**DEEP LEARNING BASED YOGA LEARNING APPLICATION**

A DISSERTATION

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

MASTER OF TECHNOLOGY

IN

**INFORMATION SYSTEMS**

Submitted by:

**Pavitra Gandhi**

**2K19/ISY/12**

Under the supervision of

Ms. Anamika Chauhan

**DEPARTMENT OF INFORMATION TECHNOLOGY**  
DELHI TECHNOLOGICAL UNIVERSITY(Formerly Delhi College of Engineering)  
Bawana Road, Delhi – 110042

**JUNE, 2021**

# **CANDIDATE’S DECLARATION**

I, Pavitra Gandhi – 2K19/ISY/12) student of M.Tech, Information Systems, hereby declare that the project Dissertation titled "**Deep Learning Based Yoga Learning Application**" which is submitted by me to the Department of Information Technology, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi **PAVITRA GANDHI**

June 30, 2021

# CERTIFICATE

I hereby certify that the Project Dissertation titled “**Deep Learning Based Yoga Learning Application**” which is submitted by Pavitra Gandhi 2K19/ISY/12, Department of Information Technology, DTU, Delhi in partial fulfilment of the requirement for the award of the degree of Master of Technology, is a record of the project carried out by the students with my supervision. To the best of my knowledge, this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi **MS. ANAMIKA CHAUHAN**

June 30, 2021 **SUPERVISOR**

# **ABSTRACT**

Yoga has been into practice for thousands of years and in today’s era it is highly suggested for people to practice it to revitalize them in their physical and mental wellness. Although yoga is considered safe, there are several asanas that need to be practiced under the guidance and supervision of a trained instructor. Using human action recognition techniques for yoga pose estimation, we have tested and designed an automated yoga asana identification and correction system that performs well in the absence of a trained instructor. This application finds its importance specially the times of the COVID-19 epidemic where social distancing is the new norm.

The application identifies yoga asana being practised by the user in real time with the help of deep learning techniques based on convolutional neural networks (CNN) and transfer learning which might require correction. The application corrects the user for right yoga posture using multi-person 2D pose estimation algorithm called OpenPose. It uses multiple deep learning algorithms for pose estimation.

The application is designed for 18 different asanas which the users can practice for a start. The system can predict an asana with over 87.6% accuracy considering the fact that user can face the camera with multiple different views while practicing yoga, namely left-hand side view, right-hand side view or front view; depending on the pose. This is taken into consideration so that it gets very much easier for the user to practice yoga without thinking about how the user should be facing the camera hence we can say that it is more user friendly. The system corrects the user by showing the direction in which they should move the body part, which is not in the correct position in real time, as a feedback to the user. As and when the user changes its posture the feedback to the user changes till the user gets the right posture.

# **ACKNOWLEDGEMENT**

The completion of this work has been possible only because many people came forward for their guidance and support. I am extremely grateful to them who helped us reach this point with our work and we would like to acknowledge their help for the same.

I am extremely grateful to our supervisor, **Ms Anamika Chauhan**, for helping me along from the beginning. She responded keenly to all my queries and helped me resolve my doubts. I heartily thank her for her help and support to me, in order to complete this project.

I am also thankful enough to get all the help and support from the whole of the IT Department, who encouraged me and provided their help in much needed times. I would also like to thank the staff from the computer labs who helped me get all the required technical resources.

PAVITRA GANDHI 2K19/ISY/12

TABLE OF CONTENTS

[CANDIDATE’S DECLARATION i](#_Toc40137144)

[CERTIFICATE ii](#_Toc40137145)

[ABSTRACT iii](#_Toc40137146)

[ACKNOWLEDGEMENT Iv](#_Toc40137147)

[LIST OF TABLES v](#_Toc40137148)I

[LIST OF FIGURES vIi](#_Toc40137149)

[Chapter 1: introduction 1](#_Toc40137150)

[1.1 Principle behind Working of Human Action Recognition](#_Toc40137152) 2

[1.2 Organization of the Thesis](#_Toc40137152) 3

[Chapter 2: YOGA: IMPORTANCE, HURDLES AND SOLUTION 4](#_Toc40137164)

[2.1 Why yoga is critically important and related problems?](#_Toc40137166) 4

[2.2 Techniques to address the issue?](#_Toc40137166) 6

[*2.2.1* Human Pose Estimation](#_Toc40137194) 6

[2.2.1.1 Skeleton-based model](#_Toc40137195) 6

[2.2.1.2 Contour-based model](#_Toc40137196) 7

[2.2.1.3 Volume-based model](#_Toc40137197) 7

[Chapter 3: RELATED WORK](#_Toc40137188) 8

[Chapter 4: PROPOSED methodology](#_Toc40137188) 14

[4.1 tools used:](#_Toc40137189) 14

[4.1.1 Programming Platform: Python 3.6](#_Toc40137190) 14

[4.1.2 Libraries Used](#_Toc40137194) 14

[4.1.2.1 OpenCV](#_Toc40137195) 15

[4.1.2.2 TensorFlow](#_Toc40137196) 15

[4.1.2.3 PyTorch](#_Toc40137196) 16

[4.1.2.4 Pandas](#_Toc40137197) 16

[4.1.2.5 Keras](#_Toc40137197) 17

[4.1.2.6 Scikit-Learn](#_Toc40137197) 17

[4.1.2.7 Matplot Lib](#_Toc40137197) 17

[4.1.2.8 Pickle](#_Toc40137197) 17

[4.2 Dataset Collection and Preparation 1](#_Toc40137198)8

[4.3 Architecture 1](#_Toc40137189)8

[Chapter 5: results and analysis 2](#_Toc40137150)6

[Chapter 6: conclusion AND FUTURE 32](#_Toc40137150)

[REFERENCES](#_Toc40137204) 33

# **LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Table. No.** | **Title** | **Pg. No.** |
| 4.1 | Shows list of all the asanas take in the dataset along with their views | 19 |
| 5.1 | Represents a comparison of performance of deep learning models | 26 |
| 5.2 | Shows the list of reference and actual user angles and deviation between each other of a single frame. | 28 |
| 5.5 | Represents the ideal posture of Paripurna Navasana used as reference to compare with the user’s same body pose | 29 |

# **LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Fig. No.** | **Title** | **Pg. No.** |
| 1.1 | Pose-driven Human Action Recognition and Anomaly Detection | 3 |
| 2.1 | Visual representation of yoga | 5 |
| 2.2 | a.) Shows a female practicing Parivrtta Anjaneyasana b.) shows how wrongly performing yoga can lead to negative effects like back pain | 5 |
| 2.3 | Represents a.) the skeleton-based body model b.) contour-based body model along with rectangles signifying body parts c.) volume-based body model (3D) | 7 |
| 3.1 | Range point clouds in 3 dimensions of person picking a box | 8 |
| 3.2 | Represents joint point locations and connections with respect to the human body fetched from Kinect camera device | 10 |
| 3.3 | Represents basic functionality of AdaBoost algorithm | 11 |
| 4.1 | Represents Virabhadrasana performed in different backgrounds or environments and in different attire and different conditions. | 18 |
| 4.2 | Shows left-hand side facing and right-hand side facing view of Anjaneyasana | 18 |
| 4.3 | Shows the workflow diagram of the proposed method | 21 |
| 4.4 | Modified VGG-16 architecture | 22 |
| 4.5 | Represents 18 keypoints skeletal structure of a person practicing Padmasana | 24 |
| 5.1 | Represents the training and validation accuracy plot with the rising epochs of VGG-16 model | 26 |
| F**ig. No.** | **Title** | **Pg. No.** |
| 5.2 | Represents the training and validation loss plot with the rising epochs of VGG-16 model. | 27 |
| 5.3 | Represents the interface the user sees when he practices Paripurna Navasana incorrectly | 27 |
| 5.4 | Represents the ideal posture of Paripurna Navasana used as reference to compare with the user’s same body pose | 28 |
| 5.5 | Represents the ideal posture of Paripurna Navasana used as reference to compare with the user’s same body pose | 29 |
| 5.6 | Represents the interface the user sees when he practices Virabhadrasana | 30 |
| 5.7 | Represents the reference image of the user used for Virabhadrasana | 30 |

# 

# CHAPTER 1

**INTRODUCTION**

Human action recognition has been catching eye of many researchers working in the field of Computer Vision. The core principle behind it is analysis of a video for identification of human action occurring in it. Whenever we think about a video the first thing that we can conclude is that it has some spatial properties with it i.e., each of the single frames and the temporal properties i.e., the arrangement of the frames. Some of the common activities like standing, running, etc. can be recognized just with the help of a single video frame but when it comes to more complex activities like walking or running, bending or falling, they most probably will need information from more than a single video frame for correct identification. For the purpose of differentiation between such activities, information about the local temporal parameters plays a key role. But for some of the use cases, this information won’t be sufficient and for that there is a need of long-time interval temporal information for appropriate identification of the activities.

Human action recognition has grabbed the attention of a lot of researchers because of its wide range of applications that directly impact the people, for instance real-time behavior examination, rehabilitation of athletes, elderly care, human-computer interaction and more. Elaborating one of the above; for the purpose of rehabilitation, human motion tracking or human pose estimation has been used since the 1980s. The technicalities behind it’s working are more or less similar to yoga pose estimation. It became a hot topic of research because of the rise in number of people who have gone through a stroke, or some or the other disability in motor functioning. Rehabilitation is a process involving continuous change which permits users to regain their functional capability back to normal and to reach this target, the activities of the user being monitored should be monitored repeatedly, and eventually corrected [1]. Inspired by this, human motion tracking was used for yoga pose detection and its correction, which is one of the most challenging applications in computer vision since the detection of body parts becomes highly difficult because of the complexity of yoga asanas. This is very much in need to help people maintain healthy lives.

