Smart Yoga – A study of different standing yogasanas with AI-based technique

Submitted by

Kalgi Oza(18MCEN10)

Department of Computer Engineering

Institute of Technology

Nirma University

Ahmedabad –382481

May-2020

Major Project

Report

Smart Yoga – A study of different standing yogasanas with AI-based technique

Submitted in partial fulfillment of
the requirements for the award of the degree of

Master of Technology
in
Computer Engineering(Networking Technology)

Submitted by

Kalgi Oza(18MCEN10)

Under the guidance of

Prof. Jitali Patel

Department of Computer Engineering
Ahmedabad –382481

May-2020

Certificate

This is to certify that the Major Project entitled **Smart Yoga – A study of different standing yogasanas with AI-based technique** submitted by Kalgi Oza (18MCEN10), towards the partial fulfillment of the requirements for the degree of Master of Technology in Computer science and Engineering of Nirma University, Ahmedabad is the record of work carried out by her under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this Major Project Part-II, to the best of my knowledge, haven't been submitted to any other university or institution for the award of any degree or diploma.

Prof. Jitali Patel,
Guide and Associate Professor,
CE Department,
Institute of Technology,
Nirma University, Ahmedabad

Dr. Gaurang Raval,
Professor,
Coordinator M.Tech CSE(NT),
Institute of Technology,
Nirma University, Ahmedabad

Dr. Madhuri Bhavsar,
Professor and head,
CE Department,
Institute of Technology,
Nirma University, Ahmedabad

Dr. R.N.Patel,
I/C Director,
Nirma University, Ahmedabad

Statement of Originality

I, **Kalgi Oza**, **18MCEN10**, give undertaking that the Major Project entitled **Smart Yoga – A study of different standing yogasanas with AI-based technique** submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in Computer Science & Engineering (Networking Technology) of Institute of Technology, Nirma University, Ahmedabad, contains no material that has been awarded for any degree or diploma in any university or school in any territory to the best of my knowledge. It is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. It contains no material that is previously published or written, except where reference has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

Signature of Student:
Date:
Place:

Endorsed by
Prof.Jitali Patel
(Signature of Guide)

Acknowledgments

It gives me immense pleasure in expressing thanks and profound gratitude to **Prof Jitali Patel**, Associate Professor, Computer Engineering Department, Institute of Technology, Nirma University, Ahmedabad for her valuable guidance and continual encouragement throughout this work. The appreciation and continual support she has imparted has been a great motivation to me in reaching a higher goal. Her guidance has triggered and nourished the intellectual maturity that I will benefit from, for a long time to come.

It gives me immense pleasure to thank **Dr. Madhuri Bhavsar**, Hon'ble Head of Computer Engineering Department, Institute of Technology, Nirma University, Ahmedabad for her kind support, and providing basic infrastructure and healthy research environment.

A special thank you is expressed wholeheartedly to **Dr. R.N. Patel**, Hon'ble I/C Director, Institute of Technology, Nirma University, Ahmedabad for the unmentionable motivation he has extended throughout the course of this work.

I would also thank the Institution, all faculty members of Computer Engineering Department, Nirma University, Ahmedabad for their special attention and suggestions towards the project work.

Kalgi Oza

18MCEN10

Abstract

Human activity recognition is well managed by human pose estimation. Hunan body parts can be detected by two ways. First is human pose estimation that detects the each key point of the human body. And second is human activity recognition that takes a series of inputs to train the model and test it with new input and give the accuracy. So according to human pose estimation history, human activity recognition estimates the pose first then judges the activity of the human body. Human pose estimation is useful in Human activity recognition. And Human activity recognition has been done with two methods. The first is with the pose key points and the second is with the 3dcnn model. Most of the work in human activity recognition assumes a figure centric scene where the actor is free to perform anything. The system is proficient to classify the activity with low error and high accuracy. It is a challenging task due to background problems, changes in scale, lightning, and frame resolution. Some actions are impulsive as well as under habit so might not be accurate as desired [1]. Human pose estimation is one of the most essential applications in computer vision. Human pose estimation helps in AI technology. Human pose estimation typically follows the assumption of human body parts. In this study, we have applied a system for performing smart yoga standing postures. Tadasana, Vrikshashana, Virabhadrasana, Utkatasana, Hasta padangusthasana are studied to recognize all the body movements. The system has recognized every bend of hands and legs, and suggested the correction to be made. The system is successfully running on Jetson tegra-x2 computing device.

