Posture Evaluation for Variants of Weight-Lifting Workouts Recognition

By

Ng Jiunn

A REPORT

SUBMITTED TO

University Tunku Abdul Rahman

In partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology

(Kampar Campus)

JANUARY 2020

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: Posture Evaluation for Variants of Weight-Lifting Workouts Recognition

Academic Session: JANUARY 2020

I Ng Jiunn

(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in

Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

(Author's signature)

(Supervisor's signature)

Address:
1105, Jalan Seksyen 1/2,
Taman Bandar Barat,
Kampar, Perak.

Aun YC

Supervisor's name

Date: 24-04-2020

Date: 24/4/2020

Posture Evaluation for Variants of Weight-Lifting Workouts Recognition

By

Ng Jiunn

A REPORT

SUBMITTED TO

University Tunku Abdul Rahman

In partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology

(Kampar Campus)

JANUARY 2020

DECLARATION OF ORIGINALITY

I declare that this report entitled "**POSTURE EVALUATION FOR VARIANTS OF WEIGHT-LIFTING WORKOUTS RECOGNITION**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : 

Name : Ng Jiunn

Date : 24-04-2020

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Dr. Aun YiChiet who has given me this bright opportunity to engage in AI field project. It is my first step to establish a career in AI field. A million thanks to you.

Special thanks to Dr Tan Hung Khoon, for his patience, and passionate in teaching and helping me when I encountered implementation issues. Finally, I must say thanks to my parents and my family for their love, support and continuous encouragement throughout the course.

ABSTRACT

Weight lifting is a flow of body movement pack in an organized exercise to force the body muscles to contract under tension by using weights such as barbells, dumbbells or even body weights in order to trigger growth, strength, endurance and power. Performing wrong posture is a very common issue for every gymnast, either beginner or even professional. Computer Vision (CV) is a field of computer science that seeks to develop techniques in enabling computers to see, identify, understand and process the content of digital images in the same way that human vision does, then provide appropriate output. Object detection and object recognition, which are two of the famous CV technologies, have been applied in this project. Posture performing workout will be detected then evaluate the posture. KNN classifier has been trained from calculating angles between joint keypoints of the user to recognise the workout type. The system with the function of detect and recognize the workout type from the input video had been tested with multiple workout type under different environments and achieved around 98% accuracy. The system is also able to classify different types of improper posture with the accuracy of 80.69% for Bicep Curl class, 65.35% for Front Raise class and 89.75% for Shoulder Press class.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CV	Computer Vision
SVM	Support Vector Machine
AI	Artificial Intelligence
DTW	Dynamic Time Warping
KNN	K-Nearest Neighbour
CNN	Convolutional Neural Network
RGB	Red Green Blue
ANN	Artificial Neural Network
ETL	Extract, Transform, Load
FPS	Frame Per Second

CHAPTER 1 INTRODUCTION

1.1 Problem Statement

In the domain of Artificial Intelligence (AI) based fitness coaching, weight lifting is a flow of body movement pack in an organized exercise to force the body muscles to contract under tension by using weights such as barbells, dumbbells or even body weights in order to trigger growth, strength, endurance and power. This exercise can be helpful to personal health. However, it can also be potentially dangerous and inefficient if the exercise is perform in wrong posture by the trainees. Performing wrong posture is a very common issue for every gymnast, either beginner or even professional. This is because it is a big challenge for trainees to perform a series of complex movements in weight lifting exercise. Posture issues can manifest into many different conditions that can put trainee health in danger, such as neck pain, back pain, shoulder pain or knee pain. Correct posture is able to minimize the strain on the human body by maintaining the balance between muscles and skeleton. This balanced musculoskeletal state is very important because it prevents further damage or any progressive deformation in weight lifting training and protects the supporting structures in the body. Therefore, performing correct posture is a very crucial aspect for proper arrangement of supporting structures, maintaining balance of the body and effective functioning of the body.

1.2 Background and Motivation

Computer Vision (CV) is a field of computer science that seeks to develop techniques in enabling computers to see, identify, understand and process the content of digital images in the same way that human vision does, then provide appropriate output (Jason 2019). CV has few important and famous technologies that will be use in this optimal posture detection system such as object detection, object recognition and etc.

In the last few years, object detection has become one of the main tasks in CV which involves assigning a class label to an image and drawing a bounding box around one or more objects in an image. The technique of object detection had been widely used in human's life and certain industry areas such as tracking objects, facial detection and etc. Object detection technology is applied to detect the human posture when performing exercise.

There are few common ways for the system to detect the posture. First of all, camera has object detection feature such as facial or hand detection few years ago. It is quite common to buy a smartphone with a camera that able to detect face and hand. With a little bit of improvision, the camera can detect the human posture too with the help of few programming language libraries such as OpenPose, wrnchAI, Caffe, Tensorflow and etc. For the second way to detect the posture will be wearing sensor on the body. By interpreting and analysing the data generated from the sensor, it is possible to detect the human posture. However, the object detection technique is far away from perfection and contains various challenges. According to the survey about the problem of object detection in real images that conducted by Dilip in 2012, the accuracy of any detection algorithms is below 16% for over 22,000 object categories.

The another main task in computer vision is image recognition, it is a process of identifying and detecting certain people, animals, features, single or multiple objects in an image or video, through the use of algorithms and machine leaning concepts. This concept is used in many applications like systems for toll booth monitoring, security surveillance, factory automation and many more. Image recognition can be done in many different ways, such as machine learning or deep learning. The machine learning approach to image recognition involves identifying and extracting features from images and use them in building a model to classify the inputs. Other than that, many of the top techniques involve the use of convolutional neural networks, which is a deep learning approach to filter images through a series of artificial neuron layers.

For the application of this technology in this proposed method, it is required to recognise the exercise type from detected human posture. Even great efforts have been made for decades, the recognition of human activities is still an immature technology that attracted plenty of people in computer vision. The implementation of supervised learning in machine learning seems like a promising approach to achieve exercise recognition based on detected posture. For instance, according to Qian, Mao, Xiang and Wang in 2010, a recognition of human activities using Support Vector Machine (SVM) multiclass classifier is showing a promising and good result. Several of features are extracted from detected posture then the features are used to train the recognition model. Image recognition has come a long way, however it is now the topic of a lot of

controversy and debate in consumer spaces. There are a lot of discussions about how rapid advances in image recognition will affect privacy and security around the world.

These findings above are implicating posture detection, recognition have been focused in these few years, and the technologies are mature enough to be implemented in real world. But, similar topics such as posture correction and evaluation are not the famous topics for discovering, even though they have so much potentials just underneath them. The deliverable of this project is going to cover all the topics mentioned before, which are posture detection, recognition, correction and evaluation.

There are few motivations to propose this approach. Research on activity recognition has traditionally focused on discriminating between different activities. For instance, to predict which activity was performed at a specific point in time. However, the evaluation of quality in performing an activity, has only received little attention so far, even though it potentially provides useful information for a large variety of applications. Therefore, this is one of the motivations of carrying out this project, to explore the hidden potential of this technology.

According to Barbara in 2014, there are a bunch of people may avoid exercising because of embarrassment. One and main of the reasons is they do not know how to start in weight lifting and they do not know how to perform correct posture since they do not have any knowledge about it. Hence, this system will be a great tool for a beginner trainee to get started in weight lifting exercise. Trainee can master the correct posture after applying the provided correction posture feedback from this system.

In addition, this system is also applicable to advance trainees too, sometimes they lifted heavy weight because of their ego and wanted to look badass in the gym. However, in the real case, their muscles could not support such weights hence poor posture is performed by them, even professional weight lifting trainee. For problem like this, it can easily solved by using this approach system to remind them maintaining a good posture is the most important one, compared to weight lifted.

1.3 Project Objectives

- To design workout exercise recognition method using kNN
    a. To label the angle of sequential keypoints trajectory in workout videos
    b. To optimize model accuracy through parameter tuning (number of neigbhours)
- To design workout posture classifier for workout posture correction recommendation using KNN
    a. To optimize model accuracy through parameter tuning and feature selection.
- To create a new benchmark for custom self-made workout dataset.

1.4 Proposed Approach/Study

Figure 1.1 shows the general proposed approach to develop the Workout Recognition and Evaluation System.

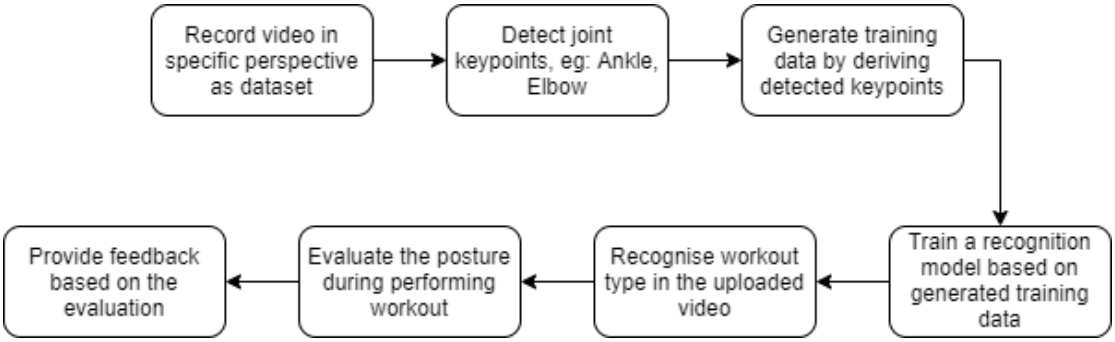


Figure 1.1: General Approach for this Project

The first step is dataset collection since data is required in order to train a model. User can record them dong workout from specific perspective such as front view or side view, depending on what kind of workout.

Secondly, further analysis will be done on the video in order to extract data. There are few pre-trained model that able to identify joint keypoints such as WrnchAI or OpenPose. Then, some operations will perform on these keypoints to generate feature

vector and train a model base on this vector. Finally, the recognition model is ready to deploy.

Another major phase will be the posture evaluation, since the workout type is predicted, the next step should be deciding whether the performed posture is correct or wrong. Once the posture is evaluated, feedback will be provided according to the evaluation type.

1.5 Highlight of what have been achieved

In FYP1, first phase of this project has been achieved, which is recognising the workout. Other than that, data preparation stage is also done such as rotating image to correct perspective to maximise the accuracy of detected joint keypoint positions.

1.6 Report Organization

The detail structure of this report are shown in the following chapters. In Chapter 2, some related approaches are reviewed. Then, system design of this project is presented in Chapter 3. Besides that, Chapter 4 will discuss the system specifications such as user or system requirements, verification plans and the implementation issues during developing this project. Moreover, Chapter 5 describes the experiment and evaluation process from Chapter. Lastly, Chapter 6, which is conclusion that summarise the whole report with few simple statements.

CHAPTER 2 LITERATURE REVIEW

2.1 Chapter Overview

After discovering some introductory papers about the human posture detection, there are few approaches to detect the human posture in weight lifting and some of them are able to evaluate and make correction for correct posture.

2.2 Pose Trainer: Correcting Exercise Posture Using Pose Estimation

An approach which named Pose Trainer has been introduced by Chen et al. (2018) for detecting and correcting human posture by using pose estimation. This approach consists of three parts that constructed by six components (Figure 2.1). The first part has only one component, Video Recording and it can only be done by user. The next part is built from Pose Estimation , Keypoint Normalization and Exercise Detection. A state-of-the-art pose estimation deep neural network, OpenPose from OpenCV is used within Pose Trainer for inference. The third part involves detecting the quality of user's predicted pose for specific exercise, which are Posture Evaluation and Feedback Correction.

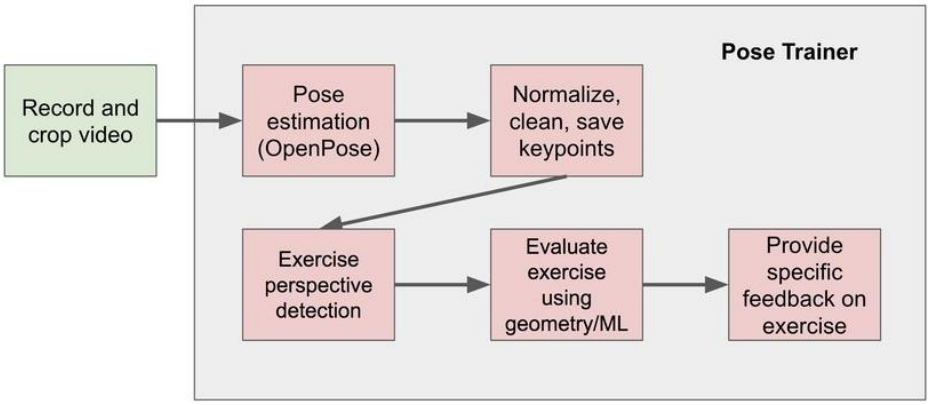


Figure 2.1: Flowchart of Pose Trainer

In the first component, the video must be recorded from a particular perspective such as facing front the camera or facing side to the camera that allows the exercise to be seen clearly. Moreover, Pose Trainer does not require distance between body and

camera or type of camera. However, Pose Trainer does require user to crop and edit the exercise video by themselves, which explaining why segmentation of replication in one exercise does not exist in this application.

For the next component, deep Convolutional Neural Network (CNN) to label Red Green Blue images. OpenPose, a pre-trained model is chose for pose detection after comparing multiple state-of-the-art pose estimators. The output of OpenPose contained coordinate of 18 keypoint locations and their respective confidence levels. The predicted keypoints include nose, neck, shoulder, elbows, wrists, hips, knees and ankles.

There are two purposes in Keypoint Normalization step. First, a parser is wrote to use the raw keypoint output from OpenPose. List of keypoints are read while 2-D coordination and confidence level are segmented into joint keypoint object. Obscure keypoint that has zero confidence level is eliminated. A Pose object which is a full skeletal estimation of a person is constructed by composing each joint keypoint in every video frame. Then, a PoseSequence is composed to chain all Pose objects from each video frame. Secondly, in order to account distance between body and camera, different body length measurements and other potential problem, the pose is normalised based on the torso's length in pixels. The torso length is calculated by average distance from neck joint to right and left hip joints.

The fourth component is Perspective Detection. In this step, the ambiguity in camera perspective is resolved. For instance, the bicep curl exercise can be recorded from the side of camera and could be performed by one or both arms (Figure 2.2). Pose Trainer is able to identify which arm is performing the exercise by calculating which keypoint is most visible.

Figure 2.2: Bicep Curl Exercise

For second last component Evaluation Exercise can be achieved by two approaches, geometric algorithm and machine learning algorithm. First, geometric heuristics are designed from keypoints of interest and personal training guidelines, in order to evaluate the body vector. Secondly, a more data-driven which is machine learning approach is introduced to evaluate the exercise posture. Since the recorded video can be any arbitrary length, this caused different keypoint vector length such as exercise speed and this is a big challenge for many machine learning algorithms. Therefore, Dynamic Time Warping (DTW) with K-Nearest Neighbour (KNN) classifier is chosen and it is a metric used to measure non-linear similarity between two time series. In DTW, keypoint in second sequence is dynamically identified that corresponded to given keypoint in first sequence. A distance matrix that consists Euclidean distance from different keypoints in two sequences is measured by iterating all points using dynamic programming. KNN classifier can predict correct or incorrect posture based from DTW distances.

In this paper, the F1 scores for bicep curl exercise is 0.85, front raise exercise in 1.00, shoulder shrug is 0.85 and shoulder press is 0.73.

There are various advantages and disadvantages found in this paper. One of the advantages is user of Pose Trainer only needs to download OpenPose and install Anaconda to make it workable, by choosing OpenPose. Unlike Tensorflow or Caffe, which require complex user installations such as CUDA and CuDNN GPU libraries. Another advantage is Pose Trainer can track multiple exercises instead of one with high accuracy, which make it has high commercial value than the other similar apps.

For the disadvantages, Pose Trainer does not designed as application, which means it is not functional on phone. In order to solve this, they should export Pose Trainer to smartphones, building an application that allows users to record video and get feedback at any place, any time. The next disadvantage is Pose Trainer only able to provide posture feedback with vocabulary format. This might be confusing to user because user may interpret the feedback with different meaning or does not even understanding the meaning of feedback. There is a way to solve, which is providing posture feedback by showing the comparison user's labelled posture diagram with ground truth trainer labelled posture diagram. A picture wins a thousand words, diagram is more understandable compare to vocabulary. For the last disadvantage, Pose Trainer is

unable to segment the repetitive exercise by itself, which left this task to the users to crop the video by themselves. This step is easy for user to perform but it is also a time-consuming step, since user has to manually crop the full exercise into segmented repetition. This problem can be tackle by providing a segmentation step to automate the process. In this case, user will not have to crop video anymore.

2.3 Machine Learning Methods for the Automatic Evaluation of Exercises on Sensor-Equipped Weight Training Machines

On the other hand, Hristo and Arnold in 2012 introduced a pattern recognition techniques for the evaluation of exercises performed on weight training machines equipped with a load cell and rotary encoder. This approach is able to evaluate the exercise movement quality regarding predefind criteria such as duration, constancy, velocity and completeness. There are totally five steps which are Signal Input from sensor data, Filtering, Segmentation, Feature Extraction and Classification (Figure 2.3).

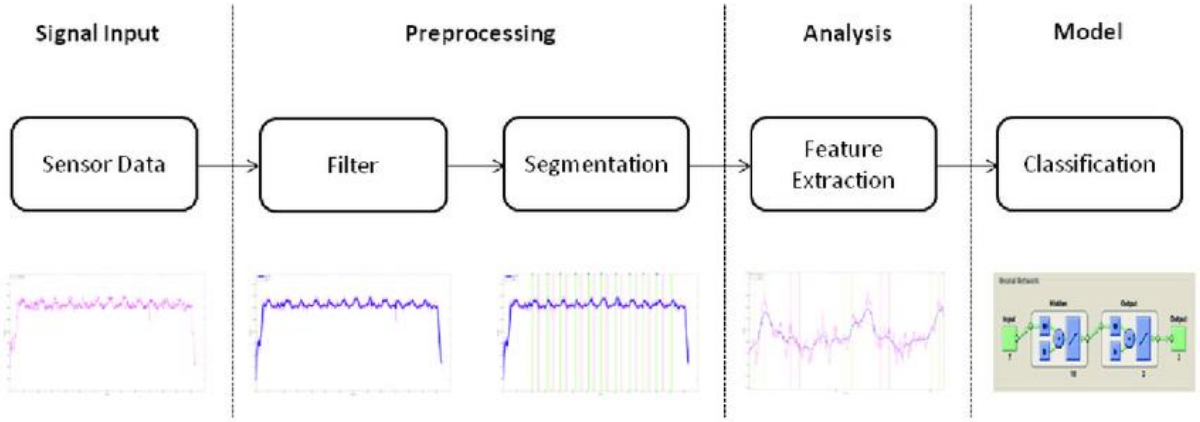


Figure 2.3: Proposed Method Flowchart

In the first step, the weight training machines are equipped with sensors such as load cell and rotary encoder for data collection purposes. There are two data that retrieved from sensor which are weight displacement value and force signal. Further determinants such as velocity, acceleration, power and repetition can be derived from sensor data.

For the data pre-processing step, it consists of cleaning, filtering and segmenting data. A strong low-pass filter is applied for smoothing the displacement value while an average low-pass filter is used for the force signal in order not to lose significant characteristics on occurred fluctuations.

The goal of feature extraction is to find and derive obvious characteristics that will identify the pre-processed data. First of all, an exhaustive set of attributes is extracted

for each repetitions in feature space. Then, the feature selection is based on time, velocity, constancy and completeness properties in weight training.

In the last step, supervised learning method is chosen because the appraisements from professional coaches are available. These assessments can label the feature vector and then assigned an exhaustive set of features to specific classes such as well-performed posture class, improper posture class and inconsistent posture class. A traditional Artificial Neural Network (ANN) approach is applied on the labelled data and the modelling results shows 93% accuracy.

The most obvious advantage of this paper is that the whole evaluation of posture exercise process is automated. It does not require any input from trainee, the evaluation process will be triggered when trainee started to use that weight training machine. This is a very user-friendly and convenient method for trainee.

On the other hand, this paper does have several disadvantages. The first disadvantage is can only evaluate the exercise that done by weight training machine because the sensors are installed inside the machine, which excluding the free weight training such as push up, squat and etc. The problem can be solved by using non-static sensor that is wearable on trainee body such as accelerator. Sensor can even replace by just using camera to video recording the performed exercise for more advance modification. In this case, the range of evaluation can expand to many exercises. Moreover, the feedback provided by this method is not complete, it only gives feedback such as speed doing exercise, completeness of exercise and etc. A good and helpful feedback should also cover posture evaluation. In order to solve this problem, detection, recognition and classification of posture should be added into this method so that trainees can know are they performing correct or incorrect exercise.

2.4 Detecting Unseen Anomolies in Weight Training Exercises

An interesting approach had been presented by Kowsar et al. in 2016 to detect unseen anomalies in weight lifting training by using motion sensor and camera. This paper presented a workflow to detect performance anomalies from observations of correct performance of an exercise (Figure 2.4). The dataset used in this paper is unilateral bicep curl exercise.