## Principle behind Working of Human Action Recognition

 Many traditional approaches are dependent on tools like object detection, dense trajectories, pose detection, and more for the purpose of action recognition. One of the tools used is Convolutional Neural Network which fetches the features from each video frame and then pools the fetched features from various video frames to achieve video-level prediction The disadvantage of this method is that it does not collect enough motion information.Along with the RGB and optical flow, other modalities information for example trajectory, audio and pose can also be leveraged. Recurrent neural networks (RNNs), specifically long short-term memory (LSTM), have gained spectacular results in the tasks which involves sequence because of the ability of long-lasting temporal modelling. Along with the 2-D CNNs utilized for image analysis and processing, 3-D CNNs were also proposed to process the frames in the videos. The 3 × 3 convolutional kernels were replaced with the 3 × 3 × 3 ones to implement 3-D convolutions over stacked frames. However, these techniques have ample number of parameters and along with that they are also required to be trained before on a large-scale data set based on video. Recent

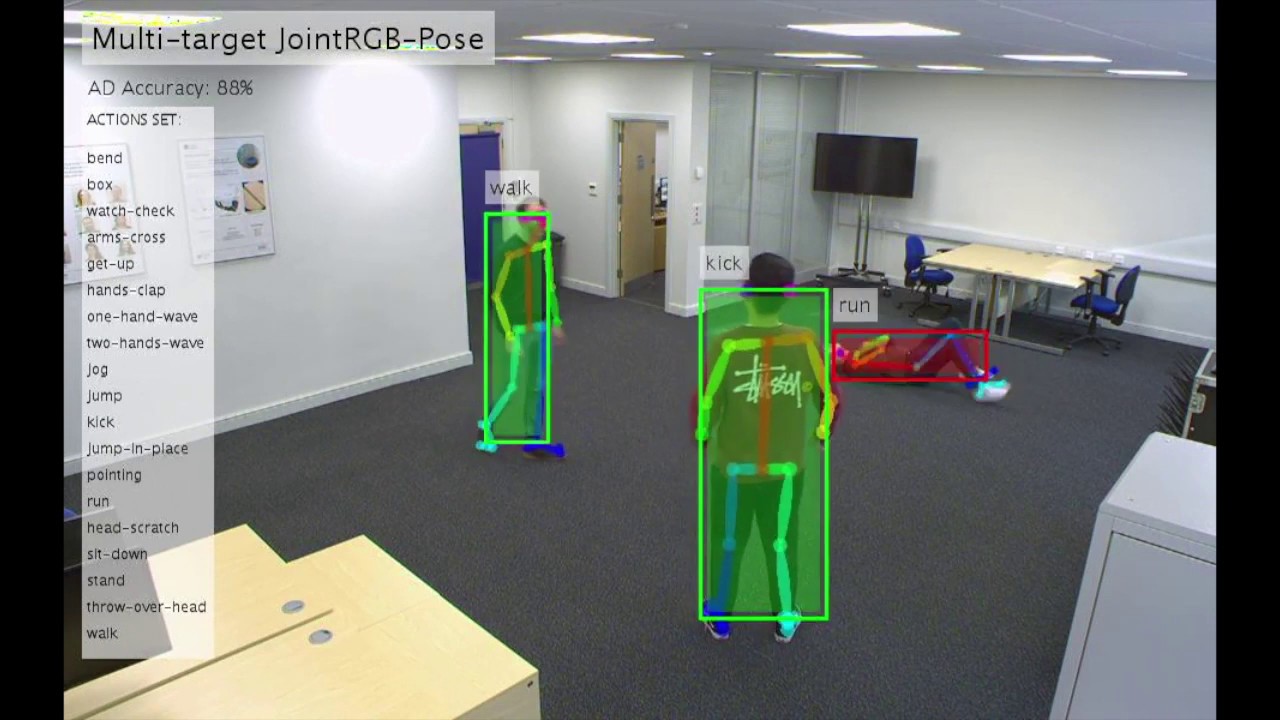
**

Figure 1.1 Pose-driven Human Action Recognition and Anomaly Detection

architectures have emphasized more on the use of attention techniques to choose important areas of the video. This helps to overcome the LSTMs inability to discriminate between different portions of the video. Several other techniques have been proposed and one of the sample outputs of it can be visualised in figure 1.1.

## Organization of the Thesis

In this thesis we have shed light on the existing methods which have been proposed to solve the mentioned problem and existing researches which can very well be used for the same in Chapter 3 and what different and better we have done to fulfill our goal. Chapter 2 introduces, how important yoga is for human health, the challenges people face while practicing yoga daily which cannot be ignored and what can be done to overcome for the betterment of lives of people. Chapter 4 gives a detailed technical information about the algorithm used in our application we have designed. Chapter 5 and 6 talks about the performance of our system, shows the user interface and the effectiveness of this application when it is in the hands of a user.

# 

# CHAPTER 2

**YOGA: IMPORTANCE, HURDLES AND SOLUTION**

## 2.1 Why yoga is critically important and related problems?

In today’s era there is an exponential rise in competition in almost all aspects of life, whether it’s work or education or family. With that, there is an exponential rise in health problems both mentally and physically especially because of the recent COVID outbreak. COVID has brought a lot of constraints on and radical changes in the daily lives of humans and with that the need to practice yoga has also increased. Yoga is estimated to be existing since around 10,000 years because of a wide variety of exercises offering mental and physical strength. Figure 2.1 shows visual representation of yoga. With the advent of smartphones people have started practicing yoga by learning various asanas online as per their needs. Unfortunately, that is not enough since the individual doesn’t realize if they are practicing precisely or not. It is sometimes hard to figure out, to what extent the user should bend their required body parts.

It is important for yoga to be practiced the right way, peculiarly in the correct pose. When it comes to complex poses individuals aren’t aware of the correct technique of practicing yoga because of no proper guidance and information. Eventually it turns out that they harm themselves because of the misguided stance. Even breathing in a wrong way while performing yoga, erroneously stretching and extending one’s body or in other words performing wrong yoga postures, can prove to be quite harmful to one’s health.

Doctors have been saying that wrong postures can become a good reason for severe pain and long-lasting chronic problems. For instance, while performing “Parivrtta Anjaneyasana” or “Revolved Crescent” pose, one needs to be very careful about the twists



Figure 2.1 Visual representation of yoga.

involved. They are very much essential part of the group of asanas and one can make those twists quite difficult for him/her to perform than they should be. More amount of force is directed in one specific area of the spine while physically pressing onto a deeper twist which is evident in figure 2.2-a. Consequently, if it is done again and again, which is what yoga practice is about, it is highly probable that it will lead to either instant impairment of the spinal cord discs or eventual bruises because of wear-and-tear as can be seen in figure 2.2-b.



a. b.

Figure 2.2 a.) Shows a female practicing Parivrtta Anjaneyasana b.) shows how wrongly performing yoga can lead to negative effects like back pain

## 2.2 Techniques to address the issue

As a solution, the first thing that crosses our mind is to take the help of a yoga practitioner under whose guidance yoga can be practiced to eliminate risks. But it is not always possible for people to afford a yoga practitioner. Considering the fact that in today’s era the handiest thing is a smartphone easily available with people, they search for ways to figure out how to practice yoga irrespective of whether they are doing it right or not. We can leverage this trend to our benefit to come up with an application or software which can help users practice yoga and assist and correct them in real-time using technology. Some of the methods researchers have been using are based on Human Action Recognition or Human Pose Estimation. Let’s discuss a bit about human pose estimation.

### 2.2.1 Human Pose Estimation

It has been one of the engrossing domains of computer vision for researchers to work on. It finds many applications like activity recognition, virtual reality, human-computer interaction, video surveillance, medical assistance and many more. It is a challenging field of research because of numerous critical factors like detection of complex joint points in highly flexible human bodies, varied attire because of multiple body appearances, change in environment that may result in occlusion of same body parts, shortening of camera view and multiple views of angles.

Human pose estimation boils down to its essential element, human body model which has three different representations because of the flexible and complex kinematic human body features, which are: skeleton-based model, contour-based model and volume-based model as shown in figure 2.3. Let us describe each of them.

#### 2.2.1.1 Skeleton-based model

It is also otherwise known as kinematic model or stick-figure model mainly because of the collection of body joint points (generally 15-20) that connect together using lines or sticks to form a skeleton structure. This model is quite adaptable and simple; hence, it can be utilized widely in 2D as well as 3D human pose estimation.

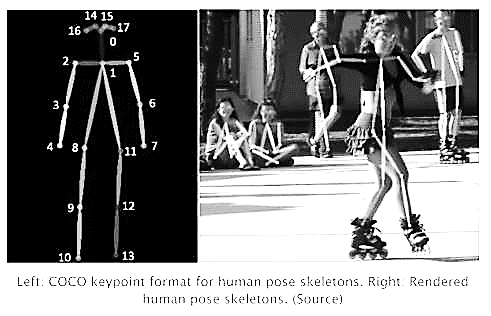


Figure 2.3 Represents a.) the skeleton-based body model b.) contour-based body model along with rectangles signifying body parts c.) volume-based body model (3D) [2]

#### 2.2.1.2 Contour-based model

This model is advantageous in human pose estimation when the body limbs including the torso are having rough width and some form of contour data. To represent the parts of the human body rectangles are used or else edges of the silhouette of the person are used [3] & Chi N.K, 2011). This representation is enriched with textual data having contour information of the target human body which is missing in skeleton-based human model.

#### 2.2.1.3 Volume-based model

It consists of 3 dimensional shapes and postures of the target human body represented using patterns, which are volume-based, using geometric design or meshes. Previously the geometric illustration of body parts was modelled using cylinders and conics. These days their modelling done is in mesh format, which are captured using the help of 3D scans.

# CHAPTER 3

**RELATED WORK**

There have been several implementations, both hardware and software, to estimate and monitor human body actions. Ray Gonsalves in [4] proposed a solution for the safety of construction workers which monitors their posture while they are working so as to find out whether they have any kind of risk or any issues related to physical health because of an incorrect posture. This solution is implemented using a 3D range image camera, since these types of cameras are capable to generate a 3D perception of the target construction worker (with the help of 3D point clouds) as visible in figure 3.1. The human objects are further segmented from the recorded videos in real time and are then converted into a star skeleton structure with the aid of image skeletonization.

