SMARTPHONE BASED VIRTUAL ASSISTANT

Thesis Report

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

Diploma in AI and ML

SUBMITTED BY

SATYAM GOVILA

40AIML526-21/1

UNDER THE SUPERVISION OF

MR. BRAHMA REDDY

UNIVERSITY OF HYDERABAD

CENTRE FOR DISTANCE AND VIRTUAL LEARNING
NAMPALLY STATION ROAD, ABIDS
HYDERABAD-500001

2022

UNIVERSITY OF HYDERABAD

Diploma in AI and ML

Smartphone Based Virtual Assistant

Approved by:
Brahma Reddy

Thesis submitted by:
Satyam Govila
40AIML526-21/1

UNIVERSITY OF HYDERABAD

Centre for Distance and Virtual Learning
Nampally Station Road, Abids
Hyderabad-500001

Declaration of Authorship

I, Satyam Govila, declare that this thesis titled, ‘Smartphone Based Virtual Assistant’ and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a Diploma degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

_____

Date:

_____

UNIVERSITY OF HYDERABAD

Centre for Distance and Virtual Learning

Certificate

This is to certify that the project entitled “Smartphone Based Gym Assistant” is a bonafide record of the work carried out by Mr. Satyam Govila (Enrolment No.40AIML526-21/1) under my supervision and guidance for the partial fulfillment of the requirements for the award of degree of Diploma in AI and ML during the academic session 2021-2022 in the Centre for Distance and Virtual Learning, UNIVERSITY OF HYDERABAD.

The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Date

Brahma Reddy

ACKNOWLEDGEMENT

*I would like to express my profound gratefulness to my supervisor **Mr. Brahma Reddy** for his guidance, support and constant encouragement. I am thankful to him for having faith in my ability and potential. His deep insight and broad knowledge provided me with valuable inputs and helped me to achieve this goal in a timely manner. I also highly appreciate the resources she provided with and that too conveniently.*

*My grateful appreciation also goes to **Srikanth Varma Chekuri** for his support and guidance throughout this project.*

I would like to thank my friends and family members who out of their comfort zone, helped me to achieve my goal in a stress-free manner.

Satyam Govila

ABSTRACT

Artificial Intelligence plays a major role in modern healthcare and fitness domain by improving and enhancing individual exercise habits, tracking health behaviours, and analysing repetitive exercise pattern and subsequently use the data to guide in fitness improvement. The thesis proposes, develops and evaluate a smart and effective AI solution based virtual gym assistant in real-time streaming video using CNN. The main objective of the proposed project is to understand and explore the art of 3D Human Pose Estimation and applying them to different aspects of the task of posture correction in exercises.

Over the time,Convolutional neural networks (CNN) algorithms have shown significant improvement in the area of human pose estimation on real-time dataset.

In this work, we have developed a production-ready application which not only helps people work out effectively at the comfort of their homes but also provides them with real-time feedback on their posture, and act as a personal virtual trainer that will help them to do their exercises in efficient manner.

Firstly, we have explored different algorithms and deep learning framework for 3D human pose estimation that could help detect different postures by representing human joints using key-points. These keypoints help in analysing the joint coordinates and calculating the angle between joints using mathematical formulation.

Secondly, we have use these joints coordinates on evaluating pose made by a person on pre-defined workouts such as biceps, leg-raise, squats and push-ups.

Thirdly, we have developed an interactive web based interface for machine learning application for the user to see a lot of useful information on dashboard. We have also added voice-based functionality in the application which uses speech recognition to ask and respond to user about pre-requisite information regarding workouts such as media type, type of exercise and other additional parameters. This feature enhances ease to use and user experience in using the fitness application

Finally, we have deployed and tested this application on different cloud architectures to evaluate the output latency ,cost involved and runtime issues in deployment. Therefore, this work considers several aspects of the human pose estimation problem and providing an end-to-end solution to human pose correction.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Motivation

Maintaining proper posture during workouts is one of the simple and yet significantly important thing to consider, but sometimes failing to maintain a good body posture can often lead to injuries like muscle strains. Mistakes in exercises are made when the person does not form a proper body pose and is not aware of correct form during exercises.

In these uncertain times of pandemic due to Covid-19, people had to avoid going to gym and stay indoors due to nationwide lockdown and hence exercising at home is an only way to maintain health and fitness.

Due to these factors, people often tend to perform exercise in improper manner due to lack of proper guidance from gym instructor which can also lead to injuries.

This project is an attempt to solve this problem by creating a smart and yet affordable AI solution which not only helps people work out effectively at the comfort of their homes but also provides them with real-time feedback on their posture, and act as a personal virtual trainer that will help them to do their exercises in efficient manner.

1.2 Proposed System : Smartphone Based Virtual Assistant

1. This system leverages the power of human pose estimation using Computer Vision and Deep Learning and aims to predict poses of individuals from videos and images data.

2. It take in account of human activity by extracting key points which represent different joints of human body and using them to evaluate postures.

3. It will help us in keeping track of repetitions , angle between joints and other important features related to different pre-defined exercises and postures.

4. The system will aim on providing instant real-time feedback on body form and alert user in case of any incorrect body posture.

5. The application also aims in maintaining low latency output and use best deployment practises , thus giving a new dimension to physical fitness and personal training.

6. The application also proposes different additional functionalities such as CLI access for development, voice based assistance for user-rich experience and exposed web API for HTML page demonstrating application dashboard .

7. There are numerous other integrations that can be made in this project where it can solve many health and fitness related real life problems, thus making an impact on the world and augmenting human capabilities in significant ways.

1.3 Human Pose Estimation

Human pose estimation represents a graphical skeleton of a human which helps us to analyse the activity of a human. The skeletons are basically a set of coordinates that describe the pose of a person.

Each joint is an individual coordinate that is known as a key point or pose-landmark and the connection between key points is known as a pair.
With pose estimation, we’re able to track humans’ motion and activity in real-world space. This opens up a wide range of application possibilities. It is a powerful technology that helps to effectively build complex applications.

1.4 Data Acquisition

In this project, we have used media files (images, videos) from different sources and webcam live feed to estimate the human pose and evaluate metrics. Most important libraries that are required will be OpenCV for computer vision processes , Mediapipe or PoseNet or Openpose ,and suitable cloud platform for deploying and productionisation of applications.

Data has been acquired from different sources listed as below :-

1. Real-time media (images and video files) data using webcam as a live feed is used extensively for this project for serving business requirements.

2. Coco Dataset (https://cocodataset.org/#home) : It is one of the most popular object detection dataset and is widely used to benchmark the performance of computer vision methods.

This dataset has 1.5 million object instances, 80 categories of objects, and around 250,000 people images.

The dataset is composed of image files and annotation files. The annotation file is a JSON containing all metadata about a single person.

The annotation JSON file comprises of different keys such as segmentation , number of key points , area of image , bounding boxes , image id and the category to which it belongs.

This dataset comes with a convenient library *pycocotools* which help us to explore and download the data and parse the annotation JSON file.

Example of annotation JSON data: “annotations”:

```
from coco_dataset import coco_dataset_download as cocod


required_object ='person'
sample_images = 50

annotations_path='/content/annotations/person_keypoints_train2017.json' #coco dataset annotations path


# cocod.coco_dataset_download(required_yobject ,sample_images , annotations_path)
```

```
# https://github.com/cocodataset/cocoapi/blob/master/PythonAPI/pycocotools/coco.py


from pycocotools.coco import COCO
coco = COCO(annotations_path)



# get category id
cat_id = coco.getCatIds(catNms = ['person'])

# get annotation ids
ann_ids = coco.getAnnIds(catIds = cat_id, iscrowd = 0)
all_ann = coco.loadAnns(ann_ids)
```

```
loading annotations into memory...
Done (t=9.03s)
creating index...
index created!
```

```
[
{ "segmentation": [ [

125.12,

539.69,

140.94, .

.

.

532.49 ]

],
“num_keypoints": 10,
"area": 47803.27955,
 "iscrowd": 0,
"keypoints": [… , 0, 0, 0, 0,142,309 , 199 , 2, …] ,
"image_id": 425226,
"bbox":[73.26 , 209.3 , 322.6 ,372.5 ] ,

"category_id": 1,
"id": 183126
},
```

FIGURE 1.1: COCO DATASET ANNOTATION: CODE AND SAMPLE DATA

3. MPII Human Pose Dataset (https://human-pose.mpi-inf.mpg.de/)

This is a state-of-the-art dataset and is used extensively for evaluation of articulated human pose estimation. It has around 25K images containing over 40K people with annotated body joints.

