Head Posture Assessment

Naru Rohith Reddy 18BIT0126, Sumanth 18BCE0473

May 2021

1 Abstract

Most people usually sit most of the time in-front of their computers nowadays because of modern lifestyle and now due to this pandemic situation this has been increased. Now as employees started working from home and students have their classes online, most of the students might lose their attention during their classes and their exams mostly being conducted online. And we are using the image processing software that generates the content from the frames that have been collected from the video feed through the live web-cam. The output is the position of the head tilted way of head with angle. Here we use the anaconda software to code the project and the flask framework is used to build GUI and to develop a well-organized interface for the software. We will be doing this project in python language on Spyder Ide that comes with the anaconda.

2 Introduction

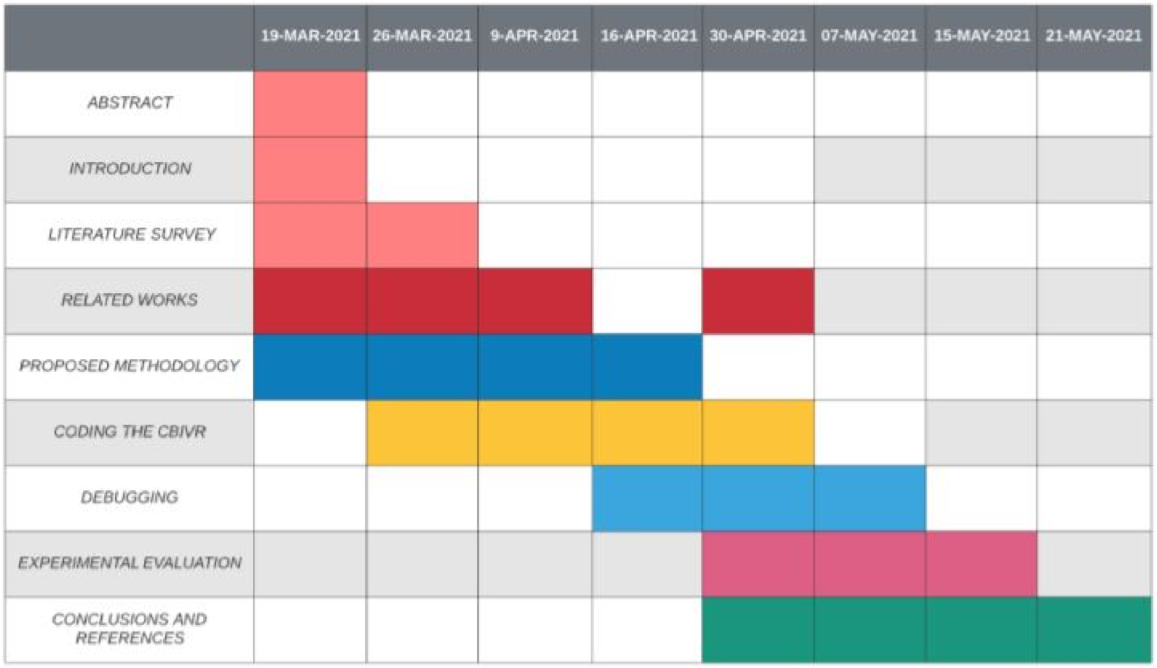
Basically, in our project here we focus on the student community so as to help the organizations to conduct fair and un-biased online examination. Here we are developing a neck posture recognition model that was developed by the python's OpenCV and flask framework, here we develop our algorithm in the way that it produces insights on the neck posture for every 5 sec. By this model We can assure the accuracy up to 85percent out of 100 sec it shows perfect insight for about 85 sec. And our model can be embedded with the other examination platforms so that the platform can use our software for proctoring their participants. Our model can warn when the user is not present in front of the device. Here the model uses open-pose model based on the Keras library to detect the face then by using the angle of head position we produce the insight. This model will be not only be focused on the student community now a days many people spend the most of the time in-front of the electronic devices. During this period their head position can be in poor condition that develop pressure on the spine and cause the neck pain or the spine related pains this system can also correct the position of the user to sit correctly and make the neck inclination angle into a specified range that it prevents neck related and spine related pains by doing so.

Here mainly we will be using Convolutional Neural Network to identify the landmarks of the face after we had successfully got our landmarks the other challaenge is to find the pose of the neck.We need five points of the face i.e. is nose tip, left and right points of ears, and left eye and right eye. We take standard 3D coordinates of these facial landmarks and try to estimate the rational and translational vectors at the nose tip. Now, for an accurate estimate, we need to intrinsic parameters of the camera like focal length, optical center, and radial distortion parameters. We can estimate the former two and assume the last one is not present to make our work easier. After obtaining the required vectors we can project those 3D points on a 2D surface that is our image.

Medical applications

we all know that now a days there are many people who spend most of the time in fornt of the laptops where due to poor sitting position and neck position many sever pains are induced in the neck and the backbone. where as on the other hand we have many patients who are suffering from the internal injuries or concutions when they are involved in the accident or any sudden injuries and hard impact on the neck. To these patients doctors usually prescribe them to strech daily which includes neck like tilting towards left and tilting towars right which helps in relaxing and contracting of the nerves that connect our head to spine and help us treating the injuries and help us relieve from sever pains of neck and spine. Our project can be used to monitor the patient streching width and can be able to monitor the patient.
https://drive.google.com/file/d/1HR10k8APtg9SCPQlGflsHF3SlaO1LL6r/view?usp=sharing

KEYWORDS
Pandemic,Flask frame-work, anaconda,GUI,Spyder,CNN,Keras,Tensorflow.



	19-MAR-2021	26-MAR-2021	9-APR-2021	16-APR-2021	30-APR-2021	07-MAY-2021	15-MAY-2021	21-MAY-2021
ABSTRACT	🟥							
INTRODUCTION	🟥							
LITERATURE SURVEY	🟥	🟥						
RELATED WORKS	🟥	🟥	🟥		🟥			
PROPOSED METHODOLOGY	🟦	🟦	🟦	🟦				
CODING THE CBIVR		🟨	🟨	🟨	🟨			
DEBUGGING				🟦	🟦	🟦		
EXPERIMENTAL EVALUATION					🟪	🟪	🟪	
CONCLUSIONS AND REFERENCES					🟩	🟩	🟩	🟩

The above is the timeline for our project completion with milestones.

3 Motivation

As when considered our own university we are making all the academic progress through the online platforms and we are having the exams on the code-tantra which comes with an inbuilt video and audio proctored examination platform. So I had the interest to develop the software that could be helpful for academic purpose

And here for the above scenario my proposed software could be helpful to monitor the student as if we are in the physical class together. And more over the main motivation to do this is it could be more helpful for the companies to monitor their employees about how much time they work and all.

Many people spend time in-front of the electronic devices in a poor head position which leads to neck pain and stress on the spine thereby they will be effected with spine related problems.so to prevent that sort of cases we here proposing this head posture assessment model.

4 Research Gap

1. Image processing as if we do in the CBIVR
2. Image identification
3. Content retrieval from the image that has been preprocessed
4. Presentation of the content that has been extracted

5 Existing System and Challenges

There are few models made with basic JavaScript libraries but they aren't that efficient as it takes time for each frame to capture, send it to the server and then assess it. Also, they don't work in the background. Here, we are using python libraries with web interface using flask framework using CV library which allows the program to run in the background also. It would have been even much better if the current model was able to solve the issue of multiple recognitions.

6 LIMITATIONS OF OUR MODEL

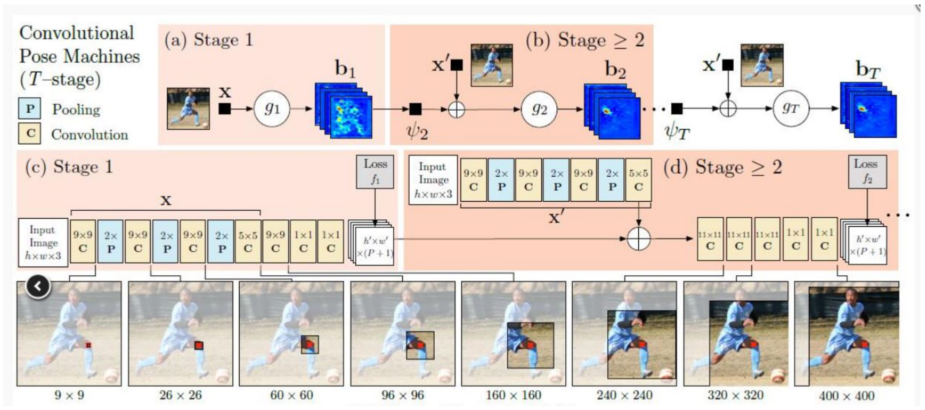
1. Light must be considerable in order to get best images for the assessment.
2. No of frames processed should be around 40-60 more than this the model takes long time to process the frames and we can't build up a good model.
3. Requires some storage space in-order to store the images.

7 ADVANCEMENTS THAT CAN BE DONE

1. Can be embedded into any application
2. Can be used for the photography like we make the neck posture as best as possible to provide with good exposure in the photos.
3. Can be joined to camera in the smartphones.

