YOGGUIDE: DETECTING, RECOGNIZING AND TRACKING YOGA POSES USING VISION TECHNIQUES

AZKA KHAN (48435)
MUZNA REHMAN (48467)

A project report submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Science Computer Science BS(CS)

Department of Computer Science
Bahria University, Karachi Campus

August 2020

DECLARATION

We hereby declare that this project report is based on our original work except for citations and quotations which have been duly acknowledged. We also declare that it has not been previously and concurrently submitted for any other degree or award at Bahria University or other institutions.

Signature : _________________________

Name : Muzna Rehman

Reg No. : 48467

Signature : _________________________

Name : Azka Khan

Reg No. : 48435

Date : Sunday, 13 December

APPROVAL FOR SUBMISSION

We certify that this project report entitled **"YOGGUIDE: DETECTING, RECOGNIZING AND TRACKING YOGA POSES USING VISION TECHNIQUES"** was prepared by **Azka Khan(48435)** and **Muzna Rehman(48467)** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Science (BSCS) at Bahria University.

Approved by,

Signature : ______________________

Supervisor : Ms. Sameena Javaid

Date : Sunday, 13 December ___

ACKNOWLEDGEMENTS

We would like to thank everyone who had contributed to the successful completion of this project. We would like to express our gratitude to our research supervisor, Ms. Sameena Javaid for her invaluable advice, guidance and her enormous patience throughout the development of the research.

In addition, we would also like to express our gratitude to our loving parent and friends who had helped and given us encouragement.

YOGGUIDE: DETECTING, RECOGNIZING AND TRACKING YOGA POSES USING VISION TECHNIQUES

ABSTRACT

In computer vision, human pose estimation is a deep-rooted issue that in the past has revealed many challenges. In many fields such as security, video games, physical therapy, etc. analyzing human activities is beneficial. One of the challenges in human pose estimation is Yoga. These days, with stress and pressure full lives, people generally prefer doing yoga at homes as yoga is said to be art of relaxation, but they feel an instructor's need to evaluate their exercise form as doing wrong posture can cause health problems. Since these resources are not always available, human pose recognition can be used to create a system of self-training exercise that allows individuals to better learn and practice exercises by their own.

This project objective is to develop an application which is an attempt to ensure correct yoga posture for three main poses which includes plank, warrior and pose reverse warrior in an intuitive way. This project uses deep learning technique for pose estimation in which different stages are involved including pre-processing stage, data augmentation, creating CNN model and training the model. Yoga-guide's ultimate aim is to use pose recognition as a tool to allow a person to practice different poses of yoga and receive feedback.

In this project, using convolution neural network (CNN) model using Keras with TensorFlow as backend a deep learning model is proposed. The key benefit of using this technique is that it offers extraction and identification of features that are appropriate for pose recognition. We are able to achieve accuracy of 97%.

.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Background

Analyzing and understanding human poses estimation is a subject that has been studied extensively in the past two decades. Therefore, this represents interest for many promising applications in different domains, such as security, video games, physical therapy, etc [1]. Issues emerge when checking human posture is required. It is about the limitation of human joints in an image or video to shape a skeletal portrayal. The automatic detection of an individual's pose in a picture is difficult as it depends on numerous factors including size and image resolution, variations in clothing and surroundings, and also interaction with other humans. One of the challenges in post estimation is yoga which has attracted many researchers in the field of pose estimation [2]. Yoga has now become a notable overall order which is a protected and viable exercise to increment active work, particularly in strength, adaptability and equilibrium, to increment physical and mental prosperity[3]. Yoga is a sort of activity that assists with profound breathing, reflection, or unwinding for the entire brain and body.

Yoga is said as the art of relaxation. From last year, people are dealing with greater pressure and suffering from stress faster than ever before. Even kids are struggling too as they are spending too much time on smartphones and tablet. In recent decades, disease has emerged with new dimensions, expressions, and manifestations. The great plagues of the past have come to an end in medical research, but we now are faceing stress-related disorders caused by our inability to adjust to the highly competitive pace of modern life.. Psychosomatic problems such as diabetes, high blood pressure, obesity, thyroid disorders, migraine, asthma, ulcers, digestive and skin disorders are said to be result from the stresses of the body and mind [4]. In developing countries, the leading causes of death such as cancer and heart disease often rise from stress. Though Modern medical science is trying to tackle these problems in different ways, but they have failed to deliver the good necessary health to man. This is because the real problem is not in the body but comes from the changing ideals of the man, his way of thinking and his feelings. Today international problem is not hunger, poverty, drugs or fear of war. It is tension, hypertension. Anyone who knows how to alleviate stress and how to control it can control high blood pressure, heart disease, etc. In the different layers of human personality, such pressures and tensions accumulate. Such pressure and tension accumulate in the different layers of human personality. We concentrate in the physical, intellectual, and emotional systems. Yoga solves tension issues with a big periscope [5]. One of the main concerns in yoga is relief from stresses and so the practise of yoga will change the nature of the mind, cure diseases and restore the creative genius.

Yoga has become very popular and well known all over the globe recently. Some sources say this is because of the advantages it offers. Power, stamina and versatility are some of the benefits that help preserve body and spirit harmony and enhance them at the same time. [3]. Yoga is helpful for people with high blood pressure, cardiac disease, pain or stress.

1.2 Problem Statements

Regular yoga practices can reduce causes of their suffering. As a result, it has become growing massively and increasingly over the past few years around the world. However, not everyone can go out to participate in yoga classes because of inconvenient public transportation systems for the old people, mostly people can’t go due to time issue. Therefore, it is important for them to practice Yoga at homes by themselves. However, it is not easy for novice Yoga people, particularly seniors, to find the incorrect parts of their Yoga poses by themselves. Usually people doing yoga poses at home without instructor either do in incorrect manner or overdo the poses which lead to acute pain and long-term chronic problems.

We know that yoga is a great form of exercise and has advantages such as increasing flexibility and strength, and reducing stress and anxiety, but if done wrongly, certain serious health issues can arise from doing wrong poses which are:

1. Backache
2. Ankle Sprain
3. Stiff neck, sprain and pain in the neck
4. Muscle Pulls

1.3 Aims and Objectives

The objectives of the project are shown as following:

i) To provide an application for correct yoga poses
ii) To work on correct pose estimation using real time image capturing.

1.4 Scope of Project

Our project will cover 3 main yoga poses which are:

1.4.1 Plank

Subject position its elbows directly under shoulders and rest of forearms on the ground. Then the subject will pop up on the toes, holding the body from head to toe in a straight line by bending its knees so that the arms and elbows are parallel to the head and each leg remains in a parallel position as shown in Figure 1.1.

Figure 1:1 Plank

1.4.2 Warrior

As far as possible, the subject stretches his/her left leg while the right leg is vertical to the ground and spreads two hands so that they fall in a line as shown in Figure 1.2

Figure 1:2 Warrior

1.4.3 Reverse Warrior

In this posture, the subject stretches the right leg as far as possible while the left leg is vertical to the ground and holds the right hand on the right knee and positions the left hand parallel to the head and vertical to the ground. as shown in Figure 1.3

Figure 1:3 Reverse Warrior

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

The literature review is a search and evaluation technique of the available literature in a given domain topic. This chapter focuses on human pose estimation technique, functions and characteristics of existing yoga corrector applications, deep learning model (CNN) and characteristics of SDLC methodology used in this project. The analysis of the literature review is used as a reference in choosing the appropriate methodology to develop the project. The structure of the literature review for this project is illustrated in Figure 2.1 below

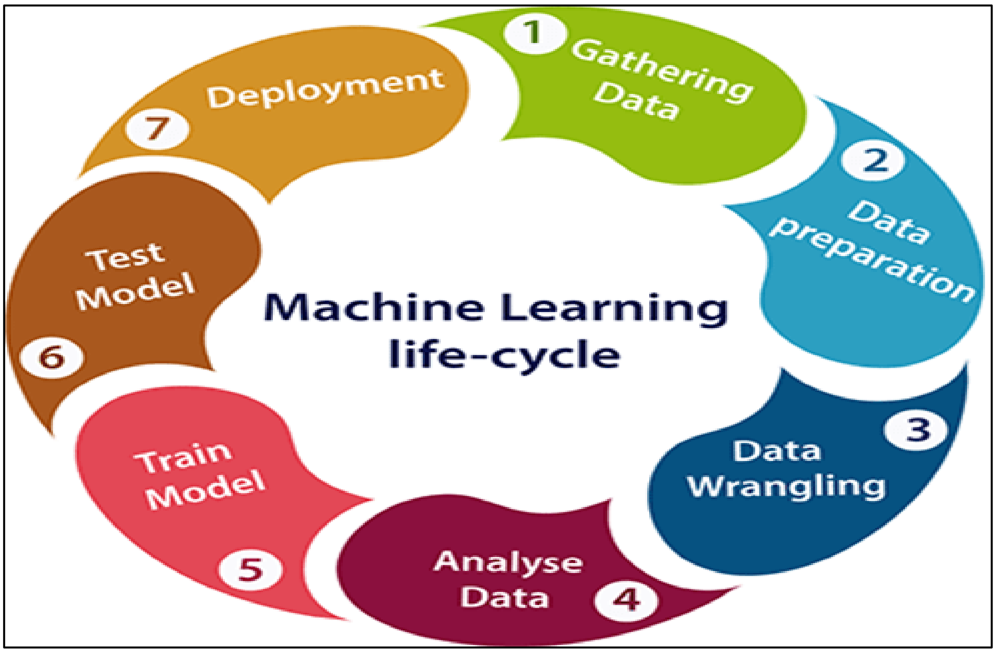

Figure 2:1 Structure of the Literature Review

2.2 Study on Human Pose Estimation

Pose recognition methods focus on detecting human figures in images and video, with the goal that one could decide, for instance, where somebody's elbow appears in a picture. There are a number of works that have been proposed for human posture recognition [6]-[12] also show in below Table 2.1. Gochoo et al. [6] developed a system for IoT-based privacy-preserving and device-free yoga postures recognition method using a deep convolutional neural networks (DCNNs) and a low-resolution infrared sensor-based wireless sensor network (WSN). They collected a total of 93,200 posture images and worked with 18 candidates to represent yoga poses. Similarly, an approach to accurately recognize various Yoga asanas using deep learning algorithms by Yadav et al. [3] has been presented in this work using convolutional neural network (CNN) and long short-term memory (LSTM) for Yoga recognition on real-time videos and a total of 6 yoga asanas were recorded that achieved a test accuracy between 98% to 99%.

Trejo and Yuan [8] proposes a technique to perceive 6 regular Yoga poses by a Kinect sensor. Initially, the Adaboost algorithm is used to build the database for poses recognition, which is provided in the tools of Kinect for windows SDK v2.0. This system also provides the user with command voices so user can easily interact with the system for purposes such as changing yoga poses and receiving instruction for a particular pose. Likewise, Islam et al. [10] have given out a system which can monitor human body parts movement of different yoga poses which aids the user to practice yoga. It has used Microsoft Kinect to detect different joint points of human body in real time and from those joint points accuracies is calculated on various angles of a certain yoga pose for a user. Three types of poses are detected and accuracy above 97% is calculated.

