A Project Report

on

Human Pose Detection like Snapchat using Deep Learning

*carried out as part of the **Minor Project IT3270** Submitted*

by

Arunim Sureka
219303090

Sarthak Garg
219302116

in partial fulfilment for the award of the degree of

Bachelor of Technology

in

Information Technology

School of Information, Security and Data Science
Department of Information Technology

MANIPAL UNIVERSITY JAIPUR
RAJASTHAN, INDIA

May 2024

CERTIFICATE

Date: 15/04/2024

This is to certify that the minor project titled **Human Pose Detection like Snapchat using Deep Learning** is a record of the bonafide work done by **ARUNIM SUREKA** (219303090) and **SARTHAK GARG** (219302116) submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology in Information Technology of Manipal University Jaipur, during the academic year 2023-24.

Ms. Smaranika Mohapatra
Assistant Professor
Project Guide, Department of Information Technology
Manipal University Jaipur

Dr. Pankaj Vyas
HoD, Department of Information Technology
Manipal University Jaipur

ABSTRACT

The advent of deep learning technologies has significantly transformed the landscape of computer vision, enabling remarkable advancements in image recognition, and understanding. One captivating application that has gained substantial attention is human pose detection, akin to the interactive filters popularized by platforms like Snapchat. This minor project embarks on the exploration of human pose detection using deep learning, a cutting-edge field that holds immense potential in various domains, from augmented reality to healthcare. The essence of this project lies in harnessing the power of deep neural networks to accurately detect and analyze human poses from images or videos. The inspiration drawn from Snapchat's interactive filters underscores the project's goal to create an engaging and immersive user experience. By leveraging deep learning techniques, we aim to develop a robust system that can accurately identify and track human body poses in real-time, contributing to the evolution of interactive applications and systems.

With the use of the Mediapipe library, we have harnessed the power of MobileNet v2 CNN architecture to explore different applications of Pose Detection. Methodologies used include data collection which involves gathering a diverse dataset of annotated images depicting various poses to train and evaluate the model or the real-time video capture. Data is then preprocessed performing data augmentation to enhance model robustness. MobileNet v2 architecture is selected for CNN. Henceforth, the training and implementation of the pre-trained model are done to perform custom poses and actions with high accuracy.

LIST OF FIGURES

Table of Contents

1. Introduction

1.1. Introduction

This project delves into the intricacies of deep learning architectures, exploring key concepts such as convolutional neural networks (CNNs) and pose estimation algorithms. As we navigate through the project's development, we anticipate not only gaining valuable insights into the technical intricacies of human pose detection but also contributing to the broader narrative of how deep learning can enhance interactive experiences in the digital realm. The motivation behind Pose detection like snapchat using deep learning is the wide variety of applications in the industry offering solutions to diverse needs ranging from entertainment and gaming to healthcare, education, and security. The ability to accurately detect and track poses in real-time opens possibilities for creating more interactive and responsive systems in various industries. The project contributes to the field of computer vision and Augmented reality by providing an efficient and accurate pose detection system. The developed project can be applied to various domains including fitness tracking and entertainment. Advantages of this project are countless, from reconstruction, virtual testing, and re-identification of individuals to animation, gaming, virtual reality, and video tracking. The use of mobileNet v2 architecture makes the CNN a lightweight model which helps in training and testing the model faster with better accuracy and also compatible for device like mobile phones that have storage and memory limitations.

1.2. Problem Statement

Through this project, we are trying to develop a deep-learning-based human pose detection system using the MediaPipe framework and OpenCV library. With the rise of social media and Augmented Reality technologies becoming readily available to the common public, there is an ever-increasing demand for real-time and accurate pose and posture detection. The filters and frames used in such applications rely on computer vision and image detection to detect a user’s body posture and key points (eg: eyes, elbows, knees) to add or augment digital elements. Achieving completely accurate pose and posture detection remains a challenge, especially when dealing with problems like occlusions (when parts of the body are hidden), complex backgrounds, variations in the human body, lighting, etc. By leveraging these tools, the project seeks to address the mentioned challenges and contribute to the development of more interactive and immersive AR experiences within social media applications.

1.3. Objectives

1.3.1. Pose Detection: Implement a robust pose detection model capable of accurately estimating the key points of a person’s body including joints and limbs.

1.3.2. Real-Time Tracking: Develop algorithms for real-time pose tracking to ensure smooth and continuous detection as the subject moves within the camera frame.

1.3.3. Data Collection and Processing: Curate a dataset of diverse poses and perform preprocessing to enhance model generalization.