8 Proposed Methodology

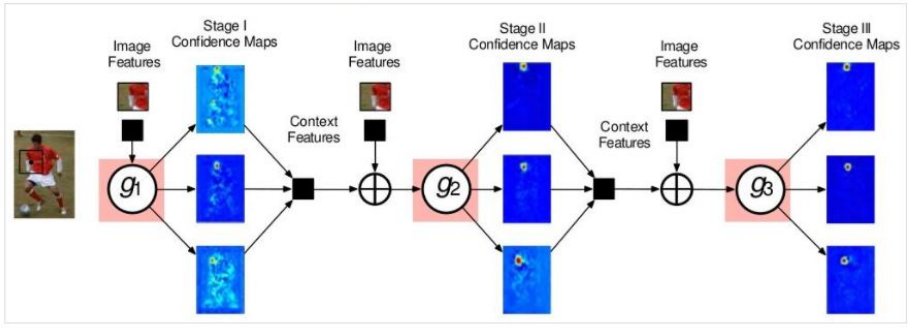
Convolutional pose estimator



simple Architecture for Convolutional pose machine

Here the Convolutional pose machine helps us to identify the keypoints we here use OpenCv which has an incorporated openpose keras model for identification of the keypoints. once we get the keypoints we will plot them in the 3-D space then we will interplot the points with the 2-D image and then we will be able to calculate the rotational and transactional vectors that help us to find the pose in our case the angle of inclination of the user.

Sequence of predictors produce increasingly refined estimates for each keypoint location.



keypoint estimation using sequence of predictors

Pose Detection

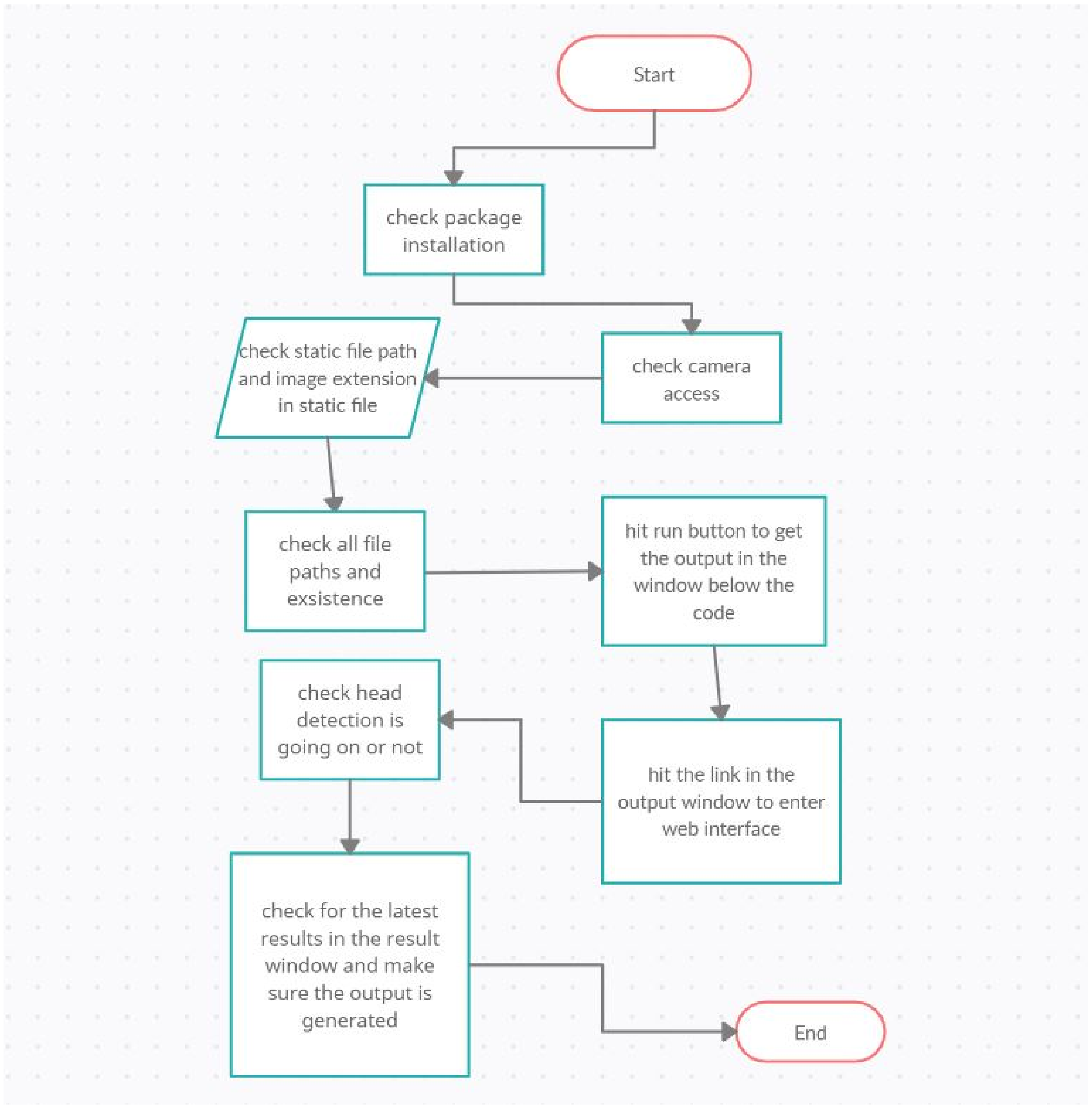
We need five points of the face i.e. is nose tip, left and right points of ears, and left eye and right eye. We take standard 3D coordinates of these facial landmarks and try to estimate the rational and translational vectors at the nose tip. Now, for an accurate estimate, we need to intrinsic parameters of the camera like focal length, optical center, and radial distortion parameters. We can estimate the former two and assume the last one is not present to make our work easier. After obtaining the required vectors we can project those 3D points on a 2D surface that is our image.

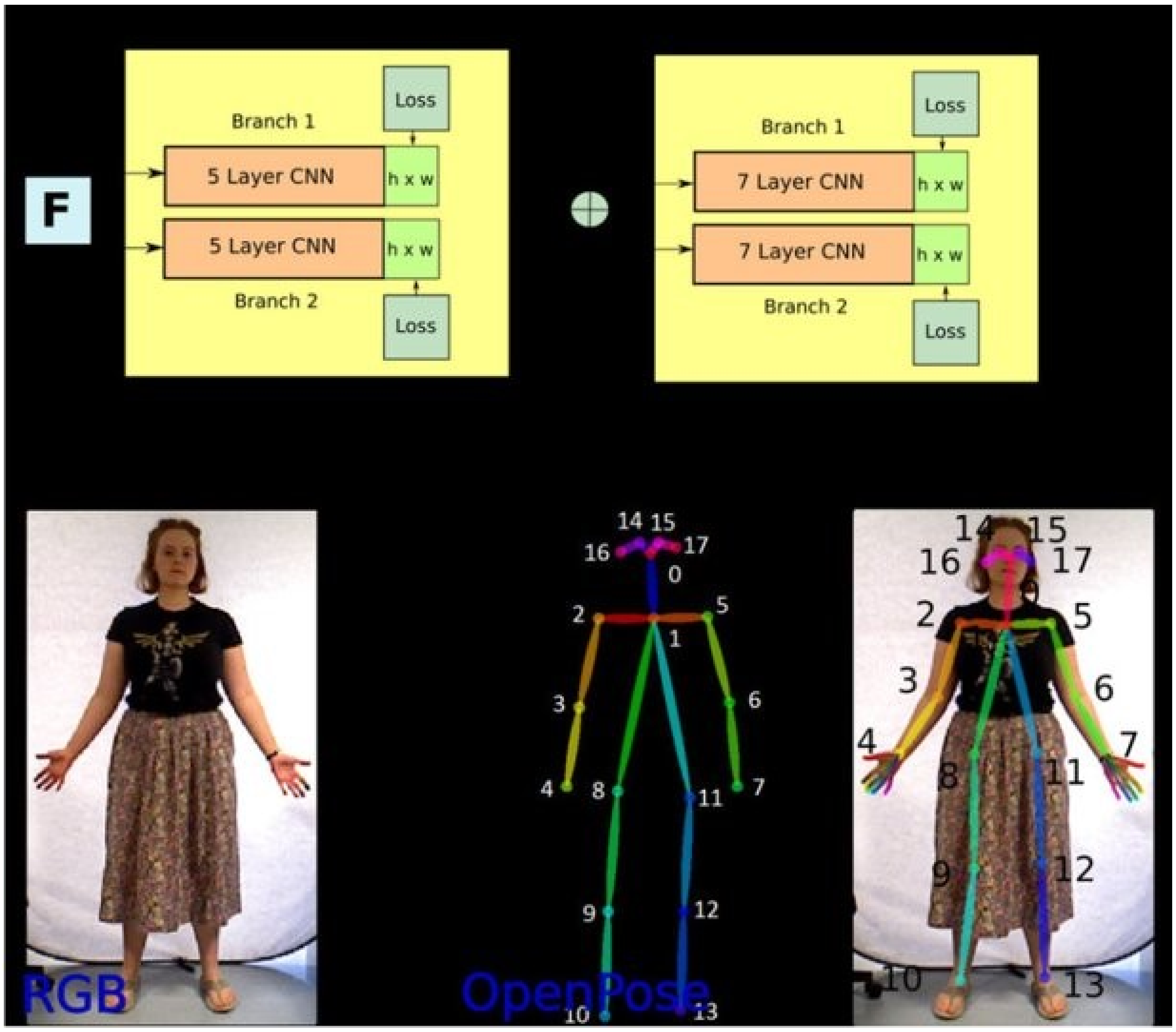
1. In first step the image is passed through baseline CNN network to extract the feature maps of the input
2. The feature map is then process in a multi-stage CNN pipeline to generate the Part Confidence Maps and Part Affinity Field
3. In the last step, the Confidence Maps and Part Affinity Fields that are generated above are processed by a greedy bipartite matching algorithm to obtain the poses Confidence Maps and Part Affinity Fields