Keywords— Human pose estimation, keypoints, human activity recognition

List of Figures

Contents

Chapter 1

Introduction

1.1 General

Now-a-days people are busy doing their job work, so they are not able to work out every day and not able to go to any yoga classes. So they require one yoga teacher that corrects their yoga at any time. So our main project is to create an AI application that identifies the moves of yoga and corrects it if it was wrong. For detecting the different posture of human there is so many pose estimation library that detects the human body, provide key points, and measure the accuracy. Also, it will give suggestions regarding correct yoga or not correct yoga. For that, first of all, we need to detect the human body and have the keypoints in our database. Then according to the key points which we are having and the key points which we had through pre-trained models, compare them, and give the result to the user.

By using this system, the user can do yoga at any time and have an accurate result through online yoga trainers. The field of human pose estimation is developing the AI technology. By using this technology the system reaches to a very good approach. And a new scenario of AI is identified in the market of computer science. This project aims to develop AI and computer vision technologies in a wide range and also get familiar with these technologies. The user gets more and more

comfortable with this technology after using human pose estimation. it is a daily workout without wasting the time of standing in traffic lines or without having a pressure of yoga teacher. This technology will provide a kind of online yoga teacher.

1.2 Human Activity Recognition

Human activity recognition is the way of predicting what a person is doing based on their body movements[2]. Those body movements are detected by human pose estimation. And activity is traced by an activity recognition system. By merging both, we can have the exact idea about human action. Through human bodyparts, human pose estimation predicts the position of keypoints. And through activity recognition, it will predict the action performed by a human with their keypoint movements[3]. So human pose estimation and human activity recognition has a joint relation in the era of computer vision. In this library, the system can predict all the actions performed by the human body. To predict those actions, the system will trace the keypoint detection with human pose estimation. And predict the action with its pre-trained model and give the output as the name of the action performed by the human body. It will also measure the accuracy and maintain the result. It will try to give the highest accuracy and lowest error while predicting the action. The 3d-CNN library is used to recognize the human action and gives an accurate result. In this paper, the system is also based on 3d cnn library and also it will predict the yoga position performed by the human body. Firstly, when a human body performs a yoga posture, human pose estimation will detect the key points. i.e bodyparts. For human pose estimation, we have used a Github library called ildoonet. Second, by implementing 3d-cnn library so that the system will identify the action performed by a human. The prediction is based on the detection that was done by ildoonet. Human activity recognition is a very challenging task as we have to focus on data in very analytical way.

The dataset collection is very large. The system will learn the pattern recognition method to train the model. The activity recognition is very useful in deep learning technology. There are many types of human activity recognition like single user activity, Multi-users activity, Sensor-based single, and group

activity. The approaches used are activity recognition through logic and reasoning, activity recognition through probabilistic reasoning, Data mining based approach to activity recognition, vision based approach, Wifi based approach, etc. Many ap- plication including video systems, human-computer interaction, robotics for recognizing human body behavior. In this system, we have divided human activity classification method into two categories[4]. The categories are based on a dataset which we have created. Also, the system will analyze the publically available dataset and examine the requirements for ideal human activity recognition. So, human activity recognition is very useful for recognizing the activity as the name suggest. And it is also based on human pose estimation.

1.3 Objective

The objectives of this study are:

- To identify the yogasana name correctly.
- To guide the person, who is doing wrong yogasana, and to correct using the system.