4. Sample videos of some of the popular Gym channels can be downloaded fromYouTube.

Example :
https://www.youtube.com/c/fitnessblender/videos

This dataset can have a wide variety of human activities and for this Project, categories containing home exercises can be used.
This can be really helpful to get information regarding proper exercise poses, correct postures , appropriate number of reps for a particular exercise and other fitness related expert advice.

1.5 Challenges and Constraints

1.5.1 Expected challenges in the dataset related to this project

I. Video data acquired through different sources should be of appropriate fps rate and ensure front view of the user so as to label all the key points correctly.
II. Adequate research in defining apt angles between joints in limbs and other body parts , and proper exercise posture should be done so that our virtual assistant gives proper advice to users.
III. The feedback given to users on different aspects should be in real-time without any latency.

1.5.2 Challenges and constraints in Computer Vision based Virtual Gym-Assistant

I. This system focuses on providing real-time feedback on exercise poses, count reps , and estimate fitness related parameters. It also aims to provide low latency output, computation at 60fps rate , easy deployment in Smartphones/web and ability to seamlessly integrate more exercise forms as per users requirements.

II. Since we are designing a virtual gym assistant that replaces a human personal trainer , we have to keep observing if it will show the same exercise estimation capabilities as humans.

III. Feedback for Human Pose correction in exercises should be highly accurate as error or inaccurate results can lead to serious health and fitness effects. Hence, we should continuously deploy new features in the system to improve its performance and additional exercises pose as per the user requirements.

IV. Men and Women have physiological differences and hence in some exercises body postures / poses have to be different. Therefore ,while developing this system we have to keep in consideration that our output feedback should depend on whether it is a male or a female.

V. Accuracy level can differ depending on the 3D position and scale in which the user is facing the camera and how far is he/she away from the camera. Ideally, a person facing the front side of the camera and at an appropriate distance gives optimal results.

VI. Output latency should be low thereby also adjusting fps rate at an optimum level.

Chapter 2

Literature Review

❖ Title of Research Paper :
“DeepPose: Human Pose Estimation via Deep Neural Networks”

Authors : Alexander Toshev and Christian Szegedy
Source : https://arxiv.org/pdf/1312.4659.pdf

- This paper proposes a method for human pose estimation based on Deep Neural Networks (DNNs).
- It is formulated as a DNN-based regression problem towards body joints.
- The location of each body joint is regressed using an input full image and a 7-layered generic convolutional DNN.
- The DNN network consists of a pooling layer, a convolution layer, and a fully-connected layer , where only convolution and fully connected layer has learnable parameters.
- It also proposes a cascade of DNN-based pose predictors which allows increased precision of joint localisation.
- This approach helps to achieve state of the art results on standard benchmarks such as the MPII, LSP, and FLIC datasets.
- Metrics used for evaluation are - Percentage of Correct Parts (PCP) and Percent of Detected Joints (PDJ).

❖ Title of Research Paper :
“Efficient Object Localization Using Convolutional Networks (2015)”

Authors : Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, Christoph Bregler
Source : https://arxiv.org/pdf/1411.4280.pdf

- In this paper , state-of-the-art performance on human-body pose estimation has been achieved with the help of Deep Convolutional Networks (ConvNets)
- This architecture helps to predict the location of human joints in monocular RGB images ● The model proposed has increased pooling layers which helps to improve computational efficiency.
- The paper also introduces a network that uses the features from the hidden layer from the heat map regression model in order to increase localization accuracy.

- Model performance is also increased with the addition of spatial dropout layer by preventing activations from becoming strongly correlated.
- The architecture is implemented with Torch7 and evaluated using FLIC and MPII-Human-Pose datasets.
- Metrics used for evaluation here is - Percentage of Correct Keypoints (PCK).

❖ Title of Research Paper :

“OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields”

Authors : Zhe Cao, Student Member, IEEE, Gines Hidalgo, Student Member, IEEE, Tomas Simon, Shih-En Wei, and Yaser Sheikh

Source : https://arxiv.org/pdf/1812.08008.pdf

- This paper presents an efficient method for multi-person pose estimation with competitive performance on multiple public benchmarks.
- It provides a real-time approach for detecting 2D human poses in images and videos.
- The model considers the bottom- up representation of association scores via Part Affinity Fields (PAFs), which is basically a set of 2D vector fields that encode the location and orientation of limbs over the image domain.
- The paper documents the release of OpenPose which is an open-source real-time system for multi-person 2D pose detection, including body, foot, hand, and facial key points.
- It also presents an annotated foot dataset with 15K human foot instances and demonstrates that a combined model with body and foot key points can be trained preserving the speed of the body-only model while maintaining its accuracy .
- Evaluation of multi-person pose estimation has been done on MPII human multi-person dataset , Coco key point challenge dataset and foot dataset(subset of Coco Dataset).

Chapter 3

System Development

3.1 Approach to Solve ML Problem

1. This problem can leverage the power of computer vision and deep learning to read the data source, process and evaluate it , and output the results to the user in real time.

2. OpenCV is a huge open-source library which will be used to process images and videos to identify objects, faces, and is used extensively in computer vision, machine learning, and image processing related applications.

3. Pose estimation in fitness applications is challenging due to the variety of possible poses , different exercise forms , numerous degrees of freedom , and single or multi-person detection.

4. Using an appropriate pose detection framework such as OpenPose , MediaPipe , PoseNet, etc, the model will identify key points which is basically a set of coordinates for each human joint (arm, head, torso, etc.,) -describing the human pose. While OpenPose and PoseNet are able to support real-time multi-person pose estimations, MediaPipe is only able to support single person pose estimation.

5. Considering all factors, we will harness the power of Blaze Pose (Mediapipe pose model framework), a fairly newly developed algorithm which allows us to infer 33 different 2D landmark key points of the human body in a single frame and hence it will help us to localise more body movements. It helps to give real-time feedback and subsequently run different machine learning models to provide state-of-the-art performance.

6. Mediapipe has an additional functionality to measure and indicate the probability of a given landmark within the image frame. The score has a range of 0.0 to 1.0, where 1.0 indicating the highest confidence level for a landmark point . Pose Detection using MediaPipe is, however ,limited in detecting single person only and not multi-person but this doesn’t have any disadvantage for our application.

7. The next step involves determining and assessing the joint coordinates and using mathematical computation to calculate the different joint angles in human pose.

8. At last, these angles are used to evaluate and analyse different body postures and provide real time feedback to users about this information.

There are three types of approaches to model the human body:

· Skeleton-based model

· Contour-based model

· Volume-based model

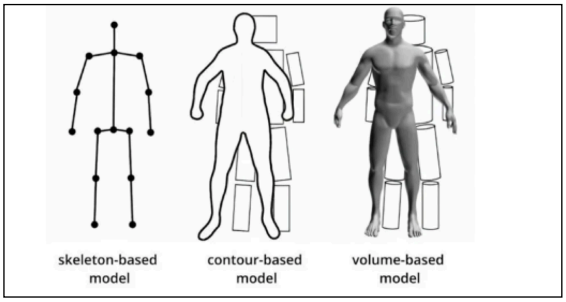


FIGURE 3.1: TYPES OF APPROACH TO MODEL THE HUMAN BODY

Here is a glimpse of 33 key points detected in human pose with the help of MediaPipe Pose Landmark feature. The output is a list of pose landmarks, and each landmark consists of x and y landmark coordinates normalised to [0.0, 1.0] by the image width and height respectively.