1.3.4. Deep Learning model: Design and train a deep learning model, potentially based on state-of-the-art architectures such as OpenPose and PoseNet to predict pose from the input images.

1.4. Scope of the Project

The scope of this project involves but not limited to real-time Pose Detection developing an efficient system that can detect and track human poses in real time using deep learning models. Ensuring accurate pose estimation even in challenging scenarios (e.g., occlusions, varying lighting conditions, complex backgrounds). Extending the model to recognize various poses, such as standing, sitting, yoga postures, or sports movements and integrating the pose detection system into user-friendly applications or devices.

Application-Specific Use Cases:
Healthcare: Monitoring patient movements for rehabilitation exercises.
Sports Analysis: Analyzing athletes form during training or competitions.
Gaming and AR: Enhancing virtual reality experiences with realistic avatars.
Security and Surveillance: Detecting suspicious activities or abnormal poses.

2. Background Detail

2.1. Conceptual Overview

Human pose detection is an application of Computer Vision. Computer vision is a field of artificial intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos, and other visual inputs. It allows them to identify and understand objects, people, and scenes in images and videos. By analyzing visual data, computer vision systems can make recommendations, take actions, and replicate certain human capabilities. Key technologies used in computer vision include deep learning and convolutional neural networks (CNNs), which help machines recognize patterns and features in visual data.

A neural network is a computational model that mimics the complex functions of the human brain. It consists of interconnected nodes or neurons that process and learn from data. These networks enable tasks such as pattern recognition and decision making in machine learning. Neural networks extract identifying features from data without pre-programmed understanding. The neural network used for image classification and detection is Convolutional Neural Network.

CNN classifies and extracts features from the image. Image is given as input in the form of 2-d matrix. This is passed through a kernel matrix or a filter that extracts features from images like vertical and horizontal edges. Dot product pf two matrix is passed on to another matrix and is further passed through pooling layer. Pooling reduces spatial dimension of feature maps while retaining essential information. After the image processing is done it is passed on to the fully connected neural network.

Mediapipe is a framework for building for building ML pipelines that process time series data like audio and video. It is pretrained library that can detect 33 key body joints and 468 points on face. It is a lightweight model trained using mobileNet v2 architecture of CNN.

MobileNetV2 is a convolutional neural network architecture designed specifically for mobile devices. It aims to achieve high performance while being lightweight and efficient. It uses Inverted Residuals and Linear Bottlenecks which employs lightweight depthwise convolutions for feature filtering and non-linearity.

2.2. Other Software Engineering Methodologies

Program execution is based on the Build and Fix model of software engineering which involves building and fixing the code continuously until correct. It is easy to implement and fast to debug and provide effectiveness in writing small to medium programs.

3. System Design & Methodology

3.1. System Architecture

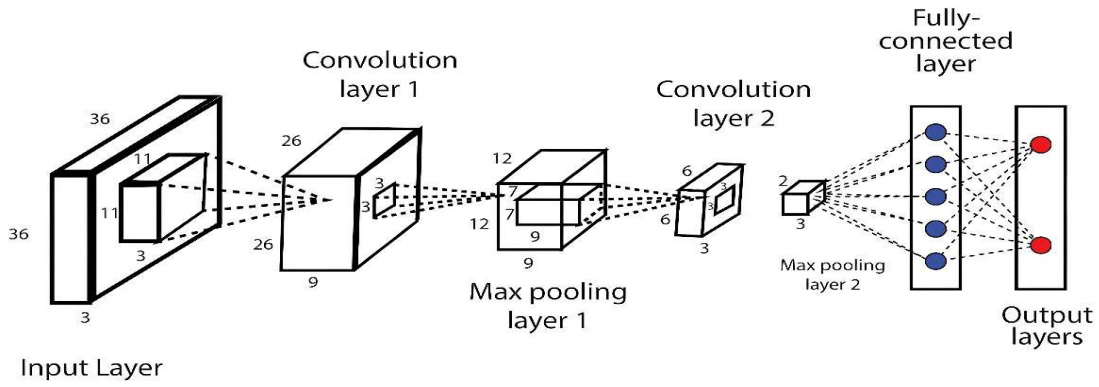


Figure 1- Working of a CNN

The above diagram is a schematic representation of the working of CNN which is the heart of this project. The input layer takes the image in the form of 2-d matrix and frames in case of video and passes it through kernel matrix for feature extraction. This comes under the convolutional layer. The output of this layer is passed through the Pooling layer which reduces the spatial dimension of feature maps retaining essential information (typically uses max pooling). It localizes the features for which the model can be trained and tested on. The output of this is passed through the fully connected layer which is a feed forward neural network consisting of input layer, hidden layer and output layer. It consists of many nodes that use activation function. The output from this layer gives us the result.