Chapter 2

Literature Study

Human pose estimation can be done on single person or on multi person as as shown below:

Figure 2.1: Single person detection and multiperson detection

Human pose estimation is used to identify the body movements of a person or gesture of a person. It also helped to find the location of key points. How many persons are there. Pose estimation is predicting the bodypart or joint position of a person.

Table 2.1: Literature references at a glance

Table showing Literature reference at a glance					
Sr.No	Publisher	Title	Year	Author	Approach referred
1	IEEE	Anticipating Human Activities Using Object Affordances for Reactive Robotic Response	2016	Hema S. Koppula and Ashutosh Saxena	By using ML algorithms like KNN, random forest, SVC
2	IEEE	Human Activity Recognition from Accelerometer Data Using Convolutional Neural Network	2017	Sang Min Yoon, Heeryon Cho	CNN approach using triaxial accelerometer data collection
3	IEEE	Automated daily activity recognition for video surveillance using neural network	2017	Aisha Hass, Muhamed Zaharadeen	Multi layerfeed forward is used to classify the activities
4	IEEE	A novel feature map for human activity recognition	2017	Guangyu He ; Xinze Luan ; Junyi Wang ; Xiaoting Wang	ML algorithm is used like SVM
5	IEEE	Abnormal human activity recognition using Bayes classifier and Convolutional neural network	2018	Congcong Liu ; Jie Ying ; Feilong Han ; Ming Raun	The technique used is Bayes classifier and convolutional neural network

Chapter 3

Implementation

3.1 Phase1:Various github libraries and data analysis

Explored various github librares such as:

1. Posenet
2. Openpose
3. Alphapose
4. Ildoonet

3.1.1 Posenet

- Posenet[5] can be used to estimate either a single pose or multiple poses, meaning there is a version of the algorithm that can detect only one person in any image or video and one version that can detect multiple persons in an image or video.
- Firstly we have to load pre-trained posenet model that is MobileNetV1 architecture. And another model is ResNet. Also we can select one of the options between two that is Single person and multi person.
- Configuration parameters: – Architecture: Either Mobilenet or Resnet

- **OutputStride**: Can be one of 8,16,32 (16 and 32 are supported in ResNet and 8,16 and 32 are supported in MobileNetV1). The smaller the value, larger the resolution and more accurate the model is.
- **inputResolution**: That specifies the size of image before padded to the model. Generally set low value of resolution to get high speed detection and accuracy.
- Multiplier: It can be 1.01, 1.0, 0.75 or 0.5 (depends upon architecture). A multiplier is used for convolutional layers. Generally set to the smaller value to achieve the highest accuracy.By default posenet loadsMobleNetV1 architecture with 0.75 multiplier and 8 outputstride.
- This is the posenet detection demo on single person:

Figure 3.1: Single person body detection

- **Keypoints detected by posenet**:

1. nose
2. Left eye
3. Right eye
4. Left ear
5. Left Shoulder
6. Right Shoulder
7. Left Elbow
8. Right Elbow
9. Left Wrist
10. Right Wrist
11. Left Hip
12. Right Hip
13. Left Knee
14. Right Knee
15. Left Ankle
16. Right Ankle

- Same way, Multiple Pose estimation can decode multiple poses.
- It is more complex and slightly slower than the single person algorithm, but has the advantage that if multiple people appear in an image, their detected keypoints are less likely to be associated with the wrong pose.
- Even if the usecase is to detect a single person's pose, this algorithm may be more desirable in that the accidental effect of two poses being joined together

won't occur when multiple people appear in the image.