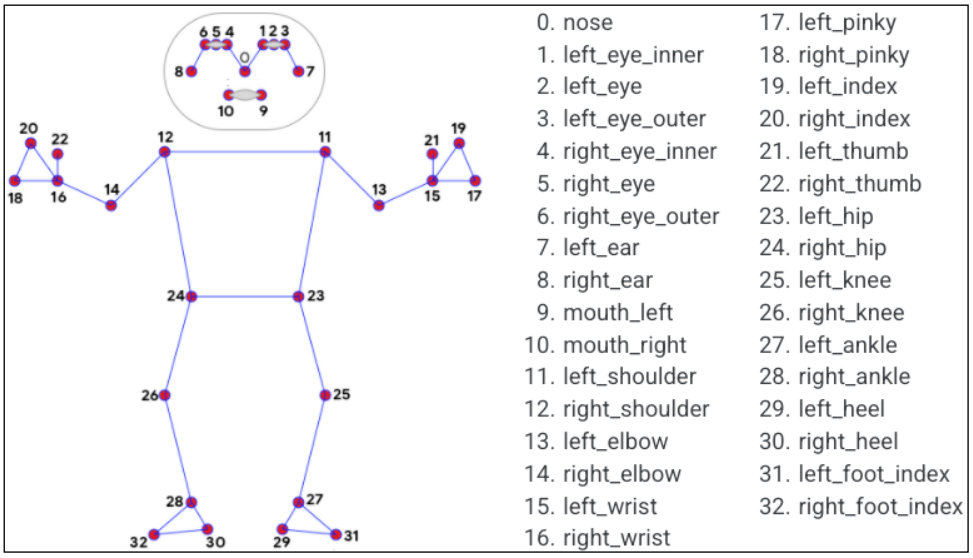


FIGURE 3.2: MEDIAPOSE LANDMARK POINTS
SOURCE: HTTPS://GOOGLE.GITHUB.IO/MEDIAPI

3.3 System Architecture

This section presents a comprehensive architectural overview of our system using a number of different modular views to depict and explain different aspects of the system.

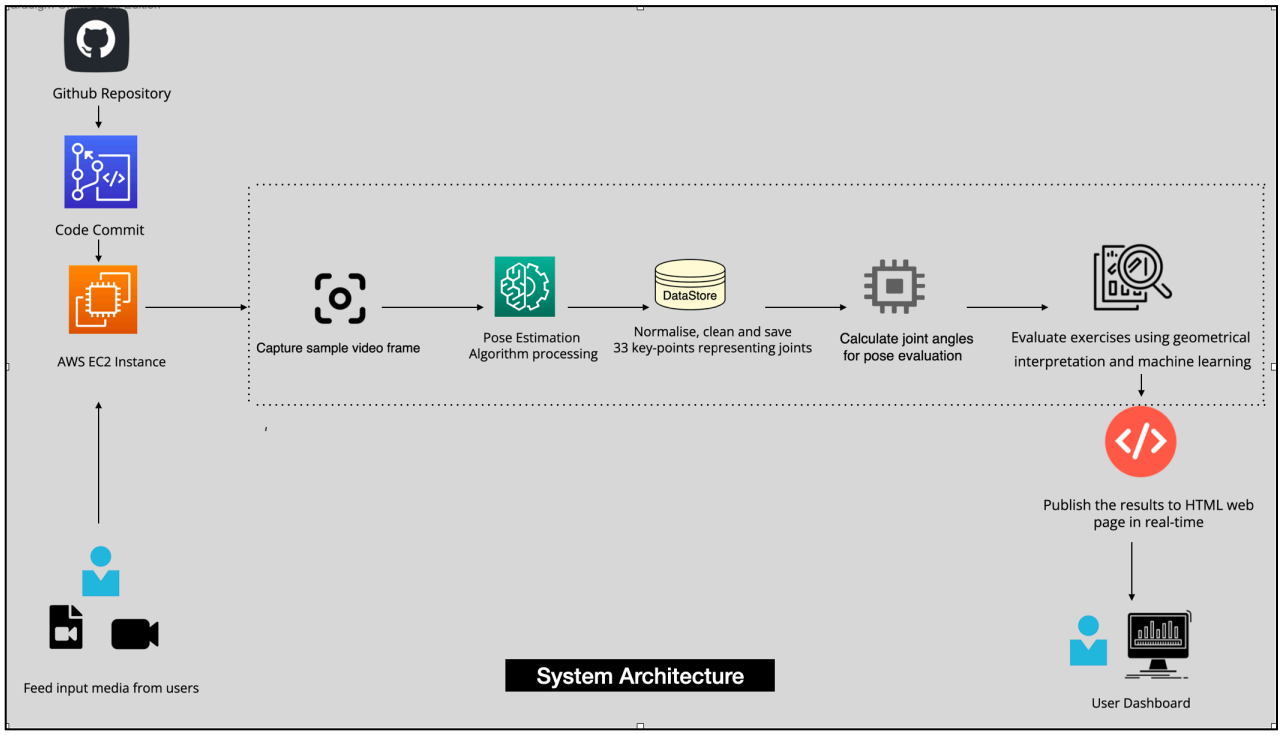


FIGURE 3.3 : SYSTEM ARCHITECTURE DIAGRAM

For analysis of the proposed architecture, we have found it convenient to consider the following components of our smartphone based gym application using machine learning:-

1. Real-time media such as images or videos from webcam or media stored in working directory is given as input feed to the application using OpenCV.

2. The application also takes other important details such as gender and type of exercise from the user. This information is important to correctly analyse the exercise according to the user's physiology and run pose estimation algorithm according to the exercise form.

3. OpenCV performs the video processing by framing i.e. taking a single instance of frames at a given time.The default colour format in OpenCV is often referred to as RGB but it is actually BGR (the bytes are reversed), and hence we perform appropriate transformation of colours to pixels before analysing the frame.

4. Then, we set up the Mediapipe instance to perform pose detection on our captured frame of human body pose. Mediapipe provides state of the art solution for high quality and low latency real-time video feeds.
5. The detector in the Blazepose model is run on the captured frame to localise a single person and provide a Region of Interest (ROI) bounding box around it and further predicts the 33 landmark points inside the box also stating the confidence value.

```
# Create DataFrame
df = pd.DataFrame(data)
df = df.style.set_caption('List of all landmark points and their respective coordinates for a given frame')

df
```

Out[12]:

List of all landmark points and their respective coordinates for a given frame

	landmark_names	x_coordinates	y_coordinates	z_coordinates	visibility_values
0	NOSE	0.44714	0.249099	-0.0676461	0.999831
1	LEFT_EYE_INNER	0.450309	0.221887	-0.107201	0.999891
2	LEFT_EYE	0.452928	0.219697	-0.107305	0.999906
3	LEFT_EYE_OUTER	0.455818	0.217383	-0.107495	0.999888
4	RIGHT_EYE_INNER	0.448415	0.222789	-0.0548563	0.999861
5	RIGHT_EYE	0.449475	0.221526	-0.0547598	0.999851
6	RIGHT_EYE_OUTER	0.450703	0.220111	-0.0547987	0.999841
7	LEFT_EAR	0.479505	0.207954	-0.202265	0.999962
8	RIGHT_EAR	0.471723	0.211543	0.0278514	0.999891
9	MOUTH_LEFT	0.462571	0.26457	-0.0966661	0.999866
10	MOUTH_RIGHT	0.460079	0.265287	-0.0299983	0.999761
11	LEFT_SHOULDER	0.521434	0.359015	-0.288062	0.99969
12	RIGHT_SHOULDER	0.50596	0.337774	0.169888	0.999326
13	LEFT_ELBOW	0.490877	0.569228	-0.210941	0.956941
14	RIGHT_ELBOW	0.492001	0.537636	0.0577586	0.270902
15	LEFT_WRIST	0.403792	0.476128	0.0613326	0.89078
16	RIGHT_WRIST	0.429504	0.442168	-0.186142	0.484888
17	LEFT_PINKY	0.381227	0.459781	0.0637302	0.83478
18	RIGHT_PINKY	0.415309	0.413767	-0.206452	0.457491
19	LEFT_INDEX	0.390569	0.430759	0.077079	0.839324
20	RIGHT_INDEX	0.413795	0.393942	-0.206798	0.438484
21	LEFT_THUMB	0.399426	0.436467	0.0783718	0.812506
22	RIGHT_THUMB	0.420495	0.399739	-0.195915	0.431996
23	LEFT_HIP	0.525413	0.746097	-0.127915	0.988428
24	RIGHT_HIP	0.497773	0.733775	0.128119	0.991752
25	LEFT_KNEE	0.507311	0.986246	-0.0904034	0.661012
26	RIGHT_KNEE	0.481952	0.963797	0.222485	0.24126
27	LEFT_ANKLE	0.503385	1.28965	0.00518739	0.252216
28	RIGHT_ANKLE	0.482704	1.23779	0.372384	0.0783406
29	LEFT_HEEL	0.515398	1.3299	0.0105907	0.425832
30	RIGHT_HEEL	0.496776	1.27571	0.383018	0.123544
31	LEFT_FOOT_INDEX	0.461946	1.35208	-0.0667203	0.23624
32	RIGHT_FOOT_INDEX	0.430417	1.29135	0.32917	0.0785742

FIGURE 3.4: LANDMARK DATAFRAME COORDINATES

6. In Pose Detection model, we initialise the pose class with the suitable arguments such as:

- min_detection_confidence : It is the minimum detection confidence with range 0.0 - 1.0 , and it states if the prediction regarding a person's detection is correct. The default threshold value is set at 0.5 which means that a detection is considered as positive if the value is greater than threshold value.
- min_tracking_confidence – It is the minimum tracking confidence with range 0.0 - 1.0 and is required to consider if the predicted landmark on the human body is valid.Increasing this value helps increasing the robustness, but also increases the latency of output. There are also other arguments such as model_complexity and smooth_landmarks which can be adjusted as per the desired performance for the model and output latency.

7. As a result of pose detection, we obtain a set of 33 key points which demonstrates the joint locations along with 3-D coordinates with visibility range.

8. These key points can also be drawn on the frame using the Mediapipe class function for clear demonstration of keypoints.

9. These landmark coordinates are useful in calculating angles between joints using *calculate_angle( )* function imported from evaluate module. The function takes in the joint coordinates as input and returns the angle in degrees.

```
angle_degrees = evaluate.calculate_angle(
                        left_shoulder_coordinates,
                        left_elbow_coordinates,
                        left_wrist_coordinates
                        )
```

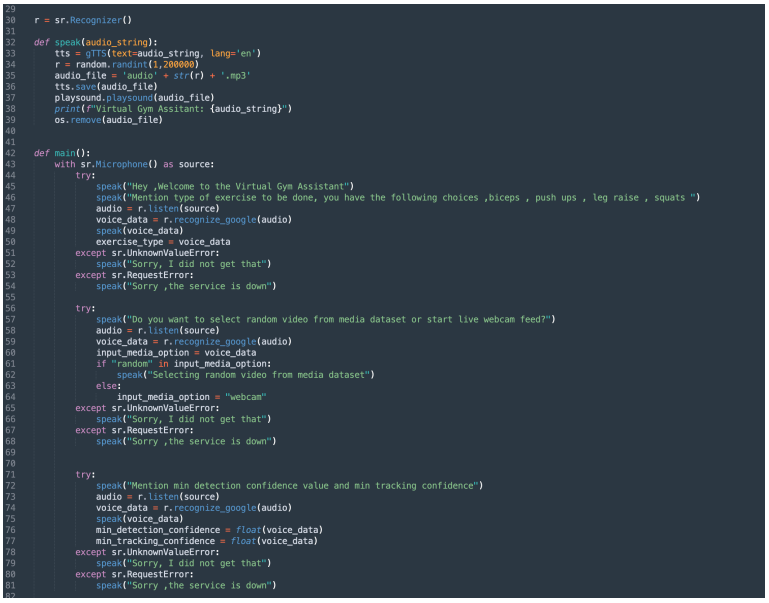
FIGURE 3.5 CODE SNIPPET - EVALUATE ANGLE

10. These joint angles are further used to count reps in exercises, evaluate the exercise form and provide the correctness of exercise done by the user, using custom pre-defined cases. These cases vary and are based on gender of user, exercise performed and the joint angle values in different exercises.
11. For instance , in bicep exercise , we have defined a use case where the application keeps track of the limb position (i.e. up or down) and the counter to count the reps depending on the position change.
12. The feedback from the application is provided to the user in real-time on the screen itself so that he/she can adjust their pose accordingly and keep count of the calculated reps.
13. The application is integrated with Streamlit which helps us to provide the web-based interface and this web application is deployed on cloud environment.

Voice-Based Functionality

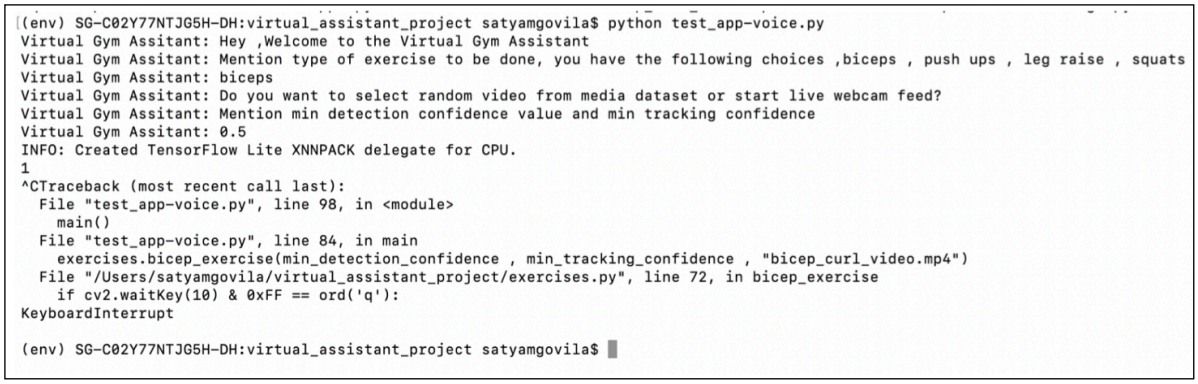
The application is integrated with voice based assistance which helps in easy to use functionality and hands free solution. It also helps to create an improved customer engagement and host subsidiary benefits such as increased brand awareness and better consumer perception.
The application in turn helps to increase productivity using speech-to-text in real-time using speech recognition and gTTS libraries in Python.