Confidence Maps: A Confidence Map is a 2D representation of the belief that a particular body part can be located in any given pixel. Confidence Maps are described by following equation:
S = (S1, S2., SJ) where Sj belongs to R$^{w*h}, jbelongsto 1...J$

Part Affinity Fields: Part Affinity is a set of 2D vector fields that encodes location and orientation of limbs of different people in the image. It encodes thee data in the form of pairwise connections between body parts.
L = (L$_1, L_2., L_C) where L_c belongs to R^{w*h*c}, c belongs to 1...C$

Workflow or Architecture Diagram

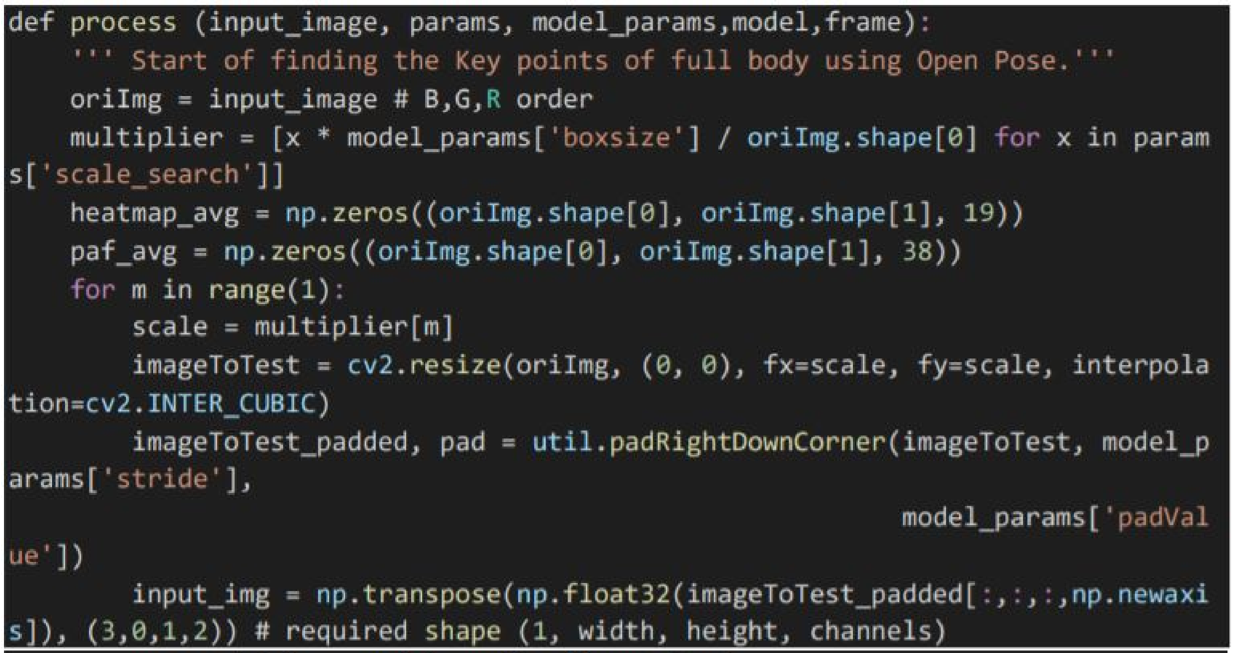

Start
check package installation
check camera access
check static file path and image extension in static file
check all file paths and exsistence
hit run button to get the output in the window below the code
hit the link in the output window to enter web interface
check head detection is going on or not
check for the latest results in the result window and make sure the output is generated
End



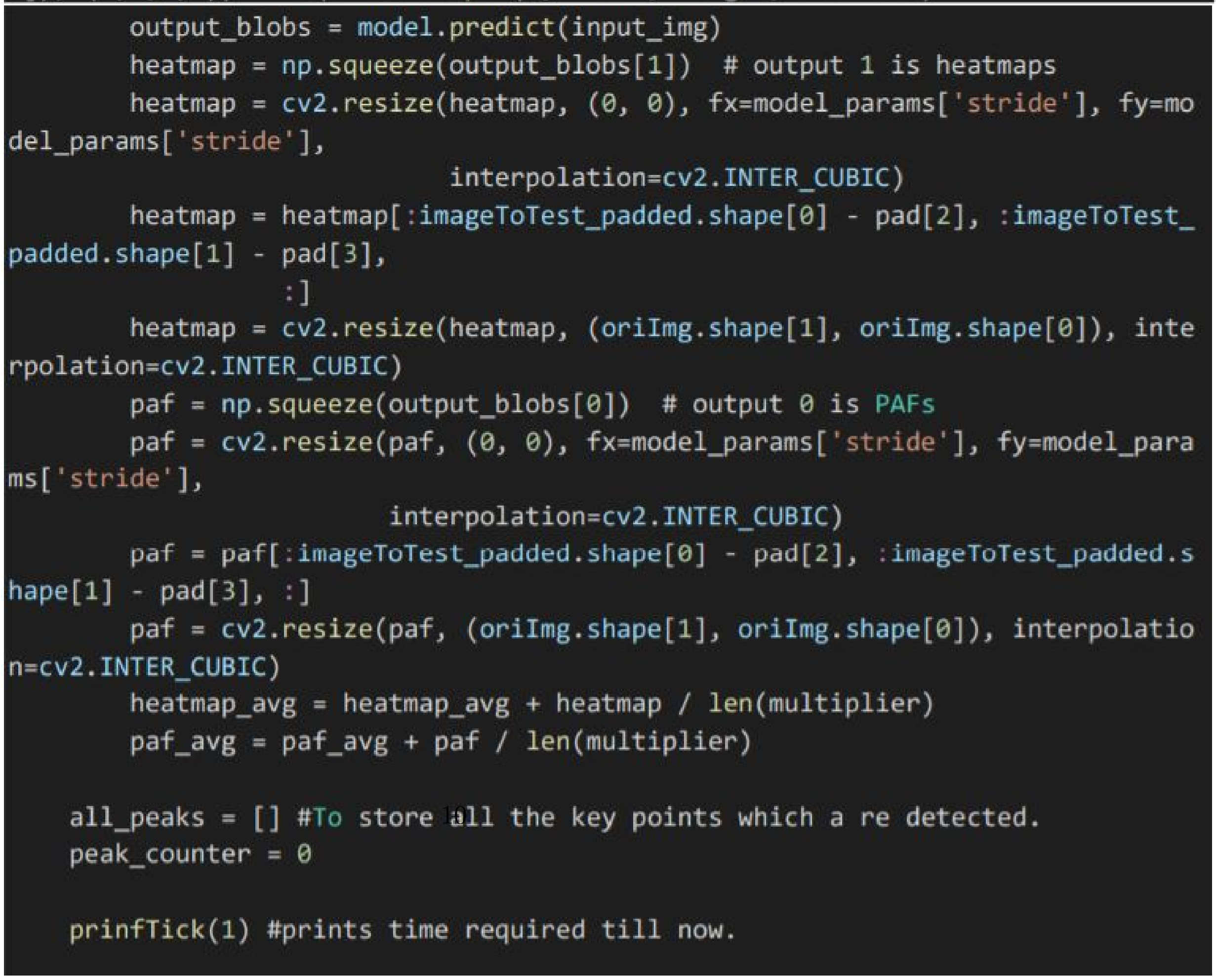
CNN example

The top branch, predicts the confidence maps of different body parts like the right eye, left eye, ears, and others.The bottom branch predicts the affinity fields which represents a degree of association between different body parts of the input image then the confidence maps and affinity fields are being processed by greedy inference.