- It uses the Fast greedy decoding algorithm.
- Both MobileNetV1 and ResNet architecture support multiperson pose estimation.
- estimateMultiplePoses() is a method for running the posenet.
- Implementation of Posenet on the Linux System:

1. Cd into the posenet folder: cd posenet
2. Install dependencies : yarn
3. Publish posenet locally: yarn build and yarn yalc publish
4. Cd into the demos and install dependencies: cd demos and yarn
5. Link the local posenet to the demos: yarn yalc link @tensorow-models/posenet yarnwatch

3.1.2 Openpose

- Openpose[6] is also a github library for human pose estimation and it detects 135 keypoints in single image.
- Openpose would not be possible without CMU Panoptic studio dataset.
- Functionality:-
  - 2D real-time multi-person keypoint detection:
    - 2D real-time multi-person keypoint detection.
    - 6-keypoint foot keypoint estimation. Integrated together with the 25-keypoint body/foot keypoint detector.
    - 2x21-keypoint hand keypoint estimation. Currently, running time depends on number of detected people.
    - 70-keypoint face keypoint estimation. Currently, running time depends on the number of detected people.

- 3D real-time single-person keypoint detection:
    - 3-D triangulation from multiple single views.
    - Synchronization of Flir cameras handled.
    - Compatible with Flir/Point Grey cameras, but provided C++ demos to add your custom input.

- It works on 30 fps, Gpu is required but still speed is slow. Also it has more dependencies while installing it. Accuracy speed is slow.
- Input: Image, video, webcam, Flir/Point Grey and IP camera. Included C++ demos to add your custom input.
- Output: Basic image + keypoint display/saving (PNG, JPG, AVI, ...), keypoint saving (JSON, XML, YML, ...), and/or keypoints as array class.
- OS: Ubuntu (14,16) , windows (8,10) , MacOS, Nvidia Tx2.

3.1.3 Alphapose

- AlphaPose[7] is an accurate multi-person pose estimator, which is the first real-time open- source system that achieves 70+ mAP (72.3 mAP) on COCO dataset and 80+ mAP (82.1 mAP) on MPII dataset.
- It has 23 fps and more accurate compare to openpose.
- It also requires GPU.
- It detects 17 keypoints and speed is high.
- Implementation of Alphapose on the linux system:

1. Get the code and build related modules:
   - Clone the repository
   - cd AlphaPose/human-detection/lib/
   - make clean
   - make
   - cd newnms/
   - make
   - cd ../../../
2. **Install Torch and TensorFlow(verson ¿= 1.2). After that, install related dependencies by**:

```
chmod +x install.sh
./install.sh
```

3. **Run fetch_models.sh to download our pre-trained models. Or download the models manually: output.zip(Google drive—Baidu pan), ftnal_model.t7(Google drive—Baidu pan)**

```
chmod +x fetch_models.sh
./fetch_models.sh
```

4. **Demo**:

```
./run.sh –indir examples/demo/ –outdir examples/results/ –
```

5. Result for COCO (17 body parts):

$$\{0, "Nose"\}$$

$$\{1, "LEye"\}$$

$$\{2, "REye"\}$$

$$\{3, "LEar"\}$$

$$\{4, "REar"\}$$

{5,"*LShoulder*"}

{6,"*RShoulder*"}

{7,"*LElbow*"}

{8,"*RElbow*"}

{9,"*LWrist*"}

{10,"*RWrist*"}

{11,"*LHip*"}

{12,"*RHip*"}

{13,"*LKnee*"}

{14,"*Rknee*"}

{15,"*LAnkle*"}

{16,"*RAnkle*"}

Figure 3.2: Multi person body detection

3.1.4 Ildoonet

- It has 10 fps, Gpu requires , more accurate but speed is slow.
- It detects 18 keypoints [8].
- **For implementation of Ildoonet**:
  - **Clone the repo and install 3rd-party libraries**:

```
git clone https://www.github.com/ildoonet/tf-pose-
estimation cd tf-pose-estimation
```

* **Build c++ library for post processing.**[See : https://github.com/ildoonet/tf- pose-estimation/tree/master/tf pose/pafprocess:]

```
cd tf pose/pafprocess
swig -python -c++ pafprocess.i && python3 setup.py build ext –
inplace python setup.py install
```

