AttentiveJoe

Akash Amin
Software Engineering
San Jose State University
San Jose, California
akash.amin@sjsu.edu
SJSU ID: 015272933

Brinda Rajeshbhai Pujara
Software Engineering
San Jose State University
San Jose, California
brindarajeshbhai.pujara@sjsu.edu
SJSU ID: 014844180

Nalini Mehendale
Software Engineering
San Jose State University
San Jose, California
nalini.mehendale@sjsu.edu
SJSU ID: 015312206

Saketh Gali
Software Engineering
San Jose State University
San Jose, California
saketh.gali@sjsu.edu
SJSU ID: 015504853

***Abstract*— Learning through online video is extremely popular. But there's no way to determine whether a student is actively participating in a lecture. An algorithm for real-time eye and lips state classification employing a simple webcam is presented. Here a simulation of an online class platform, to detect the Attentiveness of students, is developed during which the eyes and lips of the person seated ahead of the camera are detected employing python libraries.**

***Keywords*—attentive, machine learning, student, fatigue**

I. Introduction

In online classrooms, students can be inattentive and fatigued sometimes. Nowadays, online video education is steadily becoming more important. The reasons are not only geographical differences between the teacher and the student, but also new teaching concepts such as “flipped classroom” and various other factors as well. In the U.S.A. nearly 60% of the students use online education. Recently the pandemic has made all the modes of teachings virtual which even more emphasizes the need to build this system.

II. Literature Review

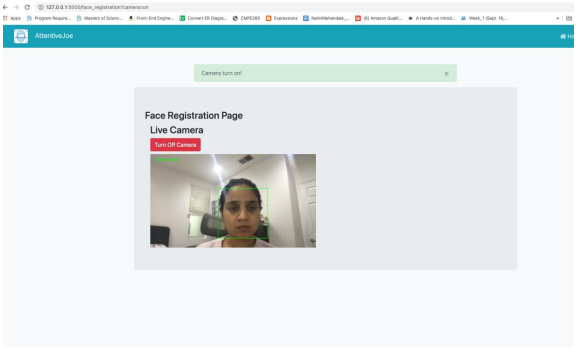
We explored the various image processing libraries like Dlib, cv2. On further reading we found that Dlib is the fastest and most accurate predictor, but it does not detect small sized faces. To overcome this challenge we did use OpenCV cv2 which, irrespective of size, provides accurate results. Also detects faces at various angles.

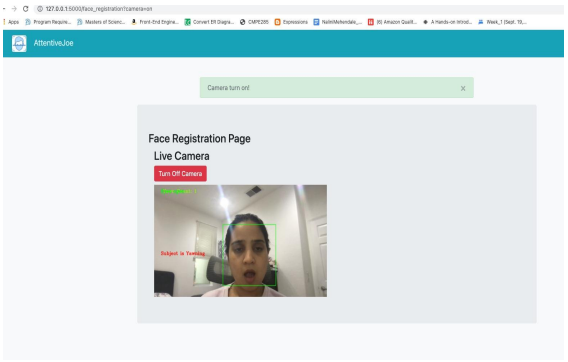
III. Project Description

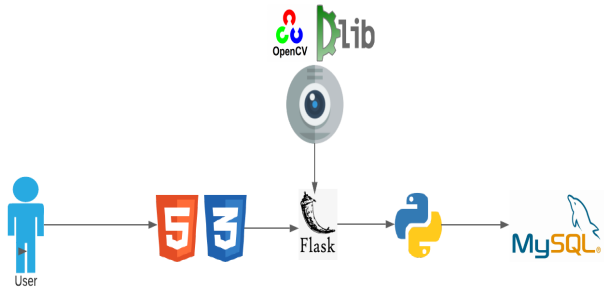
We have built a web application that prototypes video conferencing tools that actively detect the behavior (drowsiness, yawn) of the user during the time the person is attending the meeting.