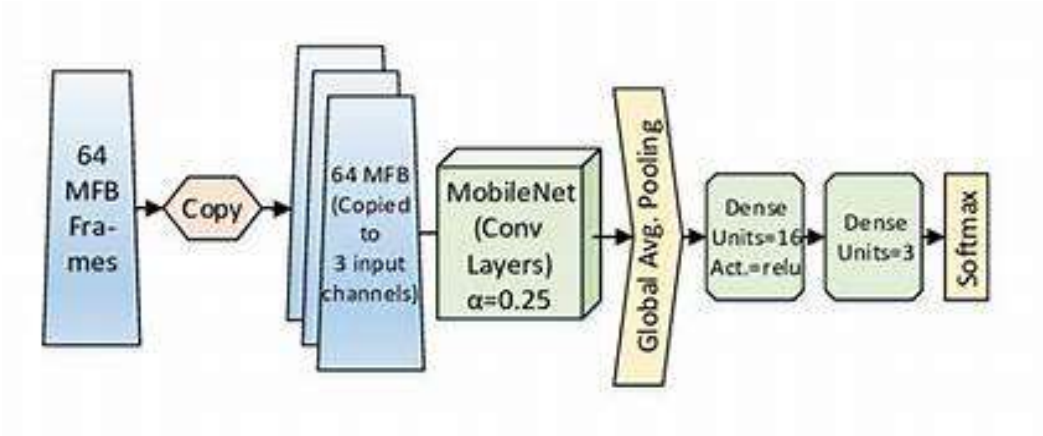


Figure 2- Working of mobileNet v2 architecture

Figure 2 is a schematic representation of the working of mobileNet v2 architecture. It consists of an input layer with an input of 64 MFB (Mel-frequency cepstral coefficients) frames. Next comes the convolutional layer. The notation α=0.25 indicates that it uses width multiplier of 0.25, it thins the network reducing its size and complexity. This output is passed through Global average pooling layer. It applies average pooling. The next layer is dense layer which consists of fully connected layer. The final layer is softmax which typically serves as the activation function for the last layer. The output is normalized into probability distribution over predicted output classes.

3.2. Development Environment

3.2.1. Hardware Requirements

- Processor: Intel Core i5 or Higher
- Disk Space: 80 GB (minimum)
- RAM: 8 GB or Higher

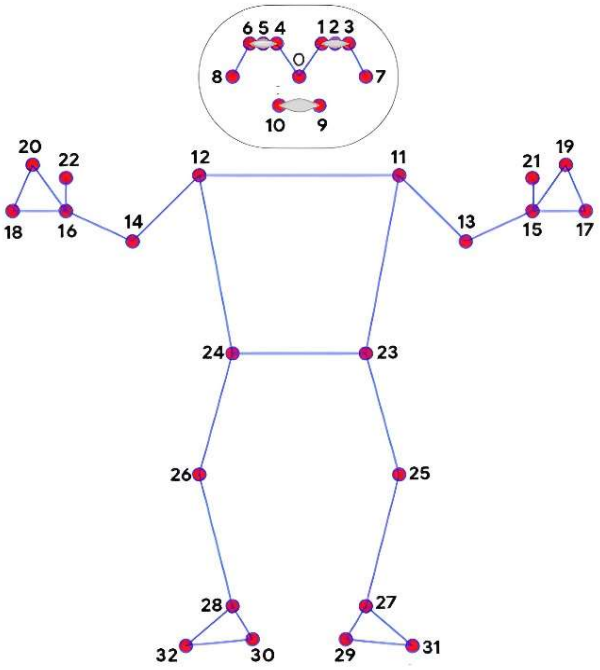
3.2.2. Software Requirements

- Operating System: Windows 10 or above
- Programming Language: Python 3.8.0 or above
- IDE: VsCode, PyCharm, Jupyter Notebook
- Deep Learning Framework: TensorFlow, PyTorch, Keras
- Pose Detection Model: PoseNet and Mediapipe
- Python Libraries: NumPy, OpenCV and Matplotlib

3.3. Methodology

Project uses Mediapipe library of the pose detection. It is a pretrained library capable of detection body joints, face mesh and palms of hand. It uses mobileNet v2 architecture which is a lightweight CNN model. CNN uses convolutional and pooling layer for feature extraction and the output from which can be passed on to fully-connected neural network or on to other Machine Learning algorithms like Support Vector Machines (SVMs) and Random Forests for better accuracy. The dataset is usually split into training and testing data but in our case the model is pretrained so it only needs testing data. The data can either be a image or video or it can be also a real time video captures that breaks the images into frame and passes through CNN for classification.