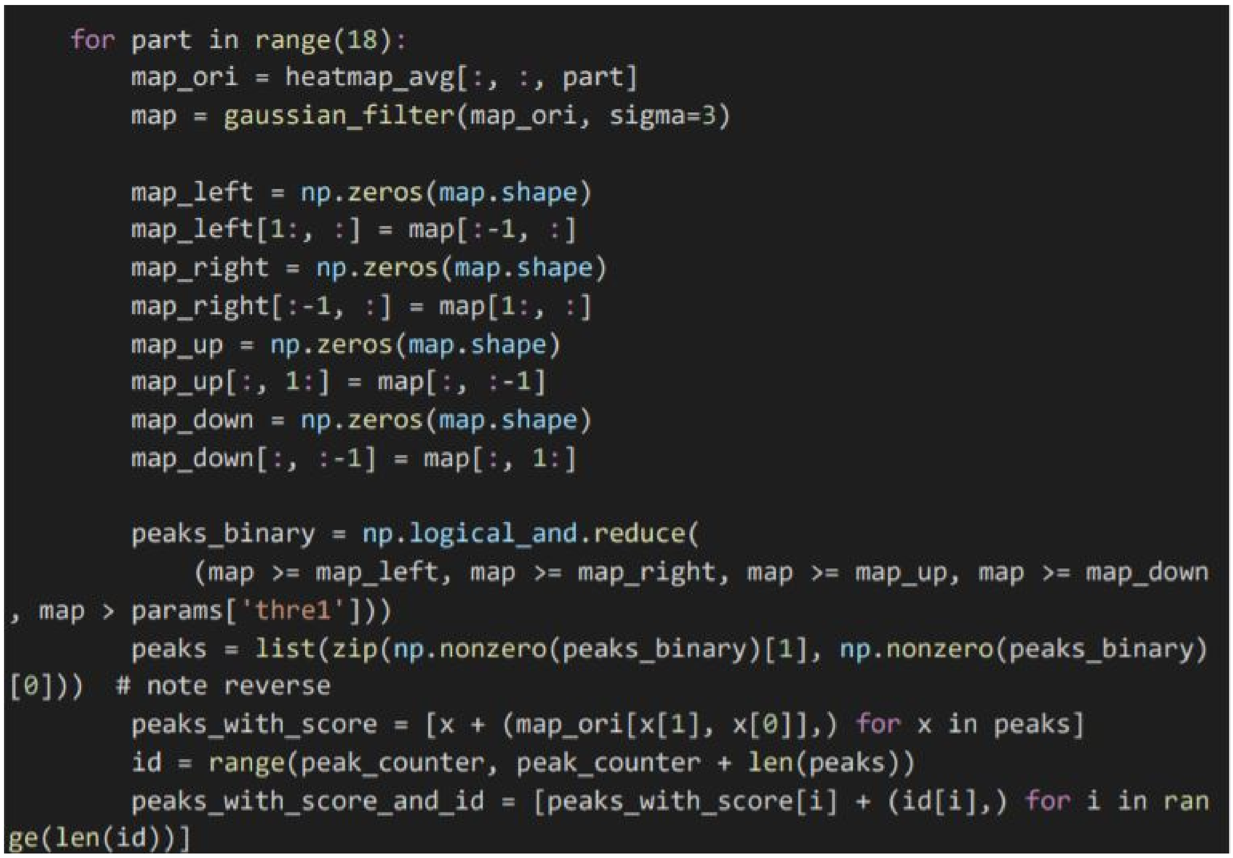
keypoint detector sample code



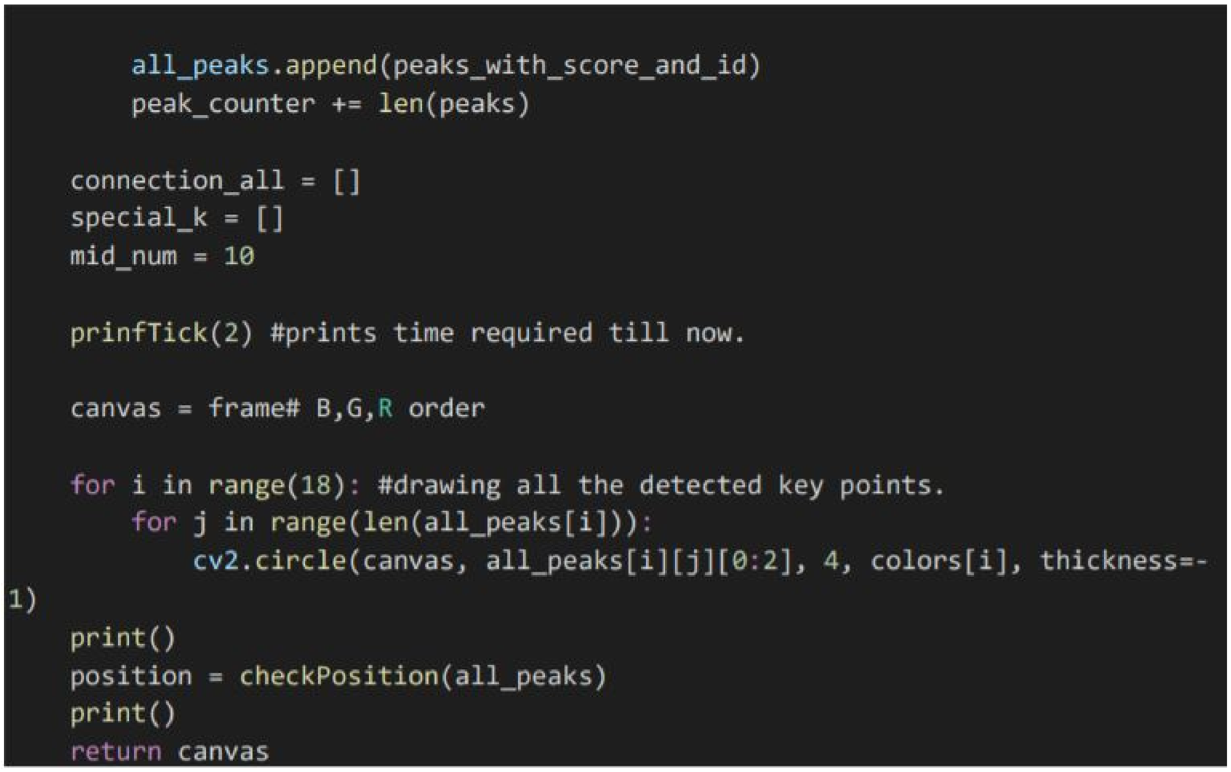
```
def process (input_image, params, model_params,model,frame):
    ''' Start of finding the Key points of full body using Open Pose.'''
    oriImg = input_image # B,G,R order
    multiplier = [x * model_params['boxsize'] / oriImg.shape[0] for x in param
s['scale_search']]
    heatmap_avg = np.zeros((oriImg.shape[0], oriImg.shape[1], 19))
    paf_avg = np.zeros((oriImg.shape[0], oriImg.shape[1], 38))
    for m in range(1):
        scale = multiplier[m]
        imageToTest = cv2.resize(oriImg, (0, 0), fx=scale, fy=scale, interpola
tion=cv2.INTER_CUBIC)
        imageToTest_padded, pad = util.padRightDownCorner(imageToTest, model_p
arams['stride'],
                                                          model_params['padVal
ue'])
        input_img = np.transpose(np.float32(imageToTest_padded[:,:,:,np.newaxi
s]), (3,0,1,2)) # required shape (1, width, height, channels)
```



```
        output_blobs = model.predict(input_img)
        heatmap = np.squeeze(output_blobs[1])  # output 1 is heatmaps
        heatmap = cv2.resize(heatmap, (0, 0), fx=model_params['stride'], fy=mo
del_params['stride'],
                             interpolation=cv2.INTER_CUBIC)
        heatmap = heatmap[:imageToTest_padded.shape[0] - pad[2], :imageToTest_
padded.shape[1] - pad[3],
                  :]
        heatmap = cv2.resize(heatmap, (oriImg.shape[1], oriImg.shape[0]), inte
rpolation=cv2.INTER_CUBIC)
        paf = np.squeeze(output_blobs[0])  # output 0 is PAFs
        paf = cv2.resize(paf, (0, 0), fx=model_params['stride'], fy=model_para
ms['stride'],
                         interpolation=cv2.INTER_CUBIC)
        paf = paf[:imageToTest_padded.shape[0] - pad[2], :imageToTest_padded.s
hape[1] - pad[3], :]
        paf = cv2.resize(paf, (oriImg.shape[1], oriImg.shape[0]), interpolatio
n=cv2.INTER_CUBIC)
        heatmap_avg = heatmap_avg + heatmap / len(multiplier)
        paf_avg = paf_avg + paf / len(multiplier)

    all_peaks = [] #To store all the key points which a re detected.
    peak_counter = 0

    prinfTick(1) #prints time required till now.
```