Table 1 Study on Posture Recoginition

Author/ Paper/Year	Work	Dataset	Technique	Number of poses/ gestures	Accuracy
Gochoo et al., 2019 [6]	This paper proposes an IoT-based privacy-preserving yoga posture recognition system.	In total, 93,200 posture images are employed.	Deep Convolutional Neural Network (DCNN).	Nearly 26 yoga postures from 18 candidates were collected.	Results had 98%-99% accuracy, respectively.
Yadav et al., 2019 [3]	An approach to accurately recognize various Yoga asanas using deep learning algorithms has been presented in this work.	A dataset of six Yoga asanas has been created using 15 individuals.	Convolutional neural network (CNN) and long short-term memory (LSTM).	A total of 6 Yoga asanas were recorded in real-time.	The system achieves a test accuracy of around 99.04% to 99.38% accuracy.

Maddala et al., 2019 [7]	A system developed for recognition of yoga asanas using Joint Angular Displacement Maps (JADM).	Video sequence of yoga.	Single stream deep CNN model.	20 yoga poses.	Accuracy of around 90%.
Trejo and Yuan, 2018 [8]	Microsoft Kinect device is used to sense the posture and give enhancement instructions.	Dataset was collected by Kinect sensor.	Kinect sensor is used to perceive yoga poses.	Six common asana yoga poses are used for this study.	Final database showed above 94.78% in terms of accuracy.
Chen et al., 2018 [9]	This system integrates computer vision techniques and analyzes body contour, skeleton, dominant axes, and feature points.	Images, videos captured in real time.	C++ implementatio n in OpenCV.	The propose d system can analyze up to 12 poses.	Accuracy between 92% and 99% for front and side view.

Islam et al., 2017 [10]	System can monitor human body parts movement of different yoga poses which aids the user to practice yoga.	Video data is captured by using Kinect device.	Microsoft Kinect used to gather joint point information.	Three types of poses: Goddess Squat, Warrior, Reverse Warrior.	The accuracy is above 97% for every angle between different body parts.
Jiang et al., 2017 [11]	This paper proposes a novel multi-layered gesture recognition method with Kinect.	Kinect with ChaLearn dataset.	Hand glove motion to record images.	More than 50, 000 gesture sequenc es were recorded with Kinect.	The proposed method achieves high recognition accuracy 88%.
Liu et al., 2017 [12]	A system which uses hand gesture recognition using kinetic.	Dataset contains 1200 different data samples.	Radial Basis Function (RBF) neural network.	10 different gestures.	The correct classification rates are reported to be 95.83% and 97.25%, respectively.

Dennis et al., 2018 [13]	This work presents a CNN algorithm for recognition of hand movements on images acquired by a single colored camera.	Dataset is based of wrists and hand images in total 6000.	Pictures of hand gestures were collected by a single-colored camera and based on the extracted hand, a NN based regressor is executed to estimate the wrist position.	Consists of 10 static classes of images and size 38x38 is used as input.	The authors concluded that different methods of capturing images produced different accuracy. Mainly between 87% to 97%
Seung-Ho Han et al., 2017 [14]	According to paper, this presents a novel model that corrects the improper postures of the patient by extracting the human skeleton using Microsoft Kinect using deep neural network.	Skeleton data is collected from Microsoft Kinect	An idea is presented that corrects the patients posture when rehabilitating, and uses Kinect SDK to extract 20 major skeleton points.	The idea was to not work on a particular pose but to gain information on patients posture so the could be helped.	This paper did not produce any result because the team formed a evaluation plan but did not work on it.

Zheng et al., 2018 [15]	A system was developed to find the behavior of animals on basis of their movements in an automatic detection system.	A dataset was created on animals consists of a Kinect v2 sensor that acquires depth images and a program that identifies sow postures and locates its bounding-boxes.	The depth images of testing dataset of a sow were acquired at 5 frames per second in 24 h on the 15th day of postpartum and the data is trained on RCNN model.	This system trained five postures standing , sitting, sternal recumbency, ventral recumbency and lateral recumbency and obtain sows accurate location in loose pens.	The accuracy on this paper came out different for night and day time postures which is between 84.1% to 92.9%.
Kothari 2020[16]	A self-instructed exercise system was built on various machine learning and deep learning approaches to	Dataset consists of videos (6yoga poses) which were worked on frame by frame.	Performed bottom-up and top-down by combining them with network feed-forwarding.	A total of 6 yoga poses were used.	The accuracy shows that the system learns more in bottom-up approach than top-down and performs better

	yoga poses on prerecorded videos.				in challenging situations.

Madala et al. [7] has given a system developed for recognition of yoga asanas using Joint Angular Displacement Maps (JADM). To improve the recognition accuracy with reduced training times, JADMs are tested with a single-stream deep CNN model. In total around 20 yoga poses are tested on video sequenced data. Accuracy turned out to be above than 90%. Chen et al. [9] in the paper proposed a yoga self-training system, which aims at instructing the practitioner to perform yoga poses correctly, assisting in rectifying poor postures, and preventing injury.

Integrating computer vision techniques, the proposed system analyzes the practitioner's posture from both front and side views by extracting the body contour, skeleton, dominant axes, and feature points. Then, based on the domain knowledge of yoga training, visualized instructions for posture rectification are presented so that the practitioner can easily understand how to adjust his/her posture using OpenCV with C++. The proposed system can analyze up to 12 poses with the accuracy being between 92% and 99%.

Jiang et al. [11] categorized data gatherings into two groups wearable sensor-based methods and optical camera-based methods. Kinect recorded ChaLearn Gesture Dataset comprising more than 50, 000 gesture sequences and used hand glove motion to record gestures. The proposed method achieves high recognition accuracy 88%. Similarly, Liu et al. [12] used RBF neural network to approximate hand motion dynamics underlying motion patterns of different gestures. Dataset contains 1200 different data samples and 10 different gestures. The correct classification rates are reported to be 95.83% and 97.25%, respectively.

2.2.1 Modelling based on Neural Network

There are a number of works that have been proposed for human posture recognition using Neural Networks. Gochoo et al. [6] developed a system for IoT-based privacy-preserving and device-free yoga postures recognition method using a deep convolutional neural networks (DCNNs) and a low-resolution infrared sensor-based wireless sensor network (WSN). They collected a total of 93,200 posture images and worked with 18 candidates to represent yoga poses.

Similarly, an approach to accurately recognize various Yoga asanas using deep learning algorithms by Yadav et al. [3] has been presented in this work using convolutional neural network (CNN) and long short-term memory (LSTM) for Yoga recognition on real-time videos and a total of 6 yoga asanas were recorded that achieved a test accuracy between 98% to 99%.

2.2.2 Using Kinect sensor by Adaboost Algorithm

Trejo and Yuan [8] proposes a technique to perceive 6 regular Yoga poses by a Kinect sensor. Initially, the Adaboost algorithm is used to build the database for poses recognition, which is provided in the tools of Kinect for windows SDK v2.0. This system also provides the user with command voices so user can easily interact with the system for purposes such as changing yoga poses and receiving instruction for a particular pose.

Likewise, Islam et al. [10] have given out a system which can monitor human body parts movement of different yoga poses which aids the user to practice yoga. It has used Microsoft Kinect to detect different joint points of human body in real time and from those joint points accuracies is calculated on various angles of a certain yoga pose for a user. Three types of poses are detected and accuracy above 97% is calculated.

2.2.3 Joint Angular Displacement Maps

Madala et al. [7] has given a system developed for recognition of yoga asanas using Joint Angular Displacement Maps (JADM). To improve the recognition accuracy with reduced training times, JADMs are tested with a single-stream deep CNN model as depicted in the given Figure 2.2 In total around 20 yoga poses are tested on video sequenced data. Accuracy turned out to be above than 90%.

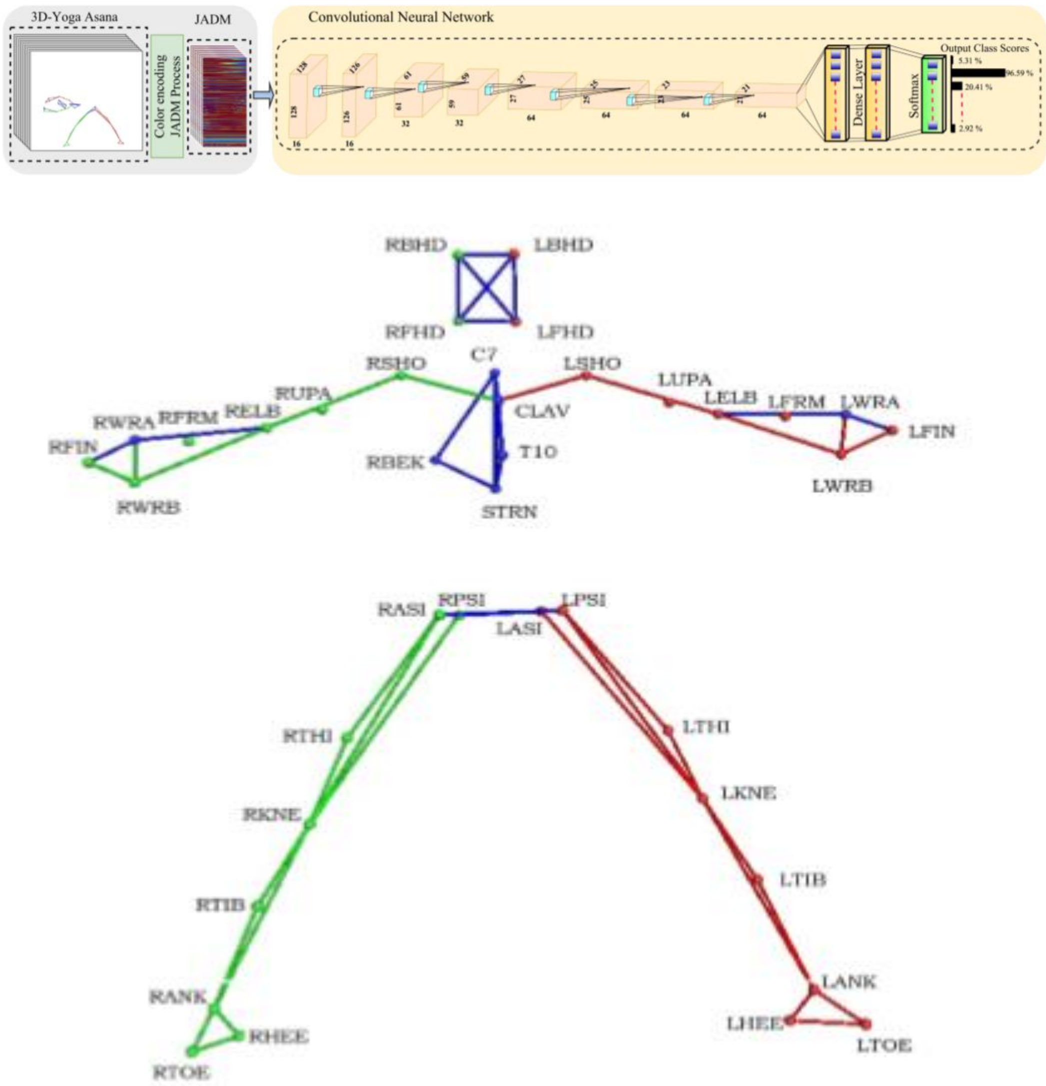

Figure 2:2 Angular Displacement Maps

2.2.4 Wearable sensor-based and optical-camera based methods

Jiang et al. [11] categorized data gatherings into two groups wearable sensor-based methods and optical camera-based methods as illustrated in given Figure 2.3. Kinect recorded ChaLearn Gesture Dataset comprising more than 50, 000 gesture sequences and used hand glove motion to record gestures. The proposed method achieves high recognition accuracy 88%.