We first import OpenCV (Open-Source Computer Vision Library) that is equipped with functions like image processing and object detection. It takes the input in the BGR format. Next comes the Mediapipe. It provides a robust solution capable of predicting thirty-three 3D landmarks on a human body in real-time with high accuracy even on CPU. It utilizes a two-step machine learning pipeline, by using a detector it first localizes the person within the frame and then uses the pose landmarks detector to predict the landmarks within the region of interest. For the videos, the detector is used only for the very first frame and then the ROI is derived from the previous frame's pose landmarks using a tracking method. Also, when the tracker loses track of the identify body pose presence in a frame, the detector is invoked again for the next frame which reduces the computation and latency. Figure 3shows the thirty-three pose landmarks along with their indexes.



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right_knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

Figure 3- Key body joints detected by Mediapipe

Mediapipe consists of five arguments namely static image mode, min detection confidence, min tracking confidence, model complexity and smooth landmarks. The description of the following arguments are:

- Static image mode: It is a Boolean value that is set to false, the detector is only invoked as needed, that is in the very first frame or when the tracker loses track. If set to true, the person detector is invoked on every input image.
- Min detection confidence: It is minimum detection confidence with range (0.0, 1.0) required to consider the person-detection model's prediction correct. It's default value is 0.5 means if the detector has a prediction confidence greater than or equal to 50% then it will be considered as positive detection.
- Min tracking confidence: It is the minimum tracking confidence [0.0, 1.0] required to consider the landmark-tracking model's tracked pose landmarks valid. If the confidence is less than the set value then the detector is invoked again in the next frame/image, so increasing its value increases the robustness, but also increases the latency. Its default value is 0.5.
- Model Complexity: It is the complexity of the pose landmark model. As there are three different models to choose from so the possible values are `0`, `1`, or `2`. The higher the value, the more accurate the results are, but at the expense of higher latency. Its default value is `1`.
- Smooth landmarks: It is a boolean value that is if set to `True`, pose landmarks across different frames are filtered to reduce noise. But only works when static image mode is also set to `False`. Its default value is `True`.

After these arguments have been initialized the image input is now converted to RGB format as OpenCV read it in the BGR format. After performing pose detection, list of thirty-three landmarks are generated where each landmark has:

- x: it is x coordinate normalized to [0.0, 1.0] by image width
- y: it is y coordinate normalized to [0.0, 1.0] by image height
- z: it is z coordinate normalized to roughly same scale as x. It represents depth with respect to midpoint of hips being the origin.
- Visibility: It is a value with range [0.0, 1.0] representing the possibility of the landmark being visible (not occluded) in the image. This is a useful variable when deciding if we want to show a particular joint because it might be occluded or partially visible in the image.