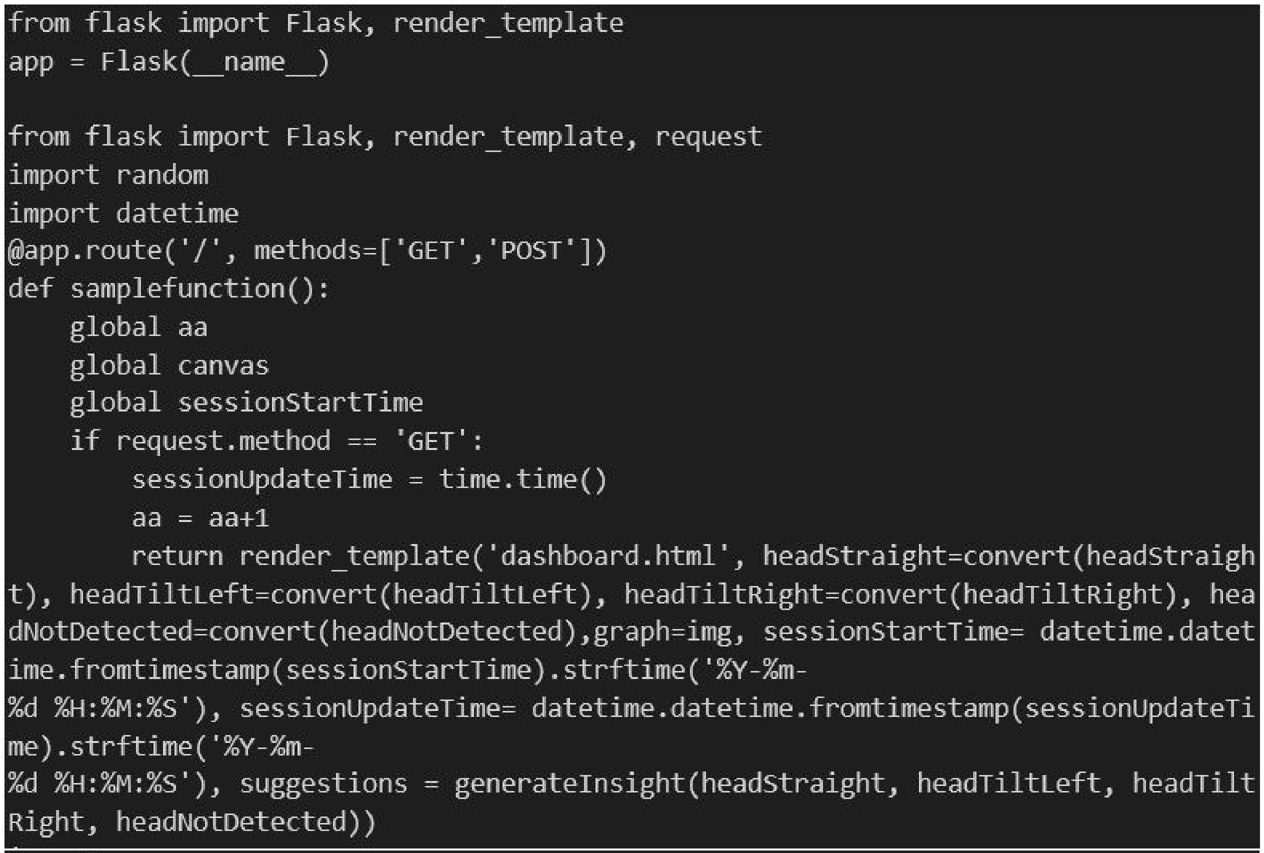


```
    for part in range(18):
        map_ori = heatmap_avg[:, :, part]
        map = gaussian_filter(map_ori, sigma=3)

        map_left = np.zeros(map.shape)
        map_left[1:, :] = map[:-1, :]
        map_right = np.zeros(map.shape)
        map_right[:-1, :] = map[1:, :]
        map_up = np.zeros(map.shape)
        map_up[:, 1:] = map[:, :-1]
        map_down = np.zeros(map.shape)
        map_down[:, :-1] = map[:, 1:]

        peaks_binary = np.logical_and.reduce(
            (map >= map_left, map >= map_right, map >= map_up, map >= map_down
, map > params['thre1']))
        peaks = list(zip(np.nonzero(peaks_binary)[1], np.nonzero(peaks_binary)
[0]))  # note reverse
        peaks_with_score = [x + (map_ori[x[1], x[0]],) for x in peaks]
        id = range(peak_counter, peak_counter + len(peaks))
        peaks_with_score_and_id = [peaks_with_score[i] + (id[i],) for i in ran
ge(len(id))]
```

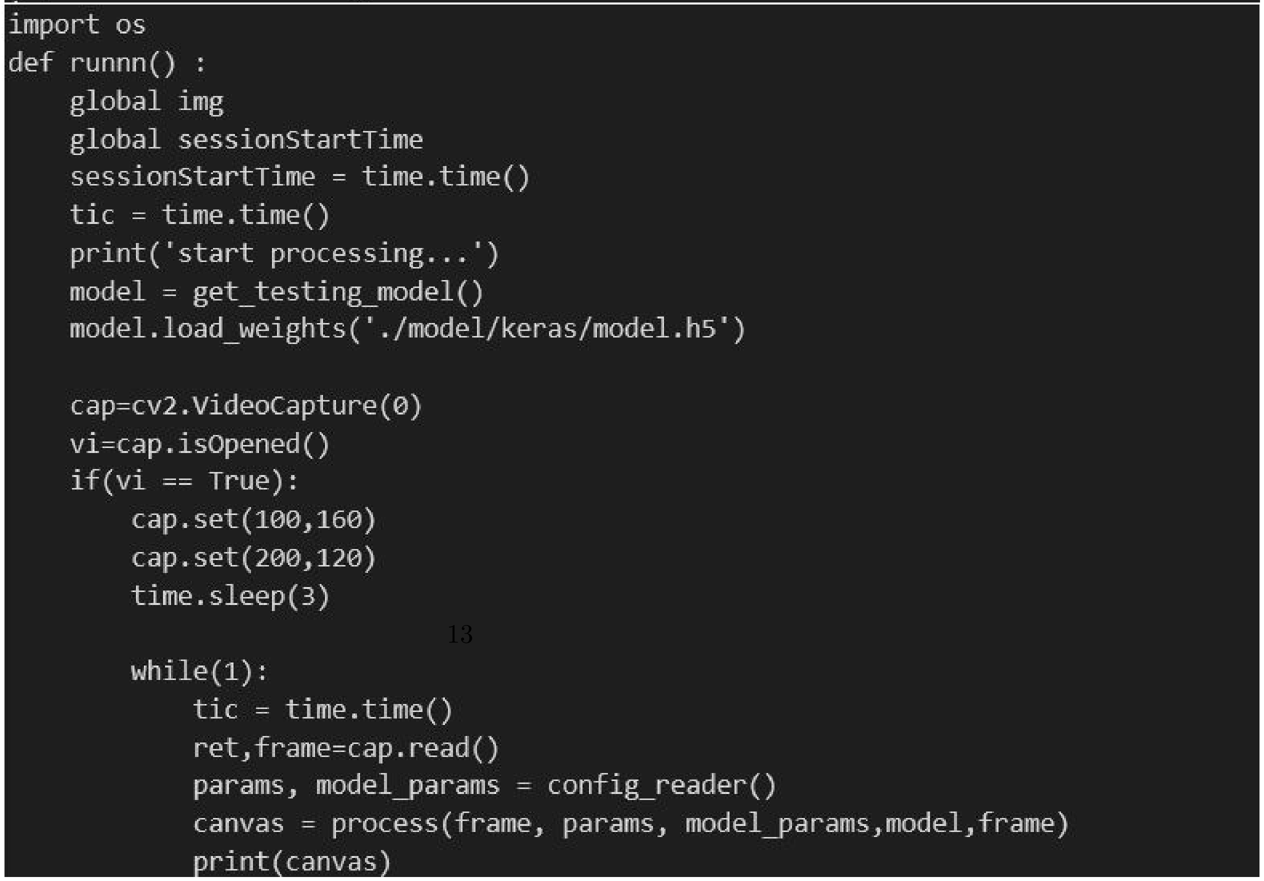


```
        all_peaks.append(peaks_with_score_and_id)
        peak_counter += len(peaks)

    connection_all = []
    special_k = []
    mid_num = 10

    prinfTick(2) #prints time required till now.

    canvas = frame# B,G,R order

    for i in range(18): #drawing all the detected key points.
        for j in range(len(all_peaks[i])):
            cv2.circle(canvas, all_peaks[i][j][0:2], 4, colors[i], thickness=-
1)
    print()
    position = checkPosition(all_peaks)
    print()
    return canvas
```