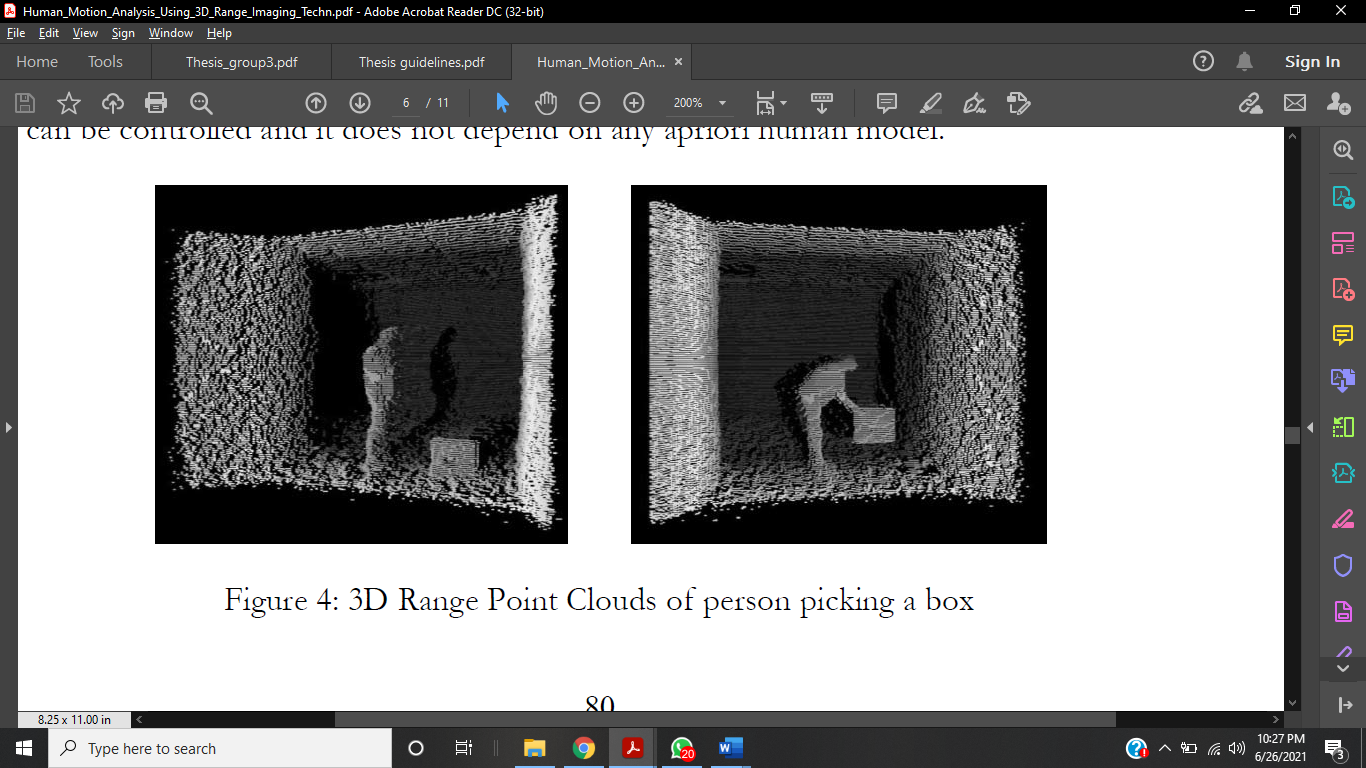
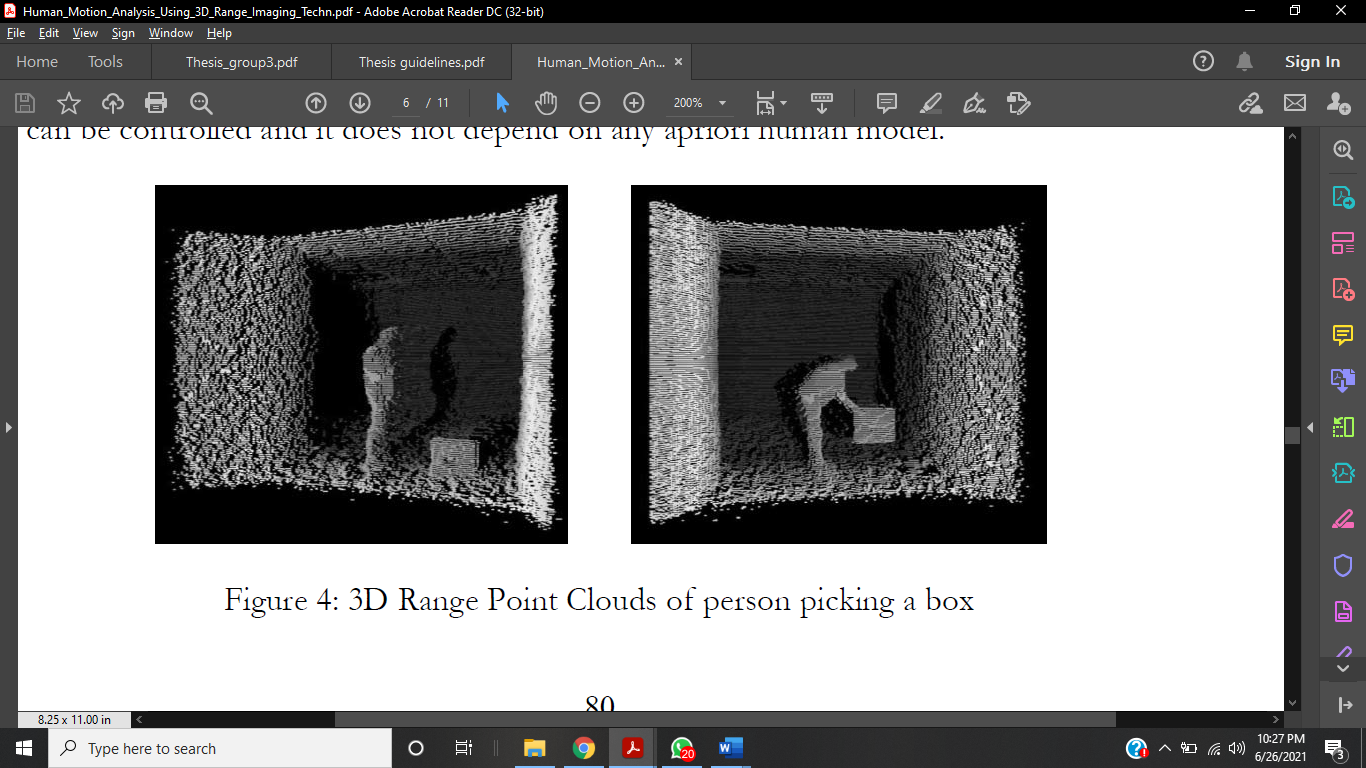


Figure 3.1 Range point clouds in 3 dimensions of person picking a box

They have not explicitly aimed at yoga pose estimation but the method which they have proposed can be used for the same. One major constraint here is that there is a limit to the maximum distance at which the 3D range image camera works which limits the scope of its usage to an indoor environment.

Q. Cai and J. K. Aggarwal, in 1996, proposed a framework for indoor environment which tracks human motion from numerous fixed cameras giving sequences of grayscale images which are monocular [5]. Dinesh et al., in 2015, presented a decent approach for the purpose of human activity recognition having an acceptable level of performance based on the conjecture that there is some or the other type of rotation and translation information with every human action or activity [6]. The gait energy images are very well enriched with structural, translation and rotational information integrated altogether. The transform used gives gradient’s edge distribution, directional pixels and information about the orientation which in turn gives the shape information of the activities being monitored. In the method proposed, the local features and global features of silhouette are merged which is an advantage for human activity recognition since it gives representation of discriminative features. For the purpose of classification of the above information put together, a SVM classifier is used for more than 2 classes. The model was experimented on two publicly available datasets which are Weizmann and KTH and they showed good performance and accuracy results. This approach for implementing human activity recognition, can be directly used for yoga pose estimation and rectification.

Islam et.al, 2017 in [7] have used Microsoft Kinect gadget for the purpose of yoga posture identification and correction. A Kinect device has infrared laser projector, built-in, which combined with a monochrome CMOS sensor, captures video information. In addition to the capability of Kinect to facilitate color images and depth images, it also has a tool to track skeletal information that can recognize 32 joint points of a human body in 3-dimensions as evident in figure 3.2. It also gives status information which signifies whether the joint is tracked or not.

By tracking of joint points of human skeleton by Kinect sensor it is possible to extract the coordinates of the joint points. Using the coordinates, it is possible to design a skeleton structure based on which a particular yoga pose is detected. For a particular pose that we need to detect, there are a set of joint points that will not have any importance value for precise detection. Only necessary and sufficient set of points are taken into consideration for yoga posture identification. All the three x, y and z coordinates of the joint points help estimate the structure of every single yoga pose. Dataset comprised of the ideal reference structure for three different yoga asanas (namely Goddess Squat,

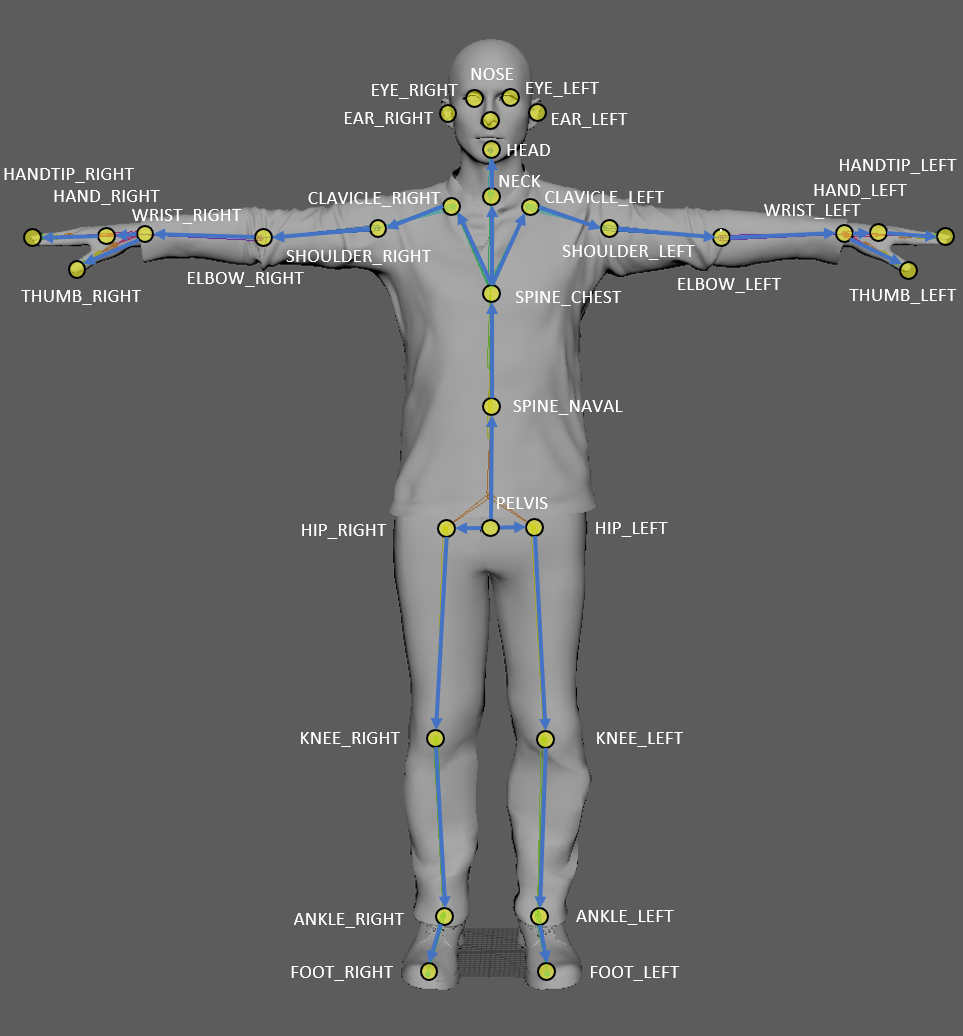


Figure 3.2 Represents joint point locations and connections with respect to the human body fetched from Kinect camera device [8]

Reverse Warrior and Warrior II) were made by collecting information of the joint points from human pose of 5 gymnastics under guidance of certified physicist (raw data). Then the raw data of these 5 models were manipulated to come up with a standard reference model (so called ideal) for comparison by taking the average angular distances of the joint points. Yoga poses were identified with 97% accuracy of identifying angle between different user body parts. Even though this is a good technique that serves the purpose, the gadget itself is costly which makes it unaffordable for majority of the users which is downside.