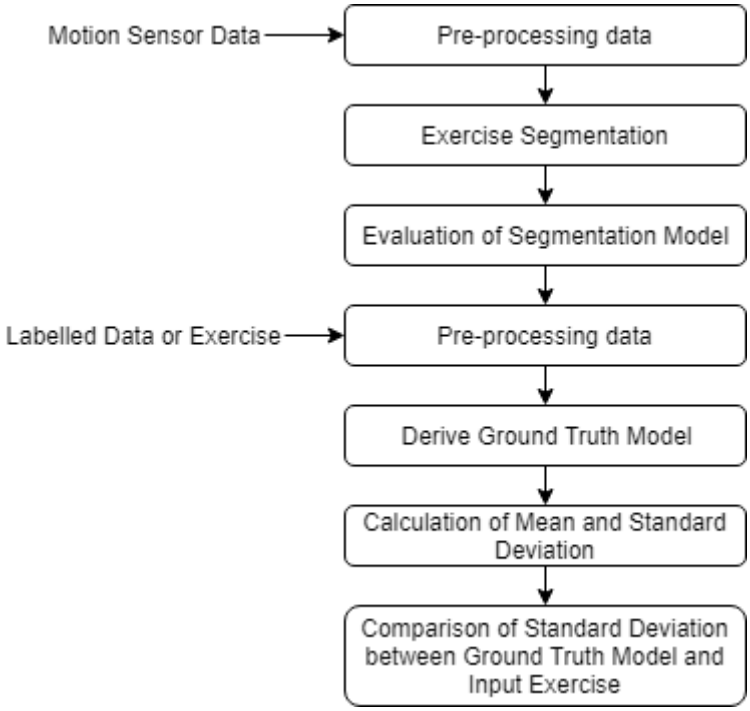


Figure 2.4: Proposed Method Flowchart

The first step of this approach is to pre-processing motion sensor data. It is common that motion sensor showed some degree of white noise that came from nature of the sensor and the white noise showed as small perturbations around the actual value. Any successful data analysis is start from removing noise from the data (Yun et al. 2006). Therefore, Kalman Filter had been chosen as a high-pass filter to eliminate noise with high accuracy (Figure 2.5).

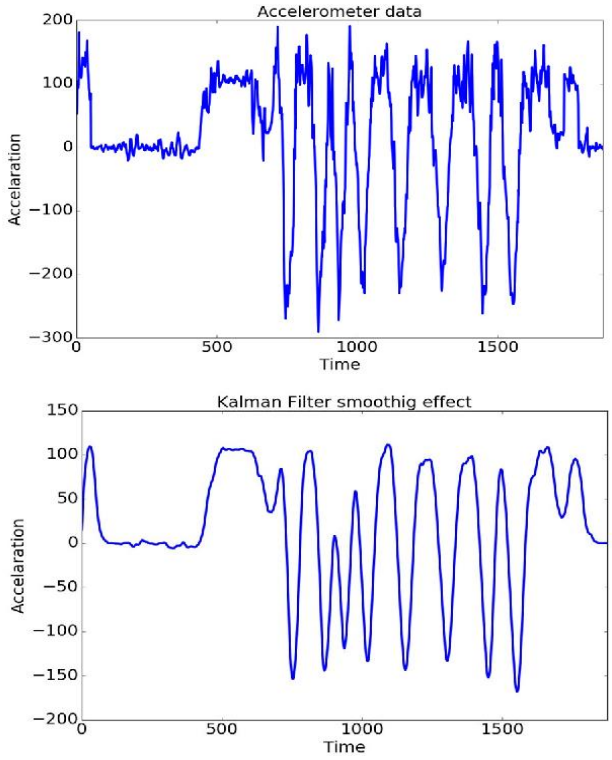


Figure 2.5: Smoothing Sensor Data with Kalman Filter

Weight lifting exercise are repetitive tasks where same move were performed for a few repetitions. In this paper, the exercise is segmented to each every repetition by using motion sensor data (Figure). A function "*f*" is defined as change-of-position in space then chain rule is applied into this function. Minimum point indicating the start point of each move and it can be estimated from *f*. After finding out all minimum points on the acceleration-time graph, segmentation can be done based on those points. An evaluation is carried out to test the accuracy of *f* function and I has 96.5% of precision which is an acceptable result.

For the classification part, supervised learning is chosen as data for correct executions of exercise are available from personal trainer. Before entering classification part, pre-processing is performed on labelled data by using a cluster algorithm to put the segments with high similarity into the same group (Figure 2.6). This step is necessary because usually starting and ending repetitions are just carrying or releasing weights, which are not actual exercise. A ground truth model is derived to act as classifier.

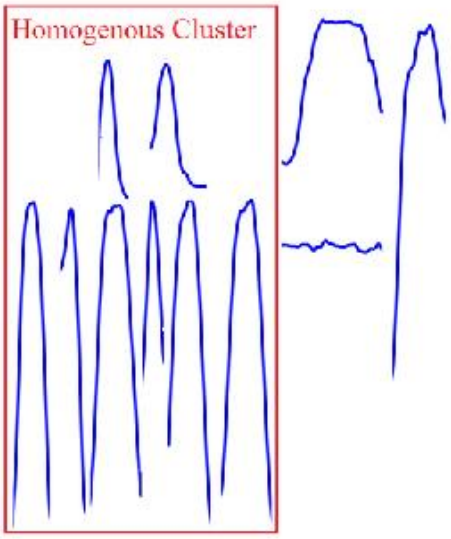


Figure 2.6: Cluster Similar Repetition Segments

In order to derive a ground truth model, mapping one time-series to another is necessary so that distance over every two pairs in the two time-series is calculated (Figure 2.7).

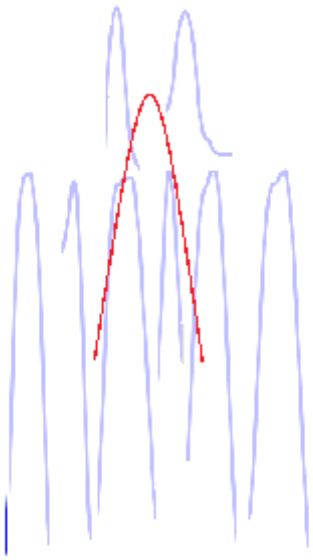


Figure 2.7: Deriving Ground Truth Model

Standard deviation for ground truth is calculated for detecting anomalies in input weight lifting exercise (Figure 2.8). Each associated trajectory of segment of input exercise is

calculated. Then, mean and standard deviation for each point in time in the trajectory set are also calculated. Mean and standard deviation from input exercise are compared with mean and standard deviation from ground truth model. If any point is found outside the margin from the new trajectory, the segment is labelled as wrong posture (Figure 2.8). This approach gave a result with 99.4% True Positive.

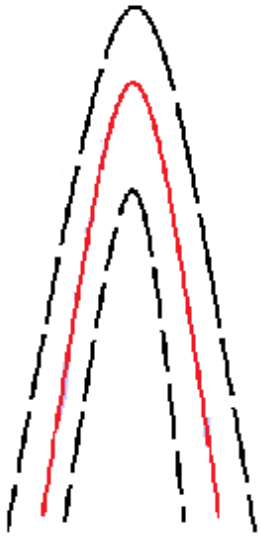


Figure 2.8: Calculation of Standard Deviation (Dashed Lines) for Ground Truth Model

One of the advantage of this proposed method is the scope of the system is not only limited to weight lifting exercises, even though the main focus is on weight lifting exercise. This system is applicable where the users must perform some repetitive task and their performance must be monitored, such as physical rehabilitation, physiotherapy, elderly fitness, health insurance and etc. The post-surgery patient usually will be required to perform physiotherapy daily routine even discharged from hospital. This system can be seen as providing the foundation monitoring tool doctor to follow up patient's progress.

The disadvantage is the trainee has to wear wearable sensor all the time when performing exercise, as it is necessary to learn the posture of trainee and segment the exercise. Some trainee might not feel comfortable wearing or unaffordable to buy the sensor. This paper can improvise from using sensor into using machine learning

algorithm to segment exercise and detect posture, instead of depending the sensor. A model can trained to automate the segmentation process from machine learning. Moreover, this system does not provide a feedback back to trainee, it can only inform trainee with correct or incorrect posture. This will leave confusion to trainees if the performed posture is classified as incorrect because they do not know how to improve their posture. Therefore, the ground truth model that trained by personal trainer in this paper should enlarge the scale so that it will not just limited into classifying correct or incorrect posture, instead providing constructive feedback to user.

2.5 Qualitative Activity Recognition of Weight Lifting Exercises

This paper is introduced by Eduardo et al. in 2013. In this paper, two approaches were tried out to specify exercises. In the first one, the exercise was specified by using machine learning techniques. Videos of trainee performing a workout in different posture are recorded, some of them corresponding to common mistake made in performing this particular workout. Therefore, the exercise recognition and evaluation are achieved with the classifier training data. The goal of first approach was to assess whether detection of mistakes in weight-lifting exercises by using activity recognition techniques could be achieved. Secondly, the specification of exercise is formalised by training a model that able to translate the instructions in the workout specification into implementable components. The second method able to recognize and evaluate the performed exercise.

2.5.1 Machine Learning Technique Approach

For the first approach, there are few steps taken to recognise and evaluate performed exercise (Figure 2.9). The main concept of this approach is to classify the exercise into different class, such as correct posture class or incorrect posture class. In this paper, 5 classes are defined while first class corresponds to correct performance of the workout while the remaining classes corresponds to other common mistakes made by trainee.

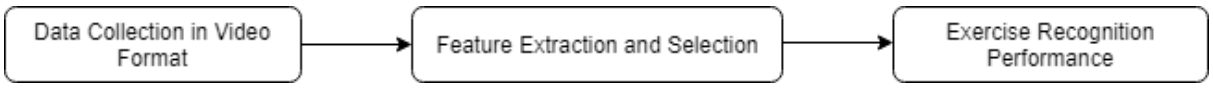


Figure 2.9: Flowchart of Machine Learning Approach

In the first step, video of executing the same workout correctly and with a set of common mistakes with some wearable sensors and machine learning is used to predict every mistake. In this way, training data is used as the activity specification and the classification algorithm as the means to compare the execution to the specification.

For feature extraction, an approach in sliding window with different lengths is used. Features are calculated on the Euler angles, as well as the gyroscope, accelerometer and readings of magnetometer for each step of sliding window. Eight features such as mean, variance, standard deviation and many more are calculated from Euler angles for each four sensors, generated in total 96 derived feature vectors. In order to identify the useful features, feature selection algorithm based on correlation was used. The algorithm was trained to use a “Best First” strategy based on backtracking.

For the final step, the exercise will be classified into different class based on extracted and selected features from previous step. In exercise recognition performance step, Random Forest approach is used due to the noise characteristics in the sensor data. Ten random forests were used and ten trees were implemented in each forest. The trained classifier is tested with 10 fold cross-validation and multiple windows sizes, with 0.5s overlapping. Lastly, the overall result in testing data was 78.2% (Figure 2.10).

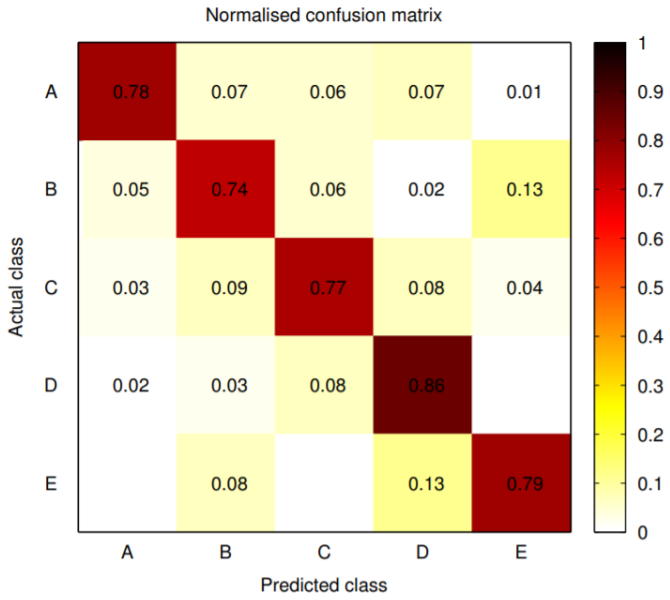


Figure 2.10: Confusion Matrix of Random Forest Algorithm

2.5.2 Exercise Specification Approach

For second approach, there are also few steps taken to recognise and evaluate the performed exercise (Figure 2.11).

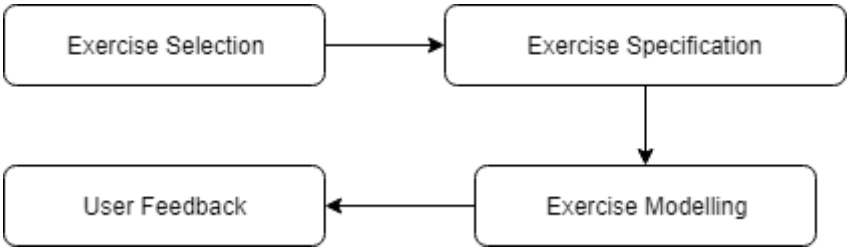


Figure 2.11: Flowchart of Exercise Specification Approach

In exercise selection, the repetitive exercise is segment into each repetition separately. This wat the analysis of each repetition can be done separately. For instance, a front raise exercise involves raising and lowering the dumbbell or barbell by using hands with multiple repetitions, therefore the beginning of the activity is defined as when the trainee started to lift it and the end as when it reached the initial position again.

Next, the exercise should be specified as clearly as possible in natural language. This is because the clearer the specification, the easier to model the exercise. In this paper, the exercise specifications are based on weight lifting book.

In Exercise Modelling step, a model of the recognition mechanism is created according to the components in the framework. There are five components Joint, Operator, Feature, Counter and Classifier (Figure 2.12).

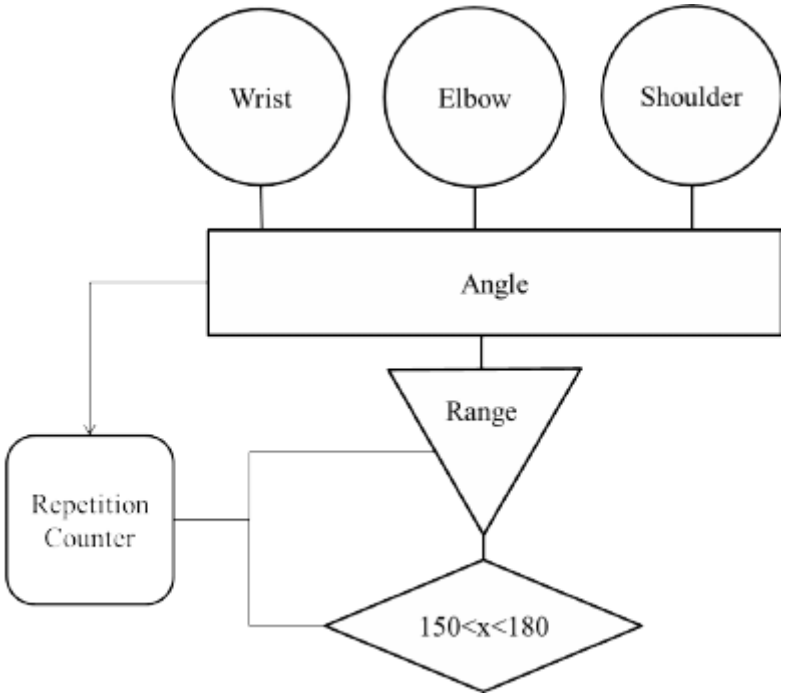


Figure 2.12: Recognition Mechanism Model

In this model, Joint is the coordinate X, Y and Z position of each of the 20 joints such as wrist, elbow, shoulder and many more are provided by Microsoft Kinect. Operators are representing the operations that had been operated on of the XYZ coordinates position of either a single joint or a set of joints. The implemented operators include the distance, angles between human joints and XYV coordinates. Operator can be used to describe the movement in terms angle between joints. Data that provided by operators is then buffer by feature components and perform statistical analysis such as standard deviation, mean, energy, range and etc.

The classification is triggered by Counters. In this approach, there are two ways to classify an exercise: continuously with features being sampled in short intervals or discretely with features being sampled after every repetition. Use Clock Counter if a feature to be monitored after a specified time interval is needed. Use Repetition Counter if the feature to be extracted for each repetition is wanted, you can use a Repetition Counter, which triggers events after detecting a repetition. Lastly, the classification of the quality of the execution of the instruction is performed by Classifier components. These can range from performing simple threshold operations to running more complex machine learning algorithms.

In the user interface, the system is able to give feedback to the user such as giving comments on the posture based on detected joint such as back, feet, elbow and many more to improve the technique for performing the exercise (Figure 2.13).

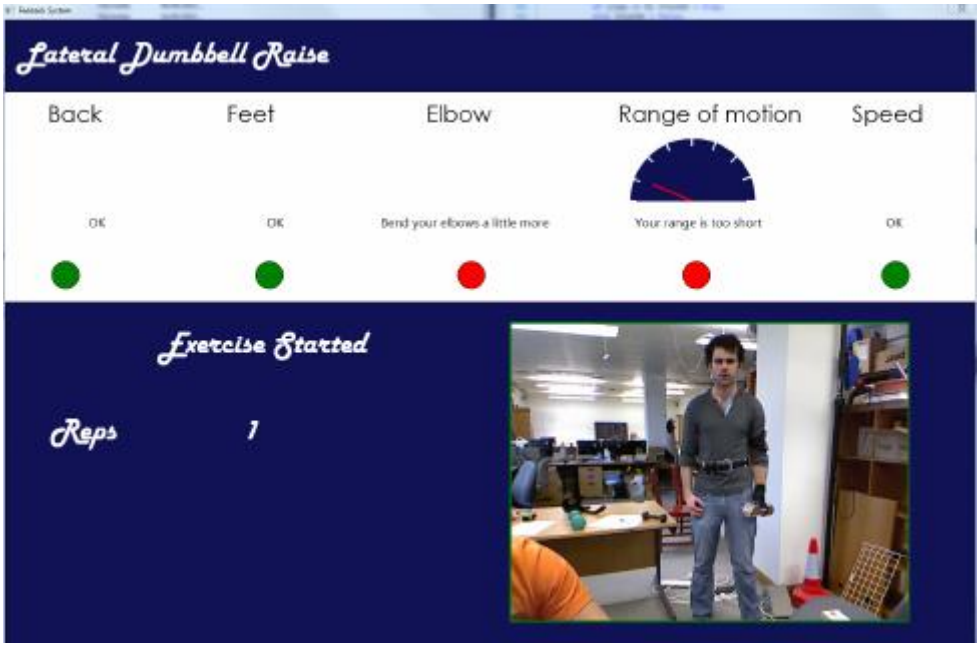


Figure 2.13: User Interface and Feedback System

This approach proved to have few advantages. For second approach, the Classifier component is kept as simple as possible so that it can be easily changed in the future. This is useful for a real life scenario which the trainee might want to make small modification in executing exercise to target different muscle groups. For instance, the general execution way of bicep burl is raising the dumbbell by lower limb of hand all the way to the top. But, there is a possibility that the trainee want to curl the dumbbell halfway to the top to simulate hand grip muscle. Moreover, each instruction is specified separately, therefore the system is suit for many kinds of users, by using smaller or larger instruction subsets. For instance, a trainee with neck or back problems might include specified instructions that designed to monitor of the spine cord. This approach also allows instructions reuse among different common exercises. For instance, there are few exercises that require the trainees not to move their hips. Therefore, the same modelling can be used and implemented for both exercises.

There are few disadvantages in first approach, which is the machine learning technique. First, the classifier training can become a quite troublesome task because it is necessary to record several repetitions including correct and incorrect executions of the same exercise, just to train a robust, user-independent classifier. Here come the problem, recording all possible incorrect execution might be an endless task because of complexity of human posture and motion. Not to mention it is a must to record the training data all over again if the specification changes. This problem is solved by applying second approach in this paper, exercise specification.

Table 2.1: Overall Comparison Between All Approaches

Functionality	2.2.1	2.2.2	2.2.3	2.2.4.1	2.2.4.2	My Proposed Approach
Sensor Dependency	No	Yes	Yes	Yes	No	No
Camera Dependency	Yes	No	No	Yes	Yes	Yes
Evaluation Process	Partial Automation	Full Automation	Partial Automation	Partial Automation	Partial Automation	Partial Automation
Limitation to Certain Types of Exercise	No	Yes	No	No	No	No
Exercise Recognition	Yes	No	Yes	Yes	Yes	Yes
Evaluation Feedback	Detailed	Partly Detailed	Not Detailed	No Feedback	Detailed	Detailed
Segmentation of Repetitive Exercise	Done by user	Done by system	Done by system	Done by system	Done by system	Done by user

CHAPTER 3 SYSTEM DESGIN

3.1 Chapter Overview

This chapter will explain the general methodologies and list down the technologies used in the methodologies step.

3.2 System Overview

This project follows standard machine learning pipeline process. (Figure 3.1)

Figure 3.1 Machine Learning Pipeline Flowchart

3.2.1 Data Collection

Before started to collect the dataset, we need to define the data that we need to collect. Therefore, a short interview is conducted with Mr Seong, a certified workout trainer that currently works in FEEX Fitness Ideal Sdn Bhd. A lot of data has been gathered during the interview included examples of incorrect posture performed by trainee. A list of common improper postures was listed down and will be recognised and evaluated by the trained model. Mr Seong also demonstrated the improper and proper postures to make sure the dataset contains correct proper and improper posture workout video.