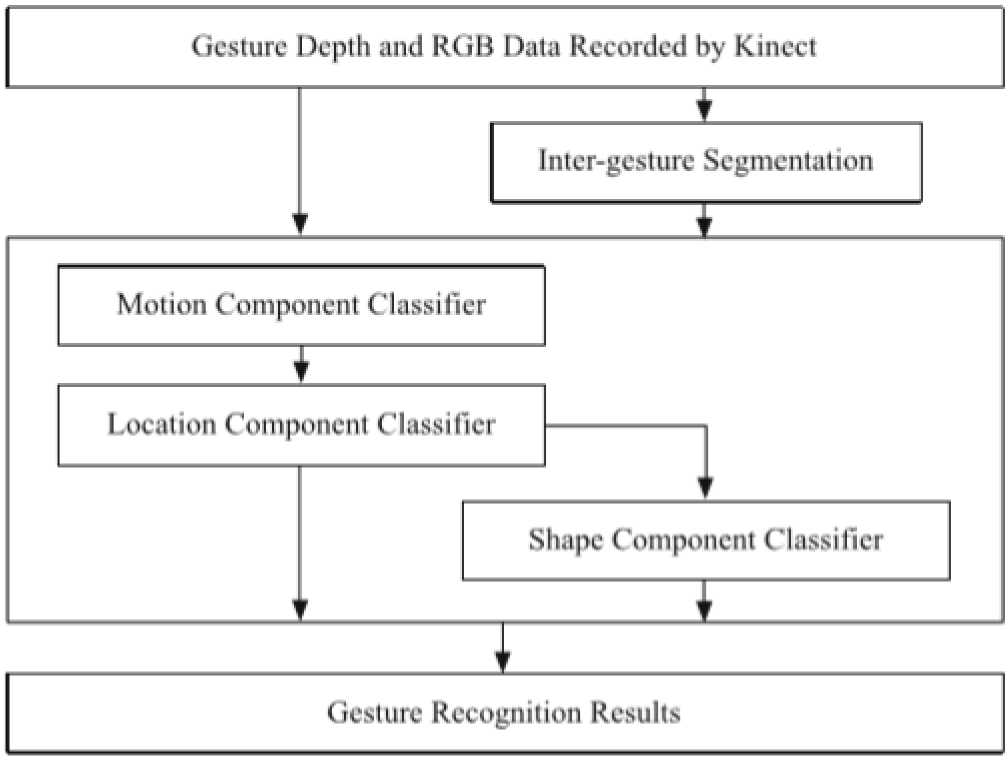

Figure 2:3 Sensor Based & Optical Camera Based Methods

Similarly, Liu et al. [12] used RBF neural network to approximate hand motion dynamics underlying motion patterns of different gestures. Dataset contains 1200 different data samples and 10 different gestures. The correct classification rates are reported to be 95.83% and 97.25%, respectively.

Dennis et al., 2018 [13] presents a CNN algorithm for recognition of hand movements on images acquired by a single colored camera. Dataset is based of wrists and hand images in total 6000 and pictures of hand gestures were collected by a single-colored camera and based on the extracted hand, a NN based regressor is executed to estimate the wrist position which consist of 10 static classes of images and size 38x38 is used as input. Similarly Seung-Ho Han et al., 2017 [14], presents a novel model that corrects the improper postures of the patient by extracting the human skeleton using Microsoft Kinect using deep neural network which corrects the patients posture when rehabilitating, and uses Kinect SDK to extract 20 major skeleton points.

Zheng et al., 2018 [15] developed a system to find the behavior of animals on basis of their movements in an automatic detection system. A dataset was created on animals consists of a Kinect v2 sensor that acquires depth images and a program that identifies sow postures and locates its bounding-boxes. Similarly, Kothari 2020 [16] developed a self-instructed exercise system on various machine learning and deep learning approaches to yoga poses on prerecorded videos. She performed bottom-up and top-down by combining them with network feed-forwarding and the accuracy shows that the system learns more in bottom-up approach than top-down and performs better in challenging.

2.3 Study of SDLC Methodology

This section discusses SDLC methodology used for this project and reason to for its selection.

2.3.1 Iterative Model

In iterative model, iterative process begins with simply implementing a small set of software requirements and iteratively enhances the evolving versions until the complete system is implemented and ready for deployment. Figure 2.9 shows iterative model structure.

Following process are involved in the iterative model:

1. Requirement Phase: In this phase requirements of the software are gathered and analysed. Iteration should eventually lead to a requirements phase which produces a complete and final requirements specification
2. Design Phase: In this step software solution is designed to meet the requirements. This could be a new design, or an extension of a previous design.

3. Implementation & Test Phase: Upon coding, integration and testing of the software.
4. Review Phase: The process in which the software is tested, the existing requirements are reviewed and the modifications and changes to the new requirements [17].

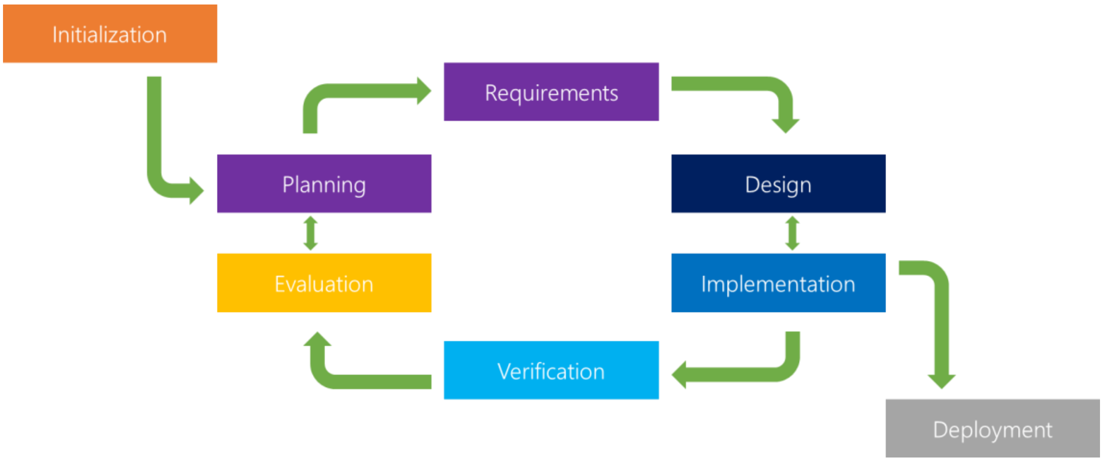

Figure 2:4 Iterative Model

2.3.2 Justification on the Selected Methodology

Before selecting the methodology to be used for this project, some considerations regarding the situation of the project must be taken into account, such as:

i. We are only 2 person who are directly in charge of the development of this project. Besides, there is a project manager who supervises the project but not directly involved in developing the system.
ii. There is no user/customer testing in our project.

Based on the above situations and also from the study of SDLC models below methodologies are not chosen to be used in this project:

1. Waterfall and V-model are not chosen because they both are quite rigid and inflexible.

2. Spiral is not chosen since it is quite complicated, whereby it requires risk analysis in each iteration. With only 2person in this project, this methodology is not feasible.
3. RAD and Agile depends on user requirements and testing. Therefore, no user are involved.

After eliminating all the choice left is iterative and according to us iterative methodology will be more suitable here as introducing iteration externally our algorithm can minimize the error margin and therefore help in accurate modelling. It's easy to use, iteration allow model to correct itself every time there is an error thus improve our model accuracy as shown in Figure 2.9

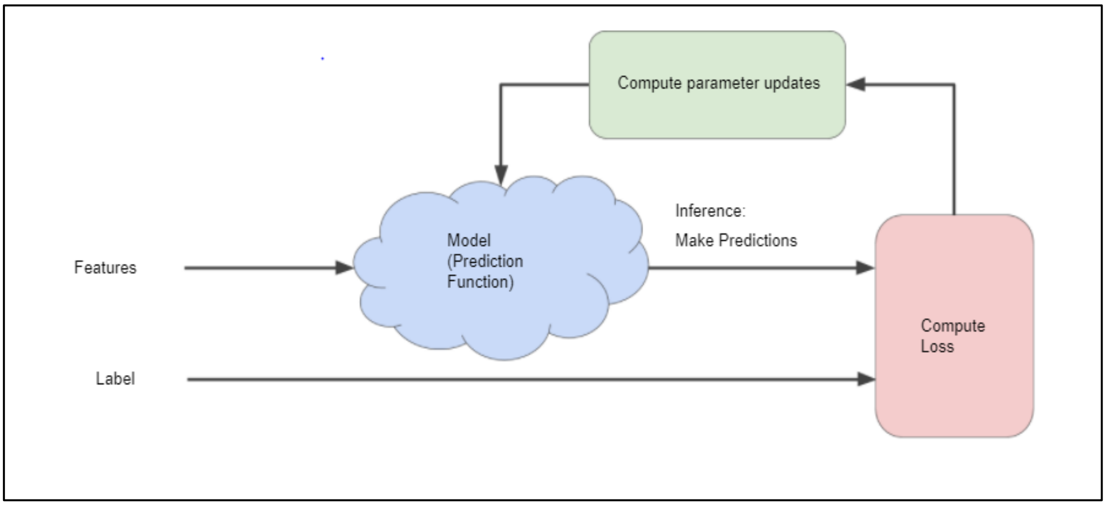

Figure 2:5 Deep Learning & Iterative Methodology

CHAPTER 3

DESIGN AND METHODOLOGY

3.1 Overview of the proposed method

Our approach is based on deep learning approach to classify yoga images. **A** CNN (Convolutional Neural Network) is an approach which is used for pose estimation of multi-stage classifier where each stage improves the results of the previous one. For the problem of yoga pose recognition two solutions are proposed and enforced. To detect exercise, one uses simple sequential model and the other uses two machine learning models which is posenet pre-build tensorflow model for the 18 body joints for pose estimation then move through second method which is CNN sequential model for the detection of exercise after doing some prepossessing.

In System Development Life Cycle (SDLC), we are going to use iterative model in which is best thought of as a cyclical process. After an initial planning phase, a small handful of stages are repeated over and over, with each completion of the cycle incrementally improving and iterating on the software. Therefore, enhancements in iterative model can quickly be recognized and implemented throughout each iteration.

3.1.1 CNN Sequential Model

The goal of our project is to classifying the different yoga poses Our approach is based on deep learning approach to classify yoga images. **A** CNN (Convolutional Neural

Network) is an approach which is used for pose estimation of multi-stage classifier where each stage improves the results of the previous one.

The following pipeline is followed for classification: Data Collection, Data Pre-processing, Training of model and Testing Model and results. Following figure 3.1 shows the flowchart of workflow of our proposed solution.

