behaviours and alertness of students during the lecture. By using the Caffe model the accuracy and efficiency of our system has increased significantly and it also reduces the number of false negative predictions.

The detailed student attentiveness data will be sent to the teacher along with the graphical representation for easier understanding thereby allowing the teacher to gain complete knowledge as to which student is paying attention and who is not. With this information the teacher can analyze the interest of students on the topic and accordingly change the mode of delivery of content for the betterment of the students.

REFERENCES

[1] Bidwell, Jonathan, and H. Funchs, "Classroom Analytics: Measuring Student Engagement with Automated Gaze Tracking,"The University of North Carolina at Chapel Hill, Chapel Hill, NC ,2011.

[2] Teachscape Classroom Walkthrough Teachscape.com Web.10.20.2010. http://www.teachscape.com

[3] Lim H, Lee S G, Nam K. Validating E-learning factors affecting training effectiveness[J].

[4] S. O. Ba and J. M. Odobez, "Recognizing visual focus of attention from head pose in natural meetings", IEEE Transactions on Systems, Man, and Cybernetics, Part B: (Cybernetics),Vol. 39,Issue.1, pp.16 – 33,2009.

[5] Ba, Sileye O., Hayley Hung, and Jean-Marc Odobez, "Visual activity context for focus of attention estimation in dynamic meetings," IEEE International Conference on Multimedia and Expo(ICME),pp.1424- 1427,2009

[6] Voit, Michael, and Rainer Stiefelhagen,"3D user-perspective, voxel- based estimation of visual focus of attention in dynamic meeting scenarios," International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction, pp.51, 2010.

[7]Stiefelhagen and Rainer, "Tracking focus of attention in meetings,". Proceedings of the Fourth IEEE International Conference on Multimodal Interfaces, pp. 273-280, 2002.

[8]Murphy-Chutorian, Erik, and Mohan M. Trivedi, "3D tracking and dynamic analysis of human head movements and attentional targets," Second ACM/IEEE International Conference on Distributed SmartCameras(ICDSC), pp.1-8, 2008.