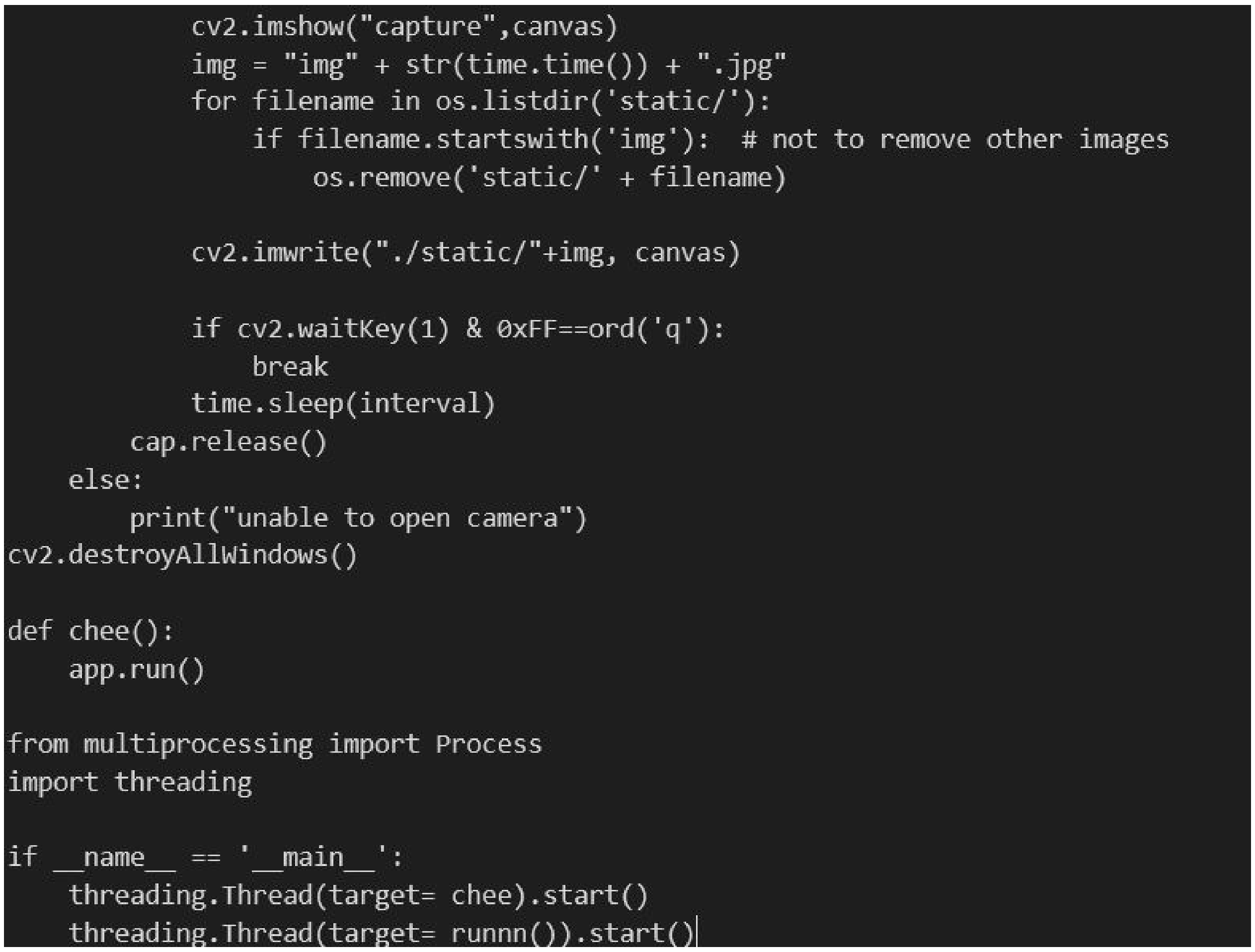
Flask code to make GUI



```
from flask import Flask, render_template
app = Flask(__name__)

from flask import Flask, render_template, request
import random
import datetime
@app.route('/', methods=['GET','POST'])
def samplefunction():
    global aa
    global canvas
    global sessionStartTime
    if request.method == 'GET':
        sessionUpdateTime = time.time()
        aa = aa+1
        return render_template('dashboard.html', headStraight=convert(headStraigh
t), headTiltLeft=convert(headTiltLeft), headTiltRight=convert(headTiltRight), hea
dNotDetected=convert(headNotDetected),graph=img, sessionStartTime= datetime.datet
ime.fromtimestamp(sessionStartTime).strftime('%Y-%m-
%d %H:%M:%S'), sessionUpdateTime= datetime.datetime.fromtimestamp(sessionUpdateTi
me).strftime('%Y-%m-
%d %H:%M:%S'), suggestions = generateInsight(headStraight, headTiltLeft, headTilt
Right, headNotDetected))
```

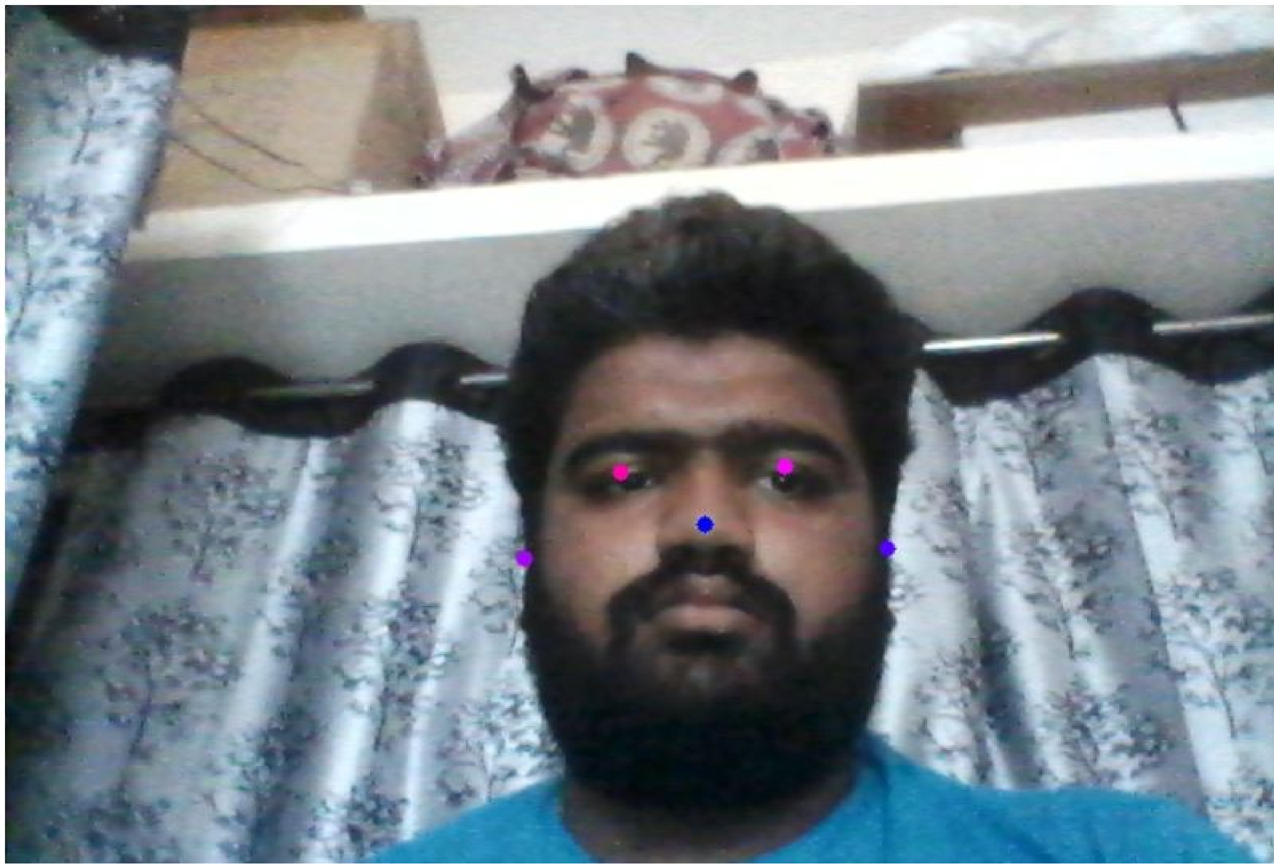


```
import os
def runnn() :
    global img
    global sessionStartTime
    sessionStartTime = time.time()
    tic = time.time()
    print('start processing...')
    model = get_testing_model()
    model.load_weights('./model/keras/model.h5')

    cap=cv2.VideoCapture(0)
    vi=cap.isOpened()
    if(vi == True):
        cap.set(100,160)
        cap.set(200,120)
        time.sleep(3)

        while(1):
            tic = time.time()
            ret,frame=cap.read()
            params, model_params = config_reader()
            canvas = process(frame, params, model_params,model,frame)
            print(canvas)
```

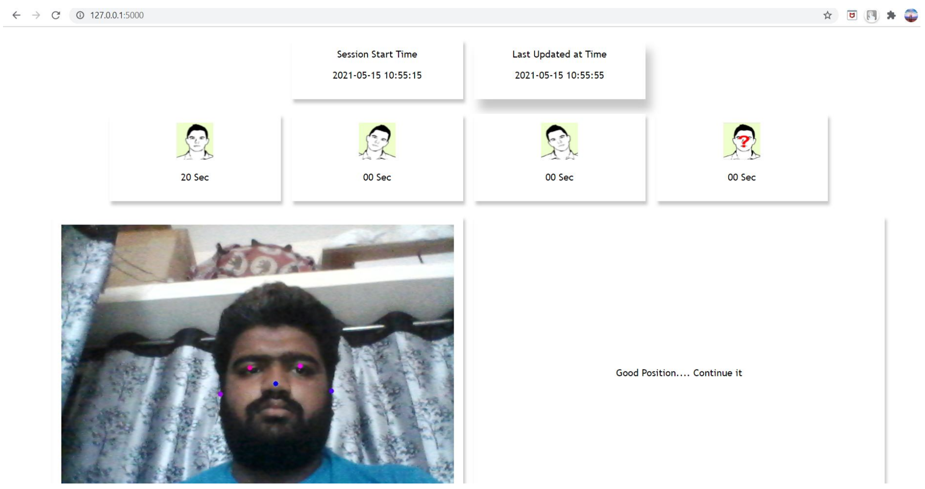


```
            cv2.imshow("capture",canvas)
            img = "img" + str(time.time()) + ".jpg"
            for filename in os.listdir('static/'):
                if filename.startswith('img'):  # not to remove other images
                    os.remove('static/' + filename)

            cv2.imwrite("./static/"+img, canvas)

            if cv2.waitKey(1) & 0xFF==ord('q'):
                break
            time.sleep(interval)
        cap.release()
    else:
        print("unable to open camera")
cv2.destroyAllWindows()

def chee():
    app.run()

from multiprocessing import Process
import threading

if __name__ == '__main__':
    threading.Thread(target= chee).start()
    threading.Thread(target= runnn()).start()
```