The application asks multiple inputs such as information about the exercise type, feed input time-media from database or webcam , confidence interval value, etc. and in turn gives personalised exercise results based on the received inputs.



```
r = sr.Recognizer()

def speak(audio_string):
    tts = gTTS(text=audio_string, lang='en')
    r = random.randint(1,200000)
    audio_file = 'audio' + str(r) + '.mp3'
    tts.save(audio_file)
    playsound.playsound(audio_file)
    print(f"Virtual Gym Assitant: {audio_string}")
    os.remove(audio_file)


def main():
    with sr.Microphone() as source:
        try:
            speak("Hey ,Welcome to the Virtual Gym Assistant")
            speak("Mention type of exercise to be done, you have the following choices ,biceps , push ups , leg raise , squats ")
            audio = r.listen(source)
            voice_data = r.recognize_google(audio)
            speak(voice_data)
            exercise_type = voice_data
        except sr.UnknownValueError:
            speak("Sorry, I did not get that")
        except sr.RequestError:
            speak("Sorry ,the service is down")

        try:
            speak("Do you want to select random video from media dataset or start live webcam feed?")
            audio = r.listen(source)
            voice_data = r.recognize_google(audio)
            input_media_option = voice_data
            if "random" in input_media_option:
                speak("Selecting random video from media dataset")
            else:
                input_media_option = "webcam"
        except sr.UnknownValueError:
            speak("Sorry, I did not get that")
        except sr.RequestError:
            speak("Sorry ,the service is down")


        try:
            speak("Mention min detection confidence value and min tracking confidence")
            audio = r.listen(source)
            voice_data = r.recognize_google(audio)
            speak(voice_data)
            min_detection_confidence = float(voice_data)
            min_tracking_confidence = float(voice_data)
        except sr.UnknownValueError:
            speak("Sorry, I did not get that")
        except sr.RequestError:
            speak("Sorry ,the service is down")
```

FIGURE 3.6 : CODE SNIPPET -VOICE RECOGNITION FEATURE



```
(env) SG-C02Y77NTJG5H-DH:virtual_assistant_project satyamgovila$ python test_app-voice.py
Virtual Gym Assitant: Hey ,Welcome to the Virtual Gym Assistant
Virtual Gym Assitant: Mention type of exercise to be done, you have the following choices ,biceps , push ups , leg raise , squats
Virtual Gym Assitant: biceps
Virtual Gym Assitant: Do you want to select random video from media dataset or start live webcam feed?
Virtual Gym Assitant: Mention min detection confidence value and min tracking confidence
Virtual Gym Assitant: 0.5
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
1
^CTraceback (most recent call last):
  File "test_app-voice.py", line 98, in <module>
    main()
  File "test_app-voice.py", line 84, in main
    exercises.bicep_exercise(min_detection_confidence , min_tracking_confidence , "bicep_curl_video.mp4")
  File "/Users/satyamgovila/virtual_assistant_project/exercises.py", line 72, in bicep_exercise
    if cv2.waitKey(10) & 0xFF == ord('q'):
KeyboardInterrupt

(env) SG-C02Y77NTJG5H-DH:virtual_assistant_project satyamgovila$
```

FIGURE 3.7 : WORKING DEMO OF VOICE RECOGNITION FEATURE

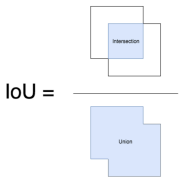
Chapter 4

Blazepose Model Network Architecture

4.1 Network Architecture

1. Blazepose is a lightweight convolutional neural network architecture for human pose estimation that is tailored for real-time inference. It is a pose detection model that can compute (x, y, z) co-ordinates of 33 skeleton keypoints and can be used extensively in fitness applications.

2. For most of the times, majority of modern object detection solutions have relied on the Non-Maximum Suppression (NMS) algorithm for their last post-processing step, which works well for rigid objects with few degrees of freedom.But this algorithm fails for certain scenarios that include highly articulated human poses because of multiple, ambiguous boxes that satisfy the intersection over union (IoU) threshold for the NMS algorithm.



3. To overcome the above said limitation, the new topology focusses on detecting the bounding box of a relatively rigid body part like the human face or torso, as it is the strongest signal to a neural network.Hence in order to make person detection fast and light-weight, the architecture takes in the assumption that the head of the person should always be visible for our single-person use case.

4. The Blazepose model consists of two machine learning models: a *Detector* and an *Estimator.* The *Detector* cuts out the human region from the input image, while the Estimator takes a 256x256 resolution image of the detected person as input and outputs the keypoints.

5. The *Detector* is a Single-Shot Detector(SSD) based architecture. Given an input image (1,224,224,3), it outputs a bounding box (1,2254,12) and a confidence score (1,2254,1).

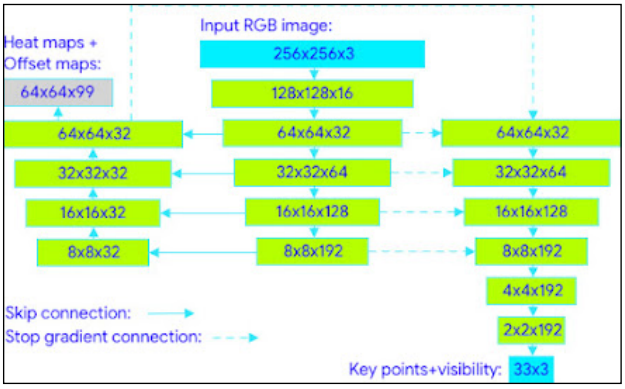


FIGURE 4.1 :NETWORK ARCHITECTURE-REGRESSION WITH HEATMAP SUPERVISION

6. The *Estimator* uses heatmap for training, but computes keypoints directly without using heatmap for faster inference.

7. The first output of the *Estimator* is (1,195) landmarks , the second output is (1,1) flags. The landmarks are made of 165 elements for the (*x ,y ,z , visibility, presence*) for every 33 keypoints.

8. The *visibility* and *presence* are stored in the range of [*min_float,max_float*] and are converted to probability by applying a sigmoid function. The *visibility* returns the probablity of keypoints that exist in the frame and are not occluded by other objects.

9. The model is trained over large dataset and as an evaluation metric,Percent of Correct Key-Points with 20% tolerance (PCK@0.2) is used. PCK is used as an accuracy metric that measures if the predicted key point and the true joint are within a certain distance threshold. The PCK is usually set with respect to the scale of the subject, which is enclosed within the bounding box. PCK@0.2 implied that the distance between predicted and true joint < 0.2 * torso diameter

Chapter 5

Implementation of Exercises Module in System

5.1 Bicep Curl Exercise

● The biceps curl is a highly recognisable and simple weight-training exercise that works the muscles of the upper arm, and is a great exercise for seeing results in strength and definition.

● This exercise is implemented in the project where the user stands in front of the camera at an appropriate distance with good lightning conditions.

● The model identifies the landmark key points of shoulder, elbow and wrist in real-time and uses mathematical formulation to calculate the angle between the joints in real-time.

● It also helps in keeping the count of reps, and current stage using the geometric interpretation from the angle feature being calculated.

● The application displays the stage of exercise, count of curls made and the angle between the joints on the user's screen which helps the user to observe and correct its form and also keep track of reps being made.

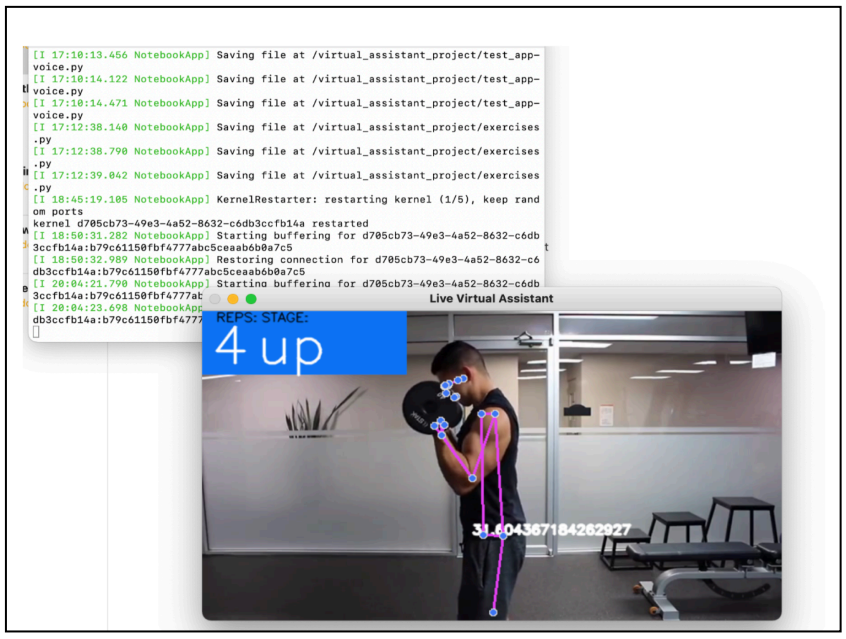


FIGURE 5.1 : WORKING DEMO OF BICEPS EXERCISE

5.2 Push-Ups Exercise

- Push ups are a great form of exercise that are beneficial to develop strength and muscles in the upper body but at the same time a person should be aware of safety measures that have to be taken care of while performing this exercise such as keeping appropriate angle between joints, taking proper rest and doing reasonable reps.