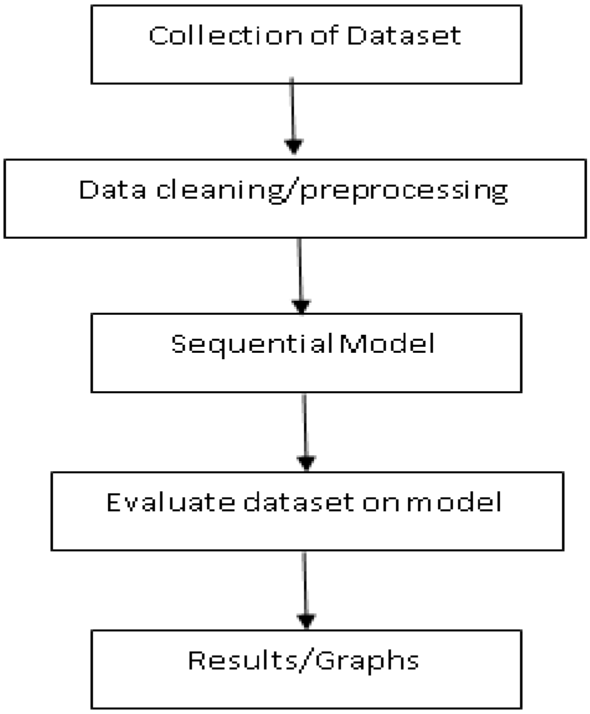

Figure 3:1 Workflow of Proposed Solution

3.1.1.1 Data Collection

Our dataset is based on 3 classes which is collected from various sources such as Google, Shutterstock, Kaggle etc. and because of the limitation of Covid’19 we couldn’t go outside and collect data ourselves from person to person as we planned. The dataset is of images at the resolution of 600 × 600 and all of them contain the same format of ‘.jpg’.

Table 2:2 Dataset Collection

Yoga Pose	Images	Augmented images
Plank	88	2150
Warrior	102	8292
Reverse-Warrior	98	10442

3.1.1.1.1 Augmented Images Sample

Figure 3:2 Augmented Dataset Example

3.1.1.2 Data Pre-processing

We used different OpenCV technique for the data preprocessing starting from resizing all the images to 600 × 600 keeping in view the model input image. And put each pose in their respective label class.

3.1.1.2.1 Dataset Split

We apportion the data into training and test sets, with an 80-20 split so with the train data contain in total 14574 Images whereas the test data consist of 14574 Images of all 3 poses.

3.1.1.2.2 Joint Points Extraction

There's a very popular method of key joint point extraction using library of OpenPose. For example; we use any picture or frame and apply the code or run the open pose code over it. Through that it tells us the point of joints on 18 locations on the body. Fig 3 shows the 18 joint points captured by OpenPose. Whereas fig 4-8 show how it worked on our dataset. Following is the list of body parts and pose pairs that were incorporated in Figure

```
BODY_PARTS = { "Nose": 0, "Neck": 1, "RShoulder": 2, "RElbow": 3, "RWrist": 4,
               "LShoulder": 5, "LElbow": 6, "LWrist": 7, "RHip": 8, "RKnee": 9,
               "RAnkle": 10, "LHip": 11, "LKnee": 12, "LAnkle": 13, "REye": 14,
               "LEye": 15, "REar": 16, "LEar": 17, "Background": 18 }

POSE_PAIRS = [ ["Neck", "RShoulder"], ["Neck", "LShoulder"], ["RShoulder", "RElbow"],
               ["RElbow", "RWrist"], ["LShoulder", "LElbow"], ["LElbow", "LWrist"],
               ["Neck", "RHip"], ["RHip", "RKnee"], ["RKnee", "RAnkle"], ["Neck", "LHip"],
               ["LHip", "LKnee"], ["LKnee", "LAnkle"], ["Neck", "Nose"], ["Nose", "REye"],
               ["REye", "REar"], ["Nose", "LEye"], ["LEye", "LEar"] ]
for i in range(len(BODY_PARTS)):
    # Slice heatmap of corresponging body's part.
    heatMap = out[0, i, :, :]

    _, conf, _, point = cv.minMaxLoc(heatMap)
    x = (frameWidth * point[0]) / out.shape[3]
    y = (frameHeight * point[1]) / out.shape[2]
    points.append((int(x), int(y)) if conf > thr else None)

for pair in POSE_PAIRS:
    partFrom = pair[0]
    partTo = pair[1]
    assert(partFrom in BODY_PARTS)
    assert(partTo in BODY_PARTS)
```

Figure 3:3 Joint Point Detection Code

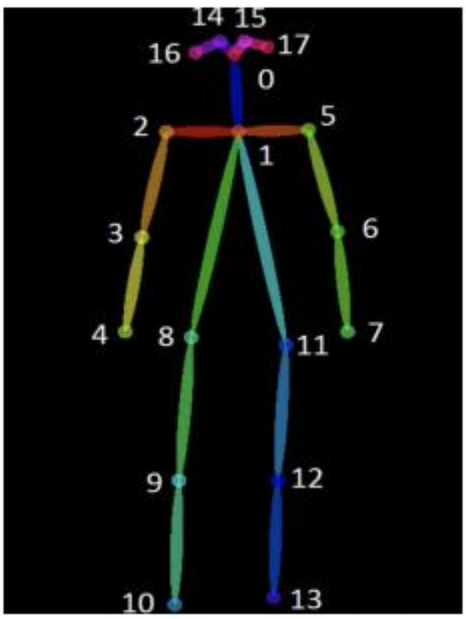

Figure 3:4 Joint Point Detection-1

Figure 3:5 Joint Point Detection-2

Figure 3:6 Joint Point Detection-3

Figure 3:7 Joint Point Detection-4

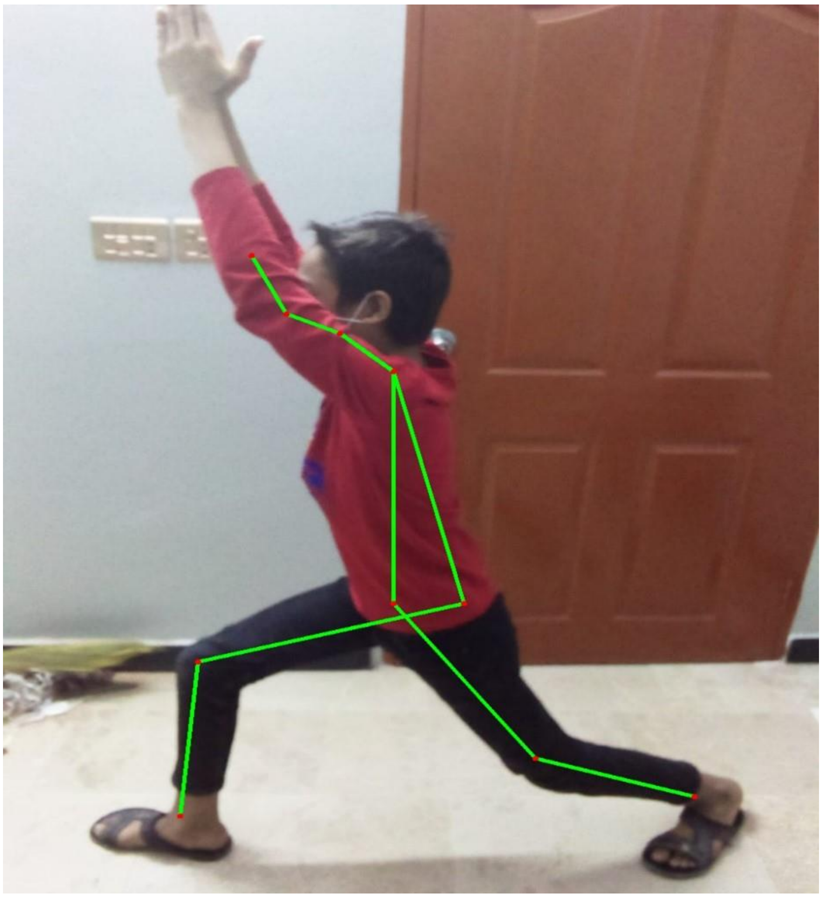

Figure 3:8 Joint Point Detection-5

Figure 3:9 Joint Point Detection-6

3.1.1.3 Layers in Sequential Model

Our system was built on libraries of tensorflow from Keras. It is a stack of layers, each layer has weights that correspond to the layer that follows it. It is called sequential as it has a single path for forward and back-propagation. There are different type to layers like Conv2d, MaxPooling2d, Dropout, Flatten, Dense, Activation etc

1) **Conv2d:** This layer is to convolve an convolution kernel on the input to produce a tensor of outputs conv2d is for spatial data where as conv1d is for temporal data. There are many input parameters in this layer such as padding, kernel and if its the first layer then we have to define the input size of the data set.
2) **Activation Function:** the activation function is used to give non linearity in the model we here use ReLU (Rectified Linear Unit) function. It’s cheap to compute as there is no complicated math. The model can therefore take less time to train or run. Mathematically, it is defined as y = max(0, x).

3) **MaxPooling2d:** This layer is used reducing its dimensionality and enhance the features contained in the sub-regions binned. The parameters are subregions matrix size.
4) **Dropout:** This layer used for preventing over fitting and regularization of data. With the addition of this layer randomly selected neurons are ignored during training and so do not update the weights during back propagation.
5) **Flatten:** This layer plays a simple role just reshape the input to single row of equal number of elements in the input matrix.
6) **Dense:** A simple name of this layer is fully connected layer all the output of previous layers are connected to the input of all the neurons in the dense layer specially use in the ending of the model to converge the solution.

3.1.1.3.1 Architecture of Sequential Model

```
model = Sequential()
#32 filters with 3x3 matrix of pixels which generate dot product with the random 3x3 matrix, done for all 32
# Note the input shape is the desired size of the input image 128,1288 with 3 bytes color
model.add(Conv2D(32,kernel_size= (3, 3), activation='relu',input_shape=( 128, 128,3),padding='same'))
model.add(Conv2D(64,kernel_size= (3, 3), activation='relu',padding='same'))

model.add(MaxPooling2D(pool_size=(2, 2), padding='same'))#reduce image size and overfitting
model.add(Dropout(0.25))
model.add(Conv2D(64,kernel_size= (3, 3), activation='relu',padding='same'))
model.add(Conv2D(64,kernel_size= (3, 3), activation='relu',padding='same'))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dense(3)) #because of 3 classes
model.add(Activation('softmax'))

model.compile(loss=keras.losses.categorical_crossentropy,
          optimizer=keras.optimizers.Adam(),
          metrics=['accuracy'])
model.summary()
```

Figure 3:10 Architecture of Sequential Model

3.1.1.3.2 Hyper parameters

1. **Learning rate:** The learning rate of the model is set to 0.001 by default.
2. **Epochs:** The model was run on 25 and 50 epochs for proper training, testing and validation.
3. **Batch size:** The batch size for the weight update is set to 32.
4. **Loss Function:** Categorical cross-entropy is used to calculate the average difference between the predicted and actual probability distributions for all problem classes. It is minimized on the completion of each batch and the best score is 0.

$$-\frac{1}{N}\sum_{i=1}^{N}\log p_{model}\left[y_i \in C_{y_i}\right]$$

Figure 3:11 Categorical Cross-Entropy Loss Function

5. **Adam Optimizer:** It’s a back-propagation algorithm in which learning rate is maintained for each network weight (parameter) and separately adapted as Fig 12: Architecture Model of First solution Fig 13: Categorical cross-entropy loss function learning unfolds. Whereas in Stochastic gradient descent learning rate is maintained for weight updates throughout the training.

3.1.1.4 Evaluate Model/Model Fit

After fitting the model and gaining training, testing, validation accuracy we display out output on a graph where x-axis is “no. of Epochs” and y-axis is the “Accuracy value %” or “Loss value %”.