A. Image Recognition

To recognize the user behavior we are using 68 face landmarks. Based on the users face movements we are calculating the euclidean distances between the facial landmarks. To detect the drowsiness we are calculating the vertical and horizontal distances between the 6 eye landmarks and if the vertical distance is reduced and horizontal distance increases to an extent. This helps us predict drowsiness. Similarly for the Yawn detection the dimensions of upper lip and lower lip are taken into account and based on movement and distance between the points of both lips, we can detect Yawning. Moreover, to increase the accuracy of the image recognition we have used the Dlib and cv2 model as they are faster and accurate. Also they help recognize faces from far distances and various angles, which increases the precision.

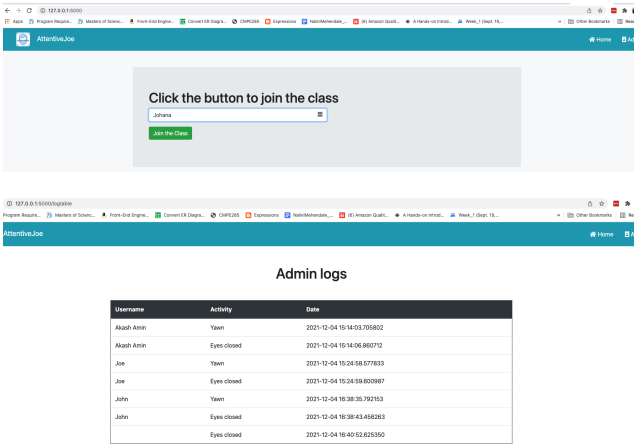






B. *Web Application*

To provide camera access in the browser we are using OpenCV and generating frames that continuously send video feeds for processing the behavior of the user. Flask encapsulates the user interface and python image recognition into one application. We are using mysql to store the logs of behavior from the video feed and further authorized users can access those logs and check the overall class activeness during the session.



IV. ARCHITECTURE DIAGRAM

The Architecture of the application is a 3 tier system. We have used HTML, CSS, JavaScript and Bootstrap at the presentation layer. The application tier uses python for processing the video streaming model and generates the output based on streamed input. The flask framework provides the application interface and camera access via browser. The data tier uses MySQL to store the logs of user activity during online classes.

V. TESTING

We used white box and black box testing approach. We tested the actual algorithm and code paths in the white box. In the black box we did the GUI testing and scenario testing.

In white box testing we wrote a small test driver and checked for algorithms by feeding actual frames, data uploads and query, front end and back end connectivity.

```
def test_pipeline():
    cap = cv2.VideoCapture(0)
    yawns = 0
    yawn_status = False
    closed = False


    while True:
        ret, frame = cap.read()
        img = yawn_detection_wrapper(frame)
        img = drowsiness_detection_wrapper(img, closed)
        cv2.imshow('Yawn Detection', img)

        if cv2.waitKey(1) == 13: #13 is the Enter Key
            break

    cap.release()
    cv2.destroyAllWindows()
```

The black box test cases involved checking for camera ON/OFF, face recognition, eye detection, eye blink, eye blink count, yawn detection, admin login, student activity report and student activity report accuracy.

VI. CONCLUSION

Attentive Joe was accurately able to measure the student attentiveness during the class. Irrespective of far and near distances we were able to detect the yawn and drowsiness of the students and based on that the students were being shortlisted as being attentive or not. Also the application interface generated a report for the professor to check the students behavior during the class.

Acknowledgment

The authors would like to thank Prof. Rakesh Ranjan and Teaching Assistant Apoorv Mehrotra for guiding and providing a platform to discuss and develop this project.

References

[1] https://medium.com/analytics-vidhya/yawn-detection-using-opencv-and-dlib-e04ba79b9936

[2] https://github.com/twinklejaswani/Yawn-Detection-using-OpenCV

[3] https://towardsdatascience.com/face-detection-models-which-to-use-and-why-d263e82c302c