Edwin W. Trejo and Peijiang Yuan (in 2018) [9] proposed a way to identify 6 different yoga poses with the help of Kinect device. This was achieved using one of the detection technologies of Visual Gesture Builder (VGB) (in Microsoft Kinect v2.0 SDK tool), AdaBoost algorithm (short for Adaptive Boosting) which is an ensemble learning method. In this a certain number of weak classifiers bring about a strong classifier. Initially or we can say, in the phase of early learning, the algorithm assigns even or equal weights all over the dataset and then predicts over it. Eventually when each iteration terminates, the data which is misclassified are assigned high weights depending upon the feedback obtained when the prior classification rules are applied. And this is how learners get added till it becomes a strong classifier giving high accuracy to the model. The same is represented in figure 3.3.



Figure 3.3 Represents basic functionality of AdaBoost algorithm

Additionally, an interactive system having voice recognition, for the purpose of executing system’s predefined voice instructions, is integrated into it. For training using AdaBoost algorithm, 10 clips of an expert yoga practitioner were recorded in different environment at 30 frames per second and accordingly strong classifiers were build. Accuracy lied in the range of 80.6 % to 98.92% based on the pose and classifiers build. Each of the 6 yoga asanas had varied difficulty degree which served the purpose of analysis of how robust the system is or can be but all it demands a high computation time for a complex model.

In 2018, Pullen, P., & Seffens, W. used similar core technique as described above for yoga pose detection by utilizing Kinect device [10]. They developed an exergame having machine learning algorithm which helped the people to get better and better at practicing yoga or we can say acquire the skillset of practicing yoga. During the training process of AdaBoost, only the features which improves the capability of prediction of the model are selected. One should make sure that those features also should reduce dimensionality. The execution time is hence improved because the irrelevant features are not being computed. Precision values while testing ranged from 0.6 (some values) to more than 0.9.

There are several methods proposed by researchers where they have irradicated the use of Kinect device and have used or implemented human yoga pose classification and estimation algorithms. J. Palanimeera, K. Ponmozhi, in 2020 [11], proposed four different machine learning algorithms - Naïve Bayes, Support Vector Machine, Logistic Regression and KNN, to classify set of 12 sun salutations asanas and pose estimation algorithm to get body joint points. Shrajal et al., in 2020 [12], proposed 3-dimensional deep learning technique based on CNN for recognition of yoga asanas. They created a dataset, with the help of 27 people (8 male and 19 females), of 10 different yoga asanas which are – Dandasana, Tadasana, Ananda Balasana, Kumbhakasana, Janu Sirsasana, Hasta Uttanasana, Anjaneyasana, Paschimottanasana, Malasana and Uttanasana. The architecture in use is a revised version of Convolution 3D (C3D) model, widely in use, because of its capability of recognizing features from video. The layers which require intensive computation are replaced with batch normalization layers and average pooling layers which are supplementary layers to improve efficiency of computation. This designed architecture is based on 3D CNN model with validation accuracy of around 91.15% on the dataset they have created. On the dataset which is publicly available, this model could achieve a good accuracy of 99.39%. The dataset which achieved the stated results though lacked the flexibility of detecting yoga asana practiced from multiple views of user facing the camera.

In another implementation in 2019 by Yadav et al. [13], an algorithm based on deep learning models is proposed to identify six different yoga asanas namely Tadasana, Bhujangasana, Trikonasana, Padmasana, Vrikshasana and Shavasana. A dataset with the help of 15 individuals was created using a HD 1080p RGB webcam*.* The core components of the deep learning algorithm proposed, consist of hybrid of long short-term memory (LSTM) and convolutional neural network (CNN) for yoga pose recognition in real-time. Features from the key points/body parts fetched from OpenPose model were extracted with the help of the CNN layer and they were fed to LSTM to get the temporal predictions. We will further elaborate on the details about OpenPose in the implementation section. Talking about the performance, the algorithm attained 99.04% single frame test accuracy and when it comes to videos, an accuracy of 99.38% was achieved after predictions were polled on 45 frames. The use of temporal data gave the benefit of using information from the prior frames which resulted in accurate and more robust results.

Chiddarwar et al. in [14] have proposed another method for yoga pose estimation using OpenPose, where they have used an android application available for the user. Lai et al. in [15], proposed a similar method using OpenPose where the trained their model on 10 different yoga asanas, along with a random asana for no-pose detection, achieving 78% accuracy.

# CHAPTER 4

**PROPOSED METHODOLOGY**

## 4.1 Tools Used

In this section, we give an overview of all the software requirements that were necessary for implementing our work, such as the programming language(s), the imported libraries etc.

### 4.1.1 Programming Platform: Python 3.6

Python is widely used for programming. Developed by a programmer named Guido van Rossum in 1991, it has been extensively developed and used for many large-scale projects. Also, it is an interpreted language. An interpreted language is a high-level language run and executed by an interpreter (a program which converts the high-level language to machine code and then executing) on the go; it processes the program a little at a time. It involves programming at high level, great for beginners, and a programmer can focus on what to do, and less on how to do that, due to its easy syntax and huge variety of import libraries.

**4.1.2 Libraries Used**

Multiple libraries and open-source packages, that are required to implement the framework, involves python’s open-source OpenCV package, TensorFlow with Keras as backend and other libraries useful for machine learning applications such as scikit and pandas.

**4.1.2.1 OpenCV**

Open-CV (Open-Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. In simple language it is library used for Image Processing. It is mainly used to do all the operation related to Images. It can do a lot of tasks including, read and write Images, Detection of faces and its features, Detection of shapes like Circle, rectangle etc. in an image. E.g., Detection of coin in images, Text recognition in images. e.g., Reading Number Plates, modifying image quality and colors example Instagram, Cam-Scanner, Developing Augmented reality apps and many more. Some of the pros of using OpenCV involves:

* One of the powerful, platform independent library and supports C++, Java, Python.
* Provides good support for basic shape detection e.g., Circle, rectangle etc.
* HaarCascade: Cascade classification is one of the powerful features of OpenCV, if you are looking for shape or object detection e.g., Face detection,
* Video support is good, compatible with almost all the webcams.
* Feature detection like detection exact shape of the face like a contour representing lips or eyelash there are many algorithms available like Active shape model, active appearance model
* Widely used in Augmented reality application.
* Support camera calibration.

### 4.1.2.2 TensorFlow

With the growth of computer science, we saw a boom in the amount of data. As this happened deep learning began to beat all other machine learning. To use this opportunity, Google, to improve its services, thought to use these neural networks:

* Gmail
* Photo
* Google search engine

They came up with TensorFlow so that researchers and developers can work together on AI. In its developed and scaled mode, it gave a chance to a lot of people to bring it to use. The first public version was released in 2015, and the first stable one was released in 2017. Apache Open-Source license has made it open source. We can use, modify and redistribute without any fee. TensorFlow architecture has three parts for working:

* Pre-processing the data
* Build the model
* Train and estimate the model

The word TensorFlow came up as it uses multi-dimensional arrays called Tensors as inputs. We construct a kind of flowchart called the graph of functions that we want to do on that input. Data is input from one side and we perform multiple operations on it and get an output from the other side. This is the reason it is called TensorFlow as input flows.

### 4.1.2.3 PyTorch

OpenPose is primarily based on PyTorch’s open-source optimised functions. It is a tensor library principally used for the purpose of Deep Learning based applications which use GPUs and CPUs. The research team of Facebook AI developed this open-source library for machine learning for Python. After Keras and TensorFlow it is one of the broadly used machine learning libraries. It mainly finds its use in computer vision and (NLP) natural language processing.

### 4.1.2.4 Pandas

Pandas is a basic tool for our data. Pandas familiarizes you with your data by cleaning, transforming, and analyzing it. Pandas help us to know our data as we can clean, transform, and analyses the data. Like, if we have to use a dataset in CSV format, Pandas extracts data from the CSV file to a Data-Frame (table). Then we can perform things like:

* Calculation of statistics:
* Average, median, maximum and minimum of columns.
* Correlation of columns
* Distribution of data
* Removing missing values and filtering rows and columns using some criteria.
* Visualization of data like plot bars, lines, histograms, bubbles, etc.
* Storing the clean and transformed data into a CSV file or other form of a database.

### 4.1.2.5 Keras

Although the deep neural networks are growing to become more and more popular, many frameworks are so complex that they have become a barrier to use it. Many high-level APIs have been proposed which are simple and better for developing neural networks, which look same but truly are very different on examining.

One of the highly popular high-level neural networks API is Keras. It supports many backend neural network engines. Keras provides a user-friendly experience which is modular, easily extendible and easy to work with Python. Standalone modules like optimisers, activation functions, cost functions, neural network layers, etc. can be permuted to obtain newly designed models. Modules can be easily added like classes and functions.

### 4.1.2.6 Scikit-Learn

Many supervised and unsupervised learning algorithms are provided by Scikit-learn with a good interface in python. It focuses on robustness and support requirements in production systems. It implies focus on ease of use, quality coding, performance, documentation, etc.

### 4.1.2.7 Mat-plot Lib

It gives an outstanding visualization in python for 2D plots. Matplotlib is built on NumPy arrays for multiplatform visualization and uses SciPy stack which is for broader use. It was presented by John Hunter in 2002. Visualization’s greatest advantage is that we can visually see large data in easy-to-understand graphs, etc. It has plots like line, bar, scatter, histogram, etc.

### 4.1.2.8 Pickle

Pickle is a module which helps to change or modulate the object structures in a way that is friendly to python and which makes it easy to work upon. All types of python objects can be pickled, with the help of pickle library and then written and stored on the disk.