After I defined the list of class, I need to collect dataset in order to train the model, therefore several workout videos are recorded and will be used in Model Training stage. The system requires trainees to upload their exercise videos with any common format such as MP4, MOA, AVI and etc.

A total workout dataset is recorded in MP4 format with 2.19GB that consists of three different workouts (Figure 3.2). There are total 845 repetitions with 265 repetitions in Bicep Curl class, 280 in Front Raise class and 300 repetitions in Shoulder Press class. There are totally seven classes including six improper postures and one proper posture in Bicep Curl class, six classes including five improper postures and one proper posture

in Front raise class and six classes including five improper postures and one proper posture in Shoulder Press class.

Figure 3.2: Dataset Video

3.2.2 Data Preparation

After the collection of dataset, I have performed some operations in Data Preparation Stage (Figure 3.3).

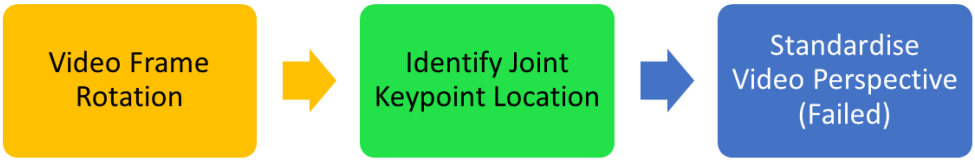


Figure 3.3: Overview Process of Data Preparation Stage

First, I realised OpenCV is having problem in orienting video in correct angle. Some videos that captured in portrait by phone will view as landscape in OpenCV (Figure 3.4).

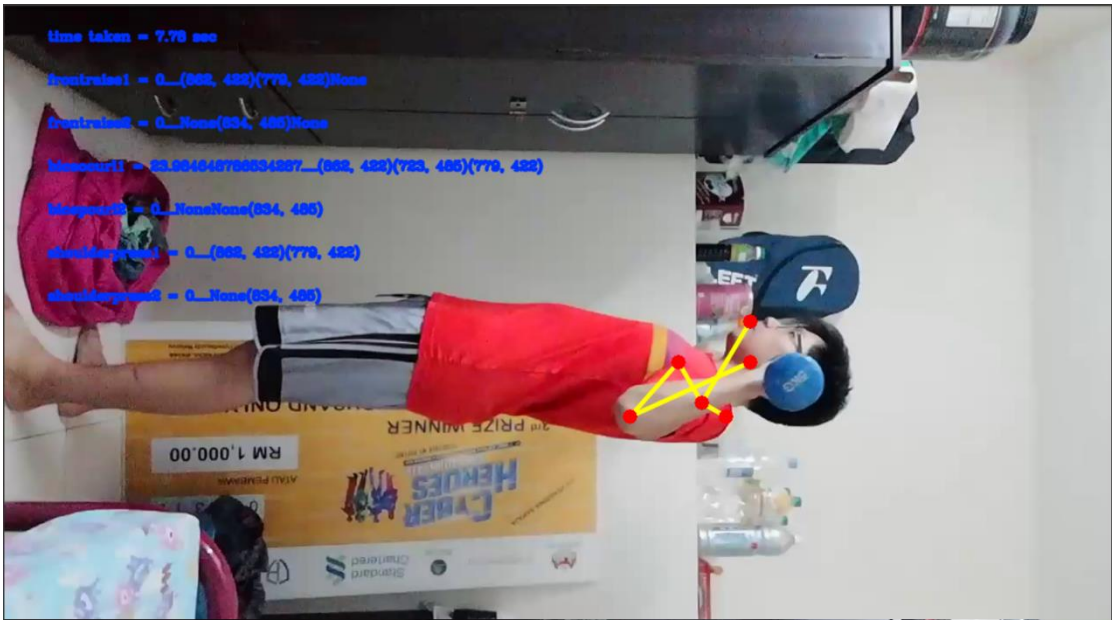


Figure 3.4: Showing in Landscape Instead of Portrait

OpenPose has a flaw which it does not work well in disoriented frame. As shown in Figure 3.4, OpenPose was unable to locate correct keypoints and identifying correct keypoints is the important step for workout recognition.

To solve this challenge, the system must read the metadata that name rotation, this metadata tells the video player to rotate the video for how many angles. Therefore, moviepy is deployed to read the value of rotation metadata is rotate the video according to the value (Figure 3.5).

```
import subprocess
import shlex
import json
from moviepy import *

def get_rotation(file_path_with_file_name):
    cmd = "ffprobe -loglevel error -select_streams v:0 -show_entries stream_tags=rotate -of default=nw=1:nk=1"
    args = shlex.split(cmd)
    args.append(file_path_with_file_name)
    # run the ffprobe process, decode stdout into utf-8 & convert to JSON
    ffprobe_output = subprocess.check_output(args).decode('utf-8')
    if len(ffprobe_output) > 0:  # Output of cmdis None if it should be 0
        ffprobe_output = json.loads(ffprobe_output)
        rotation = ffprobe_output

    else:
        rotation = 0

    return rotation

get_rotation("/content/drive/My Drive/FYP/Test/VID_20191120_020337.mp4")
```

```
270
```

Figure 3.5: Solution to Disoriented Video

In Figure 3.5, the value of rotation metadata is 270 angle. All left to do is to rotate the video frame (Figure 3.6).

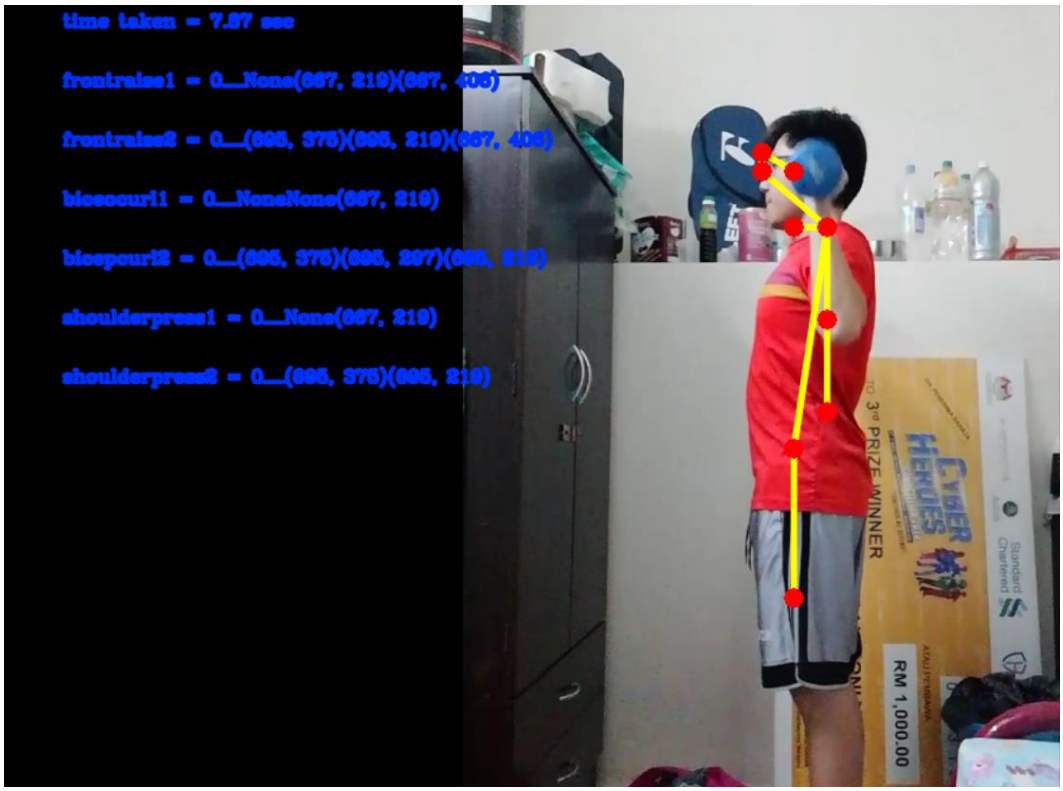


Figure 3.6: Correct Orientation of Video

OpenPose model will apply on these videos to identify the joint keypoints. A total of eighteen keypoints will be identified in every video if a human object is detected (Figure 3.7). To be specific, the output of OpenPose model is a list of geometric position of 18 human keypoints and these geometric positions are still considered as raw data that do not provide a lot of useful information for Model Training stage.

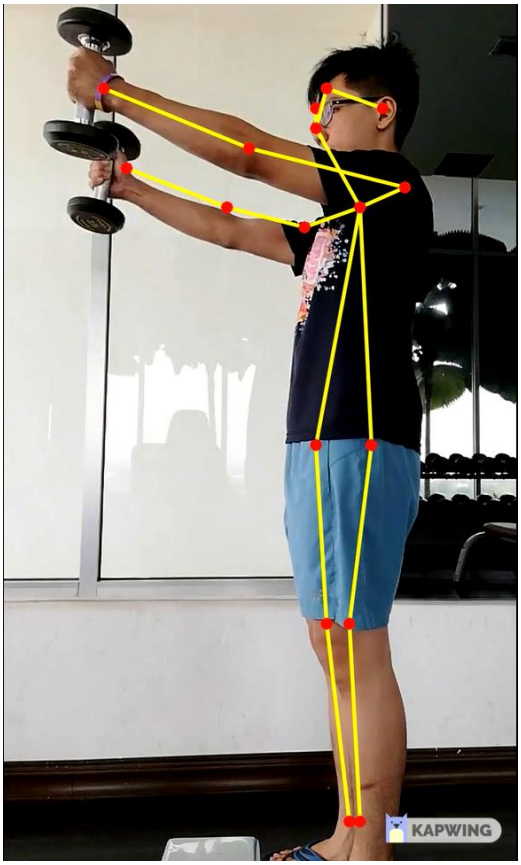


Figure 3.7: 18 Keypoints Detected

The next problem will be standardising the video perspective. For example, Figure 3.8 and Figure 3.9 are showing two different perspectives even both users are facing same side to the camera.

Figure 3.8: Record Video from Upper View

Figure 3.9: Record Video from Lower View

Situation like this may cause slightly inaccurate calculated angles. Figure 3.8 and Figure 3.9 are posing the same posture but the red line show that angle are slightly different between these two figures. This may lead to wrong prediction from classifier. Therefore, in order to solve this problem, we need to standardise the perspective of every uploaded video. Multiple solutions are brought to my mind and I decided to try one of them.

I used the OpenCV warp perspective function to adjust the frame. After deploying this solution, I realised this method is only good for object that has sharp edge in every corner but did not performed well on human object (Figure 3.10).

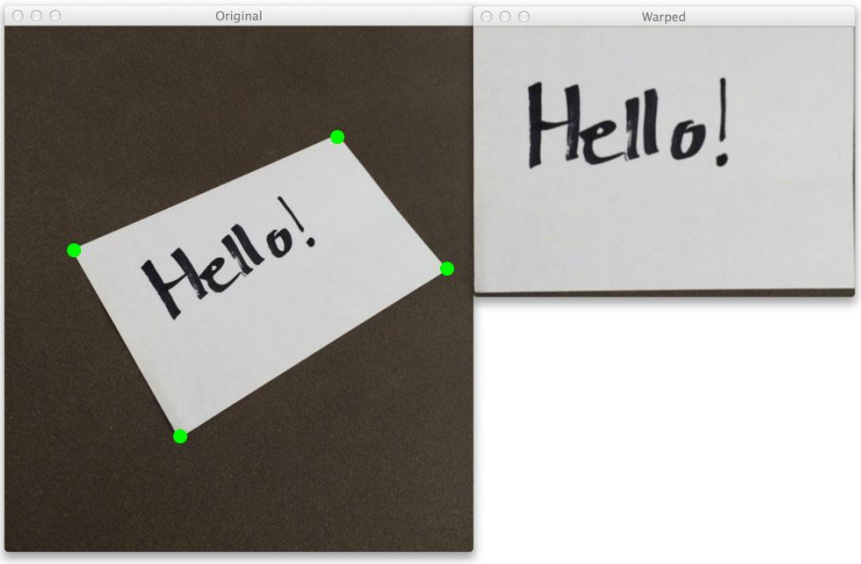


Figure 3.10: Applying Warp Function on Object that Contains Sharp Edge

I tried this solution to uploaded video and the result is not good (Figure 3.11). I tried to use different combinations of keypoints but neither of them returned a satisfactory answer.

Figure 3.11: Applied Warp Function on Human Object

The next solution is building up another model to evaluate the perspective to get the correct ratio of human torso length between human leg. The most common way is to use a Gaussian function to record down the ratio, if the ratio is not placed in normal distribution then the video need to adjust its perspective. The solution itself is a very complicated process and this problem looks like another big project instead of treating it like a sub-project. Therefore, due to time constraint I cannot work on this solution.

In conclusion, the whole situation leaving me no choice to leave this implementation issue in my system. Nevertheless, as I mentioned above, the calculated angle is just slightly different and it does not bring a huge negative impact on classifier performance.

3.2.3 Feature Selection

In order to train a model that able to recognise different type of workout and improper posture, characteristics of each workout repetition need to extract from raw data which is the keypoints location as feature vector. Since I need to train two different types of model for performing different purpose, the selected features for these two types model are different as well.

For model that classify three types of workout, the obvious characteristics of each workout are the angles between angles between the whole workout movement. For instance, the characteristics for Front Raise will be the angle between hand and torso, while Bicep Curl will be the angle between lower hand and upper hand. These angles are calculated for each frame should the flow of movement can be captured. The differences between these workout types are very huge and easy to spot, therefore calculated angles are enough to be the feature vector.

Secondly, for the model that evaluate different kind of improper posture within same workout, the characteristics between these postures are difficult to spot therefore I need to extract higher level of feature vector. I created some thresholds to hold a range of angles or even the maximum or minimum angle in that repetition. For instance, Bicep Curl has a common improper posture which is swinging the hand from the back to gain momentum to lift the weight up. The higher level feature is a boolean counter that identifies the angle between hand and torso, if it is negative then the hand swing behind the back (Figure 3.12). Model theoretically should be easier to classify the incorrect posture with the help of high level feature vector.

Figure 3.12: Angle between Hand and Torso

An issue happened in this phase, which is the angles calculated between keypoints are not accurate (Figure 3.13).

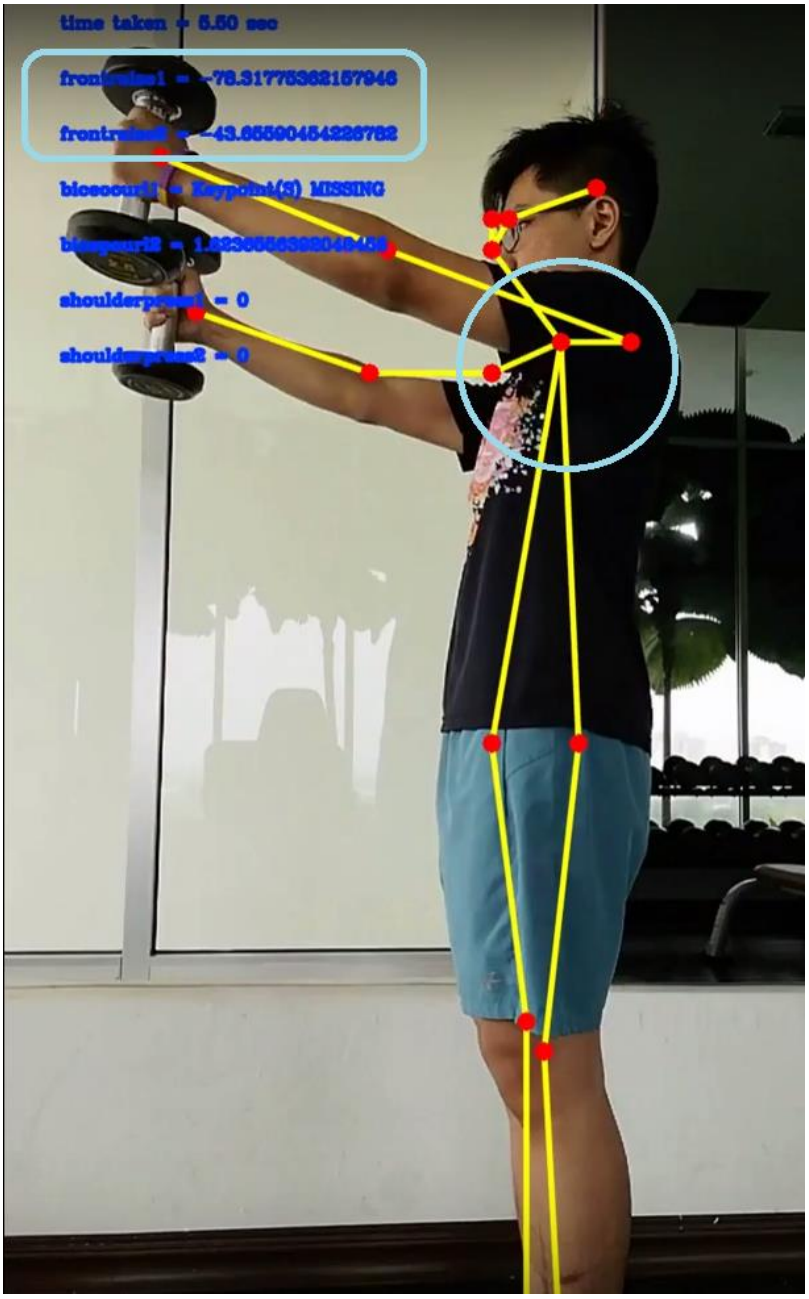
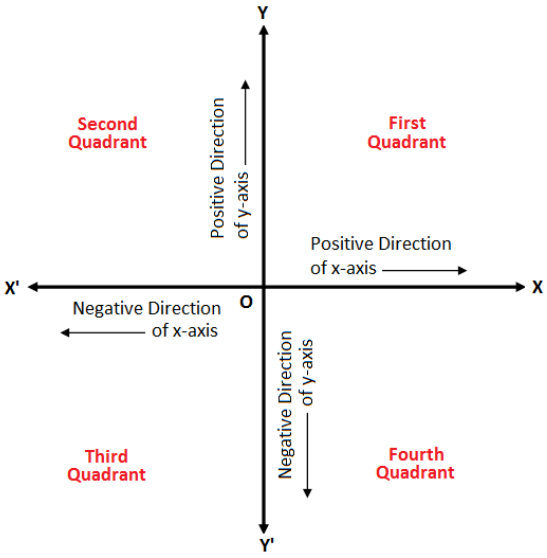


Figure 3.13: Wrongly Calculated Angles between Keypoints

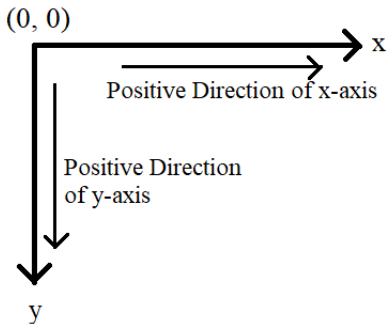
The calculated angle in light blue circle are stating -76 degrees and -43 degrees, while by just looking at it the angle shouldn't be negative at all. After various time of debugging, I realised that the method in coordinating between OpenCV and traditional math is different. In order to calculate the angle, I applied this formula:

$$\tan\theta = \left|\frac{m1 - m2}{1 + m1m2}\right|$$

Where this formula is workable in condition like this:



In order to calculate correct and positive angle, the coordinates must always stay in First Quadrant. However, in OpenCV, the coordination is stored in a way like this:



Therefore, when I use the coordination generated by OpenCV, it is actually staying in Fourth Quadrant, rather than First Quadrant. As the cause of problem has been found out, I started to work on thinking the solution. The solution for this issue is simple, simply change y-axis in every coordinate to positive and the coordinate is now staying

in First Quadrant. Figure 3.14 is showing that the angles are calculated correctly. The angles are 111 degrees on right hands to hip and 100 degrees on left hands to hip.