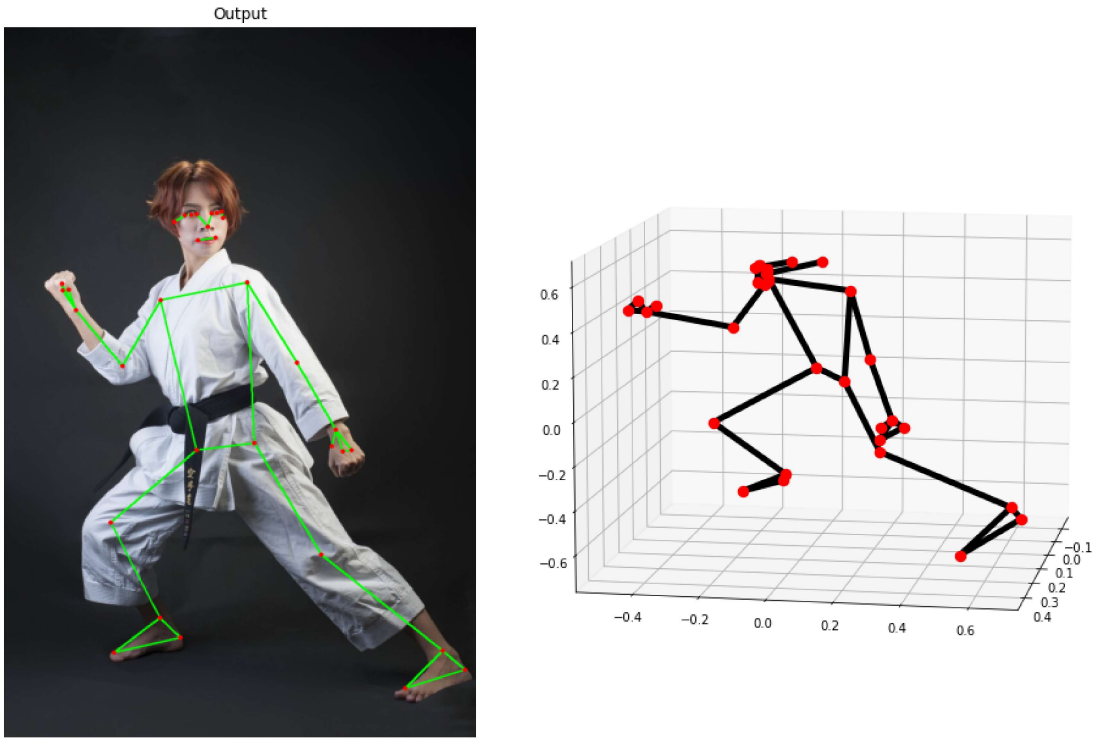


Figure 4- Output of Mediapipe

4. Implementation and Result

4.1. Modules of Project

There are 4 modules of the project. Each module corresponds to different applications. All of these uses Mediapipe for key body points detection and uses various functions to define and determine the use of those detections.

Modules consists of:

- Virtual gym trainer assistant:
  It can count number of reps for exercises like push-ups, pull ups and curls.

- Gesture based game playing:
  It uses positions of different body joints to give certain keyboard output to play games based on gesture. Arcade games like Subway Surfers can be played using this.

- Body language detection:
  Capable of detecting various body languages like happy, sad and victorious. It uses face mesh and key points on palm and hand.

- Multi-person pose detection:
  Detects pose of more than one person in the frame

4.2. Implementation Detail

Virtual gym trainer assistant:

This program is a simple 3d pose estimation application, which identifies and tracks 33 core body landmarks as defined by Blaze Pose. We have defined a function to track a selected 6 of the 33 landmarks in this application to track and count the number of repetitions of gym exercises done correctly. The application tracks and identifies whether a rep was done correctly and increases or retains the counter accordingly. The 6 key features we are focusing on for this application are the left and right shoulder, the left and right elbow, and the left and right wrist. Based on the movement of the arm, the angle between the landmark points as defined above are measured (shoulder, elbow and wrist). The decision factor for the classification of a rep as correct or incorrect is based on the measured angle. A proper rep is counted if the angle measured moves from a minimum threshold value of 50 degrees to a minimum of 140 degrees.

Gesture based game playing:

The second application uses Blaze pose library of the Mediapipe framework to identify 33 key points of the human body, using a live video capture. The program forms a stationery center aligned axis and identifies the position of the user with respect to this axis. Based on the user's movement relative to the axis a keypress is initiated. Assuming the user is currently standing at the center position, on moving left a keypress of left is executed, on moving back to center a keypress of right

is executed, on moving right from the center position a keypress of right is executed, similarly when moving back to the center from the right side a keypress of left is executed. The model also captures up and down movements and classifies them as jumping and crouching respectively. The movement is classified based on the relative position from the center of the predefined axis. When a user jumps regardless of the position on the X-axis (Left, Right or Center) a keypress of up arrow is executed and when a user crouches below the center axis a keypress of the down arrow is executed. The program can run in the background of another computer application and therefore can be used to control the applications using body movements relative to the axis. One such application is playing the Subway Surfers game which requires up down left and right movements to progress in the game. With this program we can play the game with dynamic controls like moving on the x axis and y axis simultaneously by jumping to the side, which initiates an up and left or right keypress simultaneously.

Body language detection:

The third application allows the user to train a model with their own training data inputs which can be captured using a live video capture. In this application we have used Mediapipe Holistic which pays a closer attention to landmarks points of the hand and face. It detects a total of 46 points on the body (21 on each hand and 4 additional points 2 for each elbow and shoulder) , In addition to these 44 points it detect a Face Mesh consisting of 468 landmark points detected on the face , creating a mesh like structure. The vast complexity of the face mesh allows us to train precise models and get an accurate result. We can input the training data for as many emotions and actions as required by simply changing the appropriate tag for the training data before beginning the video capture for the training data. We have captured the training data for three human emotions and actions – Happy , Sad and Victorious. The training data is captured via live video input and for increased accuracy we have taken input data from 10 people, all varying in human body factors such as build and height. The training data is pipelined to fit four machine learning models namely logistic regression, RidgeClassifier , RandomForestClassifier , GradientBoostingClassifier. The model used for final training is RandomForestClassifier as it produces the highest predicted accuracy score out of the four models used. The Model is trained using the training data captured in the CSV file with the coordinates of the landmark points detected and a live video feed is initiated. The results based on the current state of the user in the live video feed and the predicted result of the trained model are displayed together with an estimate confidence score. The program can also be implemented to give results to a prerecorded video in place of a live video feed to give results based on the labelled training data along with the confidence scores.