## 4.2 Dataset Collection and Preparation

Yoga asanas have varied range of complexity hence it can be said that it is feature-rich and this makes it difficult for the algorithm to work correctly. The lesser are the variations in the dataset the lesser is the robustness of the algorithm. The dataset which the researchers have been using are mostly created in a controlled environment lacking varied background features and have not encompassed the parameters like variation in clothing, lighting and even color factor. Since, the model also takes into consideration texture and clothing as feature while training, it is possible that if all the images have limited different colored clothing, then it may not give accurate results with the change in color and texture.

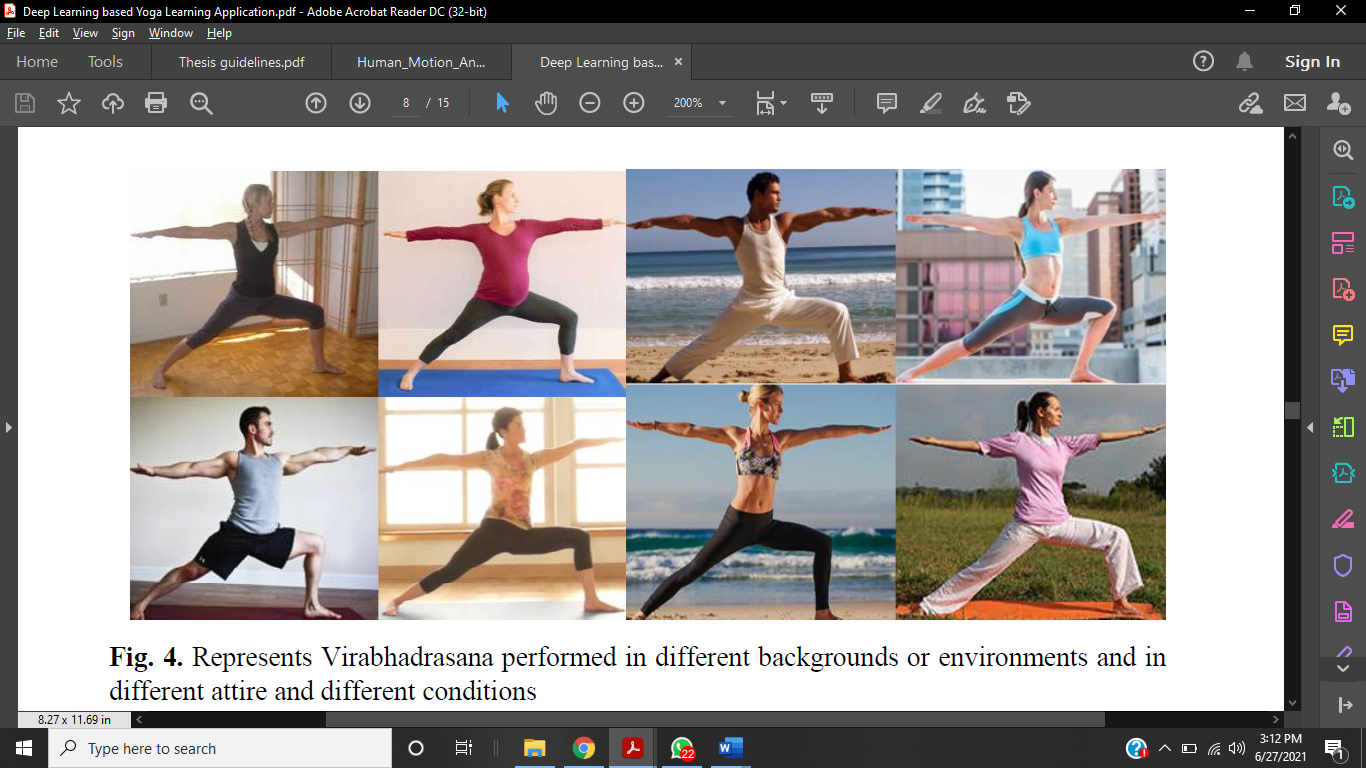


Figure 4.1 Represents Virabhadrasana performed in different backgrounds or environments and in different attire and different conditions.

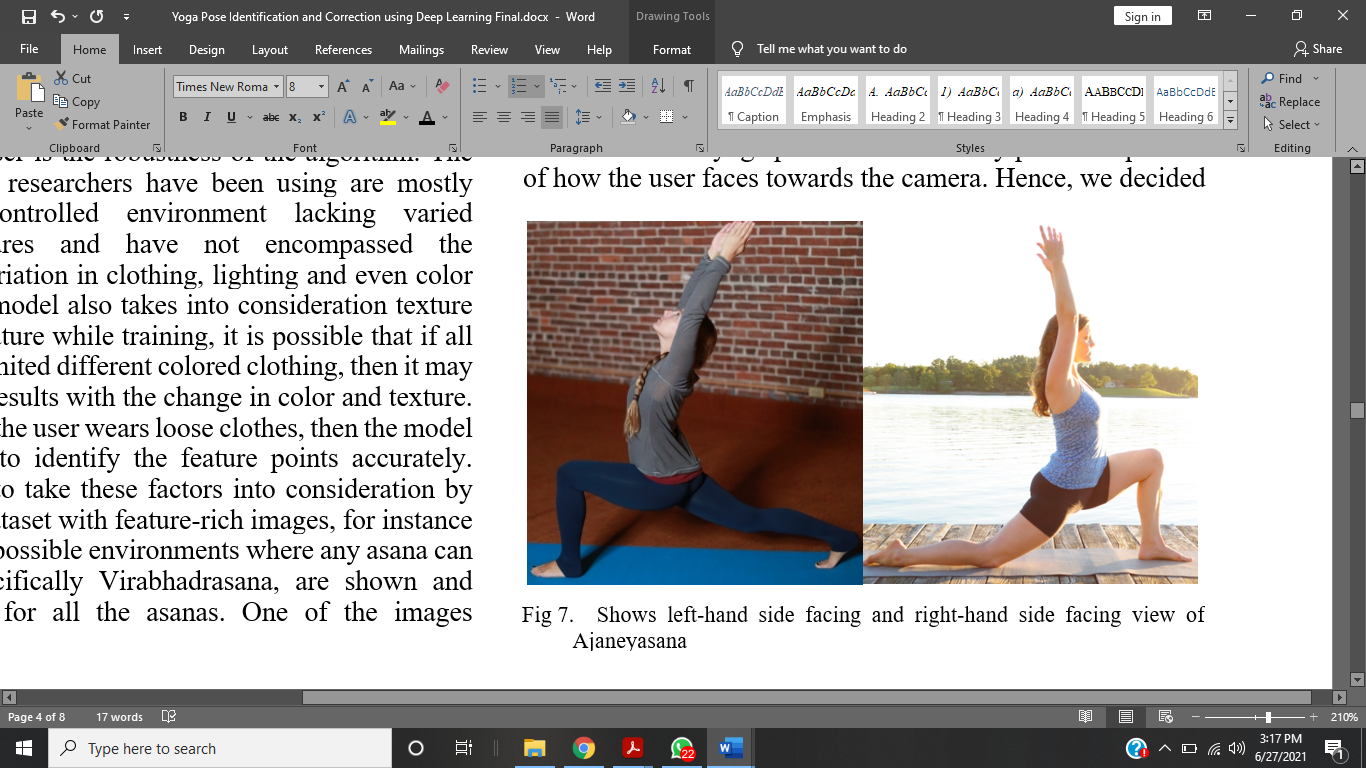


Figure 4.2 Shows left-hand side facing and right-hand side facing view of Anjaneyasana

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Yoga Asana | Front View | Left Hand Side View | Right Hand Side View |
| 1 | Agnistambhasana | Yes | No | No |
| 2 | Anjaneyasana | No | Yes | Yes |
| 3 | Ardha Uttanasana | No | Yes | Yes |
| 4 | Baddha Konasana | Yes | No | No |
| 5 | Bhujangasana | No | Yes | Yes |
| 6 | Garudasana | No | Yes | Yes |
| 7 | Hanumanasana | No | Yes | Yes |
| 8 | Malasana | Yes | No | No |
| 9 | Padmasana | Yes | No | No |
| 10 | Parighasana | Left Tilt (Y) | No | No |
| Right Tilt (Y) |
| 11 | Paripurna Navasana | No | Yes | Yes |
| 12 | Parivrtta Trikonasana | No | Yes | Yes |
| 13 | Tadasana | Yes | No | No |
| 14 | Utkatasana | No | Yes | Yes |
| 15 | Utthita Hasta Padangustasana | Left Leg (Y) | No | No |
| Right Leg (Y) |
| 16 | Utthita Parsvakonasana | Left Tilt (Y) | No | No |
| Right Tilt (Y) |
| 17 | Utthita Trikonasana | Left Tilt (Y) | No | No |
| Right Tilt (Y) |
| 18 | Virabhadrasana | Left Leg (Y) | No | No |
| Right Leg (Y) |

Table 4.1 Shows list of all the asanas take in the dataset along with their views.

Apart from this, if the user wears loose clothes, then the model may not be able to identify the feature points accurately. Hence, we chose to take these factors into consideration by diversifying our dataset with feature-rich images, for instance in figure 4.1 common possible environments where any asana can be practiced, specifically Virabhadrasana, are shown and added in dataset for all the asanas. One of the images represents a pregnant woman performing yoga which is also a different feature that can impact the performance of the model.

Another concern out here is that the user when practices yoga in front of the camera, he/she shouldn’t be bound to perform yoga asanas in a restricted view since this is not user-friendly practice. Primary focus of the user should be on practicing yoga correctly and not how the user should face the camera while practicing. To take care of this, the model should be able to detect yoga posture and the body parts irrespective of how the user faces towards the camera. Hence, we decided side facing view and right-hand side facing view of yoga pose Anjaneyasana is shown in figure 4.2.

The dataset consists of around 2400 images 18 yoga postures with three different facing views or perspectives i.e., front view, left-hand side view and right-hand side view, to train the model. The structure of the dataset is shown in Table 4.1 having the names of 18 asanas taken into consideration. In significant number of the images in dataset, noise component is also added to bring further feature variation so that the algorithm can work in environments with low lighting and high noise.

## 4.2 Architecture Used

Moving on to how we have used the dataset, our approach can be divided into two steps: first it aims at finding out, in real-time, which yoga asana the user is practicing and in the next step the algorithm estimates the correctness of the user’s posture with respect to the ideal way in which that particular yoga asana should be performed. The flow diagram of the architecture explained below is show in figure 4.3.

First let’s discuss about the classification algorithm. To train the classification algorithm for identification of yoga asana, we have used a dataset of 18 asanas but because of the multiple facing views of each asana, the total number of classes as sum up to be 31 as evident in Table I. With this dataset we have modified and trained four different deep learning models, namely Xception, ResNet-50, VGG16 and VGG-19. The best one to work here was VGG-16 giving a validation accuracy of 87.6%. We will further discuss about the performance of the models in the results section. The above models were modified in such a way that the top layer was replaced by two layers, namely Global Average Pooling layer and a Dense layer as output layer with SoftMax activation function for the purpose of prediction of multinomial probability distribution.