– **Models and Performance**:

* cmu (trained in 656x368)
* mobilenet thin (trained in 432x368)
* mobilenet v2 large (trained in 432x368)
* mobilenet v2 small (trained in 432x368)

– **Demo**:

For locally stored image:$python run.py --model=mobilenet_thin --resize=432x368 --image=./images/p1.jpg$

For webcam: $python run\_webcam.py --model=mobilenet\_thin --resize$

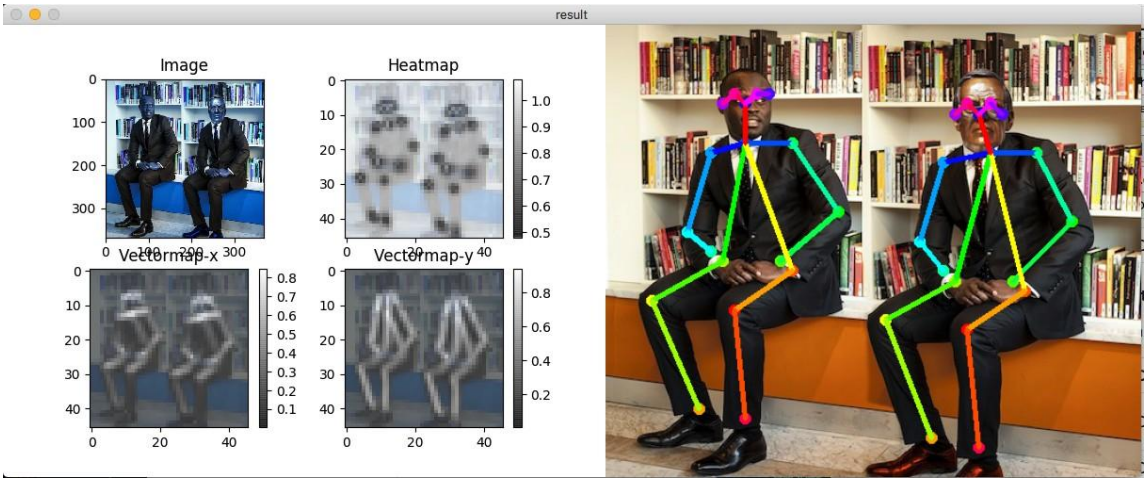

Figure 3.3: Ildoonet Output

- For the implementation of each library we have used a powerful device called as Jetson Tegra x2.
- It is the fastest, more powerful, efficient embedded AI computing device.
- It is also a GPU with fastest computing capacity. So if it shows some kind of improvement in getting high fps so detection can be also fast.
- We get around 11 fps for ildoonet while using model mobilenet v2 large. This means 11 frames per second are in process.
- Also for better accuracy we have used Movidius stick, Which is helpful in increasing fps and speed of detection.
- Movidius is computing USB stick that is to be attached with Raspberry Pi and it is kind of Virtual processing Unit.