3.1.1.4.1 Accuracy, Loss on 25 Epochs

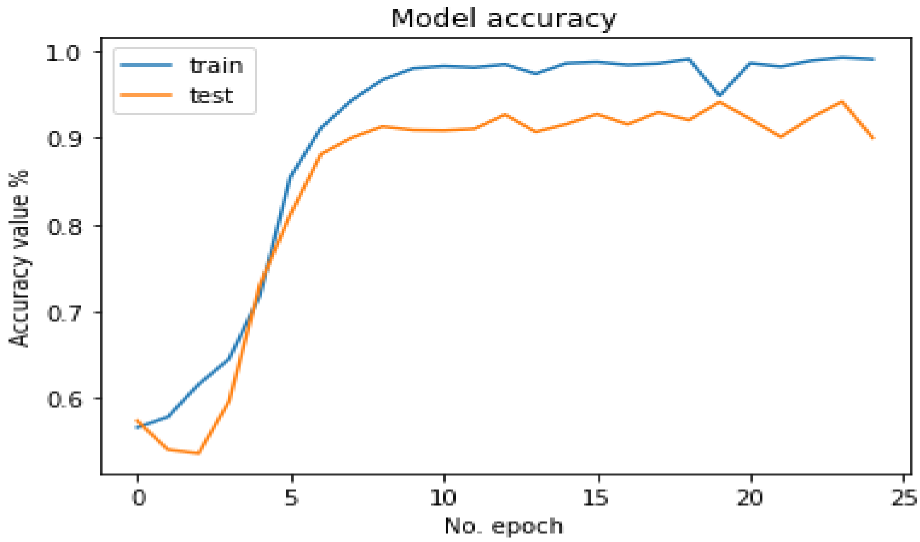

Figure 3:12 25 Epochs Model Accuracy

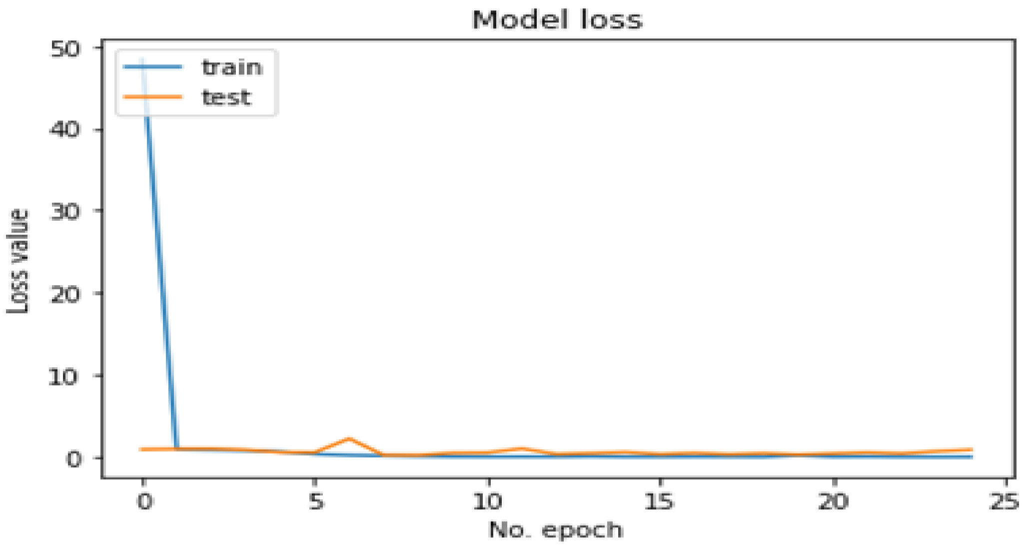

Figure 3:13 25 Epochs Model Loss

3.1.1.4.2 Accuracy, Loss on 50 Epochs

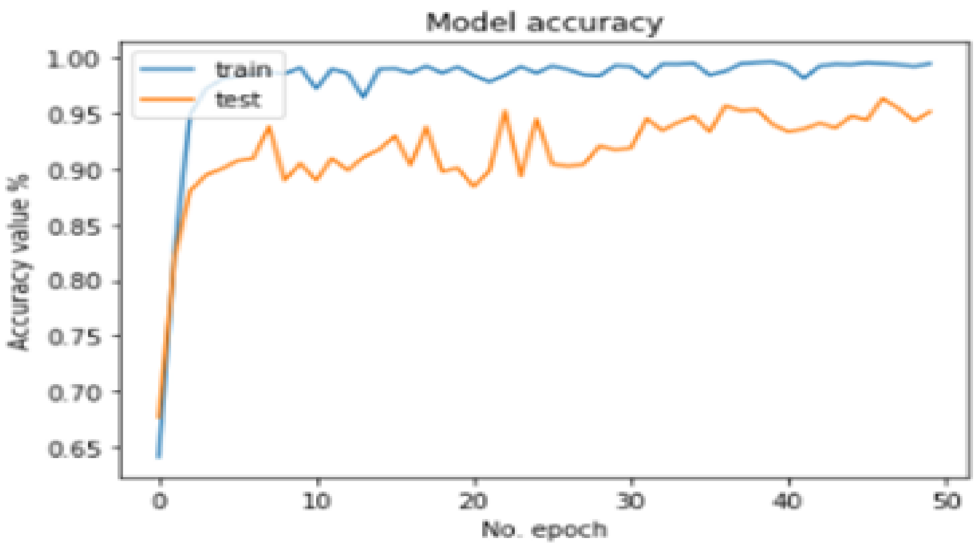

Figure 3:14 50 Epochs Model Accuracy

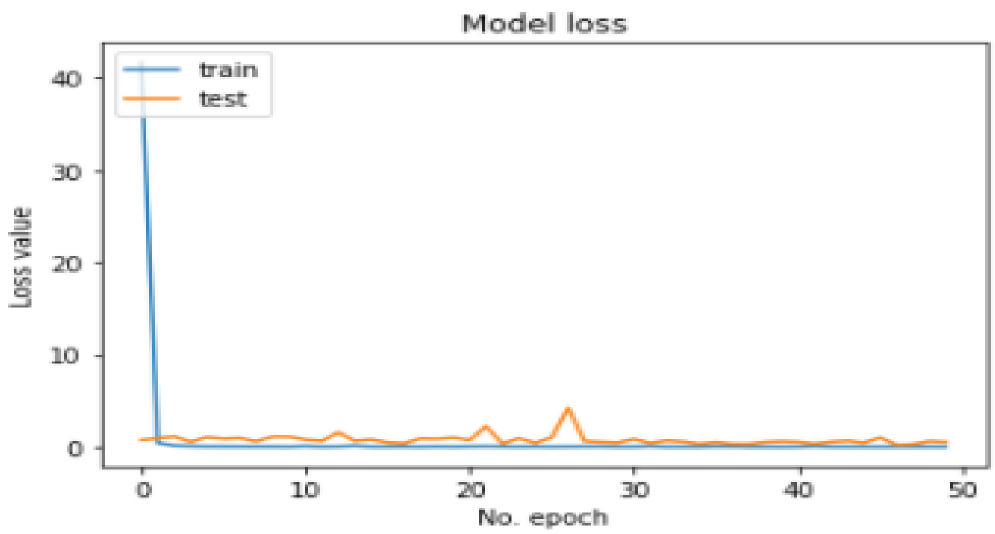

Figure 3:15 50 Epochs Model Loss

3.1.1.5 Video Testing

Figure 3:16 Video Testing Results

3.1.2 Software Development Life Cycle (SDLC)

The iterative model as shown in Figure 3.3 is a specific implementation of a software development life cycle (SDLC) that focuses on an initial, simplified implementation, which then gradually gains more complexity and a wider feature set until the final system is complete [20]. The definition of incremental growth will also often be used liberally and interchangeably when addressing the iterative process, which defines the gradual changes made during each new iteration's design and implementation.

Figure 3:17 Iterative Methodology

3.2 Model Analysis

In case of both 25 and 50 epochs, we can see that the validation loss is decreasing whereas the validation accuracy is increasing. It means model build is learning and working fine and this is how it should work for the proper image classification.

CHAPTER 4

DESKTOP APPLICATION

A desktop application is a software application that runs on a devices such as pc, laptops etc. Due to easily video making excess of these devices without any other support, these applications are very easily reachable for public.

4.1 Implementation of Desktop Module

4.1.1 Language and Framework

We have used python language. We have created CNN model using Keras with tensorflow as backend and saved that model and load it into Tkinter(Tk) framework, which is used as frontend GUI to load images/videos or capture live through camera and then perform classification.

4.1.2 Activities and Fragments

There is just 1 activity and 3 fragments, and much of the programming is performed in fragments so that in the future the code can be scaled.

4.1.2.1 Activity:

1. StartPage Activity: This is the primary operation where the entire application begins from. It only initiates the beginning fragment.

4.1.2.2 Fragments:

1. **StartPage:** This is the home page of the program where there are three buttons, one for to upload image, one for to upload videos and one for real-time exercise analysis.

Figure 4:1 StartPage

2. **ImageUploader:** When user chooses to analyze an image already saved in directory for that it presses upload image button to upload image using file

directory, then a button classify will be shown when image is uploaded so we can classify that image pose.