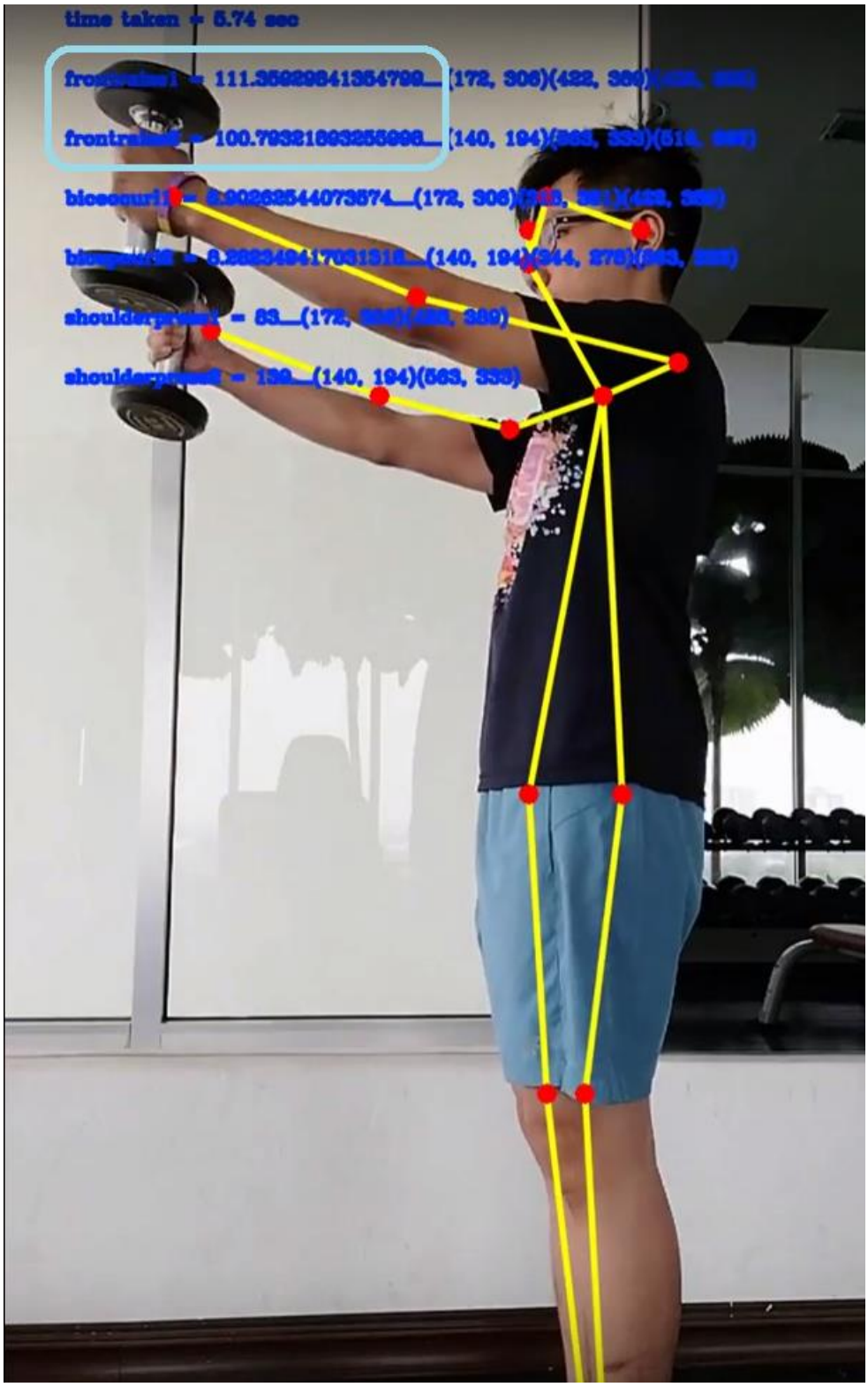


Figure 3.14: Calculated Angles Correctly Between Keypoints

Moreover, another issue popped out after fixing the inaccurate if calculated angle. The uploaded video is recorded from specific perspective, which is side to camera. In this case, half of the body will not be seen in whole video, either left or right. OpenPose model is unable to return the location for half disappeared body keypoints and system can't calculate the angle. This leads to a lot of null values or zero values in the training data, which is consider as noise. In order to maximise the model accuracy, the best scenario is to clean the data until it is noise-free.

Therefore, in order to solve this problem, first the system need to know whether the user is showing left side or right side to the camera. This can be easily done by checking the keypoints detected by OpenPose model. If the number of keypoints detected from left body is a lot higher than right body, then the system assumes the user is showing left side to the camera (Figure 3.15). The system is also able to recognise when user is facing front to the camera, when number of right side body keypoints are almost equally to the left side body. By applying this solution, the training data now contained zero null value.

Figure 3.15: Determining Body Side

3.2.4 Model Training

Multiple machine learning algorithms are tested to train the classifier, such as KNN, SVM and logistic regression, the comparison results will be discussed in Chapter 5. The algorithm with highest accuracy is chosen to train all classifiers. One classifier is trained to classify workout type while another three classifiers are trained to identify any improper posture based on the predict workout type from previous model. Parameter-tuning is also will be discussed in Chapter 5.

3.2.5 Model Evaluation

Confusion matrix, f1-score, accuracy and many more are generated to evaluate every trained model to make sure it is not overfitting or underfitting. If the model is overfitting or underfitting, Feature Selection stage will repeat again to make sure the feature vector is close to noise-free and model is able to learn something useful from it.

3.2.6 Model Deployment

I tried to use Flask, which is a micro web framework written in Python to deploy the model. Figure 3.16 shows the user interface that I implemented to deploy the model. But I faced a solution that I can't resolve.

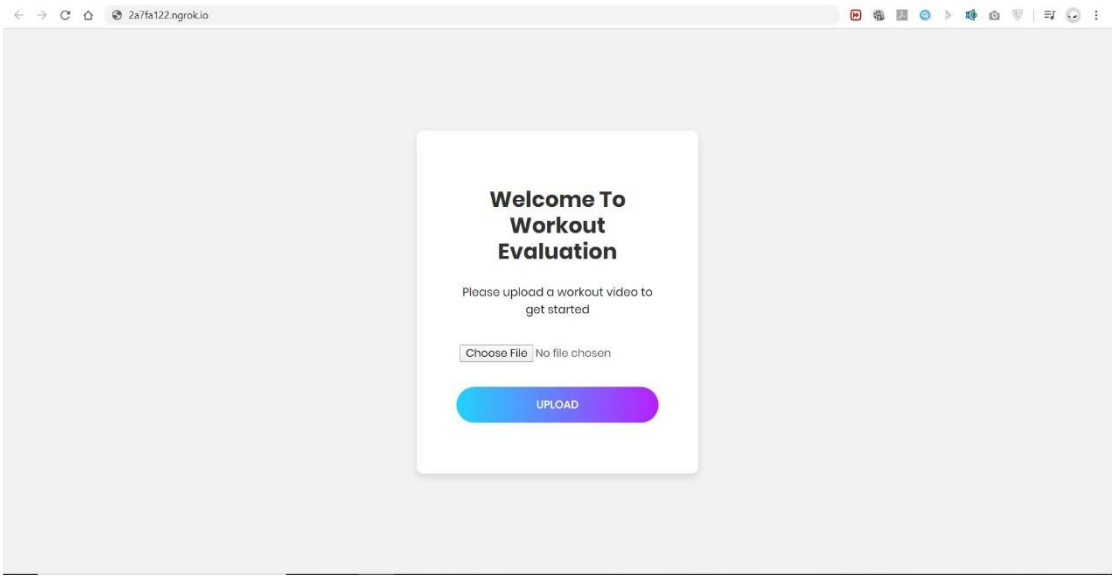


Figure 3.16: User Interface of Model Deployment

OpenPose model takes a very long time to return the keypoints location. I have done a lot of research on how to shorten the process time but I can't find anything helpful. The user is able to upload the workout video, but the prediction process is taking too long time that the server disconnected due to timed-out issue (Figure 3.17).

Figure 3.17: Error Faced in Deploying Model

Everything is working fine, the model is able to return the predicted output, model is able to deploy by using Flask. But this deployment method can only be achieved if I can find a way to shorten the time for OpenPose model to retrieve the keypoints location.

Therefore, I choose the most basic deployment method, which is launching the product by calling it in terminal. User need to download whole folder or some necessary files such as classifiers and the coding for running the classifiers. Once the user get everything ready then they can type command to launch the product inside Anaconda Prompt. This method does bring some troubles, user need to do some setups before launching the product as I mentioned in Chapter 3.2.6. Figure 3.18 shows the result from model deployment.

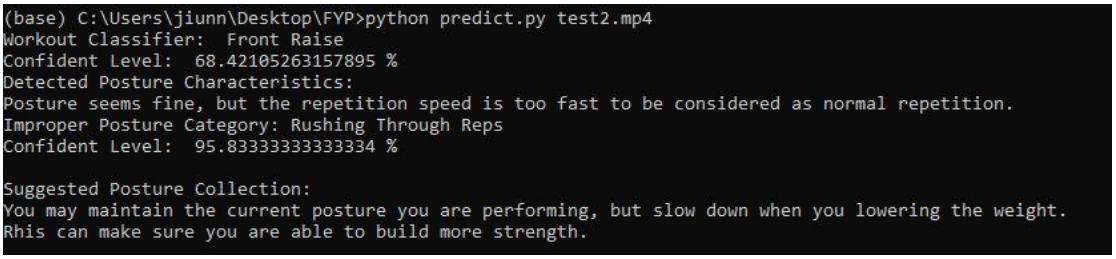


Figure 3.18: Deployment Result

Figure 3.19 shows the complete flow of deploying model.

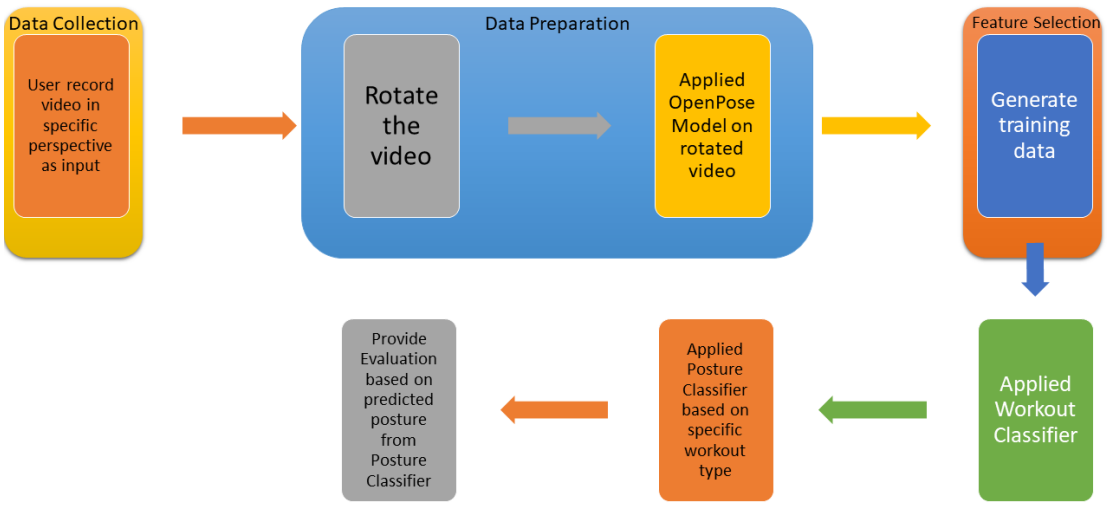


Figure 3.19 Overview of Model Deployment

The first phase of Model Deployment is Data Collection and it is similar to the Data Collection in Chapter 3.2.1, the difference is the uploaded video is input, but not dataset. User is allowed to upload video in common format such as AVI, MP4 and many more.

From Data Preparation phase to Feature Selection are very similar to Chapter 3.2.2 and 3.2.3. This is because the classifiers need the feature vector to predict the input.

For fifth and sixth stages, both of them are applying the trained model from Chapter 3.2.4. Different Posture Classifier will be selected based on output from Workout Classifier. For example, if the output from fifth stage is Bicep Curl, then Bicep Curl Posture Classifier will be selected to predict any improper posture (Figure 3.20).

```
(base) C:\Users\jiunn\Desktop\FYP>python predict.py test.mp4
Workout Classifier:  Bicep Curl
Confident Level:  66.66666666666666 %
Posture Evaluation:  B5 Bicep Curl
Confident Level:  90.38461538461539 %
```

Figure 3.20: Predicted Output from Trained Classifiers

Lastly, system will provide suitable suggestions back to user based on the output from Posture Classifier (Figure 3.21).

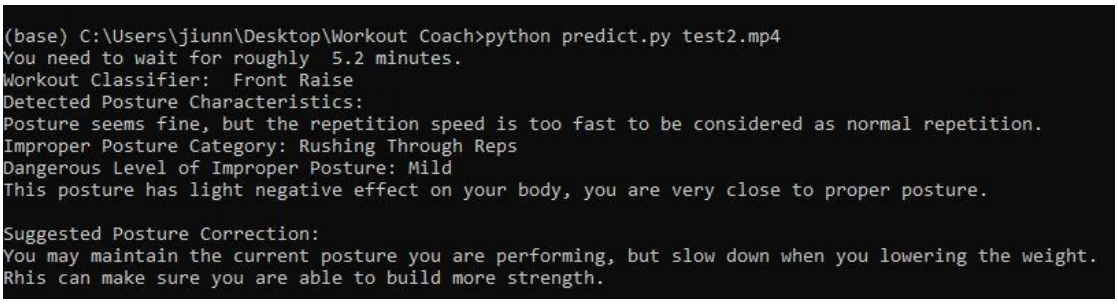


Figure 3.21: Deployment Result

In conclusion, this system is able to deploy in two platform. First, deploy in local machine. User is able to utilise the final product by downloading the whole folder from GitHub. Simply change to correct directory pathway and type "python predict.py *Uploaded Video Name*" to use the product. Please refer to Chapter 4.2 for system requirement to launch the product.

Secondly, user upload whole folder to google drive and open a Google Colab notebook to run the same prompt "python predict.py *Uploaded Video Name*" to launch the product. The differences between two platforms are listed below (Table 3.1):

Table 3.1: Differences between Google Colab Platform and Local Machine Platform

Google Colab	Local Machine
No setup needed.	Need to setup few things, such as download Anaconda Prompt, Python and etc.
Boosted by GPU that shorten the prediction process.	If local machine does not have GPU then prediction process will take longer time.
Network dependent	Network independent

Please refer to appendix for full instructions to deploy the model in two different platforms.

CHAPTER 4 DESIGN SPECIFICATIONS

4.1 Chapter Overview

In this chapter, system and user requirements will be covered. Verification plans are listed in this chapter and the tested results are discussed in Chapter 5. Some implementation issues will also be covered in this chapter too.

4.2 System Requirements

- Camera to record workout video
- Python3
- Anaconda Prompt (any version)
- OpenCV 4.1.0 (installed in Anaconda environment)
- OpenPose 1.5.1
- Scikit-Learn 0.22.2 (installed in Anaconda environment)

4.3 User Requirements

- The recorded video can only contain 1 person
- The video must record from correct perspective according to different workout.
- Avoid complex background if possible to maximise accuracy.
- Whole body of user is recommended to be fully captured in the video for achieving maximum accuracy.
- At least upper body of user is captured in the video.
- Capture the video in bright background which the visibility of user is clear.
- The resolution of video should be at least 720p.

4.4 Verification Plans

The system is able to recognize the type of workout in a video with high accuracy and consistency. However, this may affected by different situation and it may results in false detection and other problem when performing recognition. Few situations are explained as below:

- Capture a video of workout with a complex background, the model should able to find out the type of workout and display the result.
- Capture a picture of workout in a bright background, the system should able to find out the type of workout and display the result.
- Capture a picture of workout in a dark background, the system should able to find out the type of workout and display the result.

Hence, there will be few verification step to ensure the accuracy and consistency while recognizing the object. Few verification steps are explained as follow:

1. Complex background

Table 4.1 Verification P1

Procedure Number	P1
Method	Testing
Applicable Requirements	Recognise the type of workout in a background cluttered with various non-workout objects in the video frame.
Purpose	To make sure model is able to perform well in complex background
Items Under Test	Person performing workout
Precautions	The video should at least captured whole upper body started from hip to head.
Special Conditions/Limitations	The video need to recorded in correct perspective.
Equipment/Facilities	Camera
Data Recording	None
Acceptance Criteria	Model predicts the type of workout correctly
Procedures	1. Record a video of performing workout 2. Upload it to model 3. Model predict the type of workout
Troubleshooting	Repeat the procedure
Post-Test Activities	None

2. Bright background

Table 4.2 Verification P2

Procedure Number	P2
Method	Testing
Applicable Requirements	Recognise the type of workout in a bright background
Purpose	To make sure model is able to perform well in bright environment
Items Under Test	Person performing workout
Precautions	The video should at least captured whole upper body started from hip to head.
Special Conditions/Limitations	The video need to record in correct perspective.
Equipment/Facilities	Camera
Data Recording	None
Acceptance Criteria	Model predicts the type of workout correctly
Procedures	4. Record a video of performing workout 5. Upload it to model 6. Model predict the type of workout
Troubleshooting	Repeat the procedure
Post-Test Activities	None

3. Dark background

Table 4.3 Verification P3

Procedure Number	P3
Method	Testing
Applicable Requirements	Recognise the type of workout in a dark background
Purpose	To make sure the model is able to perform well with video in dark environment.
Items Under Test	Person performing workout
Precautions	The video should at least captured whole upper body started from hip to head.
Special Conditions/Limitations	The video need to recorded in correct perspective.
Equipment/Facilities	Camera
Data Recording	None
Acceptance Criteria	Model predicts the type of workout correctly
Procedures	7. Record a video of performing workout 8. Upload it to model 9. Model predict the type of workout
Troubleshooting	Repeat the procedure
Post-Test Activities	None

4.5 Implementation Issues and Challenges

The major and biggest issues of this project is OpenPose model takes a very long time to detect keypoints, by having the hardware specifications of Intel I7 6th and GTX 950, it took around 9 seconds to identify joint keypoints in one frame. If the FPS of the recorded video is 30, it will take OpenPose 45 minutes to identify all keypoints in every frame. In order to solve this time consuming issue, the project is now constructed in Google Colab, where Google provides strong GPU and TPU for free. In Goole Golab, I am able to enjoy GTX 1050, which runs faster compare to local machine. The comparison will be 9 seconds per frame in local machine and 6 seconds per frame in Google Colab. These three seconds differences might not look big, but when processing huge amount of data, it will certainly save a lot of time.

There is also an issue after using Google Colab, which is the kernel will automatically disconnect after 12 hours of running, in order to prevent data mining activities. I learnt this lesson in a hard way, the first attempt in training model was going to take approximately 14 hours. In the 12th hour in training model, which is around 90% completion, the Google Colab disconnected and I lost all the progress I made for past 12 hours. Therefore, I categorise the dataset into few parts, train them one by one, and make sure each part does not exceed 12 hours of training time.

CHAPTER 5 EXPERIMENT & EVALUATION

5.1 Chapter Overview

This chapter will briefly showing the results from this project, including the result from trained model and deployment of model.

5.2 Workout Classifier

5.2.1 Model Results

In order to get the best accuracy for workout classifier, I have tested most of the machine learning algorithms for classification problem, such as KNN, SVM, Naïve Bayes, Logistic Regression and Decision Tree.

Figure 5.1, 5.2, 5.3, 5.4 and 5.5 are showing the differences of result between them.

```
Confusion Matrix:

[[621  65   0]
 [138 246   7]
 [  2  11 186]]


CLassification Report:

              precision    recall  f1-score   support

    bicepcurl       0.82      0.91      0.86       686
   frontraise       0.76      0.63      0.69       391
shoulderpress       0.96      0.93      0.95       199

     accuracy                           0.83      1276
    macro avg       0.85      0.82      0.83      1276
 weighted avg       0.82      0.83      0.82      1276



Accuracy:
0.8252351097178683
```

Figure 5.1: Results of Logistic Regression Classifier

```
Confusion Matrix:

[[620  66   0]
 [134 253   4]
 [  0  13 186]]


CLassification Report:

              precision    recall  f1-score   support

    bicepcurl       0.82      0.90      0.86       686
   frontraise       0.76      0.65      0.70       391
shoulderpress       0.98      0.93      0.96       199

     accuracy                           0.83      1276
    macro avg       0.85      0.83      0.84      1276
 weighted avg       0.83      0.83      0.83      1276



Accuracy:
0.8299373040752351
```

Figure 5.2: Results of SVM Classifier

```
KNN:
Confusion Matrix:

[[   0 1601   15    0]
 [   0    6  719    0]
 [   0    0    0  431]]


CLassification Report:

              precision    recall  f1-score   support

    bicepcurl       1.00      0.99      0.99      1616
   frontraise       0.98      0.99      0.99       725
shoulderpress       1.00      1.00      1.00       431

     accuracy                           0.99      2773
    macro avg       0.74      0.75      0.74      2773
 weighted avg       0.99      0.99      0.99      2773



Accuracy:
0.992066354129102
```

Figure 5.3: Results of KNN Classifier

```
Decision Tree:
Confusion Matrix:

[[   9 1590   17    0]
 [   6   15  704    0]
 [   0    0    0  431]]


CLassification Report:

              precision    recall  f1-score   support

    bicepcurl       0.99      0.98      0.99      1616
   frontraise       0.98      0.97      0.97       725
shoulderpress       1.00      1.00      1.00       431

     accuracy                           0.98      2773
    macro avg       0.76      0.99      0.77      2773
 weighted avg       0.99      0.98      0.99      2773



Accuracy:
0.9830508474576272
```

Figure 5.4: Results of Decision Tree Classifier

```
Naive Bayes:
Confusion Matrix:

[[   9 1577   25    5]
 [   6  469  226   24]
 [   0   21   56  354]]


CLassification Report:

              precision    recall  f1-score   support

    bicepcurl       0.76      0.98      0.86      1616
   frontraise       0.74      0.31      0.44       725
shoulderpress       0.92      0.82      0.87       431

     accuracy                           0.78      2773
    macro avg       0.62      0.78      0.57      2773
 weighted avg       0.78      0.78      0.75      2773



Accuracy:
0.7782185358817165
```

Figure 5.5: Results of Naïve Bayes Classifier

Table 5.1 summarises the result of every machine learning algorithm.

Table 5.1: Comparison between Results of Different Machine Learning Algorithm

Machine Learning Algorithm	Accuracy
Logistic Regression	82.52%
SVM	82.99%
KNN	99.21%
Decision Tree	98.31%
Naïve Bayes	77.82%

According to Table 5.1, KNN algorithm shows the best accuracy among other six algorithms. Therefore, I chose KNN to train workout classifier.

5.2.2 Fine-Tuning of Parameter

Fine tuning of parameter is also performed on KNN classifier to maximise accuracy, by adjusting the number of nearest neighbour parameter, which is K-value. Figure 5.6 and 5.7 are showing the difference in accuracy between different parameter setup.