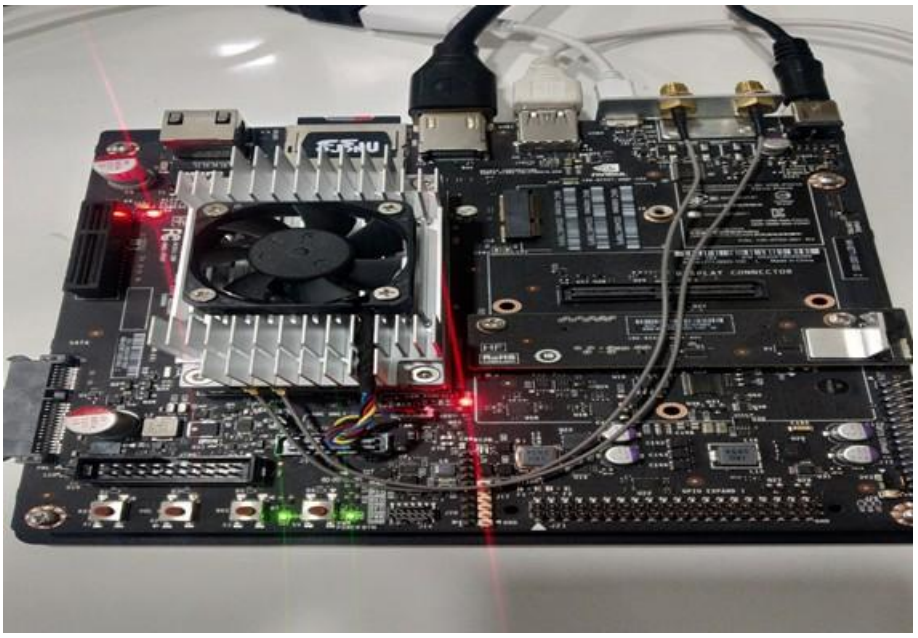

Figure 3.4: Jetson Tegra-x2

· But as per our experience, we have implemented ildoonet on this stick with

Raspberry pi, but performance was not so nice.

Figure 3.5: Movidius Neural computing stick

3.2 Phase2: Counting of hand movements

- By studying and implementing above libraries, at the end we have finalized to detect human pose with Ildoonet library.
- By doing comparisions we have finalized this library as it is slow but accurate.
- At the output it will show keypoints of a person detected.
- In this phase, after detecting keypoints properly with Ildoonet, we did the counting part. It means with the keypoint, the system is able to count the movements of particular keypoint. It will check x and y coordinate and its value[9]. If the value is differ than previous then it will identify that at this time this particular point is moving.
- This a demo when a person is moving her hands in upward direction. Then the sytem first detects that yoga is in progress. And a person is moving her right or left hand according to keypoint position.

Figure 3.6: Counting demo 1

Figure 3.7: counting demo 2

3.3 Phase 3: Identify Yogasana name

- The next phase of using other repository, that will be able to identify particular posture of a person
- That process is known as Human activity recognition. It means to recognize the action performed by any person.
- A person is free to do anything with the system and system is able to identify an action.
- For example if a person is sitting then system gives output as sitting.
- In our scenario, the system will able to identify the yoga name through the yoga posture performed by any person.
- For that one repository is used called 3d-cnn with pytorch. In which one model architec- ture is used to train the different posture of a person. And from that postures a model will identify ayoganame in validation parts.

3.3.1 Human activity recognition:

- There are so many computer-science technologies like computer vision, artificial intelli- gence, machine learning. With the help of those technologies, a new technique has been developed called Human Activity Recognition (HAR).
- In which, the system is able to identify an activity of a person.
- For that there are also various libraries to be implemented.
- For our project purpose, we have implemented 3d-cnn repository.

3.3.2 3D-CNN

- As the name implies, it has 3 dimensional activation maps.
- It is used for human activity recognition.

- Convolution neural network are type of deep model that can directly act as raw inputs[10].
- The 3d filter uses only 2 directions of input that is height and width. But the output of such operation is in 2d. Naturally, convolution layers are in 3d.
- In 3d-cnn kernels are transferred through three dimensions(height,width,depth) and pro- duce activation map.
- The modelused in 3d-cnn is residual network (ResNet).

3.3.3 ResNet Architecture

- Resnet architecture has their network with thousands of layers.
- Most of the deep neural networks has been successful by using resnet architecture because of its additional layer advantage.
- The layers are together progressively learned more complex features[10].
- The first layer learns edges, the second layer learns shapes, the third layer learns objects, the forth layer learns eyes, and so on.
- According to the dataset, it will learn the posture from a dataset, this period is known as “training” for the model.
- After that when we give the same posture but a different person then it will able to recognize the posture name with the help of training period. This period is known as the "Validation or Testing” period.
- If that validation is wrong then we have to train a model again and then validate again.
- Here is a figure of residual block that how the data will pass from one layer to the next layer.