Multi-person pose detection:

For the fourth and final implementation we have used the Movenet Multipose lightning 1 model to execute militiaperson pose detection on a live video capture as well as mp4 file video input. The model executes body landmark detection and tracking to identify human pose and posture on multiple persons simultaneously.

4.3. Results and Discussion

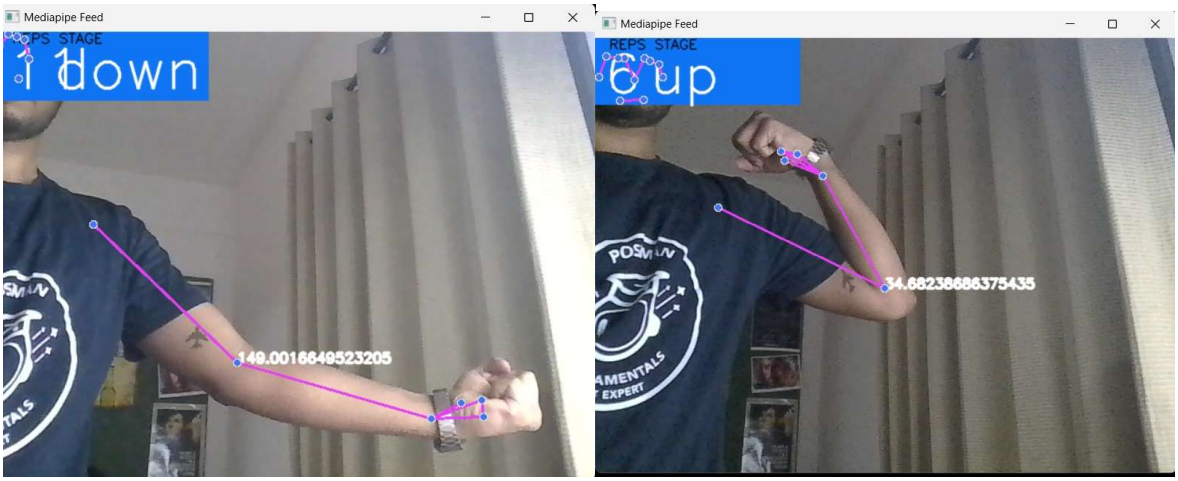


Figure 5- Output of virtual gym trainer assistant

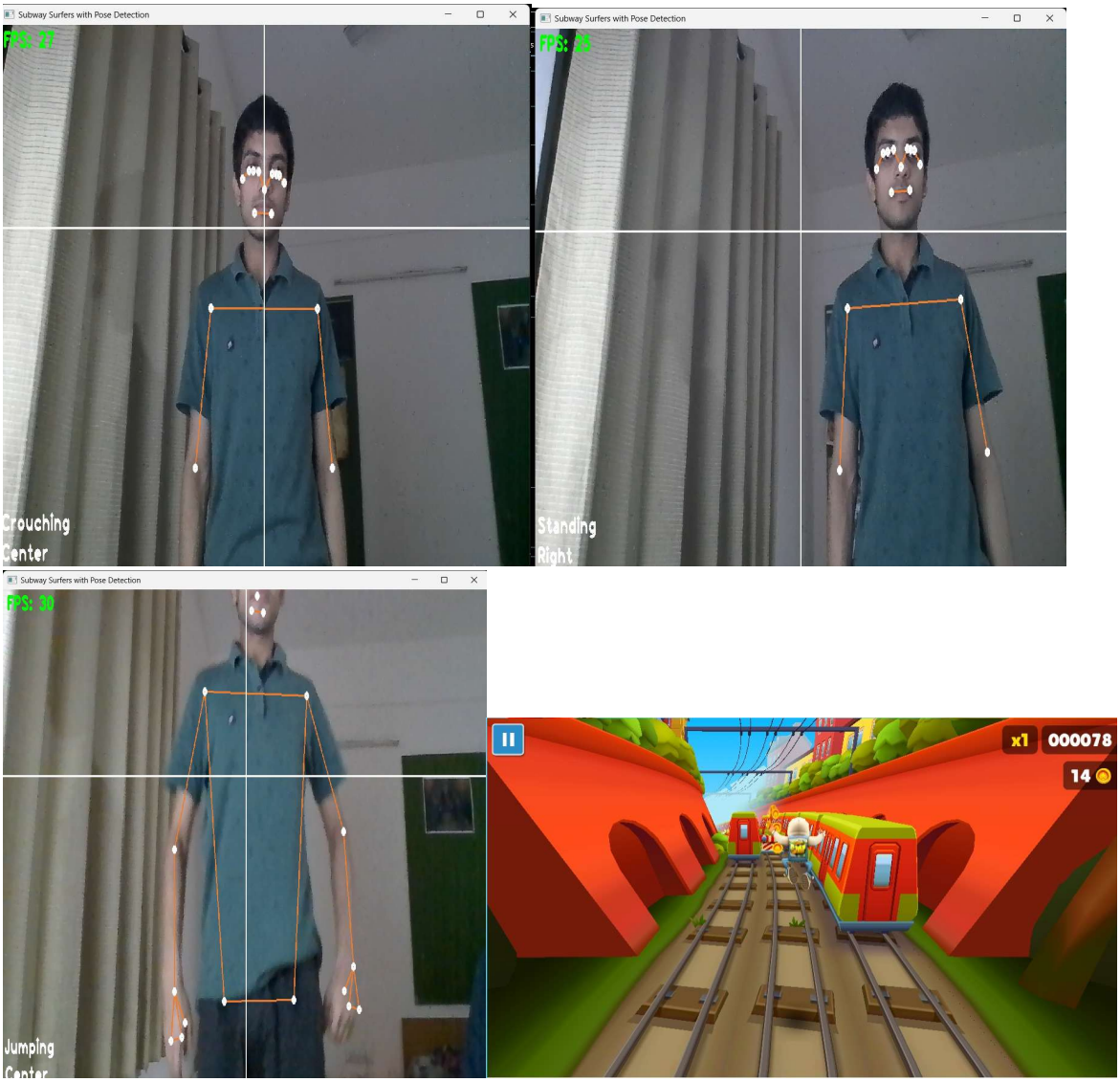


Figure 6- Output of gesture based game playing

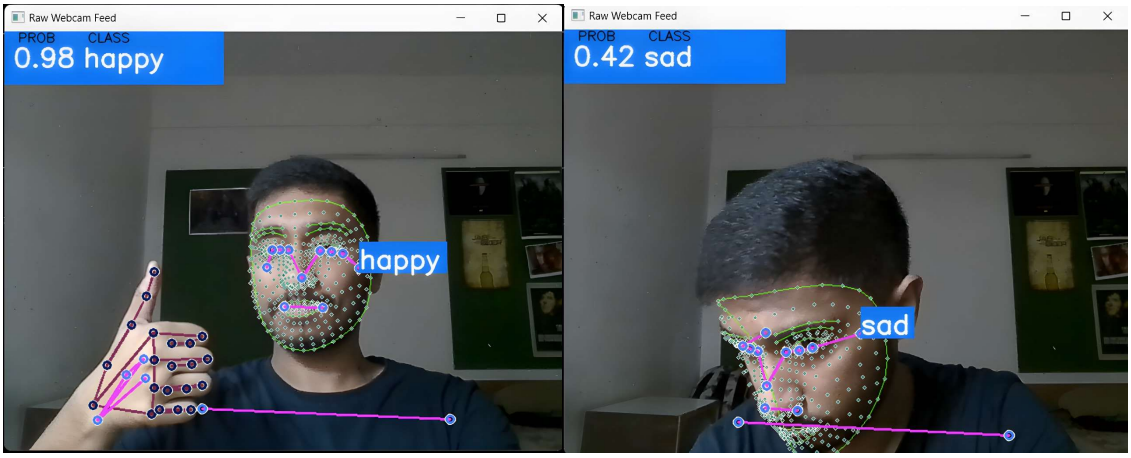


Figure 7- Output of Body language detection

Figure 8- Output of multi-person pose detection

Figure 5 to 8 represents the output of the four modules that have been implemented. In the figure it can be seen that a person is doing curls and there is a counter counting the reps due to change in the angle of elbow movements. Figure 6 shows a person controlling game movements by gesture like jumping, crouching, and moving left/right in the camera window. Figure 7 represents happy and sad body language of a person with given probability. Figure 8 shows image of two persons and algorithm detecting pose of both the pesrons.