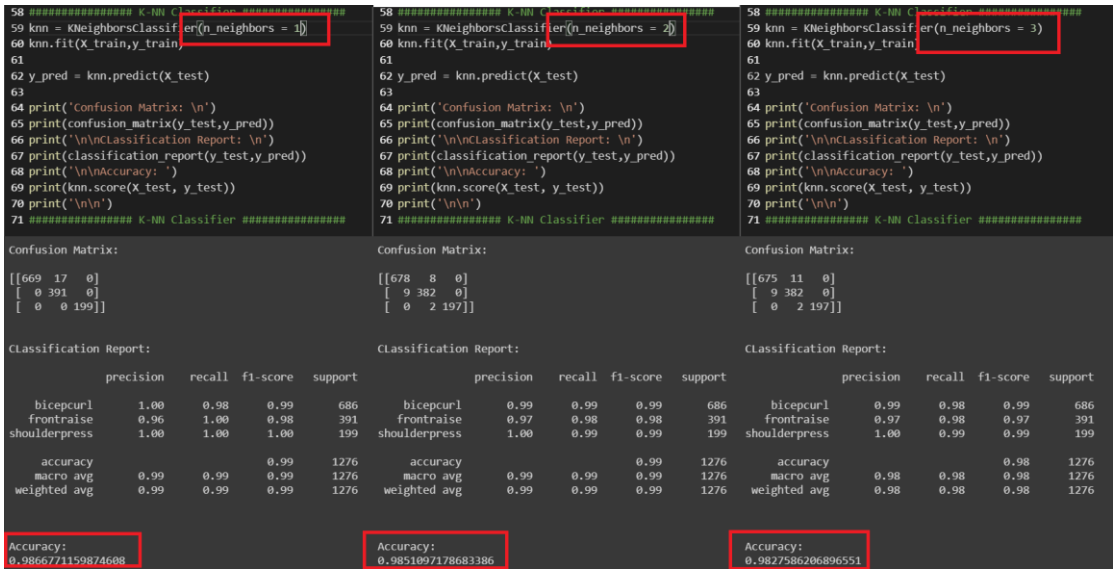


Figure 5.6: Comparison of Accuracy between Different Parameter Tuning (1)

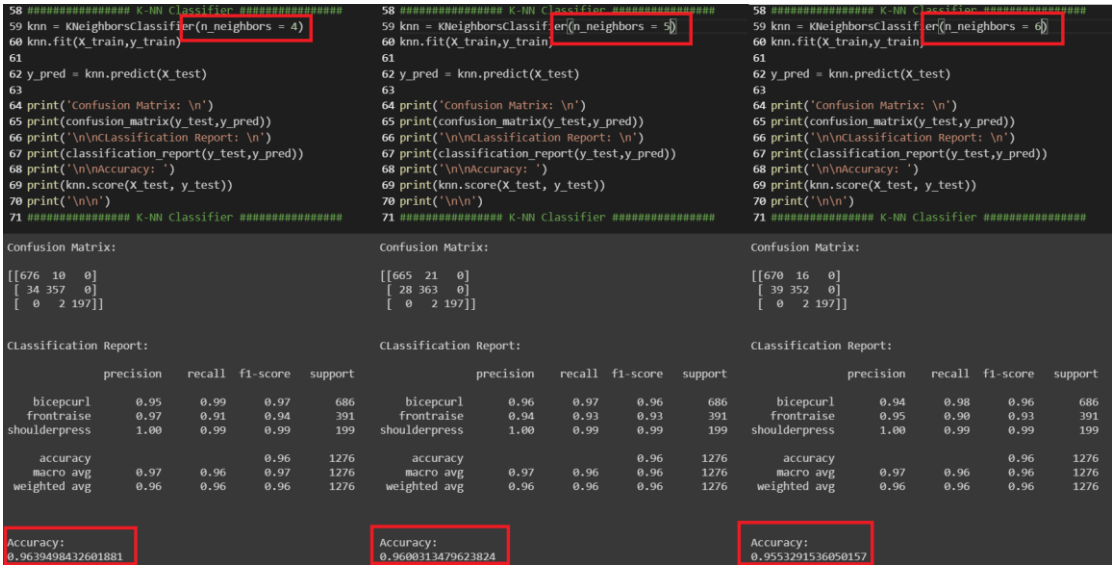


Figure 5.7: Comparison of Accuracy between Different Parameter Tuning (2)

Table 5.2 shows the result comparison between multiple values for K-value parameter.

Table 5.2: List of Accuracy According to Number of Nearest Number

Nearest Neighbour Number, K-value	Accuracy, %
1	98.7
2	98.5
3	99.2
4	96.4
5	96.0
6	95.5

5.2.3 Deployment of KNN Recognition Model

Next is the deployment of KNN recognition model. Figure 5.8 is showing that the model is able to recognise shoulder press workout correctly, even in complex background.

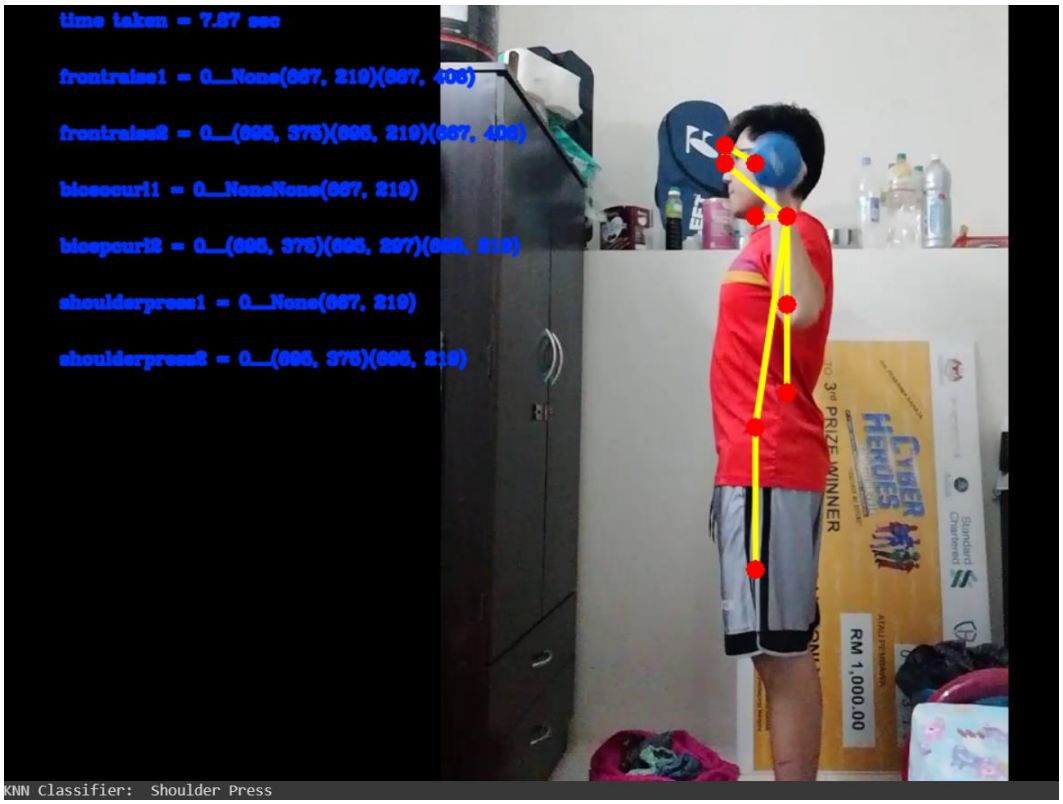


Figure 5.8: Result of Prediction from KNN Classifier on Shoulder Press Workout

There are few special cases that causing the wrong prediction from KNN recognition model. One of them is the unstable performance of OpenPose in identifying keypoints. (Figure 5.9)

Figure 5.9: Wrong Prediction from KNN Classifier (1)

As the wrongly identification of keypoints by OpenPose, the calculated angle will be inaccurate and thus KNN classifier is unable to predict correctly.

The next special case is the brightness of video. OpenPose does not work well in dark environment well the visibility of person performing workout is low. Figure 5.8 is showing OpenPose misidentify most of the keypoints and thus KNN classifier could not predict the correct workout. The workout shown in Figure 5.10 is bicep curl but predicted value is front raise.

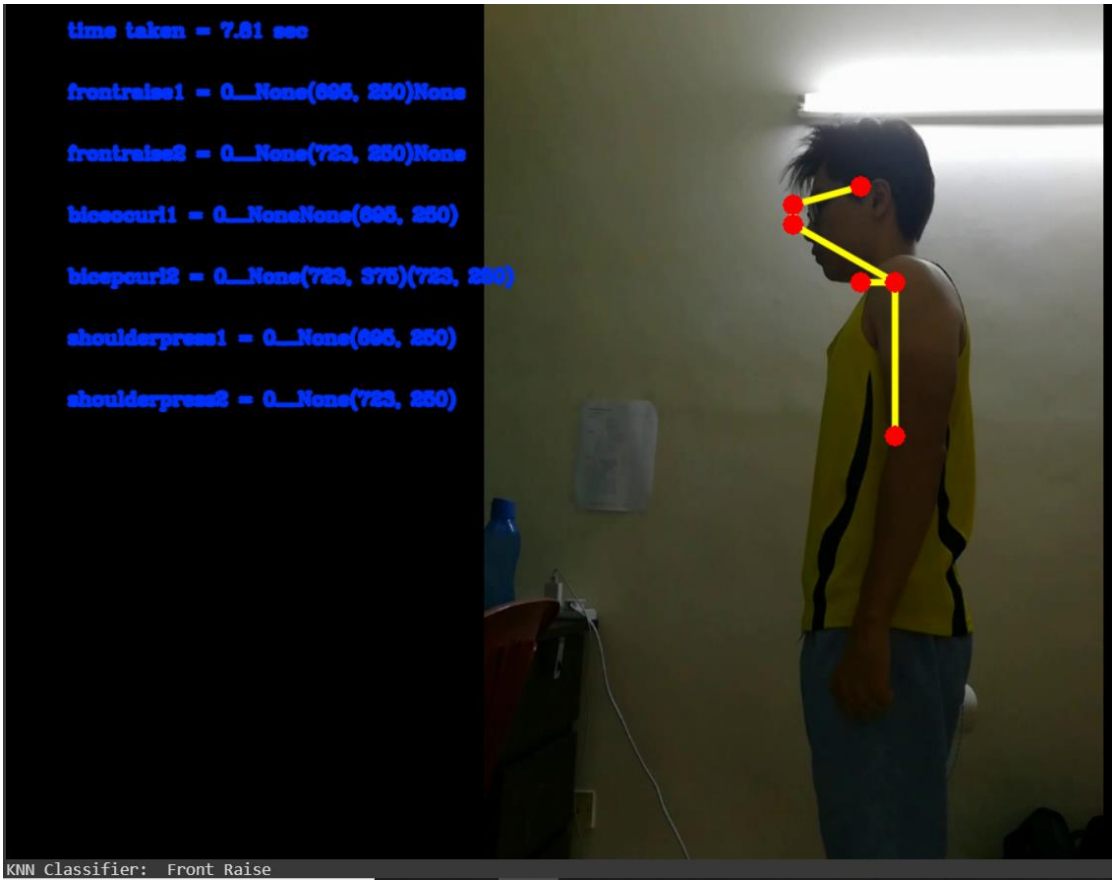


Figure 5.10: Wrong Prediction from KNN Classifier (2)

As long as the OpenPose model is able to identify keypoints correctly, KNN classifier has a very high possibility to predict the workout correctly. In Figure 5.11, the person who performing workout is in bright environment and has high visibility. KNN classifier has also predicted the workout correctly.

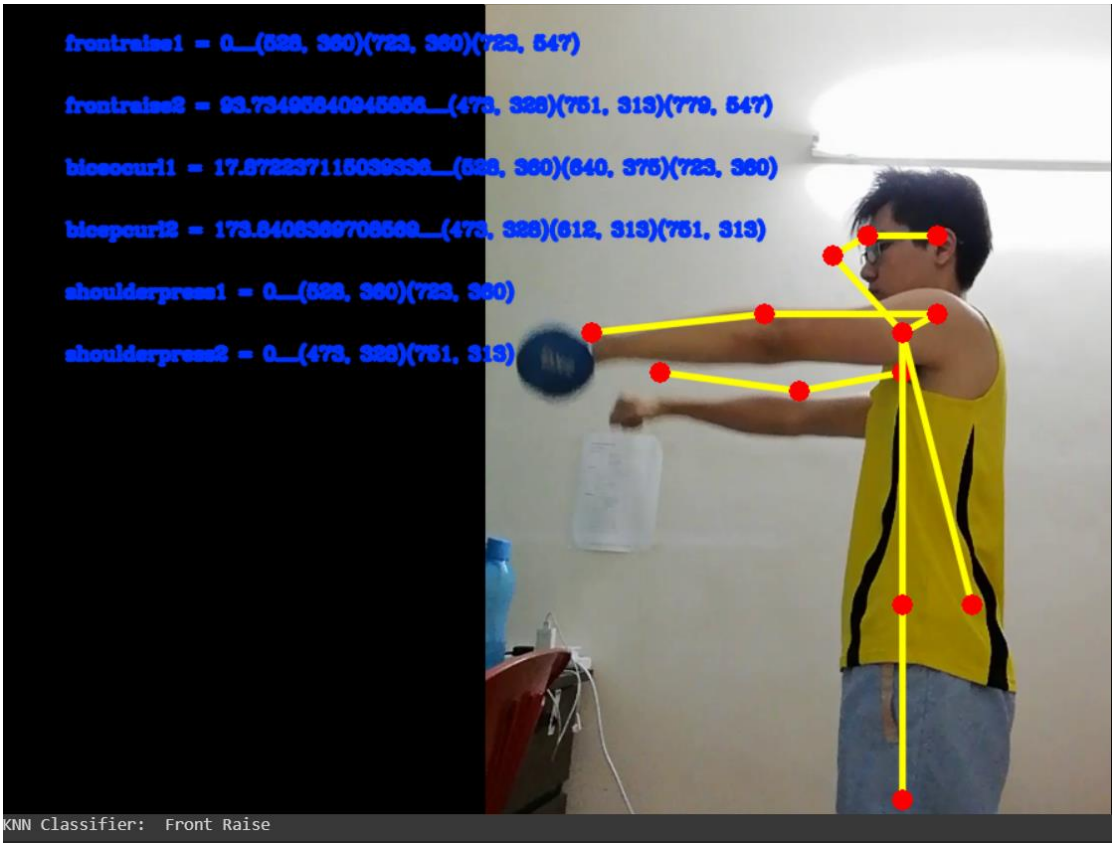


Figure 5.11: Correct Prediction from KNN Classifier

5.3 Posture Classifier

5.3.1 Model Results

As I mentioned in 5.2.1, the feature vector is generated to suit the algorithm structure for KNN, and the results have proven me I am correct. Therefore, I will not experimenting with other machine learning algorithm such as SVM, logistic classifier and many more. I will be focusing in generating higher level feature vector and fine-tuning the parameter the get the highest accuracy for every classifier. For initial training phase, I use three for K-value as default.

Figure 5.12 is showing the result for Bicep Curl posture classifier that trained by the combination of low level training data and high level training data, with the accuracy of 80.69%.

```
Confusion Matrix:

[[360   2   2   3  11  58  10]
 [  4  62  17  16   9   1   5]
 [  1   2 376   4   7   2   4]
 [  5  13   6 201   9   6   6]
 [ 17  16  15   9 439  30  34]
 [ 56   7  10   8  18 411  15]
 [  9  12  17   8  23   8 303]]


CLassification Report:

              precision    recall  f1-score   support

b1_bicepcurl       0.80      0.81      0.80       446
b2_bicepcurl       0.54      0.54      0.54       114
b3_bicepcurl       0.85      0.95      0.90       396
b4_bicepcurl       0.81      0.82      0.81       246
b5_bicepcurl       0.85      0.78      0.82       560
b6_bicepcurl       0.80      0.78      0.79       525
g1_bicepcurl       0.80      0.80      0.80       380

    accuracy                           0.81      2667
   macro avg       0.78      0.78      0.78      2667
weighted avg       0.81      0.81      0.81      2667



Accuracy:
0.8068991376077991
```

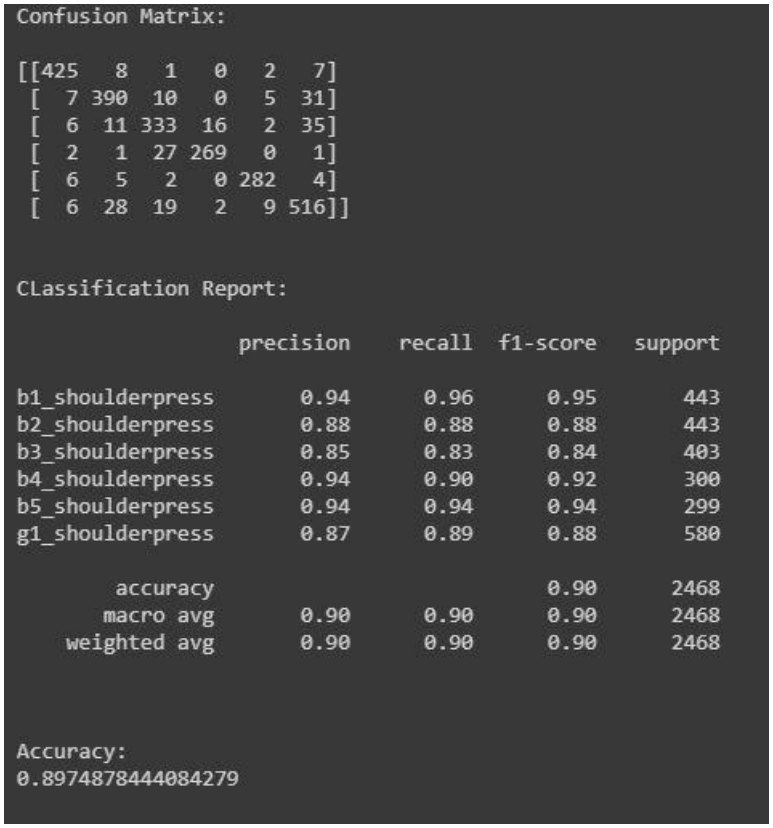
Figure 5.12: Bicep Curl Classifier with Combination of High and Low Level Feature Vector

Figure 5.13 is showing the result for Front Raise posture classifier that trained by the combination of low level training data and high level training data, with the accuracy of 64.92%.

```
Confusion Matrix:

[[337  34  20  47   2  24]
 [ 73 411  32  33   1  55]
 [ 30  19 285  35   2  50]
 [ 80  48  56 155   5  31]
 [  4   0   7   8 182  14]
 [ 40  40  58  38   3 275]]


CLassification Report:

              precision    recall  f1-score   support

b1_frontraise       0.60      0.73      0.66       464
b2_frontraise       0.74      0.68      0.71       605
b3_frontraise       0.62      0.68      0.65       421
b4_frontraise       0.49      0.41      0.45       375
b5_frontraise       0.93      0.85      0.89       215
g1_frontraise       0.61      0.61      0.61       454

     accuracy                           0.65      2534
    macro avg       0.67      0.66      0.66      2534
 weighted avg       0.65      0.65      0.65      2534



Accuracy:
0.649171270718232
```

Figure 5.13: Front Raise Classifier with Combination of High and Low Level Feature Vector

Figure 5.14 is showing the result for Shoulder Press posture classifier that trained by the combination of low level training data and high level training data, with the accuracy of 89.75%.



```
Confusion Matrix:

[[425   8   1   0   2   7]
 [  7 390  10   0   5  31]
 [  6  11 333  16   2  35]
 [  2   1  27 269   0   1]
 [  6   5   2   0 282   4]
 [  6  28  19   2   9 516]]


CLassification Report:

                  precision    recall  f1-score   support

b1_shoulderpress       0.94      0.96      0.95       443
b2_shoulderpress       0.88      0.88      0.88       443
b3_shoulderpress       0.85      0.83      0.84       403
b4_shoulderpress       0.94      0.90      0.92       300
b5_shoulderpress       0.94      0.94      0.94       299
g1_shoulderpress       0.87      0.89      0.88       580

        accuracy                           0.90      2468
       macro avg       0.90      0.90      0.90      2468
    weighted avg       0.90      0.90      0.90      2468



Accuracy:
0.8974878444084279
```

Figure 5.14: Shoulder Press Classifier with Combination of High and Low Level Feature Vector

Table 5.3 shows the overall view of posture classifier for three different types of workout.

Table 5.3: Accuracy of Posture Classifiers

Workout Type	Accuracy
Bicep Curl	80.69%
Front Raise	64.92%
Shoulder Press	89.74%

5.3.2 Model Accuracy Optimisation

I tried two ways to optimise classifier accuracy, which are testing different combination of training data and parameter fine-tuning.