- This is where the gym assistant app can be of great advantage where it can help to detect the correct pose and keep track of the above mentioned factors.

- The model requires the landmark key points such as shoulder, elbow and wrist and calculates the angle which can be further required to perform exercise analysis as demonstrated above.

- The application also notifies the user if the angle between the joints or the reps being made is not appropriate in real-time. As a part of future work, the application will take account of voice based assistance to inform the user about the pose correction.

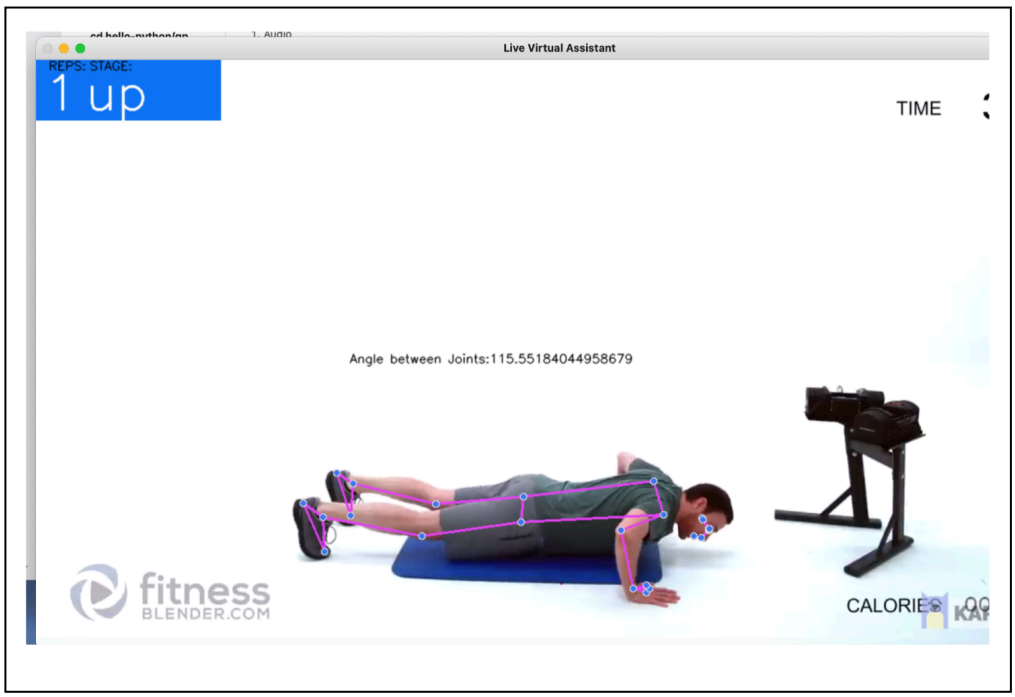


FIGURE 5.2 : WORKING DEMO OF PUSH-UPS EXERCISE

5.3 Leg Raise Exercise

- Leg raise is a challenging exercise which is designed to target the lower and upper abdominal muscles, and build strength in this region. They are proven to be a superlative workout which can help to improve flexibility across the back extensors and hips flexors.

- The application expects the user to lie down on back and lift both legs at the same time and lift them up at an appropriate angle.

- The model requires the landmark key points such as shoulder, knee and hip and calculates the angle between them to extract angle features and further perform exercise analysis as shown below in real-time.

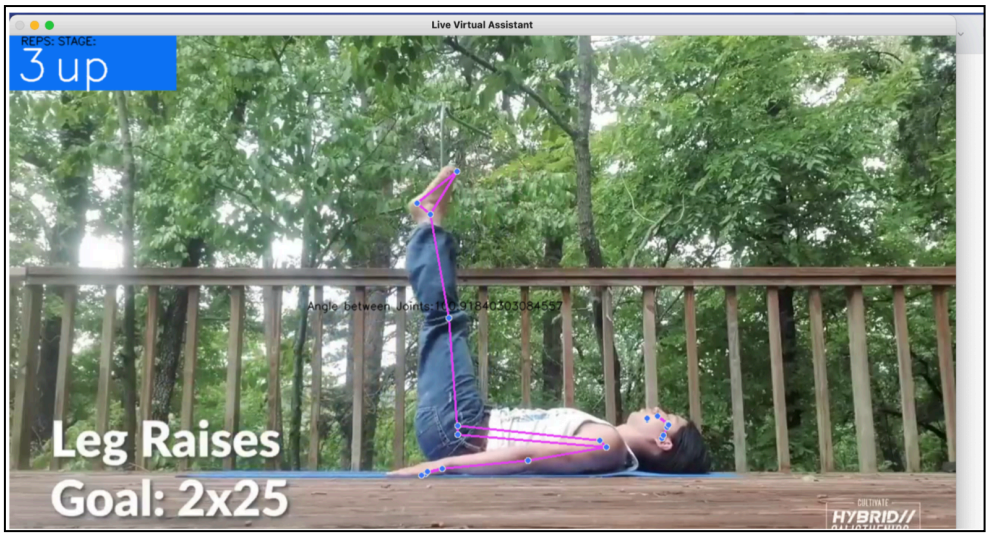


FIGURE 5.3 : WORKING DEMO OF LEG-RAISE EXERCISE

5.4 Squats Exercise

- The squat is considered a compound movement, which means it works for multiple muscle groups across multiple joints. The primary muscles involved in the movement are the quadriceps and it also work on the muscles around the knee, which helps build strength and prevent injury.

- The squat exercise expects the user to maintain a proper position and keep a shoulder-width distance between the feet.

- The model extracts the landmark key points such as hip, knee and angle and calculates the angle between them which helps in analysis and also informs the user about up-down movement.

- The application also keeps track of the reps, stage and the pose movement in real-time and display on screen.

FIGURE 5.4 : WORKING DEMO OF SQUATS EXERCISE

Chapter 6

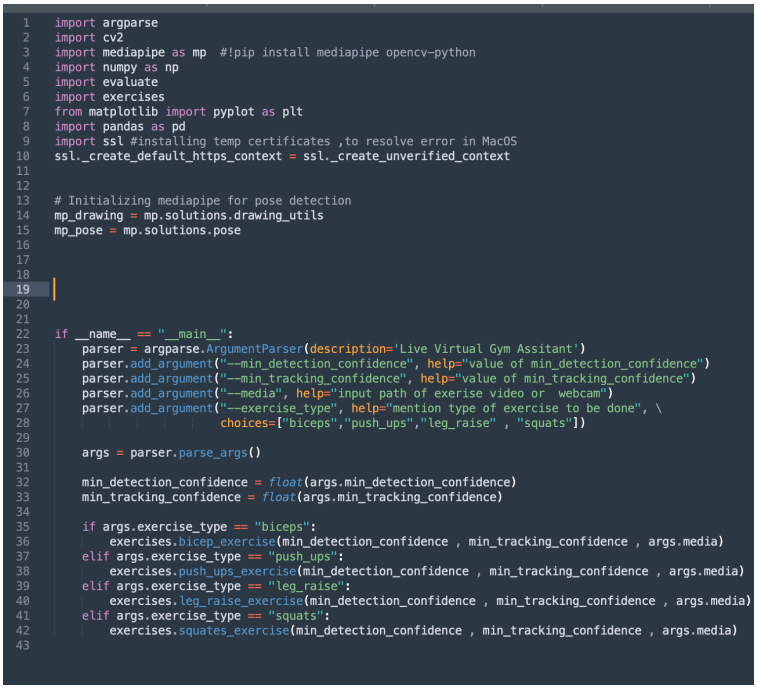
Deployment and Productionisation

6.1 Productionisation of Application

ML model deployment and productionisation actually refers to putting the model into a production environment like on a web-interface and making the model use for general users.It also involves getting and analysing feedback from users and continuously integrating new features for model improvement.