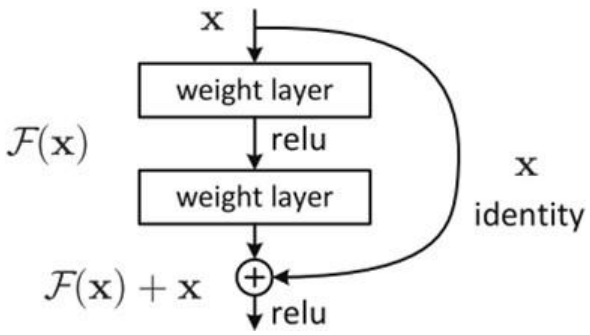

Figure 3.8: Residual block

·

· Here they just to add the output from previous layer to the layer ahead.

· Here we can back propagate to the previous layers the additional multiplication can make the gradient value very small , which will then fail to have an effect on minimizing the error. Resnet architecture was one of the ways used to solve this problem. The central idea of Resnet is to add the "shortcut connection" which skips one or more layers.

· This connection passes the gradient from previous layers to next layers. Resnets contains many such connections. we have used 3d Resnets which uses 3d convolutions. The size of kernel is 3x3x3 and initially stride used is 1, the input sizes are 3x6x112x112.

Layer Name	Architecture	
	18-layer	34-layer
conv1	$7 \times 7 \times 7, 64$, stride 1 (T), 2 (XY)	
conv2_x	$3 \times 3 \times 3$ max pool, stride 2 $\begin{bmatrix} 3 \times 3 \times 3, 64 \\ 3 \times 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 64 \\ 3 \times 3 \times 3, 64 \end{bmatrix} \times 3$
conv3_x	$\begin{bmatrix} 3 \times 3 \times 3, 128 \\ 3 \times 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 128 \\ 3 \times 3 \times 3, 128 \end{bmatrix} \times 4$
conv4_x	$\begin{bmatrix} 3 \times 3 \times 3, 256 \\ 3 \times 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 256 \\ 3 \times 3 \times 3, 256 \end{bmatrix} \times 6$
conv5_x	$\begin{bmatrix} 3 \times 3 \times 3, 512 \\ 3 \times 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 512 \\ 3 \times 3 \times 3, 512 \end{bmatrix} \times 3$
	average pool, 400-d fc, softmax	

Figure 3.9: ResNet architecture

- Brackets contain Residual blocks, after each convlutional layer there is a batch normalisationn layer and a relu layer.

3.3.4 Dataset used

- We have used custom dataset called kinetics. In which we have added 100 videos of each yogasanas, 70 for training and 30 for validation purposes.
- Resnet-34 contains its pre-trained dataset called ucf-101, it has 101 classes.
- Kinetics dataset contains 400 categories of human actions, 30,000 or more than trimmed videos. In which we have added our cutom videos.Each clip is around 10 seconds.
- The annotation quality is quite high in kinetics dataset.
- Here we have divided the whole dataset into two parts: Training and validating. And the data for both the categories are not same.

- Kinetics data structure is a group of data points which is having an attribute of the applied system.
- Kinetics data are used on a system where there is a value change with the function of time.
- Kinetics dataset allows value that is changeable with time. There will be a value that can be added, deleted, re-inserted, and updated. Depending on the demand, one can divide the data into parts.

3.4 Phase 4: Correction process

- In this phase, as the name suggests it will correct the person when he/she does it wrong.
- For example, when a person is doing Vriksasana and if he/she just bend the leg not lifted the leg then the system will indicate that the leg is not properly bent.
- So that suggestion part is also there in our system.

Chapter 4

Results

4.1 Tadasana

It is one yogasana which we have taken for the project. The yogasana posture is like both the hands are lifted in the air and people have to stand on their toe. He/she can stand at-least 10 mins for better health. Here, the model can test the yogasana within a minute. And Gives the result as Yoga name that is Tadasana.