For Bicep Curl posture classifier, I achieved 80.69% for combining high level and low level training data. Figure 5.15 shows the result for training Bicep Curl classifier with only high level training data. The accuracy for this model is 60.67%.

```
Confusion Matrix:

[[249   0   0   0   0 197   0]
 [  0 114   0   0   0   0   0]
 [ 40 234 110   0   0   0  12]
 [  0 128   0 111   7   0   0]
 [ 79   0   0   0 447  19  15]
 [ 73   0   0   0   0 452   0]
 [ 10 226   0   0   9   0 135]]


CLassification Report:

              precision    recall  f1-score   support

b1_bicepcurl       0.55      0.56      0.56       446
b2_bicepcurl       0.16      1.00      0.28       114
b3_bicepcurl       1.00      0.28      0.43       396
b4_bicepcurl       1.00      0.45      0.62       246
b5_bicepcurl       0.97      0.80      0.87       560
b6_bicepcurl       0.68      0.86      0.76       525
g1_bicepcurl       0.83      0.36      0.50       380

    accuracy                           0.61      2667
   macro avg       0.74      0.61      0.57      2667
weighted avg       0.79      0.61      0.63      2667



Accuracy:
0.6066741657292838
```

Figure 5.15: Bicep Curl Classifier with High Level Feature Vector

Figure 5.16 shows the result for training Bicep Curl classifier with only low level training data. The accuracy for this model is 73.83%.

```
Confusion Matrix:

[[341   8   4  10  12  49  22]
 [ 10  54  14  18   5   4   9]
 [  5  16 350   2  20   1   2]
 [ 14  19   9 182   6  12   4]
 [ 28  14  11  22 414  44  27]
 [ 55   6   8  12  45 372  27]
 [ 27   5  11  13  36  32 256]]


CLassification Report:

              precision    recall  f1-score   support

b1_bicepcurl       0.71      0.76      0.74       446
b2_bicepcurl       0.44      0.47      0.46       114
b3_bicepcurl       0.86      0.88      0.87       396
b4_bicepcurl       0.70      0.74      0.72       246
b5_bicepcurl       0.77      0.74      0.75       560
b6_bicepcurl       0.72      0.71      0.72       525
g1_bicepcurl       0.74      0.67      0.70       380

    accuracy                           0.74      2667
   macro avg       0.71      0.71      0.71      2667
weighted avg       0.74      0.74      0.74      2667



Accuracy:
0.7382827146606674
```

Figure 5.16: Bicep Curl Classifier with Low Level Feature Vector

For Front Raise posture classifier, I achieved 64.92% for combining high level and low level training data. Figure 5.17 shows the result for Front Raise training classifier with only high level training data. The accuracy for this model is 42.37%.

```
Confusion Matrix:

[[266 177   0   0   0   0]
 [ 25 581   0   0   0   0]
 [  0 376   0   0   0   0]
 [ 77 281   0   7   0   0]
 [  0  10   0   0 210   0]
 [  0 501   0   0   0   0]]


CLassification Report:

              precision    recall  f1-score   support

b1_frontraise       0.72      0.60      0.66       443
b2_frontraise       0.30      0.96      0.46       606
b3_frontraise       0.00      0.00      0.00       376
b4_frontraise       1.00      0.02      0.04       365
b5_frontraise       1.00      0.95      0.98       220
g1_frontraise       0.00      0.00      0.00       501

     accuracy                           0.42      2511
    macro avg       0.50      0.42      0.35      2511
 weighted avg       0.43      0.42      0.32      2511



Accuracy:
0.4237355635205098
```

Figure 5.17: Front Raise Classifier with High Level Feature Vector

Figure 5.18 shows the result for training Front Raise classifier with only low level training data. The accuracy for this model is 52.29%.

```
Confusion Matrix:

[[301  55  13  57   1  16]
 [ 75 390  35  60   7  39]
 [ 34  44 225  35  10  28]
 [ 74  89  38 124  11  29]
 [ 35  37  32  37  45  34]
 [ 70  67  57  68  11 228]]


CLassification Report:

              precision    recall  f1-score   support

b1_frontraise       0.51      0.68      0.58       443
b2_frontraise       0.57      0.64      0.61       606
b3_frontraise       0.56      0.60      0.58       376
b4_frontraise       0.33      0.34      0.33       365
b5_frontraise       0.53      0.20      0.30       220
g1_frontraise       0.61      0.46      0.52       501

     accuracy                           0.52      2511
    macro avg       0.52      0.49      0.49      2511
 weighted avg       0.53      0.52      0.51      2511



Accuracy:
0.5228992433293509
```

Figure 5.18: Front Raise Classifier with Low Level Feature Vector

For Shoulder Press posture classifier, I achieved 89.74% for combining high level and low level training data. Figure 5.19 shows the result for training Shoulder Press classifier with only high level training data. The accuracy for this model is 71.36%.

```
Confusion Matrix:

[[397   0   0   0   0  36]
 [  5 149  38   0  17 222]
 [  0   0 199  75  11 126]
 [  0   5  21 264  19   0]
 [  0   0  12   0 246  51]
 [  0   6  38  12   0 474]]


CLassification Report:

                  precision    recall  f1-score   support

b1_shoulderpress       0.99      0.92      0.95       433
b2_shoulderpress       0.93      0.35      0.50       431
b3_shoulderpress       0.65      0.48      0.55       411
b4_shoulderpress       0.75      0.85      0.80       309
b5_shoulderpress       0.84      0.80      0.82       309
g1_shoulderpress       0.52      0.89      0.66       530

        accuracy                           0.71      2423
       macro avg       0.78      0.72      0.71      2423
    weighted avg       0.77      0.71      0.70      2423



Accuracy:
0.7135782088320264
```

Figure 5.19: Shoulder Press Classifier with High Level Feature Vector

Figure 5.20 shows the result for training Shoulder Press classifier with only high level training data. The accuracy for this model is 85.33%.

```
Confusion Matrix:

[[411  12   3   4   3  10]
 [ 11 374  15   1   8  34]
 [ 10  12 323  25   1  32]
 [ 13   0  37 242   1   7]
 [  8   3   2   1 281   4]
 [ 24  37  28   5  11 475]]


CLassification Report:

                  precision    recall  f1-score   support

b1_shoulderpress       0.86      0.93      0.89       443
b2_shoulderpress       0.85      0.84      0.85       443
b3_shoulderpress       0.79      0.80      0.80       403
b4_shoulderpress       0.87      0.81      0.84       300
b5_shoulderpress       0.92      0.94      0.93       299
g1_shoulderpress       0.85      0.82      0.83       580

        accuracy                           0.85      2468
       macro avg       0.86      0.86      0.86      2468
    weighted avg       0.85      0.85      0.85      2468



Accuracy:
0.853322528363047
```

Figure 5.20: Shoulder Press Classifier with Low Level Feature Vector

Every result from tested combinations will be summarised in Table 5.3.

Table 5.4: Overview Result for Different Combinations

Workout Type	Training Data Combination	Accuracy
Bicep Curl	High Level + Low Level	80.69%
	High Level	60.67%
	Low Level	73.83%
Front Raise	High Level + Low Level	64.92%
	High Level	42.37%
	Low Level	52.29%
Shoulder Press	High Level + Low Level	89.75%
	High Level	71.36%
	Low Level	85.33%

Based on Table 5.4, every workout classifier has the highest accuracy by combining high level and low level feature vector. Other than that, I also realized that the classifier accuracy for low level is higher than high level. This condition can only be explained with the high level feature vector that extracted from low level does not contain a lot of useful information that I expected, but still contain little useful information for model to learn. This is the reason why I can get the highest accuracy by combining both of them. If I am able to extract more useful feature from low level then the classifier accuracy might increase.

In conclusion, the extracted high level training data does finish its job but does not performed perfectly as expectation.

Secondly, the another way to optimise classifier accuracy is performing parameter fine-tuning. For KNN algorithm, I can only fine tune one parameter, which is the K-value, since KNN algorithm is the easiest and simplest machine learning algorithm. In other way, it means that I need to find the most suitable K-value. Therefore, there are multiple ways to do this.

First way to find the best K-value is perform a very basic mathematic operation.

$$Kvalue = \sqrt{datapoints}$$

Datapoints means the number of class that needed to predict in classifier. There are seven classes in Bicep Curl posture classifier, 6 classes in Front Raise and Shoulder Press posture classifiers. Therefore the after applying the equation, the K-value for both classifiers are either two or three.

The second way is to perform Elbow method, which is a very common method to choose the optimum K-value. Figure 5.21 is the sample from applying Elbow Method. The graph shape looks like an arm and the point that looks like elbow is the optimal K-value (Bonaros 2019). For Figure 5.21, the optimal K-value is three.

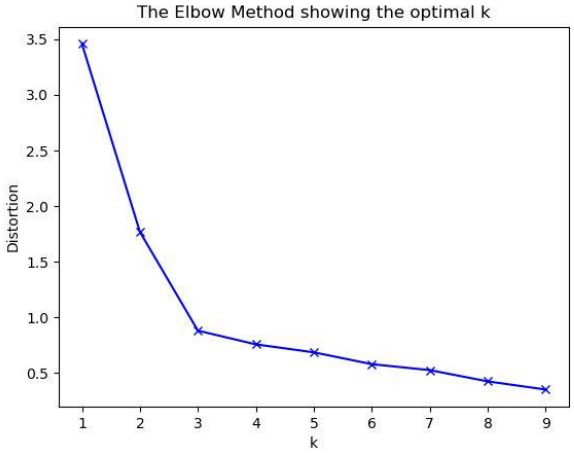


Figure 5.21: Sample of Elbow Method

Figure 5.22 shows the result from applying elbow method.

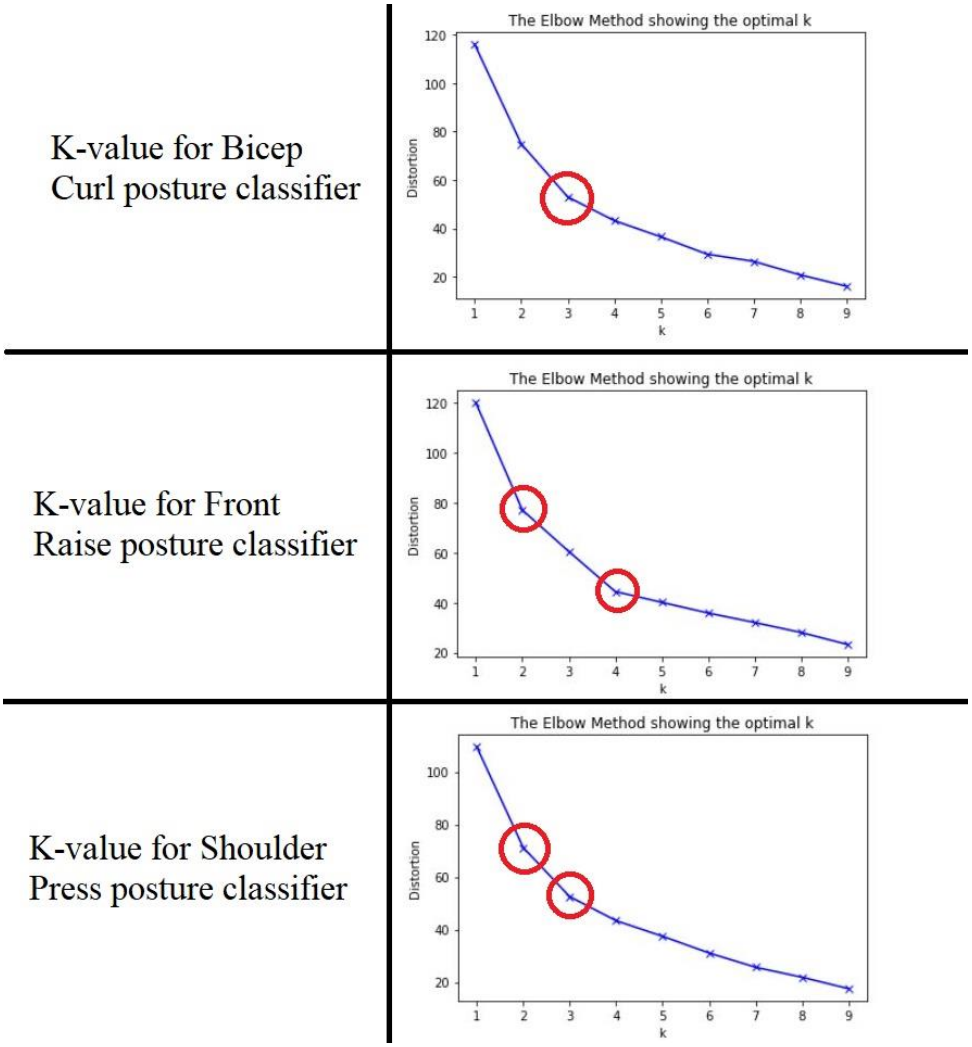


Figure 5.22: Result from Elbow Method

The red circle drew in Figure 5.22 are the potential optimal K-value for respective posture classifier, which is three for Bicep Curl, two or four for Front Raise and two or three for Shoulder Press. The list of possible K-value from these two methods is summarised in Table 5.5.

Table 5.5: Potential Optimal K-value

Posture Classifier	K-value
Bicep Curl	3
Front Raise	2, 3, 4
Shoulder Press	2, 3

In conclusion, after implementing those two methods, the best K-value for each posture classifier falls within the range from two to four. Therefore, I will try a range from two to six of K-value to determine which K-value is the best parameter to train the classifier. As you can see I extend the range from four to six to make sure I tested out K-value as many as I can.

Figure 5.23 shows the result from the range of two to six for K-value in Bicep Curl posture Classifier.

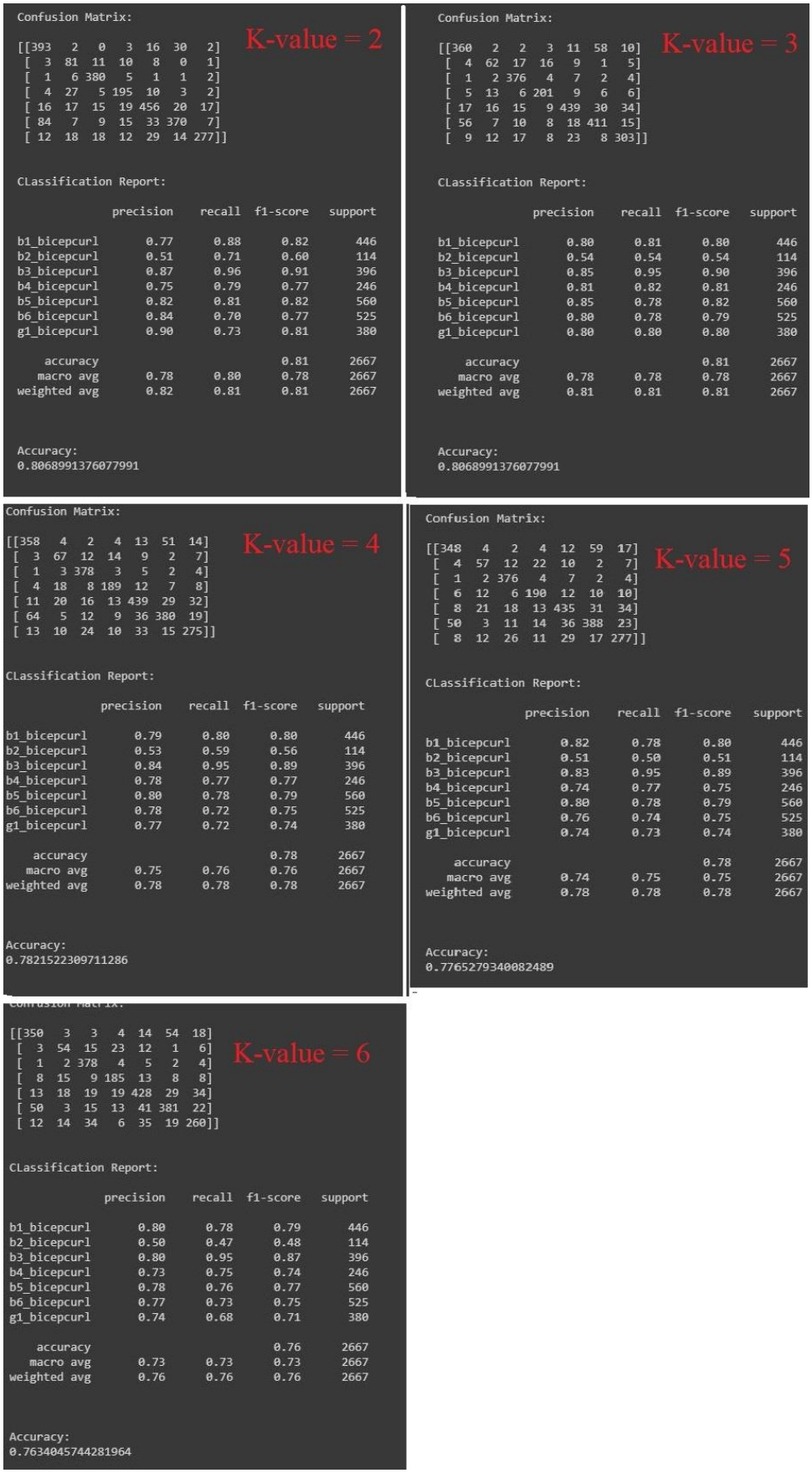


Figure 5.23: Results of Applying Different K-value for Bicep Curl Classifier

Figure 5.24 shows the result from the range of 2 to 6 for K-value in Front Raise posture Classifier.

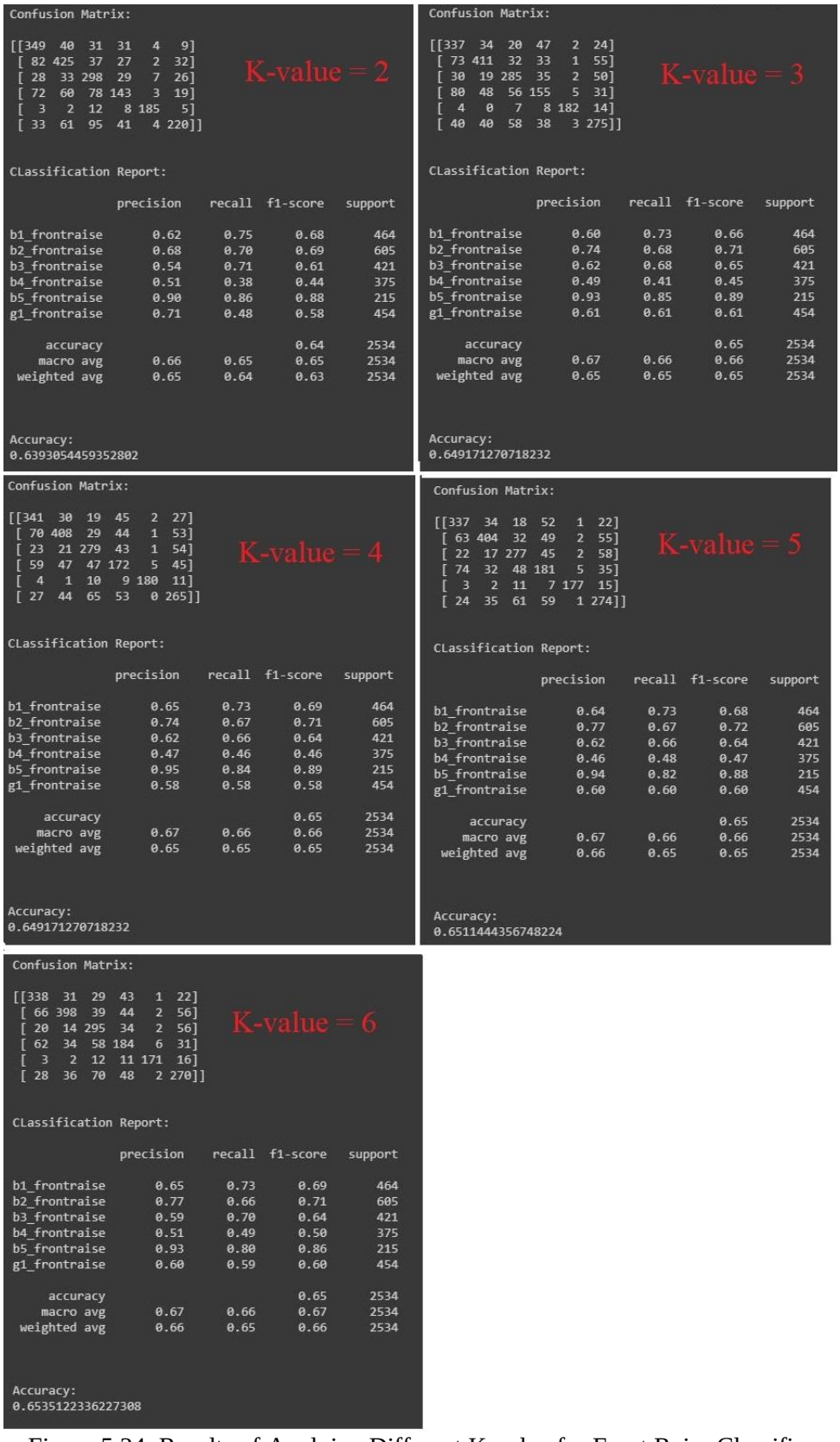


Figure 5.24: Results of Applying Different K-value for Front Raise Classifier

Figure 5.25 shows the result from the range of 2 to 6 for K-value in Shoulder Press posture Classifier.