Deployment of the ML model is an integral part of machine learning based applications where we integrate the code into a production ready environment for practical business use cases.

In this project, for the deployment phase, few discrepancies were often observed between the code in which a machine learning model is written and the code which is actually required and hence re-coding of the model and altering the initial assumption is taken into account.



```
import argparse
import cv2
import mediapipe as mp  #!pip install mediapipe opencv-python
import numpy as np
import evaluate
import exercises
from matplotlib import pyplot as plt
import pandas as pd
import ssl #installing temp certificates ,to resolve error in MacOS
ssl._create_default_https_context = ssl._create_unverified_context


# Initializing mediapipe for pose detection
mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose




if __name__ == "__main__":
    parser = argparse.ArgumentParser(description='Live Virtual Gym Assitant')
    parser.add_argument("--min_detection_confidence", help="value of min_detection_confidence")
    parser.add_argument("--min_tracking_confidence", help="value of min_tracking_confidence")
    parser.add_argument("--media", help="input path of exerise video or  webcam")
    parser.add_argument("--exercise_type", help="mention type of exercise to be done", \
                        choices=["biceps","push_ups","leg_raise" , "squats"])

    args = parser.parse_args()

    min_detection_confidence = float(args.min_detection_confidence)
    min_tracking_confidence = float(args.min_tracking_confidence)

    if args.exercise_type == "biceps":
        exercises.bicep_exercise(min_detection_confidence , min_tracking_confidence , args.media)
    elif args.exercise_type == "push_ups":
        exercises.push_ups_exercise(min_detection_confidence , min_tracking_confidence , args.media)
    elif args.exercise_type == "leg_raise":
        exercises.leg_raise_exercise(min_detection_confidence , min_tracking_confidence , args.media)
    elif args.exercise_type == "squats":
        exercises.squates_exercise(min_detection_confidence , min_tracking_confidence , args.media)

```

FIGURE 6.1 : CODE SNIPPET -APP RUN CLI MODE

6.2 Building Web Interface for Virtual Gym Assistant Application

In this project, a static web application is developed using Streamlit, which is an open-source python framework for building web apps. It helps in creating an interactive dashboard using Streamlit API in Python.

The advantage of using Streamlit are as follows-

● Easy to use as less refactoring of former code is required to create a web app with proper documentation of API usage.

● Absolutely no callbacks needed since widgets are treated as variables.

● It also comes with data caching functionality which helps in simplification and speeding up the computation pipeline.

Here's the screenshot of the working demo of web application-

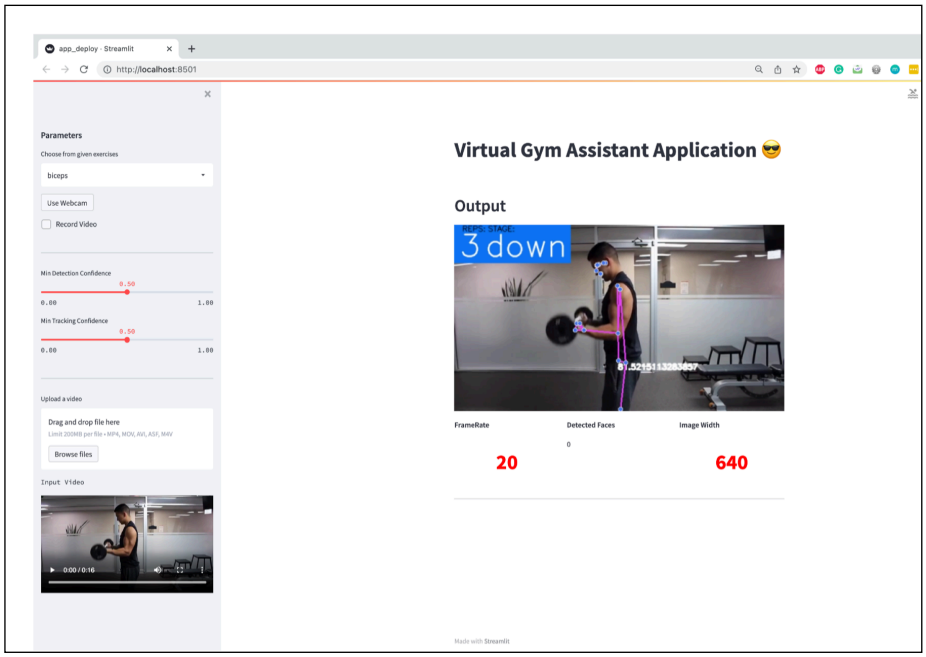


FIGURE 6.2 : WORKING DEMO OF STREAMLIT WEB APPLICATION

In this dashboard, several widgets have been integrated so as to increase user interactivity and experience such as -

● Buttons
● Sliders for changing parameters of detecting and tracking confidence
● Media uploading button
● Webcam check-box
● Live output feed window which shows real-time tracking of exercises, counter, human body key-point feature and angles between joints.
● Additional data about video such as video dimensions and frame rate.

The web application shown above is deployed on localhost and works seamlessly without any latency issues and outputs live feed in real-time.

6.3 Deploying Web Application in Cloud Environment

For productionising the web application, we primarily have two options :

- IaaS (Infrastructure as a Service)
- PaaS (platform as a service)

6.3.1 Heroku Platform

In this project, in order to host web application on a cloud based platform, Heroku is used. Heroku is a container-based cloud Platform as a Service (PaaS) which is extensively used to deploy, manage, and scale applications with a great support of multiple programming languages such as Java, Node.js, Clojure, Python, PHP, and Go.

As we can see in the images below, Heroku offers a decent dashboard with a complete set of features to observe activity, deployment , metrics and billing details.

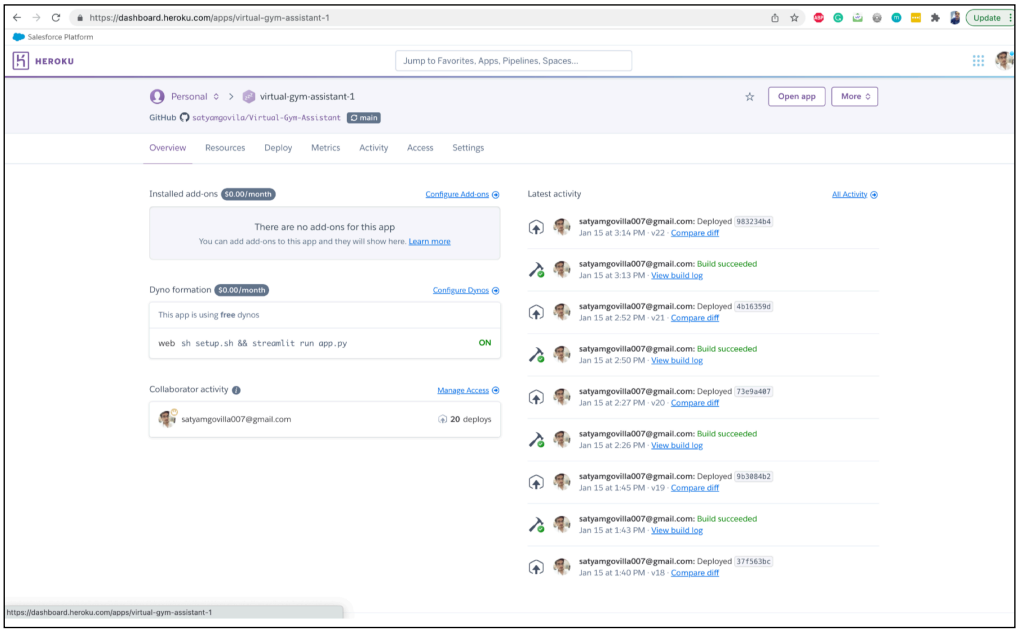


FIGURE 6.3 : HEROKU DASHBOARD

Streamlit web application is deployed on Heroku for test purpose as we can see the screenshot of the working demo below.