Figure 4.1: Posture showing Tadasana yoga

4.2 Vriksasana

Here the yogasana's posture is like any lag can bend towards the knee. And both the hands are lifted up in the air. The model can train the yoga within a

minute. Below is the image of a person doing Vriksasana.

Figure 4.2: Posture showing vikrasana yoga

4.3 Virbhadrasana

In this yogasana, person have to bend over his/her knee and opposite leg position is towards backside. A person is doing yogasana in the image and a model detects it properly. The image is as below:

In all of the above yogasana model can take 16 frames at a time. And after 4 or 5 frames it will start detecting a yoga and gives labeling at left side corner. Here we have provided 100 videos of particular yogasana for training purpose and 30 different videos for validation purpose. By learning with the dataset, when we provide an other video it detects properly

Figure 4.3: Posture showing Virbhadrasana yoga

4.4 **Utkatsana**

Here in this yogasana, a person is doing utkatasana. It is known as Chair pose, where a person is likely to perform asana in sitting chair posture.

Figure 4.4: Posture showing Utkatasana yoga

4.4 utthita hasta padangusthasana

This yogasana is performed by raising one leg horizontally looking straight. Other leg must be perfectly in vertical position.

Figure 4.5: Posture showing utthita hasta padangusthasana yoga

Chapter 5

Conclusion

5.1 Conclusion

According to the study, we can come to the result that our system is well trained with few yogasanas and gives the output properly. Also, It will correct the posture if a person does it wrong. It will indicate corrections like lift your feet, hand or make some angle at your elbow, etc.

References

[1] C. Liu, J. Ying, F. Han, and M. Ruan, “Abnormal human activity recognition using bayes classifier and convolutional neural network," 2018 IEEE 3rd Int. Conf. Signal Image Processs. ICSIP 2018, pp. 33–37, 2019, doi: 10.1109/SIPROCESS.2018.8600483.

[2] G. He, X. Luan, J. Wang, and X. Wang, “A Novel Feature Map for Human Activity Recognition,” Proc. - 2017 2nd Int. Conf. Mech. Control Comput. Eng. ICMCCE 2017, vol. 2018-Janua, pp. 216–219, 2018, doi: 10.1109/ICMCCE.2017.47.

[3] S. M. Lee, S. M. Yoon, and H. Cho, “Human activity recognition from accelerometer data using Convolutional Neural Network,” 2017 IEEE Int. Conf. Big Data Smart Comput. BigComp 2017, pp. 131–134, 2017, doi: 10.1109/BIGCOMP.2017.7881728.

[4] M. Babiker, O. O. Khalifa, K. K. Htike, A. Hassan, and M. Zaharadeen, “Automated daily human activity recognition for video surveillance using neural network,” 2017 IEEE Int. Conf. Smart Instrumentation, Meas. Appl. ICSIMA 2017, vol. 2017-Novem, no. November, pp. 1–5, 2018, doi: 10.1109/ICSIMA.2017.8312024.

[5] https://github.com/tensorflow/tfjs-models/tree/master/posenet

[6] https://github.com/CMU-Perceptual-Computing-Lab/openpose

[7] https://github.com/MVIG-SJTU/AlphaPose

[8] https://github.com/ildoonet/tf-pose-estimation

[9] H. S. Koppula and A. Saxena, “Anticipating Human Activities Using Object Affordances for Reactive Robotic Response,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 38, no. 1, pp. 14–29, 2016, doi: 10.1109/TPAMI.2015.2430335.

[10] K. Hara, H. Kataoka, and Y. Satoh, “Learning spatio-Temporal features with 3D residual networks for action recognition,” Proc. - 2017 IEEE Int. Conf. Comput. Vis. Work. ICCVW 2017, vol. 2018-Janua, pp. 3154–3160, 2017, doi: 10.1109/ICCVW.2017.373.