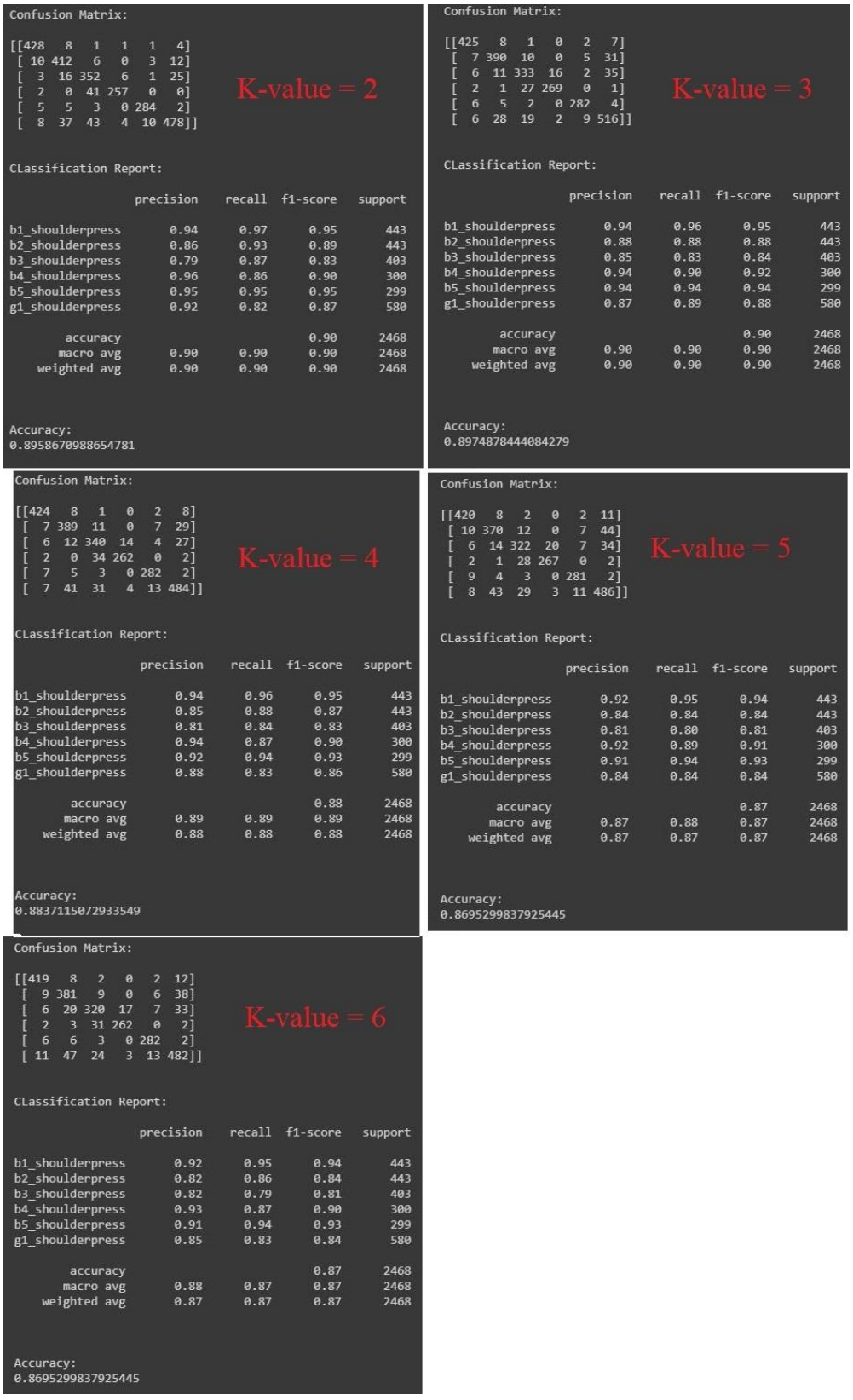


Figure 5.25: Results of Applying Different K-value for Shoulder Press Classifier

Every result from tested K-value will be summarised in Table 5.6.

Table 5.6: Overview Result for Different K-value

Workout Type	K-value	Accuracy
Bicep Curl	2	80.69%
	3	80.69%
	4	78.22%
	5	77.65%
	6	76.34%
Front Raise	2	63.93%
	3	64.92%
	4	64.92%
	5	65.11%
	6	65.35%
Shoulder Press	2	89.59%
	3	89.75%
	4	88.37%
	5	86.95%
	6	86.95%

In conclusion, by comparing Table 5.5 and Table 5.6, the calculated K-value is not all correct but it is very close to the optimal K-value. Perhaps I can perform more methods to calculate the most suitable K-value.

5.4 Compare Related Work

Table 5.7 shows the summary comparison between existing approaches with my approach.

Table 5.7: Overall Comparison Between All Approaches

Functionality	2.2.1	2.2.2	2.2.3	2.2.4.1	2.2.4.2	My Proposed Approach
Sensor Dependency	No	Yes	Yes	Yes	No	No
Camera Dependency	Yes	No	No	Yes	Yes	Yes
Evaluation Process	Partial Automation	Full Automation	Partial Automation	Partial Automation	Partial Automation	Partial Automation
Limitation to Certain Types of Exercise	No	Yes	No	No	No	Yes
Exercise Recognition	Yes	No	Yes	Yes	Yes	Yes
Evaluation Feedback	Detailed	Partly Detailed	Not Detailed	No Feedback	Detailed	Detailed
Ease of Installation	Yes	No	No	No	No	Yes

CHAPTER 6 CONCLUSION

As a conclusion, the workout recognition and evaluation system had been successful developed to achieve the project objectives. The system with the function of detect and recognize the workout type from the input video had been tested with multiple workout type under different environments and achieved around 98% accuracy. On the other hand, the system is also able to classify different types of improper posture with the accuracy of 80.69% for Bicep Curl class, 65.35% for Front Raise class and 89.75% for Shoulder Press class.

The problem of this project is the OpenPose model may tend to miss out some of the joint keypoints. This may due to some factors such as complex background or dark environment. It is essential to get the video which has better resolution or quality for the detection and recognition. The another problem caused by OpenPose model is the time taken for it to retrieve the keypoints location. Perhaps in future there are other alternative ways to retrieve body keypoints location in shorter time, maybe within one second for one frame. The problem restraints me for deploying the model in real-life scenario. There is actually another interesting problem which is the complexity of human body movement. It is impossible to list out every possible posture that made by human. Therefore, the improper postures that predicted in this project are actually just the few most common postures.

There are actually few novelties and contributions from this project. The first one will be creating new benchmark for this workout dataset. The reason I create my own workout dataset is because I couldn't find any public workout dataset. Therefore, I have uploaded the workout dataset to Kaggle so that other people can access to my dataset for free and all the annotations have been made. My approach that shown in this project can be a benchmark approach for this dataset. Hopefully by making this dataset public to CV community can raise the attention to the evaluation of quality in performing an activity field.

For the future work, there are actually few areas that can be improved. The first area is developing a faster Keypoint Detection Model so that the system can shorten the evaluation process, so that this project is commercially ready to launch into the market. A faster Keypoint Detection Model also increase the chance of deploying this system

the real-time scenario. In this case, users does not have to upload video to the system and can receive the evaluation easily. For another area will be evaluating the number of improper postures. As I mentioned above, it is impossible to list down every possible postures. Therefore, the system can be further developed into having one single proper posture in 3D. Then the system is able to convert the 2D uploaded video into 3D format. Then proper posture will compare with the converted video, evaluation can be provided based on the comparison between these two 3D models. In this case, I do not need to list down every possible posture because the solution can tackle every kind of posture made by user.

BIBLIOGRAPHY

Barbara, M 2014, *Are you too Embarrassed to Exercise?* Available from: <https://www.psychologytoday.com/intl/blog/shyness-is-nice/201401/are-you-too-embarrassed-exercise>. [05 August 2019].

Bonaros, B 2019, *K-Means Elbow Method Code For Python* Available from: <https://predictivehacks.com/k-means-elbow-method-code-for-python/>. [22 April 2020].

Chen, S & Yang, R 2018, Pose Trainer: Correcting Exercise Posture using Pose Estimation. *Department of Computer Science Stanford University*, California: Stanford.

Eduardo, V, et al., 2013, Qualitative Activity Recognition of Weight Lifting Exercises. *Proceedings of the 4th Augmented Human International Conference*, pp. 116-123.

Imran, V, 2015, How to Choose the Value of K in KNN Algorithm. Available from: <https://discuss.analyticsvidhya.com/t/how-to-choose-the-value-of-k-in-knn-algorithm/2606>. [22 April 2020].

Jason, B 2019, *A Gentle Introduction to Computer Vision*. Available from: <https://machinelearningmastery.com/what-is-computer-vision/?fbclid=IwAR375th0DO4vfPNy7n9BqGbx_gRizi3Wi_7JgyJE1RYbv9NETFWxJxKBGm4>. [28 July 2019].

Jason, B 2019, *A Gentle Introduction to Object Recognition With Deep Learning*. <Available from: https://machinelearningmastery.com/object-recognition-with-deep-learning/>. [28 July 2019].

Kim, D, Cho, M, Park, Y, & Yang, Y 2015. Effect of an exercise program for posture correction on musculoskeletal pain. *Journal of physical therapy science, vol. 27, no. 6,* pp. 1791–1794. doi:10.1589/jpts.27.1791

Kowsar, Y, Moshtaghi, M, Velloso, E, Kulik, L & Leckie, C 2016, Detecting Unseen Anomalies in Weight Training Exercises. *Microsoft Research Centre for Social NUI*, pp. 517-526.

Mike, S n.d., *Posture-Improving Weight-Lifting Exercises*. Available from: <https://www.livestrong.com/article/371062-weight-lifts-that-improve-posture/>. [01 August 2019].

Novatchkov, H & Baca, A 2012, Machine Learning Methods For The Automatic Evaluation Of Exercises On Sensor-Equipped Weight Training Machines. *9th Conference of the International Sports Engineering Association*, vol. 34, pp. 562–567.

Paul, R 2019, *A Fundamental Guide to Weight Training*. Available from: <https://www.verywellfit.com/weight-training-fundamentals-a-concise-guide-3498525>. [30 July 2019].

Qian, H, Mao, Y, Xiang, W & Wang, Z 2010, Recognition of Human Activities Using SVM Multiclass Classifier. *Pattern Recognition Letters*, vol. 31, no. 2, pp. 100-111.

Timo, R, 2018, *Python Vs. C++ for Machine Learning – Language Comparison.* Available from: <https://www.netguru.com/blog/python-vs.-c-for-machine-learning-language-comparison>. [12 August 2018].

Vikas, G 2019, *Pose Detection Comparison: wrnchAI vs OpenPose*. Available from: <https://www.learnopencv.com/pose-detection-comparison-wrnchai-vs-openpose/>. [01 August 2019].

Yun, X & Bachmann, ER 2006, Design, Implementation, and Experimental Results of a Quaternion-Based Kalman Filter for Human Body Motion Tracking. *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1216-1227.

APPENDIX

Appendix A Access to Public Workout Exercise Dataset

I have uploaded the dataset to Kaggle. Therefore, you can access to this dataset with this link [*https://www.kaggle.com/jiunn1998/workout-exercise*]

This dataset consists of three different type of workout exercises, which are Bicep Curl, Front Raise and Shoulder Press. Each exercise also consists of different type of improper and proper postures (Figure A.1).

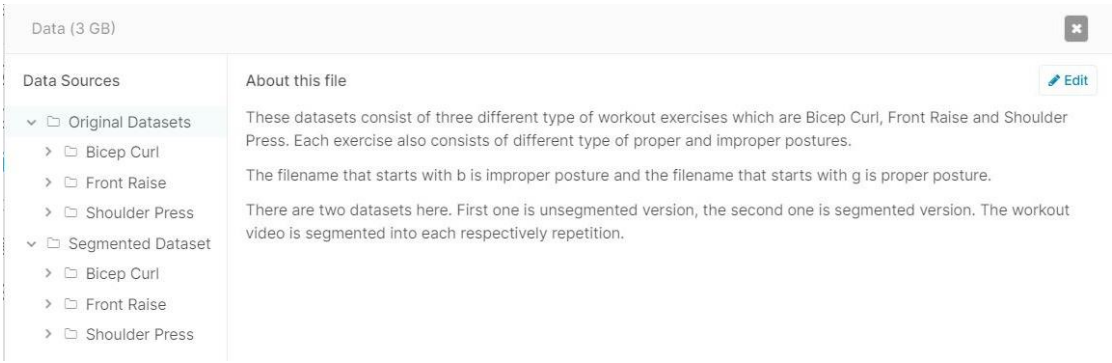


Figure A.1: Uploaded Datasets in Kaggle

Appendix B Model Deployment

B-1 Deploy in Local Machine

First, you need to download whole project directory from Google Drive, [*https://drive.google.com/drive/folders/16xEqtV9iTx7ZEwiVt2Ca1uOdct0JYBLK?usp=sharing*] (Figure B.1). I am unable to upload the directory to GitHub because GitHub does not allow to upload a single file that exceeds 20MB.

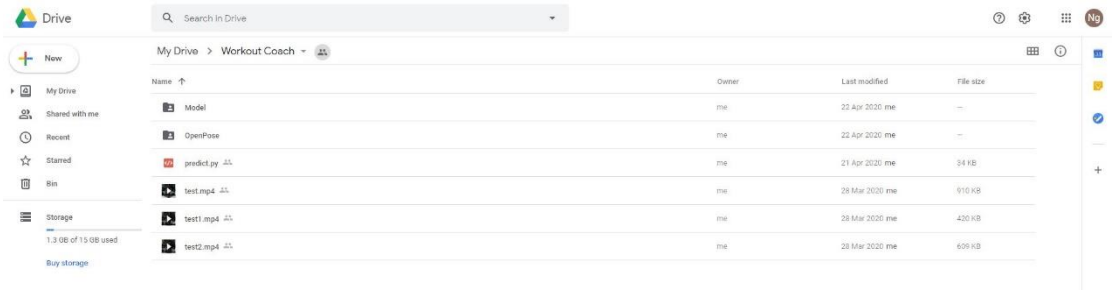


Figure B.1: Workout Coach in Google Drive

After you download the directory, you need to run the system in Anaconda Prompt. If you do not have Anaconda Prompt or Anaconda Navigator, please visit *https://www.anaconda.com/distribution/* . Please choose Python 3.7 version (Figure B.2).

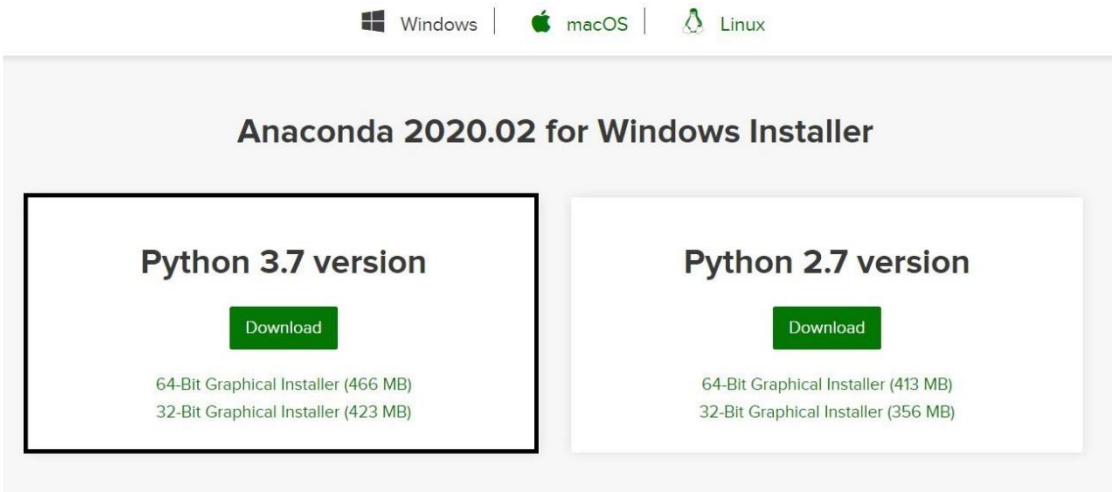


Figure B.2: Download Anaconda

Launch Anaconda Prompt if you complete the steps above. Please move to the downloaded project directory. You are required to download Python and some Python libraries before you able to run it. First, you need to download Python using Anaconda Prompt, by typing conda install python=3.7.3 (Figure B.3).

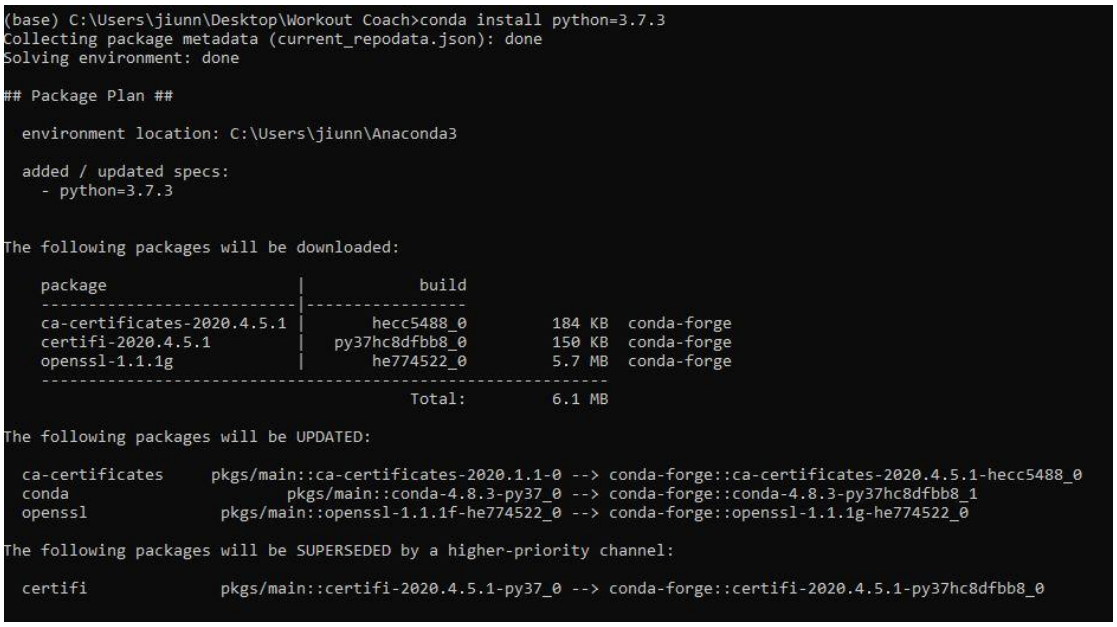
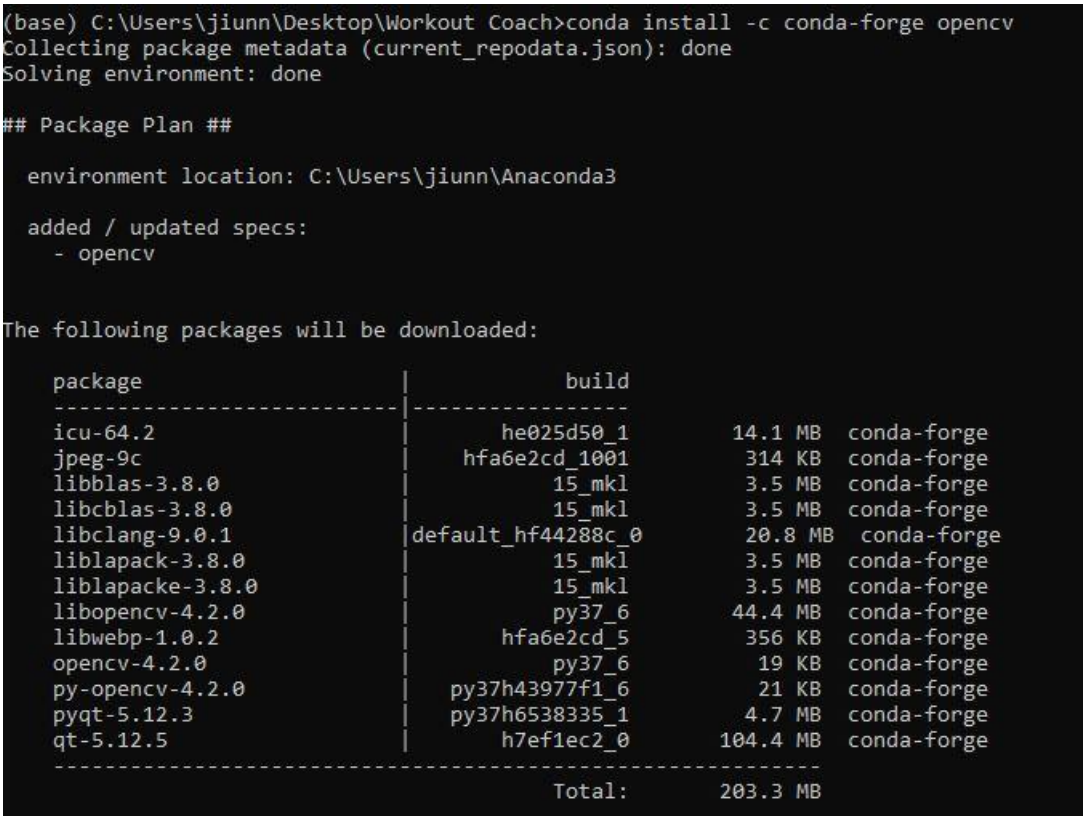


Figure B.3: Download Python3 in Anaconda Prompt

Next, download other Python libraries by running commands:

- install –c conda-forge opencv (Figure B.4)
- pip install moviepy (Figure B.5)
- pip install –U scikit-learn (Figure B.6)