The steps involve pushing all the code, media, requirements file to Git, integrating Git with Heroku platform, creating a domain for web app and deploying the application.

The web application can be accessed from the following domain provided by Heroku- https://virtual-gym-assistant-1.herokuapp.com/

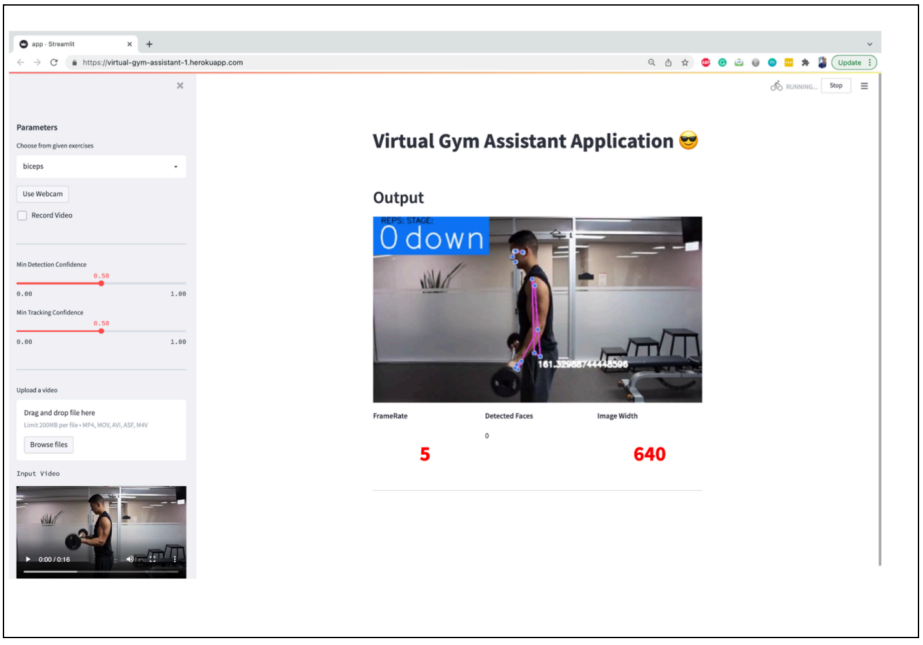


FIGURE 6.4 : WORKING DEMO OF DEPLOYMENT ON HEROKU

Advantages of using Heroku-

● Free tier version of Heroku to initially test and deploy basic web applications is available. This is very convenient for users to test Heroku features and to figure out if it is suitable for application.

● Ease-to-use interface and functionality and intuitive platform dashboard helps users perform easy scaling, management, and application monitoring.

● Easy to scale and hassle free server resource management.Its auto-scaling feature helps to easily detect traffic spikes, create more Dynos accordingly and manage horizontal scaling.

● Optimal level of security for applications and any potential issues.

● Easy integration with Git and very efficient functionality of CI-CD pipeline.

● Integrated add-ons make it effortless to perform service installations and manage configurations, billing, and data from CLI or Heroku Dashboard.

Disadvantages of using Heroku-

● Prices are expensive as compared to other cloud based platforms and since resource utilisation is not optimal, costs are relatively high.

● Latency issues observed in virtual gym assistant application in live feed and model processing.

● Limited type of instances as users can only choose from specific memory, computing, CPU share limits, and dedicated server variations.

6.3.2 AWS Cloud Server

For deployment purpose, we have also explored the AWS cloud architecture to deploy Streamlit web application.
Amazon Elastic Compute Cloud (Amazon EC2) is a IaaS web service that provides secure, resizable compute capacity in the cloud where a user can launch instances with a variety of OSs, load them with custom application environments, manage network access permissions, and run images on multiple systems.

FIGURE 6.5 : WORKING DEMO OF DEPLOYMENT ON AWS

Advantages of using AWS Cloud architecture:-

- Instant provisioning of new servers and machines with customisable RAM
- Elastic growth for any workload with lots of instance types and ready to go images to launch an OS or software without doing all the setup
- Programmatic/API access to do development work
- All accounts are in a Virtual Private Cloud by default (isolated private network for security)

Disadvantages of using AWS Cloud architecture:-

- Instance types are rigid, so we must get entirely bigger instances even if require more CPU or RAM
- Expensive at on-demand rates if elasticity is not needed or expensive upfront payment is required if not using server for entire purchase length

Chapter 7

Conclusion

7.1 Project Limitations and Constraints

Although, this project has focussed on developing and deploying an end-to-end solution very effectively but a few notable limitations and constraints has been observed too.

1. The web application works seamlessly fine on local system without any latency issues but a delayed feedback of output video is received when the application is deployed on free tier cloud architecture platform. The potential solution of this issue is to deploy the application on paid cloud server with extra resources and subsequently evaluate cost runtime analysis.

2. Voice Recognition functionality works fine on local system but fails to work inside docker container or AWS EC2 instance. This issue arises mainly because the code logic expects to use default microphone of machine which it fails to detect inside docker container or EC2 machine.

3. The system , however didn’t show any kind of ambiguity in case if multi-persons are detected in a frame but still can be improved to handle such use cases.

4. Due to the deployment of application on free tier of cloud instances with limited resources, application might lag on runtime in case of multiple requests due to in-efficient horizontal scaling.

7.2 Future Scope

Many different adaptations, tests, and experiments on real-time varied dataset have been left for the future due to lack of time. It also includes designing and implementing a rigorous framework for error analysis and logging for development purpose. Another area of improvement comes in testing and deploying web application on paid cloud architectures to avoid runtime issues.
More complex workout exercises can be implemented that can help a user to choose from wide variety of options for their session. Also, an additional functionality of storing and providing workout session information to users on mobile dashboard can be added which can help to create a database and thus can help to create a time series analysis on a given period of time.

REFERENCES

[1] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, Christoph Bregler . Efficient Object Localization Using Convolutional Networks (2015).https://arxiv.org/pdf/1411.4280.pdf

[2] Zhe Cao, Student Member, IEEE, Gines Hidalgo, Student Member, IEEE, Tomas Simon, Shih-En Wei, and Yaser Sheikh. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields.https://arxiv.org/pdf/1812.08008.pdf .

[3] Alexander Toshev and Christian Szegedy. DeepPose: Human Pose Estimation via Deep Neural Networks.https://arxiv.org/abs/2006.10204.

[4] Valentin Bazarevsky and Fan Zhang. On-device, real-time hand tracking with mediapipe. https://ai.googleblog.com/2019/08/ on-device-real-time-hand-tracking-with. html.

[5] Joao Carreira, Katerina Fragkiadaki , Pulkit Agrawal and Jitendra Malik. Human Pose Estimation with Iterative Error Feedback.https://arxiv.org/pdf/1507.06550.pdf.

[6] Valentin Bazarevsky Ivan Grishchenko Karthik Raveendran Tyler Zhu Fan Zhang Matthias Grundmann. BlazePose: On-device Real-time Body Pose tracking .https://arxiv.org/pdf/2006.10204.pdf.

”