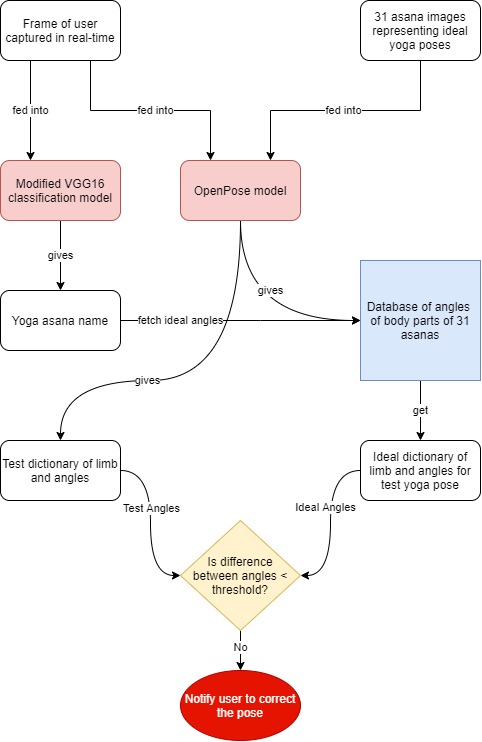


Figure 4.3 Shows the workflow diagram of the proposed method

The architecture of the modified VGG-16 model chosen for identification of yoga pose is shown in figure 4.4. As part of the modification, global average pooling layer is used to replace the fully and densely connected layers. Since the pool size here is equal to the size of input layer and the number of input feature maps to this layer are same as the total number of features, the result of this layer is a set of average values of feature maps, having the same size as the total number of features. Since we have used this layer at the end of the model, right before dense layer, it becomes equal to the number of classes. When the output values of this layer are fed another dense layer having SoftMax function we achieve a multiclass probability distribution which is what we want. This approach primarily helped in reducing overfitting and since here there is absence of any parameter yet to be learned and added, a significant improvement in the performance was observed because of the connection of a classifier to feature extractor. The results because of this made us stick to adding to this implementation.

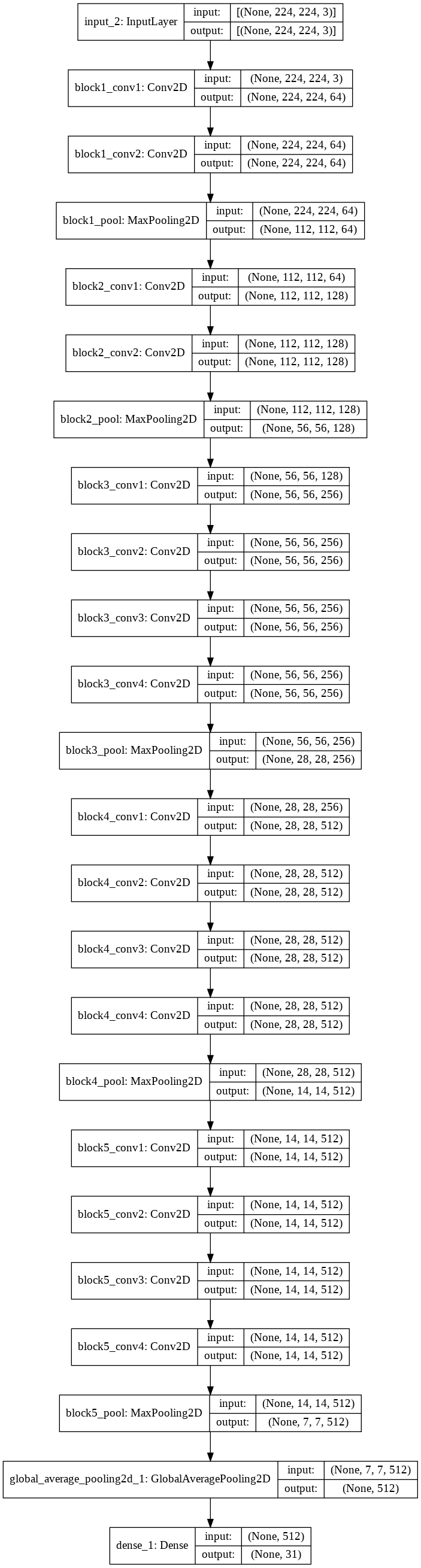


Figure 4.4 Modified VGG16 architecture

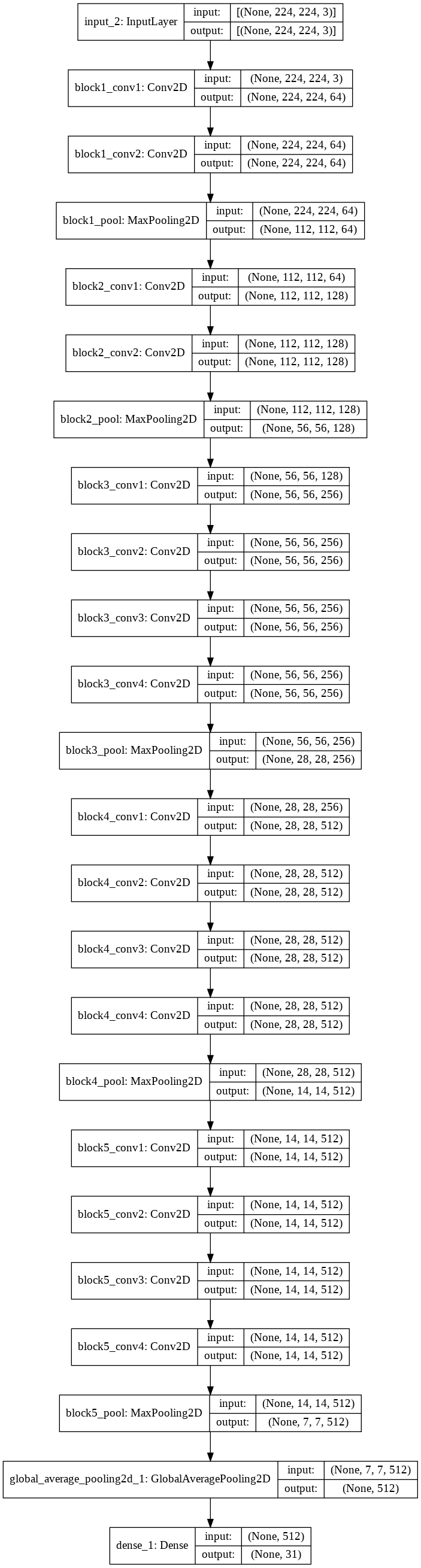


Figure 4.4 (continued)

For the purpose of training and validation, we have divided the dataset in two parts, 75% of the images were used for training and 25% of the images for validating the model, that is approximately 1768 images for training and 576 images for validation belonging to 31 classes. Talking about the necessary hyper-parameters for the purpose of training, after a lot of tweaking, the model was trained for around 100 epochs with the learning rate of 0.0001 and loss function as categorical cross-entropy, since there are 31 classes. For standardization, the input images were resized to 224 x 224. The loss function is formulated as below:

Talking about the initial steps in the architecture in figure 4.3, in real-time the frames having poses of the user are captured and fed to a classifier for identification where we get the pose name. In the next step, our aim is to find out how precise it is and correct the user for any wrong pose and for this we have used OpenPose to fetch the body point coordinates. It uses CNN architecture as its core model to identify human body joints in

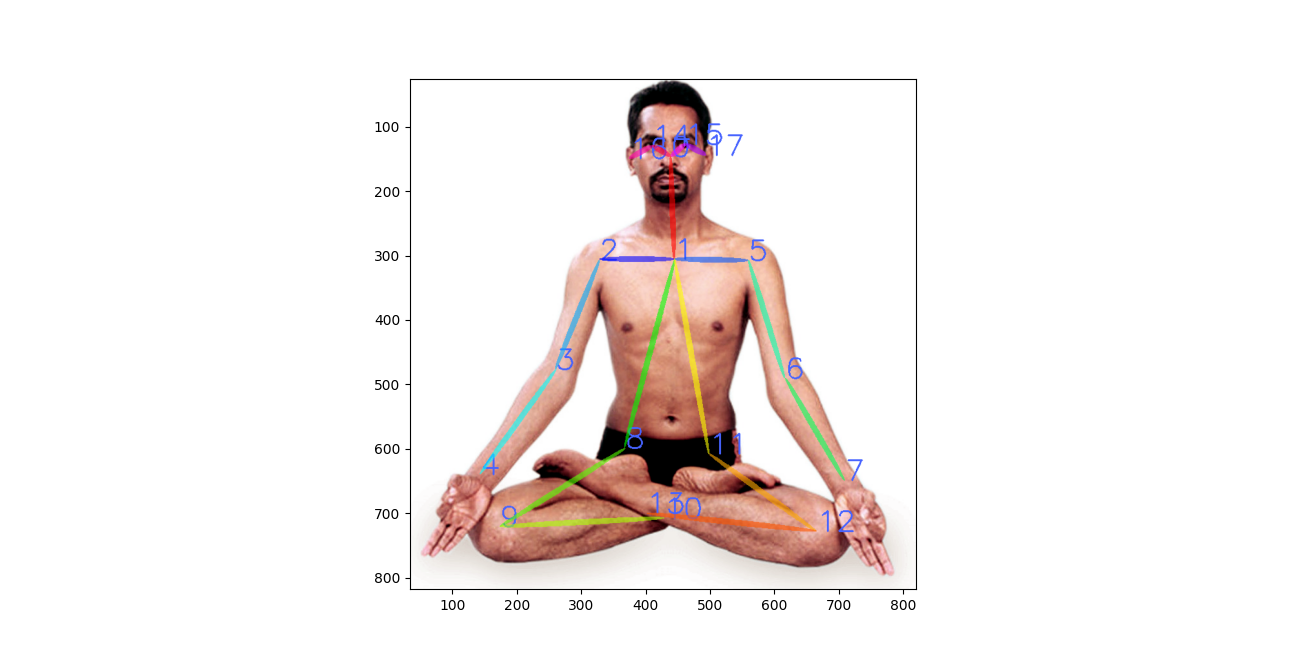


Figure 4.5 Represents 18 keypoints skeletal structure of a person practicing Padmasana

case of presence of single or multiple people [16]. It can identify 18 distinct body parts in the input frames both from a camera in real-time or from recorded videos as shown in figure 4.5. This is what attracts researchers to use it in various applications like activity detection, surveillance, sports and in our case, yoga pose estimation.