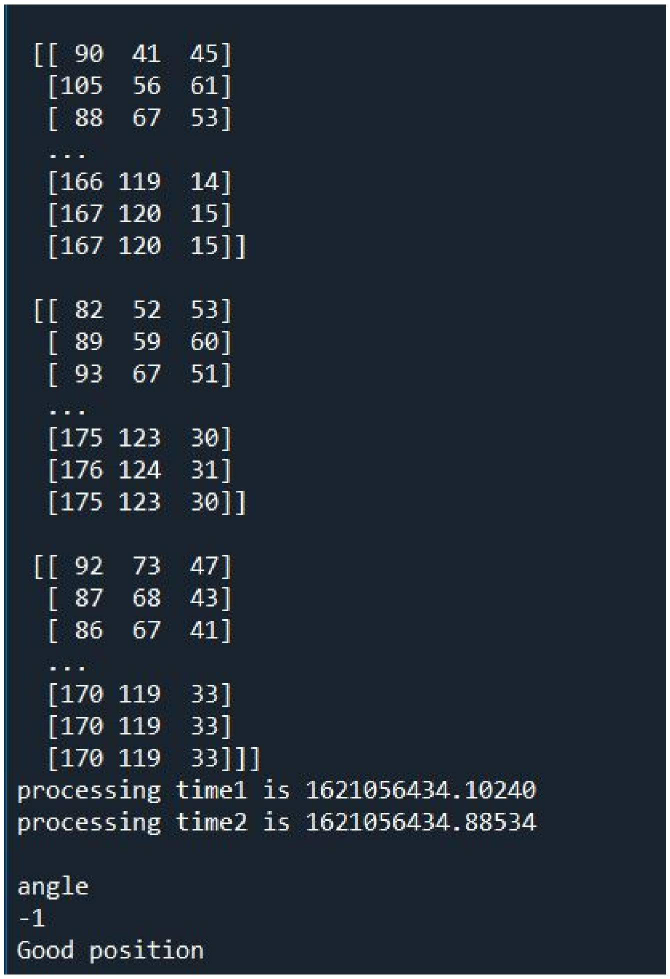
Sample how the keypoints look like on face



Sample how the GUI look like

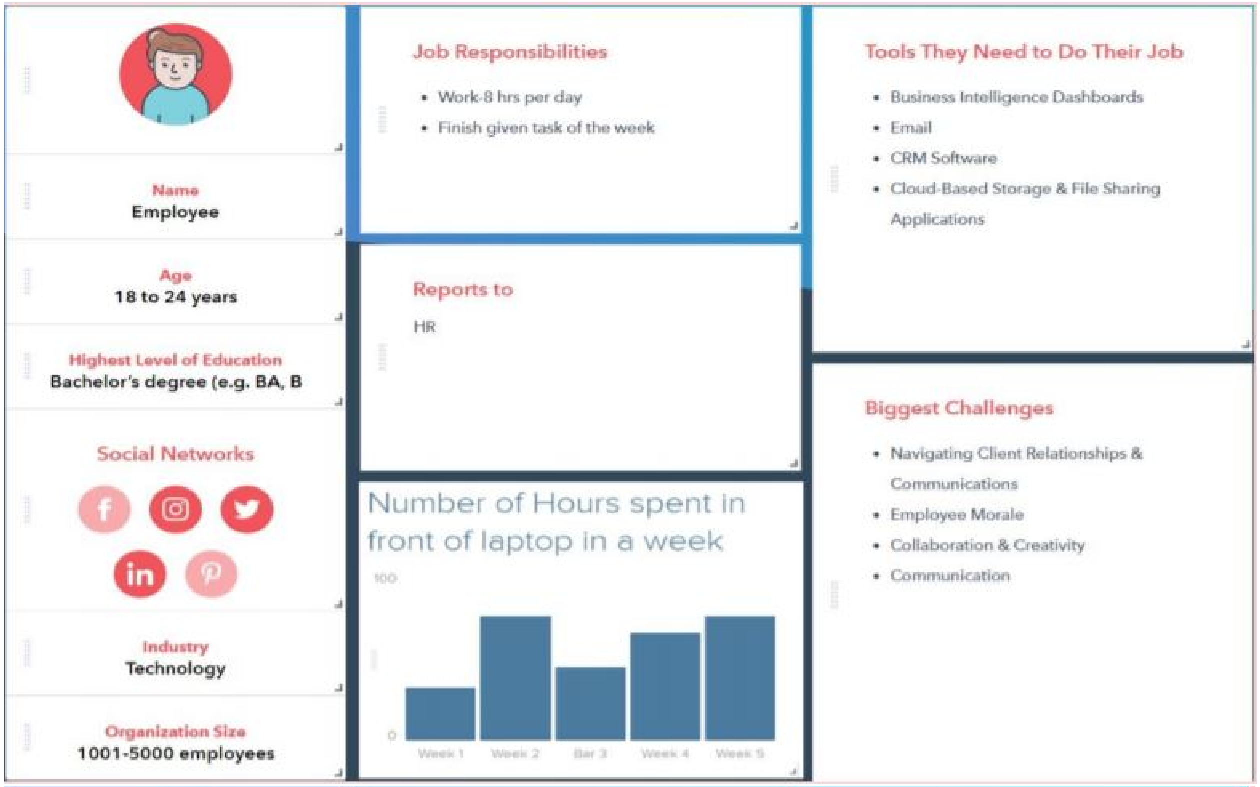


Sample how the output will be in terminal



```
 [[ 90  41  45]
  [105  56  61]
  [ 88  67  53]
  ...
  [166 119  14]
  [167 120  15]
  [167 120  15]]

 [[ 82  52  53]
  [ 89  59  60]
  [ 93  67  51]
  ...
  [175 123  30]
  [176 124  31]
  [175 123  30]]

 [[ 92  73  47]
  [ 87  68  43]
  [ 86  67  41]
  ...
  [170 119  33]
  [170 119  33]
  [170 119  33]]]
processing time1 is 1621056434.10240
processing time2 is 1621056434.88534

angle
-1
Good position
```

Personas for some core users



Name
Employee

Age
18 to 24 years

Highest Level of Education
Bachelor's degree (e.g. BA, B

Social Networks

Industry
Technology

Organization Size
1001-5000 employees

Job Responsibilities

- Work-8 hrs per day
- Finish given task of the week

Reports to

HR

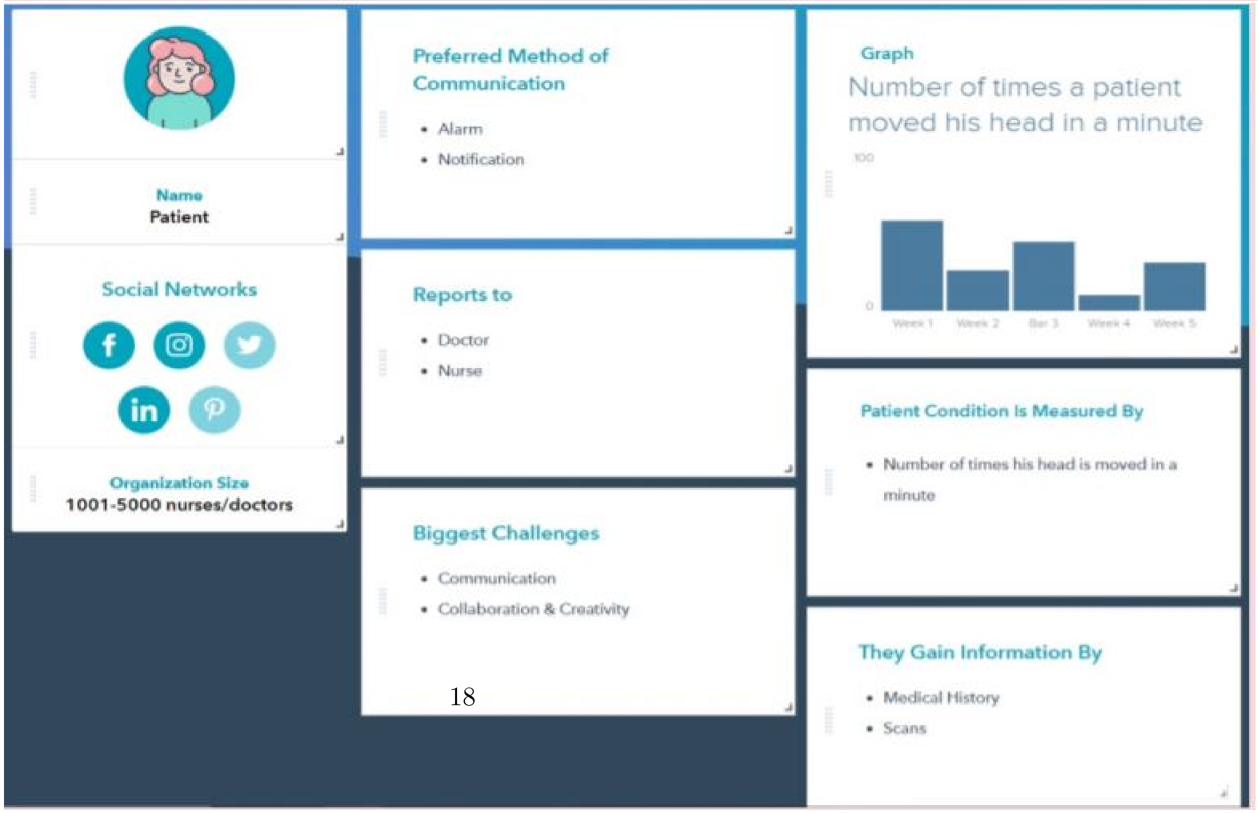
Number of Hours spent in front of laptop in a week