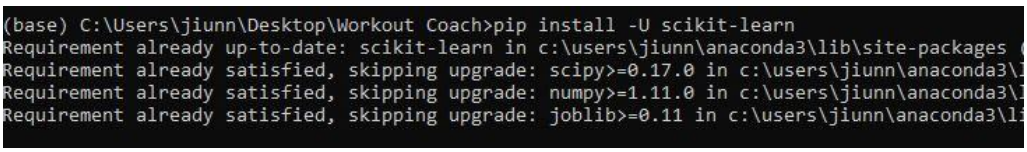


```
(base) C:\Users\jiunn\Desktop\Workout Coach>conda install -c conda-forge opencv
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\jiunn\Anaconda3

  added / updated specs:
    - opencv


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    icu-64.2                   |       he025d50_1        14.1 MB  conda-forge
    jpeg-9c                    |    hfa6e2cd_1001         314 KB  conda-forge
    libblas-3.8.0              |           15_mkl         3.5 MB  conda-forge
    libcblas-3.8.0             |           15_mkl         3.5 MB  conda-forge
    libclang-9.0.1             |default_hf44288c_0        20.8 MB  conda-forge
    liblapack-3.8.0            |           15_mkl         3.5 MB  conda-forge
    liblapacke-3.8.0           |           15_mkl         3.5 MB  conda-forge
    libopencv-4.2.0            |           py37_6        44.4 MB  conda-forge
    libwebp-1.0.2              |       hfa6e2cd_5         356 KB  conda-forge
    opencv-4.2.0               |           py37_6          19 KB  conda-forge
    py-opencv-4.2.0            |   py37h43977f1_6          21 KB  conda-forge
    pyqt-5.12.3                |   py37h6538335_1         4.7 MB  conda-forge
    qt-5.12.5                  |       h7ef1ec2_0       104.4 MB  conda-forge
    ------------------------------------------------------------
                                           Total:       203.3 MB
```

Figure B.4: Download OpenCV in Anaconda Prompt

```
(base) C:\Users\jiunn\Desktop\Workout Coach>pip install moviepy
Requirement already satisfied: moviepy in c:\users\jiunn\anaconda3\lib\
Requirement already satisfied: proglog<=1.0.0 in c:\users\jiunn\anacond
Requirement already satisfied: numpy; python_version >= "2.7" in c:\use
Requirement already satisfied: requests<3.0,>=2.8.1 in c:\users\jiunn\a
Requirement already satisfied: decorator<5.0,>=4.0.2 in c:\users\jiunn\
Requirement already satisfied: tqdm<5.0,>=4.11.2 in c:\users\jiunn\anac
Requirement already satisfied: imageio-ffmpeg>=0.2.0; python_version >=
Requirement already satisfied: imageio<3.0,>=2.5; python_version >= "3.
Requirement already satisfied: certifi>=2017.4.17 in c:\users\jiunn\ana
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1
Requirement already satisfied: idna<3,>=2.5 in c:\users\jiunn\anaconda3
Requirement already satisfied: chardet<4,>=3.0.2 in c:\users\jiunn\anac
Requirement already satisfied: pillow in c:\users\jiunn\anaconda3\lib\s
```

Figure B.5: Download moviepy in Anaconda Prompt



```
(base) C:\Users\jiunn\Desktop\Workout Coach>pip install -U scikit-learn
Requirement already up-to-date: scikit-learn in c:\users\jiunn\anaconda3\lib\site-packages
Requirement already satisfied, skipping upgrade: scipy>=0.17.0 in c:\users\jiunn\anaconda3\
Requirement already satisfied, skipping upgrade: numpy>=1.11.0 in c:\users\jiunn\anaconda3\
Requirement already satisfied, skipping upgrade: joblib>=0.11 in c:\users\jiunn\anaconda3\li
```

Figure B.6: Download scikit-learn in Anaconda Prompt

If you feel complicated by downloading these setup in terminal, you can actually finish the same job by using Anaconda Navigator, go to Environment, select the environment

you want to run the system and search any package you want to download. Figure B.7 shows I install openCV library in Anaconda Navigator.

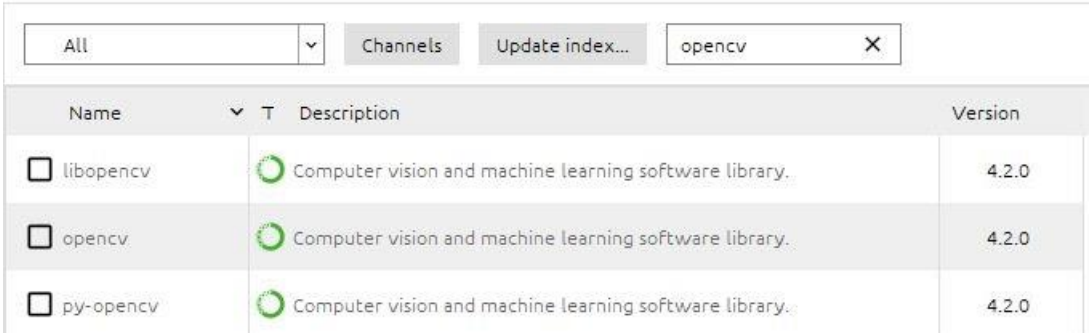


Figure B.7: Download Package in Anaconda Navigator

Until now, all the prerequisites are ready and you should be able to run the system. Please upload the workout video that you want to evaluate into Workout Coach directory. Simply type "python predict.py *your file name*" to run the system (Figure B.8).

```
(base) C:\Users\jiunn\Desktop\Workout Coach>python predict.py test.mp4
You need to wait for roughly  7.733333333333333 minutes.
Workout Classifier:  Bicep Curl
Detected Posture Characteristics:
Your elbows flare out or move behind your back as you lift the weight
Improper Posture Category: Moving Your Elbow
Dangerous Level of Improper Posture: Adequate
This posture has moderate negative effect on your body, you might need some time to perform proper posture.

Suggested Posture Correction:
Keep your elbow still and stick your upper hand to your torso and you are good to go.
Usually this improper posture is due to excessive weight, therefore I encourage you to drop some weight.
```

Figure B.8: Run Workout Coach System in Anaconda Prompt

B-3 Deploy in Google Colab

If you feel troublesome to deploy local machine, you can choose to deploy in Google Colab. The only weakness is it requires Internet every time you run it. Firstly, download the whole project directory, the same step in Appendix B-2. Then, upload whole directory to your own Google Drive. When the directory is uploaded, kindly open Workout Coach.ipnyb (Figure B.9).

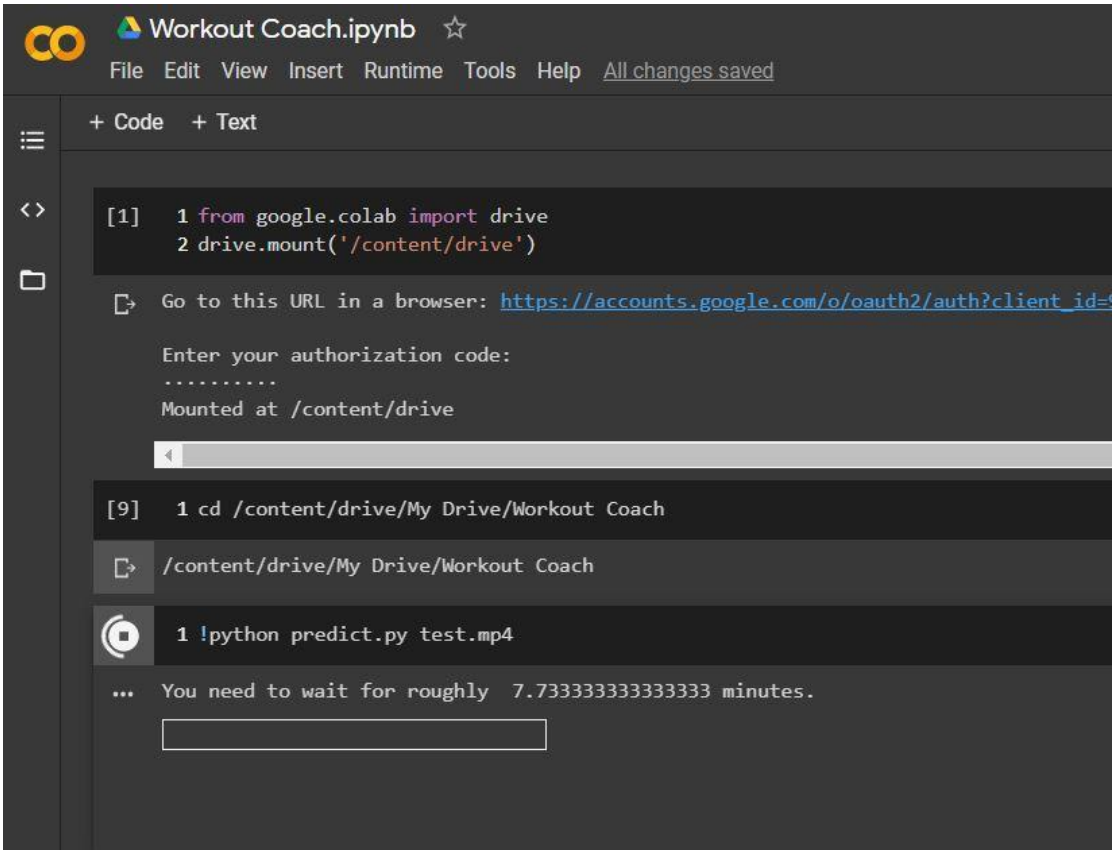


Figure B.9: Image of Workout Coach.ipnyb

The last step is to run every code cell. You will receive the evaluation result by running the last code cell (Figure B.10).

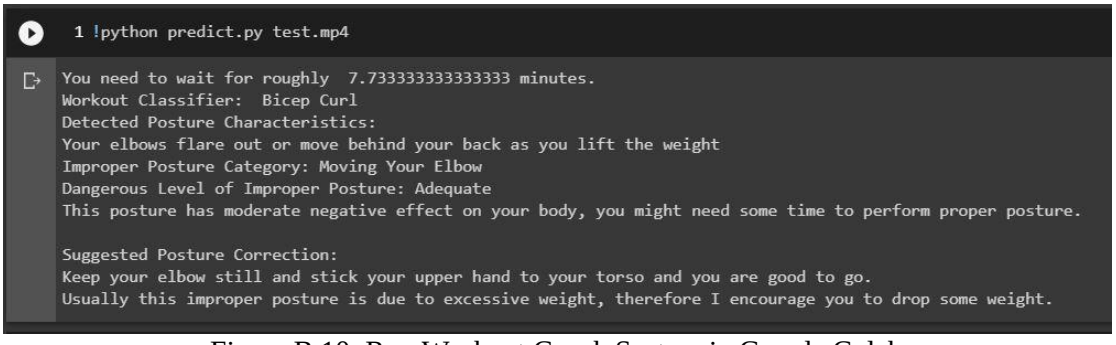
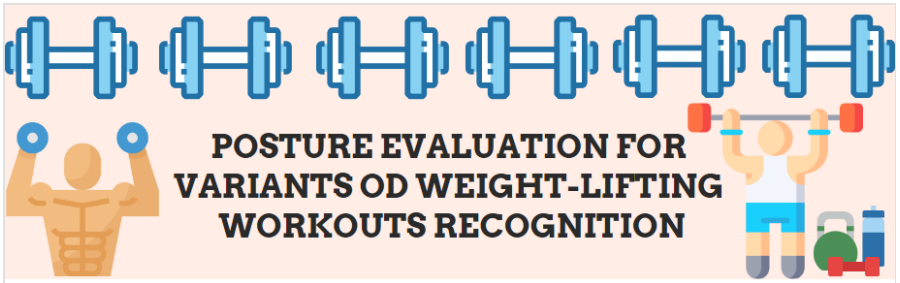


Figure B.10: Run Workout Coach System in Google Colab

This method is obviously faster and easier to deploy compare to deploy in local machine. Therefore, you can choose any method that you prefer.



POSTURE EVALUATION FOR VARIANTS OD WEIGHT-LIFTING WORKOUTS RECOGNITION

Weight lifting is a flow of body movement pack in an organized exercise to force the body muscles to contract under tension. Performing wrong posture is very common and it can increase the risk of injury. Object detection and object recognition which are under CV field are implemented in this project. The system is able to recognise the workout type and evaluate the posture

Objectives

- To design workout exercise recognition using KNN
- To design an A.I coach for workout posture correction recommendation using KNN
- Optimize model accuracy through parameter tuning and feature selection

Machine Learning Pipeline

- Follow standard machine learning pipeline.
- Heavily focused on Data Preparation stage and feature selection.
- Such as cleaning the noise, standardize every video frame perspective.
- Extract multiple level of feature vector from raw data (body keypoints geometric locations).

Parameter Fine-Tuning

- Applied two methods to find the optimal K-value for every classifier.
- First method is perform a square root of number of classes.
- Second method is apply Elbow Method.
- The results from these methods help to reduce the range of K-value for testing.

Workout Recognition Classifier Result

- Achieve 98.7% of accuracy in classifying three types of workout exercises.
- Able to perform well in bright and complex background, as long as the visibility of user is high.

Workout Posture Classifier Result

- Achieve accuracy of 80.69% for Bicep Curl, 65.35% for Front Raise, 89.75% for Shoulder Press
- Able to perform well in bright and complex background, as long as the visibility of user is high.

Model Deployment

- Able to deploy in two platforms, either in Google Colab which is on cloud platform or deploy in local machine.
- Does not need any setup for Google Colab platform.
- Need some setup such as installing Python, Anaconda Prompt and many more for deploying in local machine.

Future Work

- Implement real-time system.
- Upgrade to process posture evaluation by comparing human object in 3D format to open for countless type of postures.

Smart AI Workout Coach

To use the product, please visit
https://drive.google.com/drive/folders/16xEqtV9iTx7ZEwiVt2Ca1uOdct0JYBLK?usp=sharing
To access the dataset, please visit
https://www.kaggle.com/jiunn1998/workout-exercise

Turnitin Originality Report

Processed on: 23-Apr-2020 21:13 +08
ID: 1305008748
Word Count: 10302
Submitted: 2

Posture Evaluation for Variants of Weight-Lif... By Ng Jiunn

Similarity Index	Similarity by Source	
11%	Internet Sources:	8%
	Publications:	7%
	Student Papers:	4%

include quoted | include bibliography | excluding matches < 8 words | mode: quickview (classic) report | Change mode | print | download

3% match (Internet from 03-May-2019)
https://perceptual.mpi-inf.mpg.de/files/2013/03/velloso13_ah.pdf

1% match (Internet from 30-Aug-2017)
http://socialnui.unimelb.edu.au

1% match (Internet from 08-Apr-2019)
https://core.ac.uk/download/pdf/81171386.pdf

1% match (student papers from 18-Nov-2014)
Submitted to Universiti Tunku Abdul Rahman on 2014-11-18

1% match (Internet from 19-Jan-2020)
https://www.techopedia.com/definition/33499/image-recognition

<1% match (Internet from 19-Feb-2020)
http://www.research.lancs.ac.uk

<1% match (publications)
Eduardo Velloso, Andreas Bulling, Hans Gellersen, Wallace Ugulino, Hugo Fuks. "Qualitative activity recognition of weight lifting exercises", Proceedings of the 4th Augmented Human International Conference on - AH '13, 2013

<1% match (Internet from 30-Oct-2019)
http://sm-eb.smartfixmarburg.de

<1% match (Internet from 01-Nov-2016)
https://www.scribd.com/document/303400564/Communication-Articles-2013

<1% match (publications)
Novatchkov, Hristo, and Arnold Baca. "Machine learning methods for the automatic evaluation of exercises on sensor-equipped weight training machines", Procedia Engineering, 2012.

<1% match (student papers from 06-Apr-2015)
Submitted to Boston University on 2015-04-06

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Ng Jiunn
ID Number(s)	16ACB05121
Programme / Course	BACHELOR OF COMPUTER SCIENCE (HONS)
Title of Final Year Project	Posture Evaluation for Variants of Weight-Lifting Workouts Recognition

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 11 **%** **Similarity by source** Internet Sources: 8 % Publications: 7 % Student Papers: 4 %	
Number of individual sources listed of more than 3% similarity: 0	

Parameters of originality required and limits approved by UTAR are as Follows:

i. **Overall similarity index is 20% and below, and**
ii. **Matching of individual sources listed must be less than 3% each, and**
iii. **Matching texts in continuous block must not exceed 8 words**

Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Name: Aun YC

Date: 24/4/2020

Signature of Co-Supervisor

Name: _____

Date: _____

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	16ACB05121
Student Name	Ng Jiunn
Supervisor Name	Dr. Aun YiChiet

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Front Cover
√	Signed Report Status Declaration Form
√	Title Page
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
√	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report. (Signature of Student) Date: 24-04-2020	Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction. (Signature of Supervisor) Date: 24/4/2020

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 2**
Student Name & ID: Ng Jiunn 16ACB05121	
Supervisor: Dr. Aun Yichiet	
Project Title: Posture Evaluation for Variants of Weight-Lifting Workouts Recognition	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

i. Meet with supervisor to determine a draft idea for FYP2.

2. WORK TO BE DONE

i. Revise back FYP1 report.

3. PROBLEMS ENCOUNTERED

-

4. SELF EVALUATION OF THE PROGRESS

Self-assigned tasks are completed within expected timeframe.

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 4**
Student Name & ID: Ng Jiunn 16ACB05121	
Supervisor: Dr. Aun Yichiet	
Project Title: Posture Evaluation for Variants of Weight-Lifting Workouts Recognition	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

i. Expand the size of dataset by recording more workout video.
ii. Finish the annotation for the workout video since this is a supervised learning approach.

2. WORK TO BE DONE

i. Need to perform Data Preparation stage for new dataset.

3. PROBLEMS ENCOUNTERED

-

4. SELF EVALUATION OF THE PROGRESS

Self-assigned tasks are completed within expected timeframe.

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 6**
Student Name & ID: Ng Jiunn 16ACB05121	
Supervisor: Dr. Aun Yichiet	
Project Title: Posture Evaluation for Variants of Weight-Lifting Workouts Recognition	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

i. Performed Data Preparation on dataset like cleaning off the noise, null value, standardising the video perspective by rotating it.

2. WORK TO BE DONE

i. Proceed to Feature Selection phase.

3. PROBLEMS ENCOUNTERED

i. The noise may be outlier, difficult to distinguish them.

4. SELF EVALUATION OF THE PROGRESS

Self-assigned tasks are completed within expected timeframe.

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 8**
Student Name & ID: Ng Jiunn 16ACB05121	
Supervisor: Dr. Aun Yichiet	
Project Title: Posture Evaluation for Variants of Weight-Lifting Workouts Recognition	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

i. Successfully extract low level of feature vector from raw data such as keypoint location,
ii. Further extraction is made on low level feature vector to generate a higher level feature vector to make sure it contains useful information for model to learn.

2. WORK TO BE DONE

i. Proceed to Model Training stage.
ii. Proceed to Model Evaluation stage.

3. PROBLEMS ENCOUNTERED

ii. Having problem is deciding what to extract from low level feature vector to generate high level feature vector.

4. SELF EVALUATION OF THE PROGRESS

Self-assigned tasks are completed within expected timeframe.

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 10**
Student Name & ID: Ng Jiunn 16ACB05121	
Supervisor: Dr. Aun Yichiet	
Project Title: Posture Evaluation for Variants of Weight-Lifting Workouts Recognition	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

i. The model is trained and evaluated to make sure it is not underfitting or overfitting by generating confusion matrix, accuracy, f1-score and many more.

2. WORK TO BE DONE

i. Need to find a method to deploy these trained model.
ii. Start working on report.

3. PROBLEMS ENCOUNTERED

i. Front Raise classifier is having an unexpected low accuracy. Generate a new feature vector and start Model Training and Model Evaluation stages again.

4. SELF EVALUATION OF THE PROGRESS

Self-assigned tasks are completed within expected timeframe.

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 12**
Student Name & ID: Ng Jiunn 16ACB05121	
Supervisor: Dr. Aun Yichiet	
Project Title: Posture Evaluation for Variants of Weight-Lifting Workouts Recognition	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

ii. All technical part is done, from Data Collection to Model Deployment.
iii. Finished half of the report.

2. WORK TO BE DONE

i. Need to generate a lot of data to support statement and theory listed in report.
ii. Need to send the report to Turnitin for plagiarism checking.

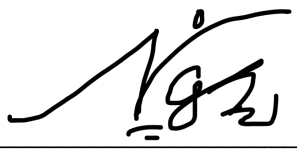
3. PROBLEMS ENCOUNTERED

i. Facing some problems in categorising chapter when writing report, consult supervisor to make sure the report organisation is correct.

4. SELF EVALUATION OF THE PROGRESS

Self-assigned tasks are completed within expected timeframe.

Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 13**
Student Name & ID: Ng Jiunn 16ACB05121	
Supervisor: Dr. Aun Yichiet	
Project Title: Posture Evaluation for Variants of Weight-Lifting Workouts Recognition	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

i. Report is finished and ready for supervisor to check.

2. WORK TO BE DONE

i. Make corrections on report based on feedback from supervisor once he done checking.
ii. Generate a final version of report and let supervisor to sign.
iii. Submit everything, like report and coding.

3. PROBLEMS ENCOUNTERED

-

4. SELF EVALUATION OF THE PROGRESS

Self-assigned tasks are completed within expected timeframe.

Supervisor's signature

Student's signature