4.4. Month wise plan of progress

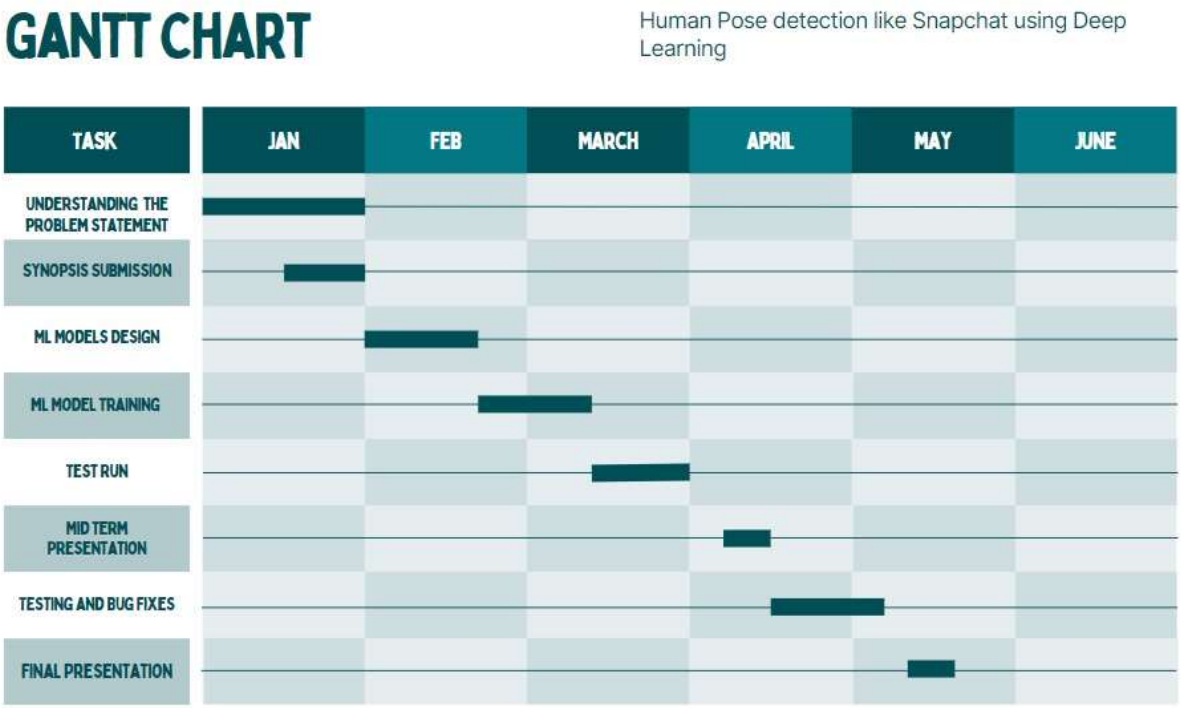


Figure 9- Gantt Chart of monthly progress

5. Conclusion and Future Plan

This project on human pose detection like snapchat using deep learning provides a comprehensive overview of usage of Mediapipe and CNN and exploring various applications of pose detection with it. Pose detection is an important application in various fields ranging from medical to scientific experiments. This project implements three different applications along with multi person pose detection. Detection accuracy has been taken into account along with the confidence score for various custom-built poses based on key body joints using by defining functions in python. This project integrates insights found from different paper on the usage of best algorithm. It was found that combination of CNN and Random Forest gave better accuracy on an average in comparison to other machine learning algorithms. Overall this project has completed all aspects related to pose detection from selecting the best architecture to increasing accuracy and showcasing various applications of it.

Future plan consists of but not limited to focusing on improving the efficiency further by leveraging the ideal CNN and machine learning combination. Multi-person pose detection accuracy and frame rate can further be improved using YOLO which segments the image into multiple parts with each part containing object in it. Furthermore, the pose detection algorithm is applied individually to these segments for the result.

6. References

1. Ce Zheng, Wenhan Wu, Chen Chen, Taojiannan Yang, Sijie Zhu, Ju Shen, Nasser Kehtarnavaz, and Mubarak Shah (2018). Deep Learning-Based Human Pose Estimation: A Survey. J. ACM 37, 4, Article 111.

2. Samkari, E.; Arif, M.;Alghamdi, M.; Al Ghamdi. M.A. Human Pose Estimation Using Deep Learning: A Systematic Literature Review. Mach. Learn. Knowl. Extr.2023, 5, 1612–1659.

3. Cao, Z., Simon, T., Wei, S. E., & Sheikh, Y (2019). OpenPose: Real-time multi-person keypoint detection library. IEEE Transactions on Pattern Analysis and Machine Intelligence.

4. A. Kendall, M. Grimes, and R. Cipolla (2015). PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. Proceedings of the IEEE International Conference on Computer Vision (ICCV).

5. Rajalingappaa Shanmugamani (2018). Deep Learning for Computer Vision: A Brief Review. Publisher: Packt Publishing ISBN-13: 978-1789341076.

6. Mediapipe, https://developers.google.com/mediapipe/ , March 2024