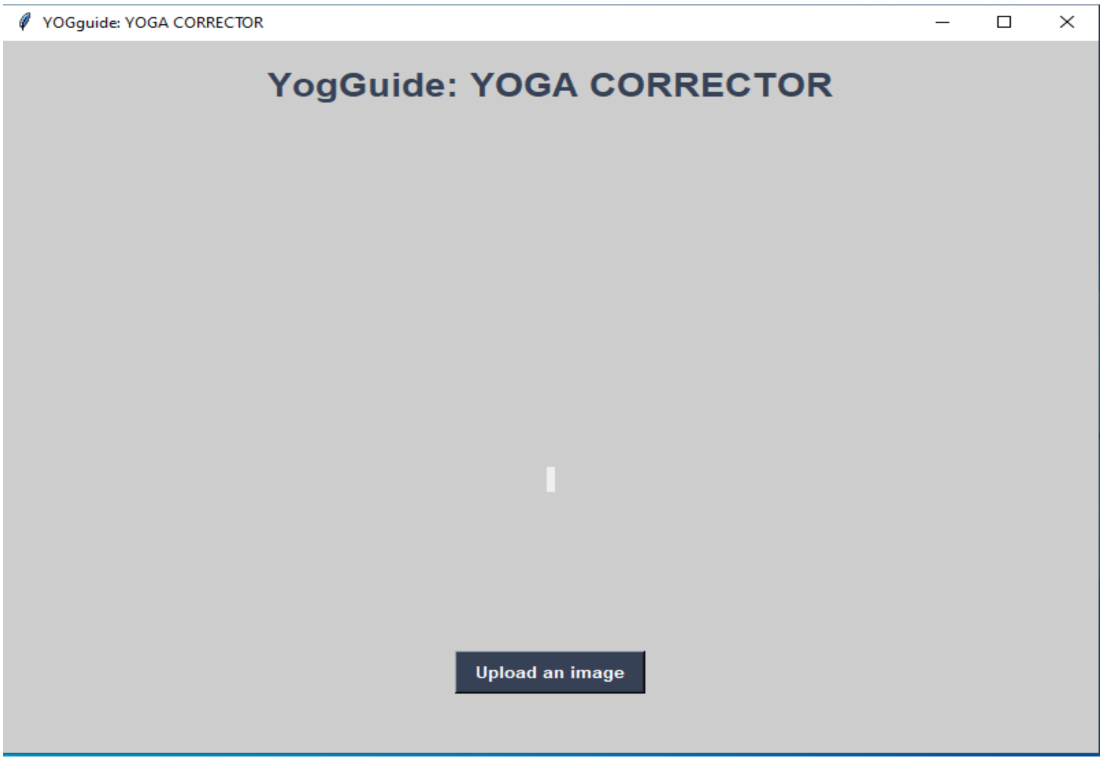

Figure 4:2 Image Uploader-1

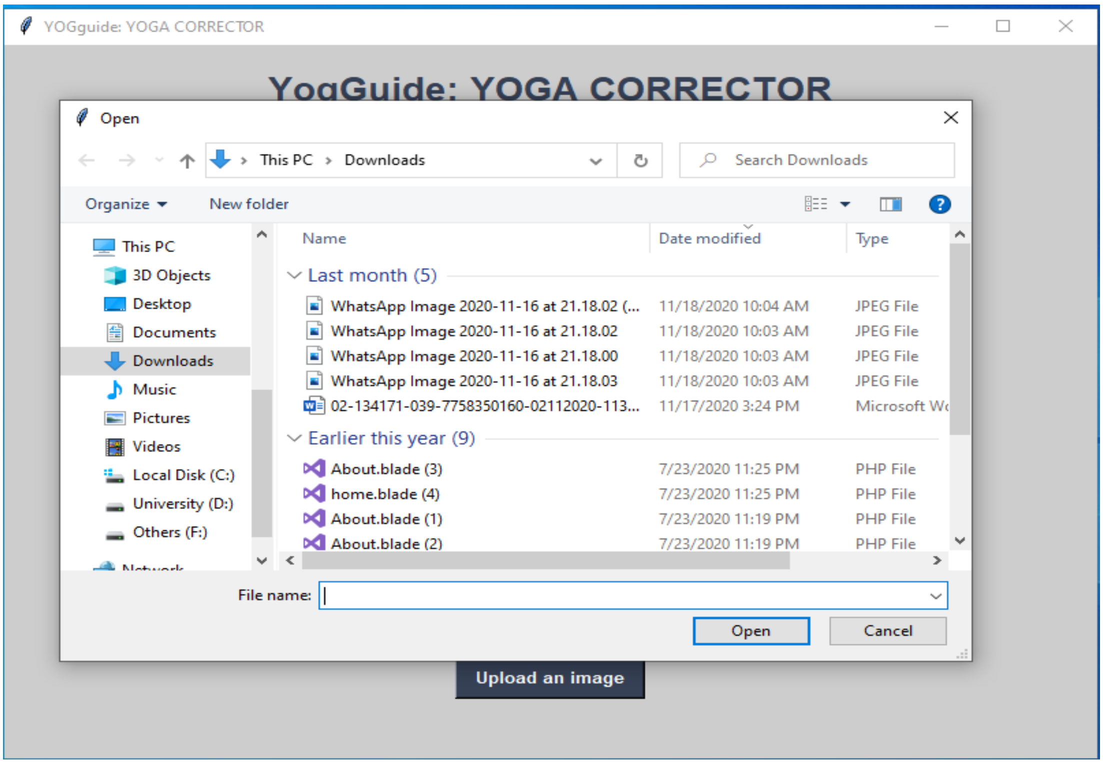

Figure 4:3 ImageUploader-2

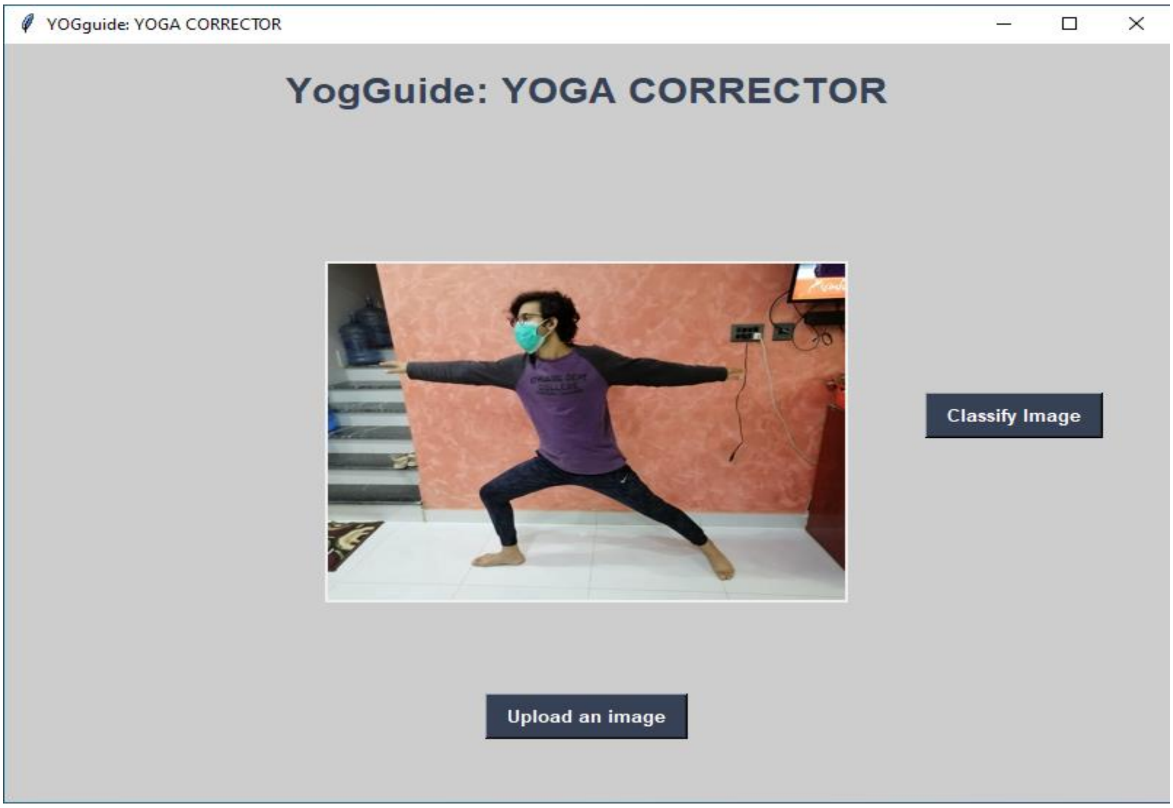

Figure 4:4 Image Uploader-3

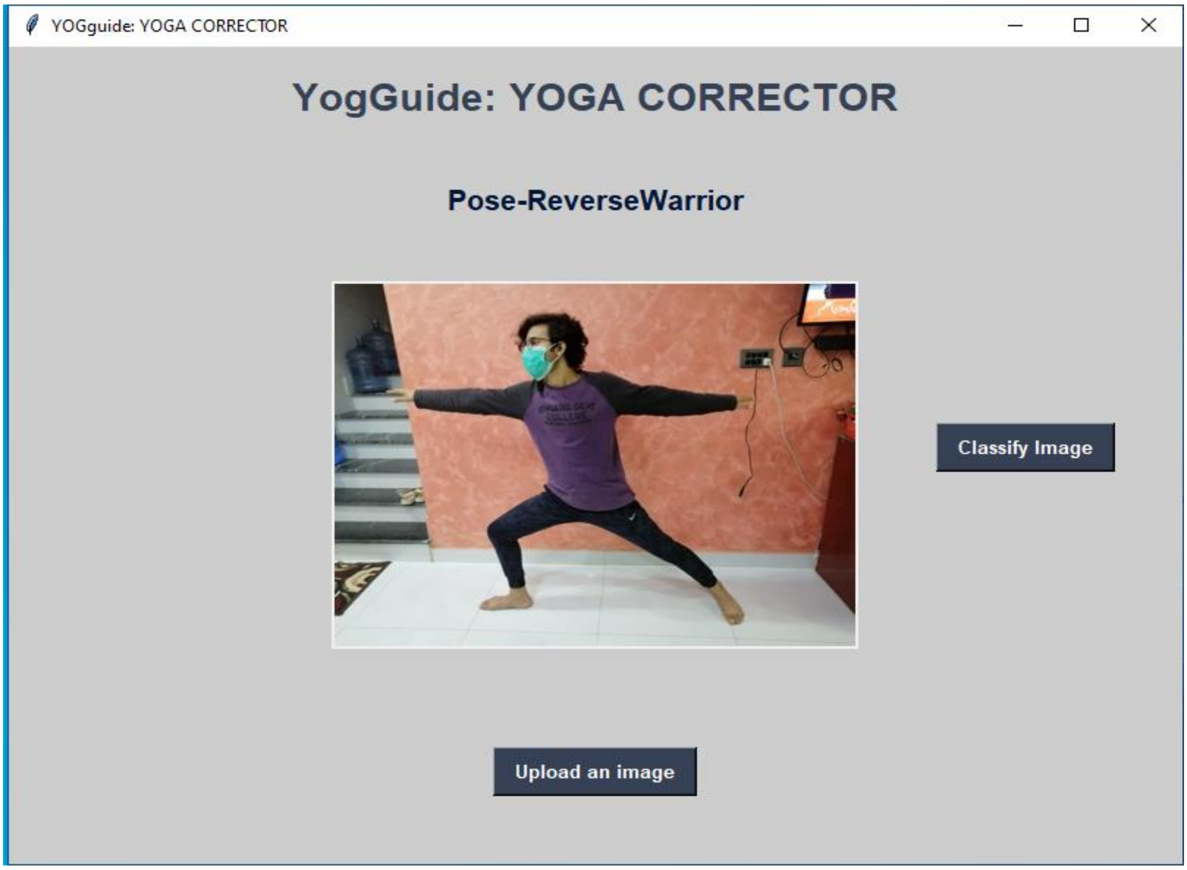

Figure 4:5 ImageUploader-4

3. **VideoPlayer:** When the user chooses to analyze previously recorded video and press upload video button then this fragment is initialized, it uses OpenCV to read video frame by frame and then after processing display the results.