Tools They Need to Do Their Job

- Business Intelligence Dashboards
- Email
- CRM Software
- Cloud-Based Storage & File Sharing Applications

Biggest Challenges

- Navigating Client Relationships & Communications
- Employee Morale
- Collaboration & Creativity
- Communication



Name
Patient

Social Networks

Organization Size
1001-5000 nurses/doctors

Preferred Method of Communication

- Alarm
- Notification

Reports to

- Doctor
- Nurse

Biggest Challenges

- Communication
- Collaboration & Creativity

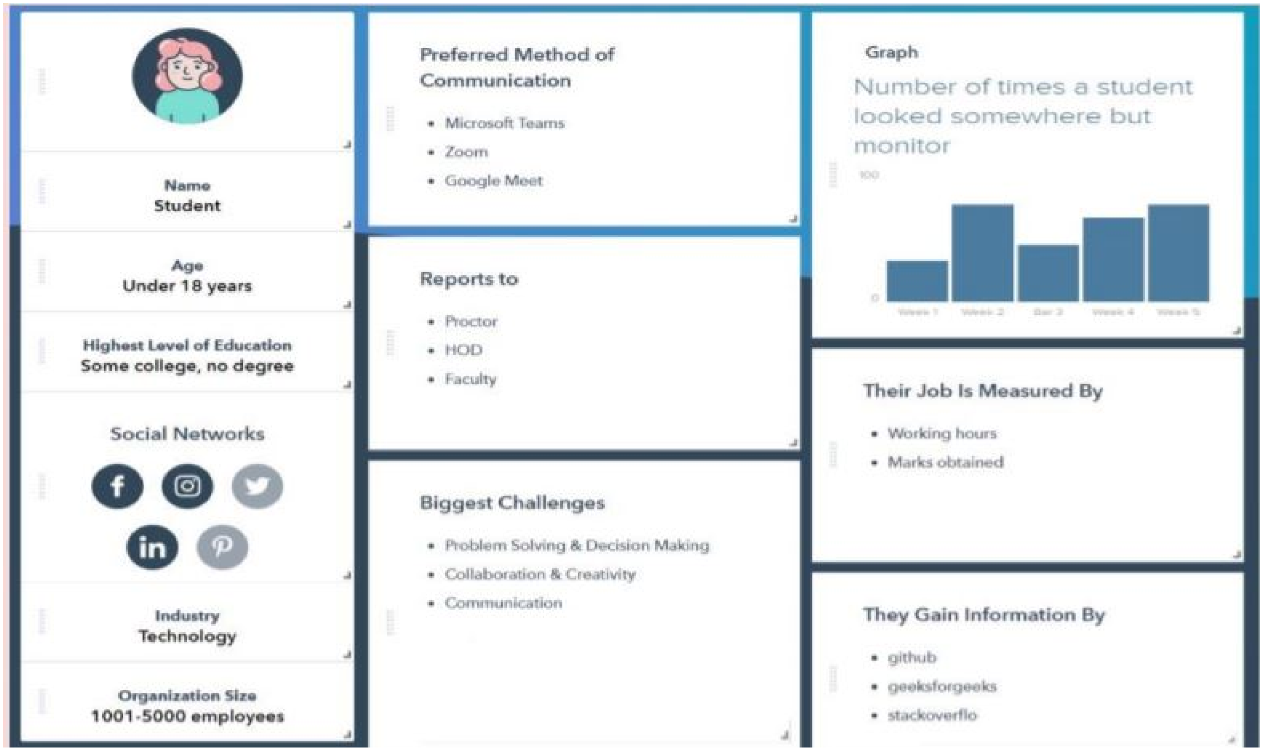
Graph
Number of times a patient moved his head in a minute

Patient Condition Is Measured By

- Number of times his head is moved in a minute

They Gain Information By

- Medical History
- Scans



Name
Student

Age
Under 18 years

Highest Level of Education
Some college, no degree

Social Networks

Industry
Technology

Organization Size
1001-5000 employees

Preferred Method of Communication

- Microsoft Teams
- Zoom
- Google Meet

Reports to

- Proctor
- HOD
- Faculty

Biggest Challenges

- Problem Solving & Decision Making
- Collaboration & Creativity
- Communication

Graph

Number of times a student looked somewhere but monitor

Their Job Is Measured By

- Working hours
- Marks obtained

They Gain Information By

- github
- geeksforgeeks
- stackoverflo

9 Results and Discussion

As far as our implementation and as per our knowledge we build a fully functioning neck monitoring system that helps many members and serves many needs. It includes the student, employees and patients. every one can use this software and we made different function like we constructed a modular code for easy understanding and better usage and finally the results that have been produced are very accurate to the pose of the head of the user. And the web interface that we had developed is also working perfectly and updating the times of the each position and the insight in the output window perfectly.

10 Conclusion

As we had developed a system that monitors the user head position and the the tilt angle it could be used to monitor the students and then the main advantage can be the doctor can look into patient log and can make suggestions out of them for the patient.And the patient can use this software to monitor himself during the stretches and let the doctor know about the logs and improvements through the software and as per the logs concern as it has the start time and end time if the session the doctor can easily identify any time differences in the stretching exercises and can help the patient better.

11 References

[1] David, E., Madec, S., Sadeghi-Tehran, P., Aasen, H., Zheng, B., Liu, S., ... Guo, W. (2020). Global Wheat Head Detection (GWHD) dataset: a large and diverse dataset of high-resolution RGB-labelled images to develop and benchmark wheat head detection methods. Plant Phenomics, 2020.

[2] Yao, C., Hu, J., Min, W., Deng, Z., Zou, S., Min, W. (2020). A novel real-time fall detection method based on head segmentation and convolutional neural network. Journal of Real-Time Image Processing, 17, 1939-1949.

[3] Moujahid, A., Dornaika, F., Arganda-Carreras, I., Reta, J. (2021). Efficient and compact face descriptor for driver drowsiness detection. Expert Systems with Applications, 168, 114334.

[4] Yolcu, G., Oztel, I., Kazan, S., Oz, C., Bunyak, F. (2020). Deep learning-based face analysis system for monitoring customer interest. Journal of Ambient Intelligence and Humanized Computing, 11(1), 237-248.

[5] Shao, X., Qiang, Z., Lin, H., Dong, Y., Wang, X. (2020, November). A survey of head pose estimation methods. In 2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics) (pp. 787-796). IEEE.

[6] Jang, H. K., Han, H., Yoon, S. W. (2020). Comprehensive monitoring of bad head and shoulder postures by wearable magnetic sensors and deep learning. IEEE Sensors Journal, 20(22), 13768-13775.

[7] Severin, I. C. (2020, October). Head Posture monitor based on 3 IMU sensors: Consideration toward Healthcare application. In 2020 International Conference on e-Health and Bioengineering (EHB) (pp. 1-4). IEEE.

[8] Severin, I. C. (2020, December). The Head Posture System Based on 3 Inertial Sensors and Machine Learning Models: Offline Analyze. In 2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI) (pp. 672- 676). IEEE.

[9] Yang, Y., Wang, C., Jia, H., Quan, L. (2020). Estimate of Head Posture Based on Coordinate Transformation with MP-MTMLSTM Network. International Journal of Pattern Recognition and Artificial Intelligence, 34(09), 2059031.

[10] Ansari, S., Du, H., Naghdy, F., Stirling, D. (2020, December). Unsupervised Patterns of Driver Mental Fatigue State Based on Head Posture Using Gaussian Mixture Model. In 2020 IEEE Symposium Series on Computational Intelligence (SSCI) (pp. 2699- 2704). IEEE.

[11] Gupta, R., Saini, D., Mishra, S. (2020, August). Posture detection using Deep Learning for Time Series Data. In 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT) (pp. 740-744). IEEE.