After the detection of key points, detected body parts are allocated to each individual in the frame in case of presence of multiple people. The OpenPose model has VGG-19 architecture in the beginning layers which helps in feature extraction from the input image. These features are then fed to two parallel running convolution layer branches. The first branch predicts 18 different confidence maps representing particular human body parts. The second branch predicts 38 different Part Affinity Fields (PAFs) which signify the degree of relation between body parts [17]. For the purpose of refining the predictions, more stages are added. The confidence maps are then used to generate bipartite graphs between several body parts and this is where the weaker connections are removed with the help of PAF values. This is how human skeletons structures are identified.

The OpenPose model gives us the body point coordinates using which we found the inclination of the identified body parts with an imaginary horizontal line. Angles were used as the important parameter to identify how deflected the body part is from ideal position it should be in for that yoga asana. To calculate the angle of each limb or body parts, atan2( ) math function was used which takes different between x0 and x1 and difference between y0 and y1 which are the coordinates of two joint points forming a limb. This function gives the angle of the limb between –π and +π which is the angle of the limb with respect to a horizontal axis. A database of yoga posture name and the ideal angle of each of 18 body parts was build using OpenPose on correct yoga poses. The angles from this database and the angles fetched from the users pose in real-time are compared and checked if they are satisfying the set threshold criteria and accordingly the user is indicated as seen in the architecture in Fig. 8. The sample outputs for both correct and wrong case of user practicing yoga are added in the results section.

# CHAPTER 5

**RESULTS AND ANALYSIS**

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Model name | Training Accuracy | Validation Accuracy |
| 1 | VGG-16 | 86.4% | 87.6% |
| 2 | VGG-19 | 84.1% | 80.5% |
| 3 | Xception | 93.7% | 76.2% |
| 4 | ResNet-50 | 88.2% | 65.4% |
| Table 5.1 Represents a comparison of performance of deep learning models | | | |

In this section we have presented a comparative description of the performance of the classification algorithms and the view of the interface the user sees for the feedback on the correctness of the posture. Each classifier has its own specialty according to which it gives results. The accuracies after training of the mentioned deep learning models are shown in Table 5.1. It can be seen that the validation accuracy of VGG-16 is the highest.

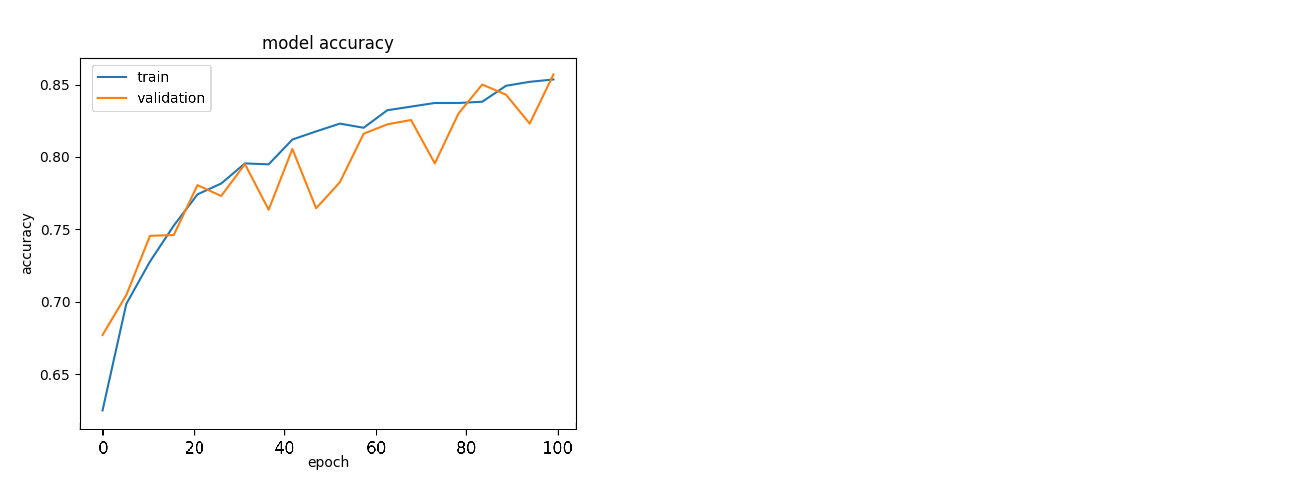


Figure 5.1 Represents the training and validation accuracy plot with the rising epochs of VGG-16 model.

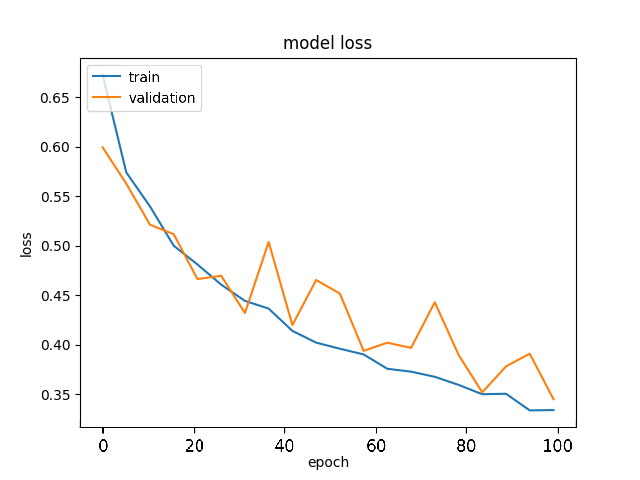


Figure 5.2 Represents the training and validation loss plot with the rising epochs of VGG-16 model.

We can visualize the rising training and validation accuracy of VGG-16 in the graph in Fig. 11 and decreasing training and validation loss in Fig.12 indicating a training loss of 0.321 and validation loss of 0.342.

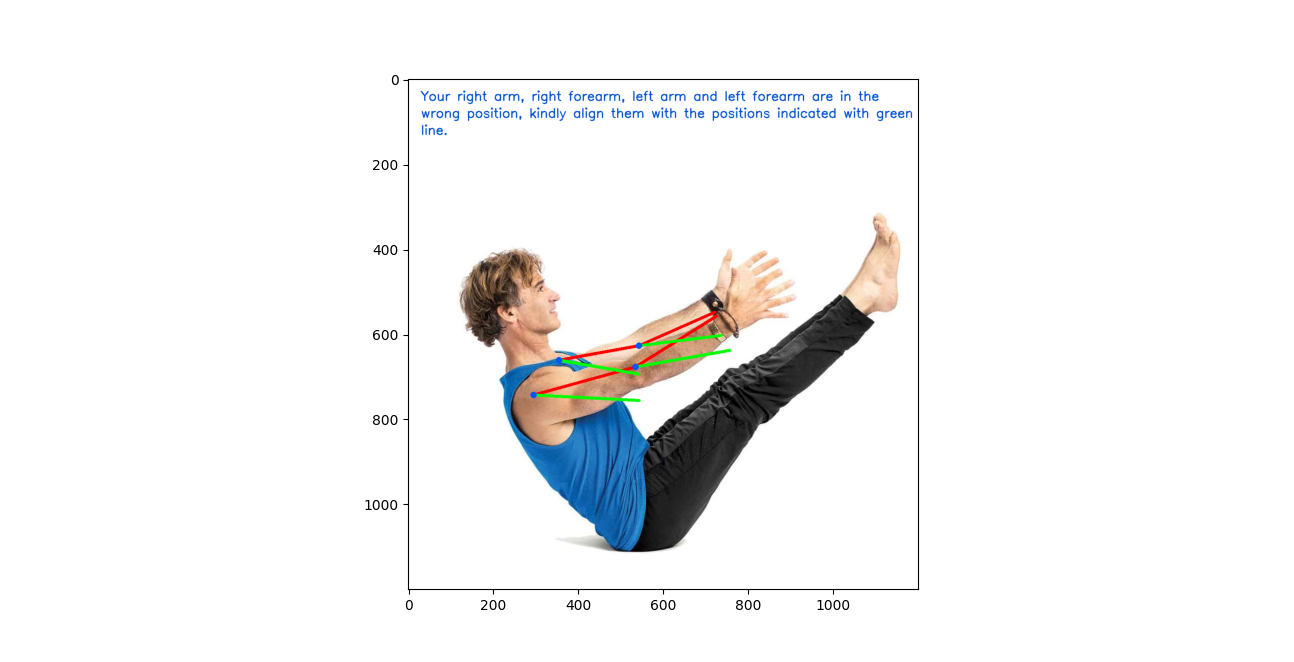


Figure 5.3 Represents the interface the user sees when he practices Paripurna Navasana incorrectly



Figure 5.4 Represents the ideal posture of Paripurna Navasana used as reference to compare with the user’s same body pose

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Limb | Actual User Angle (deg.) | Reference User Angle (deg.) | Difference Angle (deg.) |
| 1 | Right Shoulder | -53 | -39 | 14 |
| 2 | Left Shoulder | 135 | 147 | 12 |
| 3 | Right Arm | 164 | -177 | **19** |
| 4 | Right Forearm | 147 | 170 | **23** |
| 5 | Left Arm | 169 | -170 | **21** |
| 6 | Left Forearm | 156 | 173 | **17** |
| 7 | Right Abdomen | -127 | -119 | 8 |
| 8 | Right Thigh | 130 | 134 | 4 |
| 9 | Right Calf | 143 | 138 | 5 |
| 10 | Left Abdomen | -136 | -123 | 13 |
| 11 | Left Thigh | 130 | 135 | 5 |
| 12 | Left Calf | 144 | 139 | 5 |
| 13 | Neck | 95 | 104 | 9 |
| Table 5.2 Shows the list of reference and actual user angles and deviation between each other of a single frame. | | | | |

|  |  |  |
| --- | --- | --- |
| No. | Limb | Limb Connection |
| 1 | Right Shoulder | 1 to 2 |
| 2 | Left Shoulder | 1 to 5 |
| 3 | Right Arm | 2 to 3 |
| 4 | Right Forearm | 3 to 4 |
| 5 | Left Arm | 5 to 6 |
| 6 | Left Forearm | 6 to 7 |
| 7 | Right Abdomen | 1 to 8 |
| 8 | Right Thigh | 8 to 9 |
| 9 | Right Calf | 9 to 10 |
| 10 | Left Abdomen | 1 to 11 |
| 11 | Left Thigh | 11 to 12 |
| 12 | Left Calf | 12 to 13 |
| 13 | Neck | 1 to 0 |