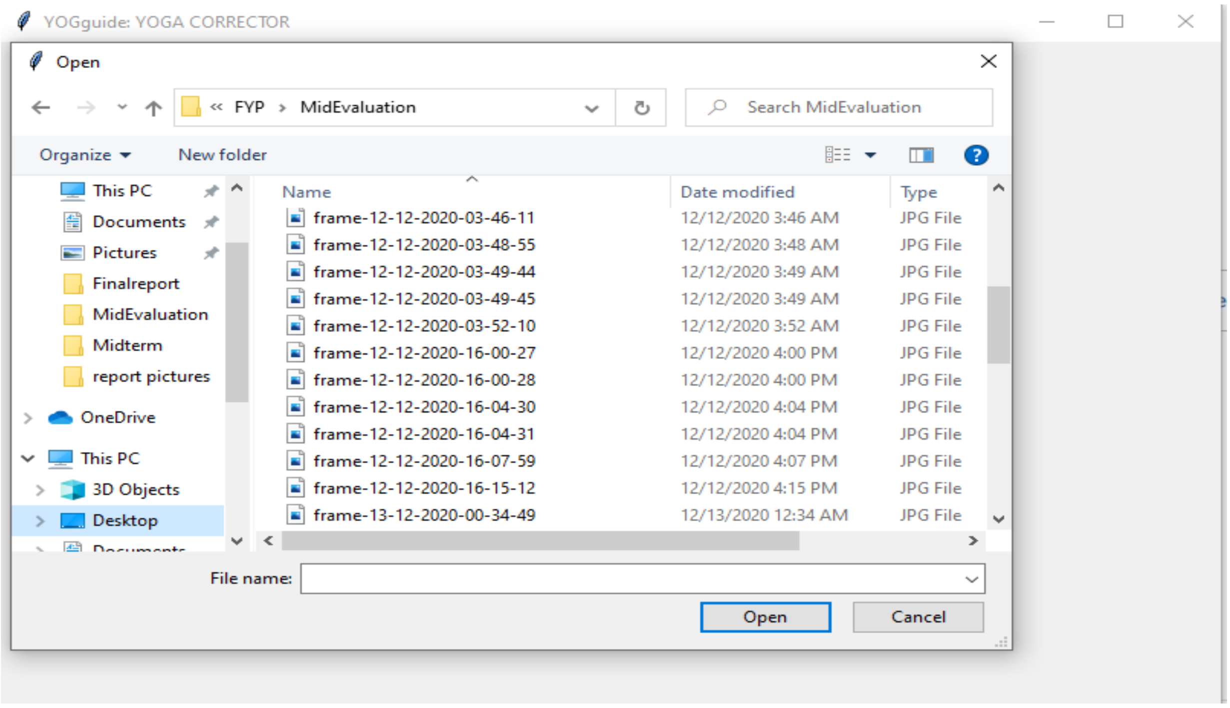

Figure 4:6 VideoUploader-1

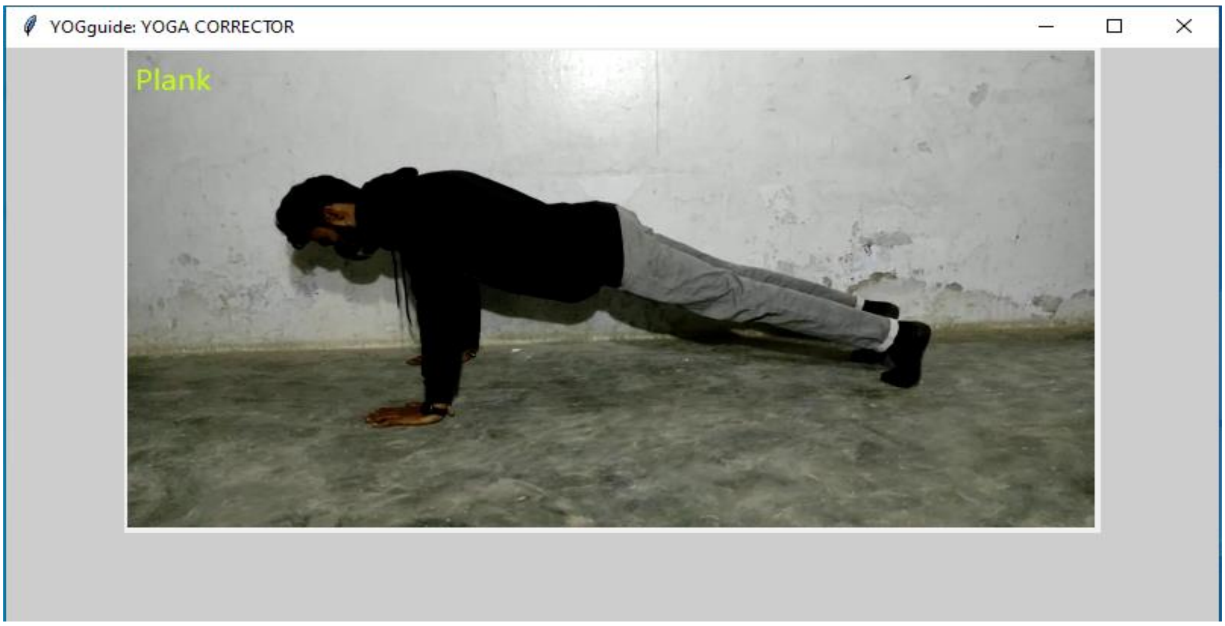

Figure 4:7 VideoUploader-2

4. **LiveCameraCapture:** When the user chose to take live picture from their webcam or any source and then classify that pose. The camera can classify pose near to 25ms.

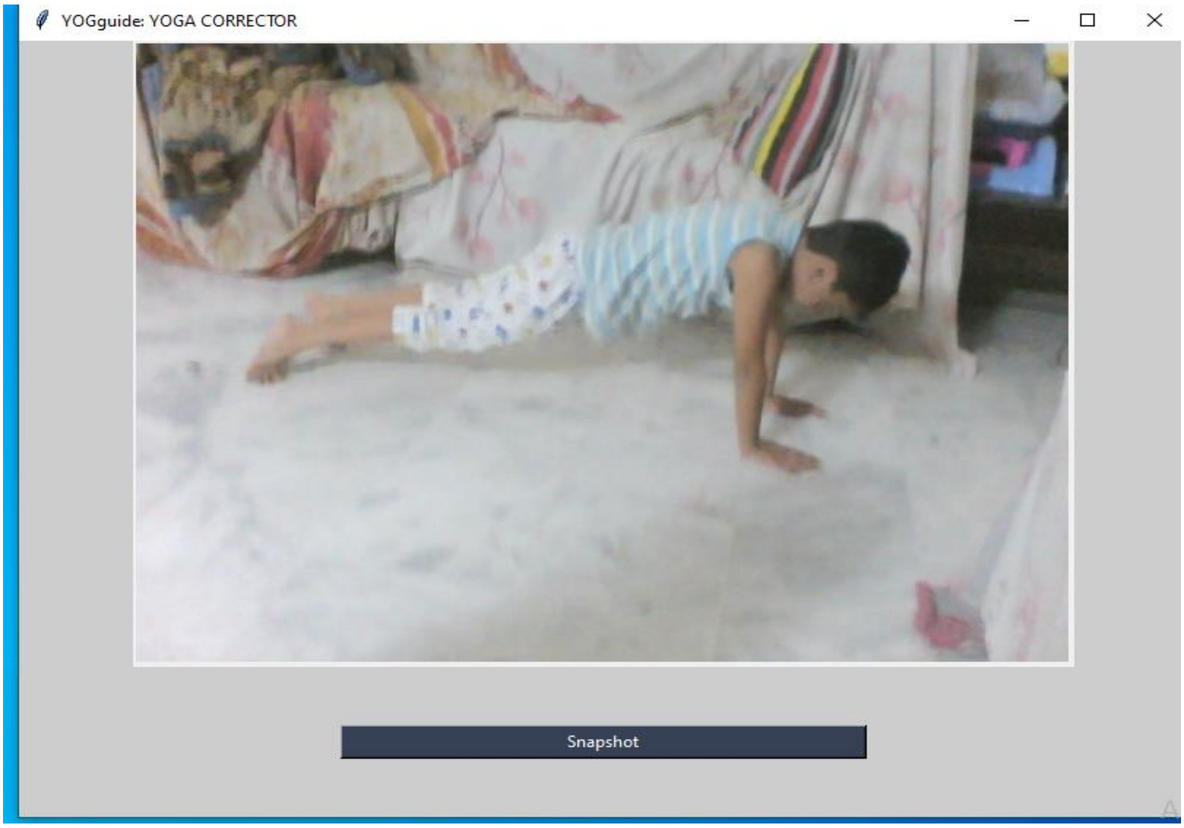

Figure 4:8 LiveCameraCapture-1

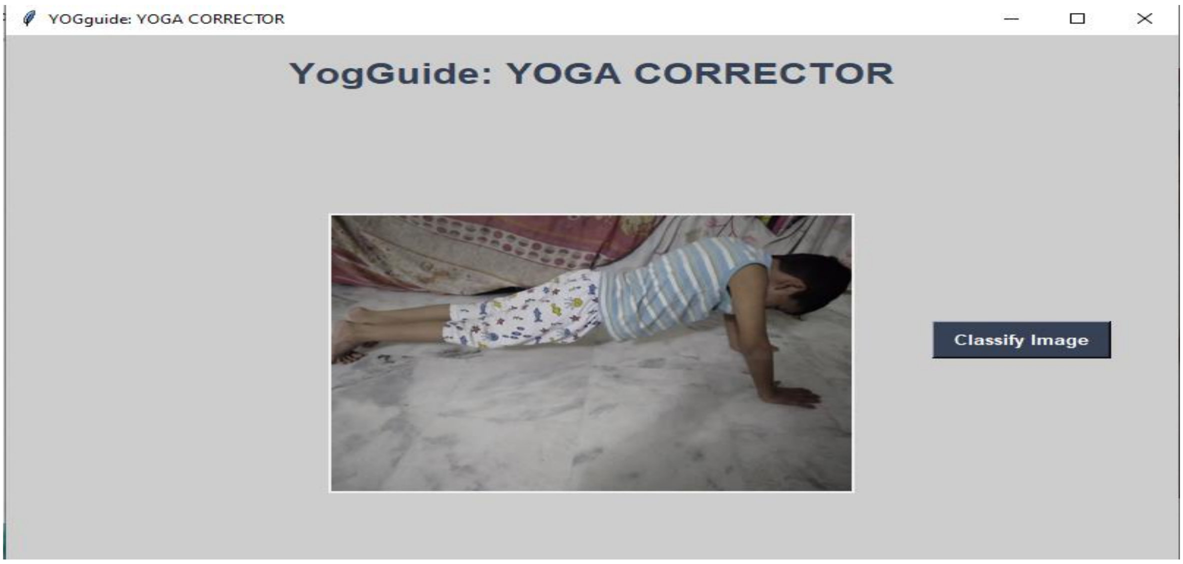

Figure 4:9 LiveCameraCapture-2

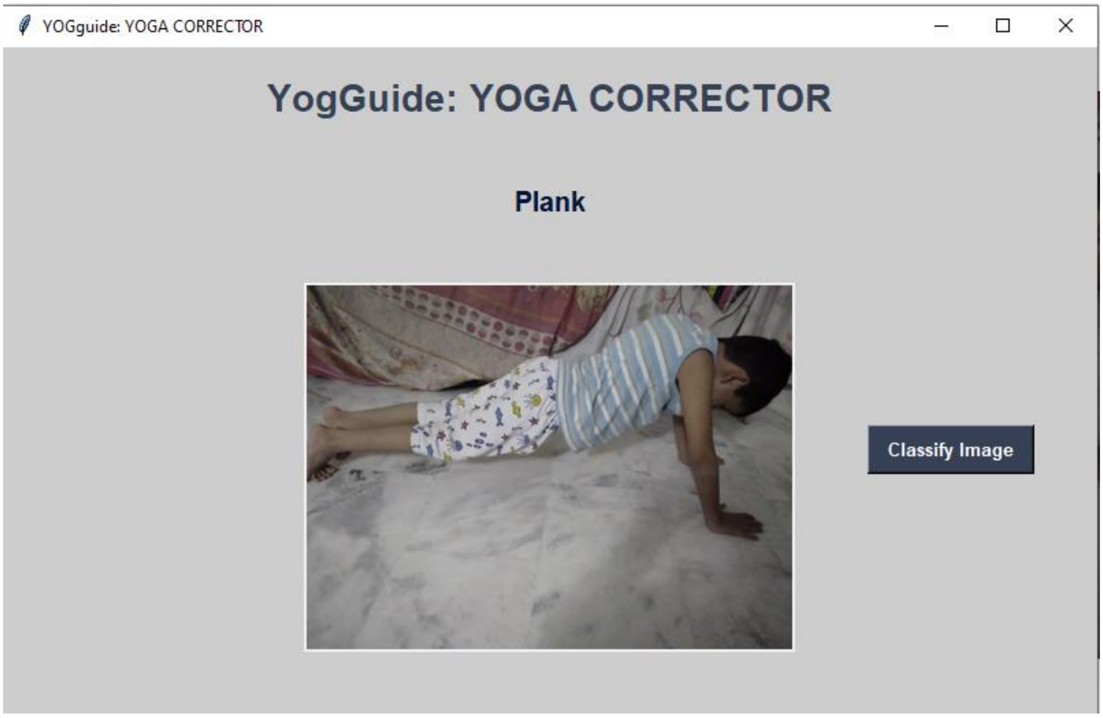

Figure 4:10 LiveCameraCapture-3

4.1.3 Application Workflow Diagram

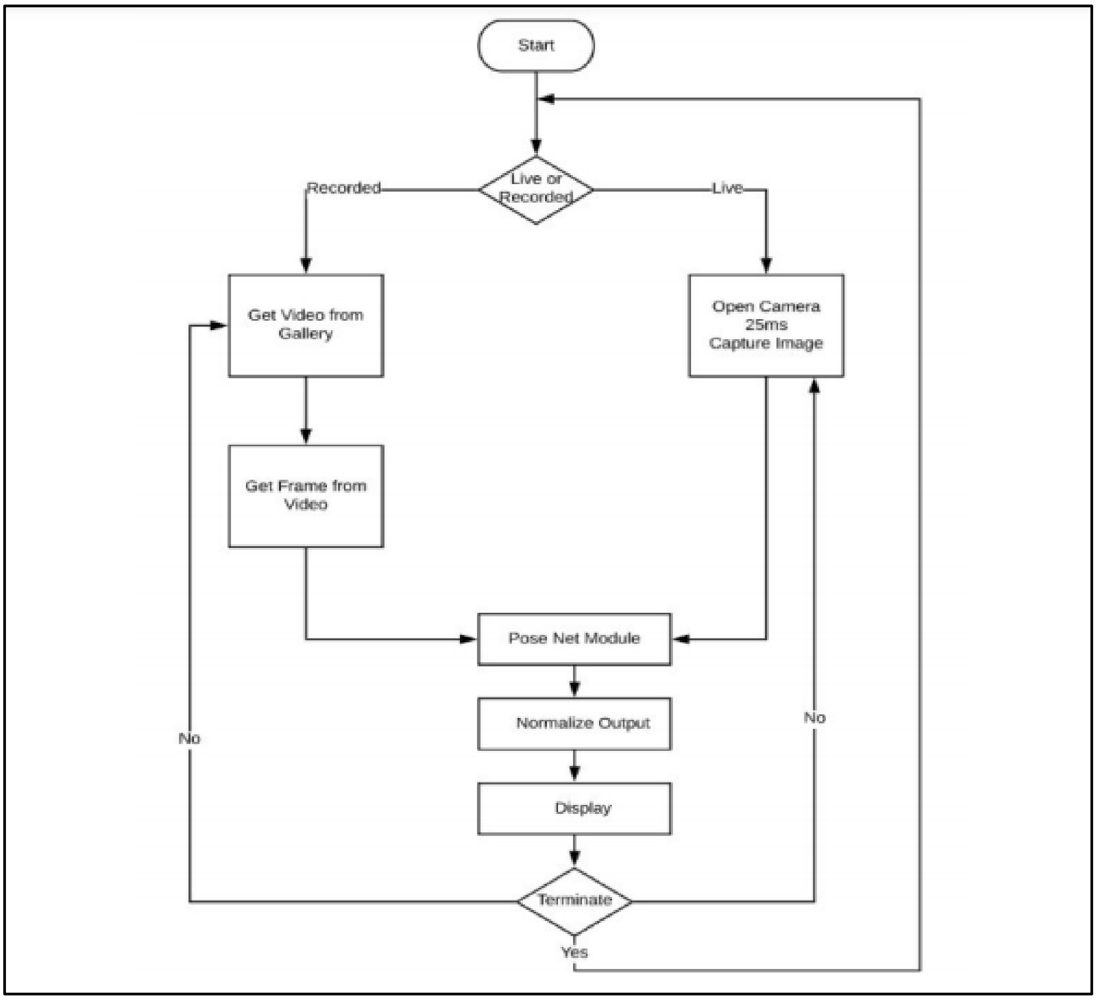

Figure 4:11 WorkFlow of Proposed Solution

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Problems

As it was taken from a wide distance, the data set used for the posenet was not good for identifying body key joints. And we were unable to recreate the entire data set due to COVID-19. For better results on our current data collection, We tried another pose estimation model (Open-pose having 18 body key joints), but these models require fast processing power that an ordinary laptop devices does not have, so it crashes when we tried to run.

5.2 Future Directions

Just 3 yoga poses are currently categorised by the proposed models. There are a variety of yoga poses, and hence developing a pose estimation model that can be accurate for all the poses is a challenging problem. The dataset can be extended by adding more yoga poses performed by individuals. Further, we can also work on multi person pose estimation.

This idea can be further extended and we can work on to detect joint points regardless of any body type. Moreover, joint points should be detected on any type of clothing either fitted or lose. Last, it can also be extended to detect human and non-human and then do pose estimation.

5.3 Summary

Our YogGuide Application is a supervised machine learning project where we uses two different techniques for our pose estimation one is CNN Image classification sequential model, and we are considering 3 different poses including plank, warrior and pose-reverse warrior. Another approach was to build sequential model over the pre-build tensor flow posenet poseestimation model.

We have created our CNN model using keras with tensorflow as back end. We created model architecture by ourselves and then perform training and validating of our model on our 3 poses. At last, we loaded and saved our model to our GUI framework which is tkinter (tk) which can be easily used by users to anaylyze their pose.

REFERENCES

[1] A. Toshev and C. Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks," 2014, pp. 1653–1660, Accessed: Aug. 15, 2020. [Online]. Available:

https://openaccess.thecvf.com/content_cvpr_2014/html/Toshev_DeepPose_Human_Pose_2014_CVPR_paper.html.

[2] Y. Chen, C. Shen, X.-S. Wei, L. Liu, and J. Yang, "Adversarial PoseNet: A Structure-Aware Convolutional Network for Human Pose Estimation," 2017, pp. 1212–1221, Accessed: Aug. 15, 2020. [Online]. Available: https://openaccess.thecvf.com/content_iccv_2017/html/Chen_Adversarial_PoseNet_A_ICCV_2017_paper.html.

[3] S. K. Yadav, A. Singh, A. Gupta, and J. L. Raheja, "Real-time Yoga recognition using deep learning," *Neural Comput. Appl.*, vol. 31, no. 12, pp. 9349–9361, Dec. 2019, doi: 10.1007/s00521-019-04232-7.

[4] "The Health Benefits of Yoga and Exercise: A Review of Comparison Studies | The Journal of Alternative and Complementary Medicine." https://www.liebertpub.com/doi/abs/10.1089/acm.2009.0044 (accessed Aug. 15, 2020).

[5] N. L. Atkinson and R. Permuth-Levine, "Benefits, Barriers, and Cues to Action of Yoga Practice: A Focus Group Approach," *Am. J. Health Behav.*, vol. 33, no. 1, pp. 3–14, Jan. 2009, doi: 10.5993/AJHB.33.1.1.

[6] M. Gochoo *et al.*, "Novel IoT-Based Privacy-Preserving Yoga Posture Recognition System Using Low-Resolution Infrared Sensors and Deep Learning," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 7192–7200, Aug. 2019, doi: 10.1109/JIOT.2019.2915095.

[7] T. K. K. Maddala, P. V. V. Kishore, K. K. Eepuri, and A. K. Dande, "YogaNet: 3-D Yoga Asana Recognition Using Joint Angular Displacement Maps With ConvNets," *IEEE Trans. Multimed.*, vol. 21, no. 10, pp. 2492–2503, Oct. 2019, doi: 10.1109/TMM.2019.2904880.

[8] E. W. Trejo and P. Yuan, "Recognition of Yoga Poses Through an Interactive System with Kinect Device," in *2018 2nd International Conference on Robotics and Automation Sciences (ICRAS)*, Jun. 2018, pp. 1–5, doi: 10.1109/ICRAS.2018.8443267.

[9] H.-T. Chen, Y.-Z. He, and C.-C. Hsu, "Computer-assisted yoga training system," *Multimed. Tools Appl.*, vol. 77, no. 18, pp. 23969–23991, Sep. 2018, doi: 10.1007/s11042-018-5721-2.

[10] "Yoga posture recognition by detecting human joint points in real time using microsoft kinect - IEEE Conference Publication." https://ieeexplore.ieee.org/abstract/document/8289047/ (accessed Aug. 15, 2020).

[11] F. Jiang, S. Zhang, S. Wu, Y. Gao, and D. Zhao, "Multi-layered Gesture Recognition with Kinect," in *Gesture Recognition*, S. Escalera, I. Guyon, and V. Athitsos, Eds. Cham: Springer International Publishing, 2017, pp. 387–416.

[12] F. Liu, B. Du, Q. Wang, Y. Wang, and W. Zeng, "Hand gesture recognition using kinect via deterministic learning," in *2017 29th Chinese Control And Decision Conference (CCDC)*, May 2017, pp. 2127–2132, doi: 10.1109/CCDC.2017.7978867.

[13] D. Núñez Fernández and B. Kwolek, *Hand Posture Recognition Using Convolutional Neural Networks*. 2019.

[14] Seung-Ho Han, Han-Gyu Kim, and Ho-Jin Choi, "Rehabilitation posture correction using deep neural network," in *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Feb. 2017, pp. 400–402, doi: 10.1109/BIGCOMP.2017.7881743.

[15] Z. Chan, Z. XunMu, Y. XiaoFan, W. LiNa, T. ShuQin, and X. YueJu, “Automatic recognition of lactating sow postures from depth images by deep learning detector.,” *Comput. Electron. Agric.*, vol. 147, pp. 51–63, 2018.
[16] S. Kothari, “Yoga Pose Classification Using Deep Learning,” p. 45.
[17] “Iterative and incremental developments. a brief history - IEEE Journals & Magazine.” https://ieeexplore.ieee.org/abstract/document/1204375 (accessed Aug. 15, 2020).

APPENDICES

APPENDIX A: Project Milestones

S. No	Elapsed time since start of the project	Milestone	Deliverable
1	February (Research)	Read scientific papers Gather data insights Decide on methodology	Detailed project execution plan
2	March (Data exploration)	Slice data in different ways Gather datasets	Report with the results
3	April (Measurement of angles)	Body joints point detection Report writing	No deliverables
4	May (1st module, Literature review)	Skeleton information Data processing Related research paper writing	Validate dataset to the supervisor Literature review writing
5	June (Posture Model Building)	Model building	Presentation and report submission Mid evaluation
MID EVALUATION			

6	July, August (Desktop application)	Application designing and development Report writing	Report with the design and front-end
7	September (2nd module, Literature review)	Application processing Related research paper writing	Report the application Literature review writing
9	October (Test the model on application)	Testing application Report writing	Test report First draft of report
10	November (Final model)	Maintenance of application	Final report
FINAL EVALUATION			

APPENDIX B: Project Gantt Chart