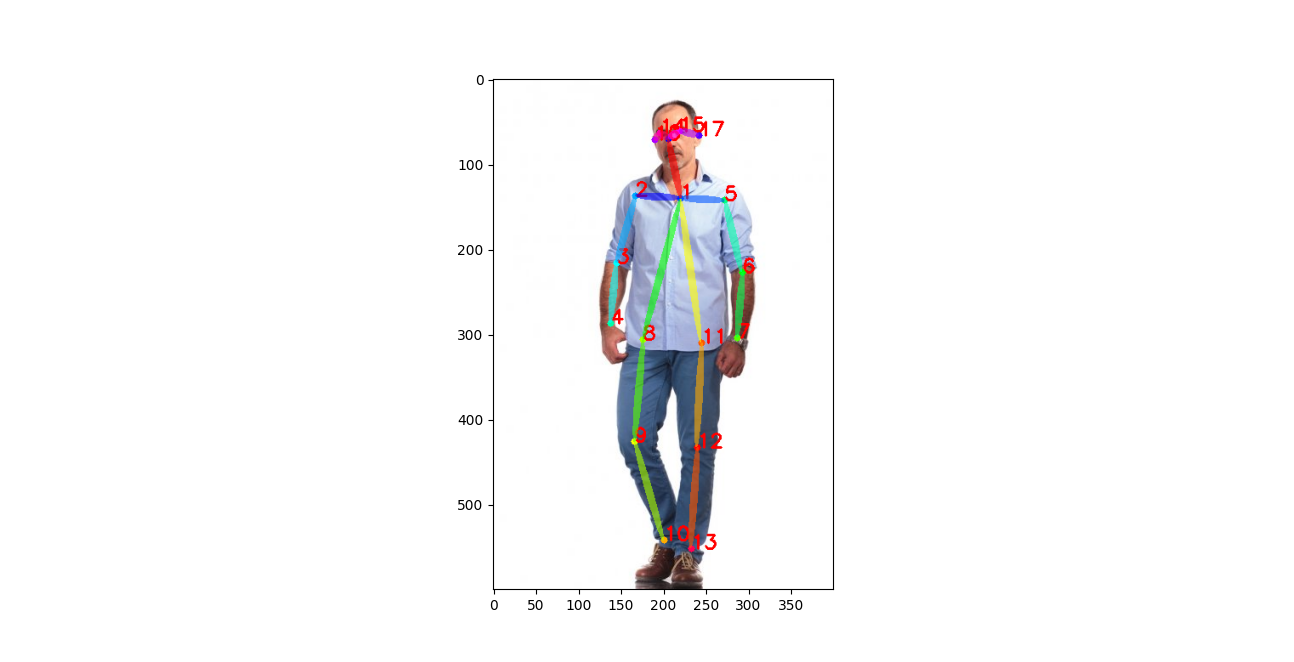


Figure 5.5 Represents the ideal posture of Paripurna Navasana used as reference to compare with the user’s same body pose

For the user interaction and feedback, the frames captured in real-time from the camera are shown back to the user on a display/screen (maybe a TV) with those body parts which are in wrong position, highlighted with red color line, as an indication of the limb. In green color another line, as a representation of the limb, is shown to the user for the user to move, which represents the right position of the body limb as can be seen in figure 5.3. As discussed, the inclination of each body part is found and compared with the reference angles. In the about yoga asana, the reference angles were found with the help of the user in the image shown in figure 5.4. With the help of OpenPose model we have fetched the reference angles as well as actual angle as listed in the Table 5.2. It also shows the difference between the angles which are then compared with a threshold value of 15 degrees. If the difference is greater than 15 degrees then that body part/limb is marked as the one in incorrect position. Four such body parts were marked: right arm, right forearm, left arm and left forearm. The mapping of the limb connection with the body part is evident in figure 5.5. Out of the total 17 limbs, only the above mentioned 13 limbs were used for yoga pose correction since the rest of them didn’t add any significant contribution for correction.

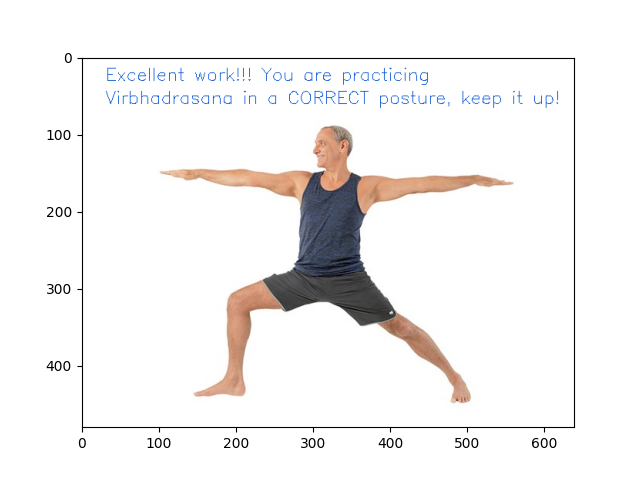


Figure 5.6 Represents the interface the user sees when he practices Virabhadrasana correctly.



Figure 5.7 Represents the reference image of the user used for Virabhadrasana

Talking about the case when the user is practicing yoga with correct posture, an indication is shown right next to the user on a display complimenting the user. This interface is illustrated in the figure 14 below where the user is practicing yoga asana Paripurna Navasana. The user gets the visuals and notification which specifies the names of the body parts in the wrong posture. This is hence in short, an application for the user to use. When the application captures the user’s video from the camera, every single frame is processed and then the user is notified in every single frame about which body part the user needs to correct. Based on the feedback the user corrects the pose and that is reflected on the display. When the user is finally in correct position, he gets a notification as shown in figure 5.6. The ideal image taken for yoga pose Virabhadrasana is shown in figure 5.7.

# CHAPTER 6

**CONCLUSION AND FUTURE SCOPE**

It is quite evident that the field of deep learning has a capability of creating a very powerful impact on the society. The algorithms used for yoga pose detection are accurate and robust enough to work in complex environments. Our efficient implementation utilizes Convolutional Neural Network (CNN) based VGG-16 for classification and PoseNet for pose rectification. Since our system has low processing power, we couldn’t run this algorithm using camera at more than 4-5 frames per second. We can increase the processing power by hosting the database of pose angles onto a high-speed server and communicate with it at a faster rate using recent technologies like 5G. We can find how powerful and helpful it can be for us in research proposed by Kapil et al. [18].

The proposed application can very well be considered as an IOMT device since this can also be used in healthcare industries like for the purpose of rehabilitation for people recovering from accidents. We can find how impactful IOMT devices can be in people’s lives in [19]. Talking about the future scope of this application, it can be very well in use for people who aren’t able to get the benefits of a yoga practitioner specially in times like these when people need to be in isolation because of COVID-19 pandemic and even maintain social distancing. As part of further improvement, we can keep on enhancing the database containing the ideal angles of yoga posture which will widen the scope of usage of this application. We can further enhance the user friendliness of this application by incorporating more and more user facing views in the dataset. Because of a feature-rich dataset which we have used, the application turns out to be highly robust and user friendly.

# REFERENCES

|  |  |
| --- | --- |
| [1] | H. a. H. H. Zhou, “Human motion tracking for rehabilitation—A survey,” *Biomedical signal processing and control,* vol. 1, pp. 1-18, 2008. |
| [2] | Y. Y. T. a. M. H. Chen, “Monocular human pose estimation: A survey of deep learning-based methods.,” *Computer Vision and Image Understanding 192,* 2020. |
| [3] | S. X. M. J. B. a. Y. Y. Ju, “Cardboard people: A parameterized model of articulated image motion,” in *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, 1996. |
| [4] | R. a. J. T. Gonsalves, “Human motion analysis using 3D range imaging technology,” in *Int. Symp. on Automation and Robotics in Construction*, 2009. |
| [5] | Q. a. J. K. A. Cai, “Tracking human motion using multiple cameras,” in *Proceedings of 13th International Conference on Pattern Recognition*, 1996. |
| [6] | D. K. A. D. R. M. a. R. K. Vishwakarma, “Human motion analysis by fusion of silhouette orientation and shape features,” *Procedia computer science,* no. 57, pp. 438-447, 2015. |
| [7] | M. U. H. M. F. B. A. I. H. a. M. K. H. Islam, “Yoga posture recognition by detecting human joint points in real time using microsoft kinect,” *IEEE Region 10 humanitarian technology conference (R10-HTC),* pp. 668-673, 2017. |
| [8] | Y. W. P. M. Tetyana Sych, “Azure Kinect body tracking joints,” Microsoft, 26 06 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/kinect-dk/body-joints. |
| [9] | E. W. a. P. Y. Trejo, “Recognition of Yoga poses through an interactive system with Kinect device,” in *2nd International Conference on Robotics and Automation Sciences (ICRAS)*, 2018. |
| [10] | P. a. W. S. Pullen, “Machine learning gesture analysis of yoga for exergame development,” in *IET Cyber-Physical Systems: Theory & Applications 3*, 2018. |
| [11] | J. a. K. P. Palanimeera, “Classification of yoga pose using machine learning techniques,” *Materials Today: Proceedings,* no. 37, pp. 2930-2933, 2020. |
| [12] | S. A. R. S. S. R. S. a. S. S. Jain, “Three-dimensional CNN-inspired deep learning architecture for Yoga pose recognition in the real-world environment,” *Neural Computing and Applications,* pp. 1-15, 2020. |
| [13] | S. K. A. S. A. G. a. J. L. R. Yadav, “Real-time Yoga recognition using deep learning,” *Neural Computing and Applications 31,* no. 12, pp. 9349-9361, 2019. |
| [14] | G. G. A. R. M. C. R. D. a. P. G. Chiddarwar, “AI-Based Yoga Pose Estimation for Android Application,” *International Journal of Innovative Science and Research Technology,* vol. 5, no. 9, 2020. |
| [15] | A. B. R. a. B. V. Lai, “Yog. ai: deep learning for yoga,” 2019. |
| [16] | Z. G. H. T. S. S.-E. W. a. Y. S. Cao, “OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields,” *IEEE transactions on pattern analysis and machine intelligence 43,* no. 1, pp. 172-186, 2019. |
| [17] | Z. T. S. S.-E. W. a. Y. S. Cao, “Realtime multi-person 2d pose estimation using part affinity fields,” *In Proceedings of the IEEE conference on computer vision and pattern recognition,* pp. 7291-7299, 2017. |
| [18] | S. K. a. K. S. Agrawal, “5G Millimeter Wave (mmWave) communication system with software defined radio (SDR),” *Proceedings of the International Conference on Recent Trends in Engineering & Science (ICRTES-16),* 2016. |
| [19] | A. M. B. a. A. C. Bajaj, “Recent trends in internet of medical things: a review,” *Advances in Machine Learning and Computational Intelligence,* pp. 645-656, 2021. |