International Journal of Engineering and Management Research e-ISSN: 2250-0758 | p-ISSN: 2394-6962
Volume-11, Issue-4 (August 2021)
www.ijemr.net https://doi.org/10.31033/ijemr.11.4.11

Yoga Posture Classification using Computer Vision

Madhura Prakash[1], Aishwarya S[2], Disha Maru[3], Naman Chandra[4] and Varshini V[5]
[1]Assistant Professor, Department of Information Science and Engineering, B.N.M Institute of Technology, INDIA
[2]Student, Department of Information Science and Engineering, B.N.M Institute of Technology, INDIA
[3]Student, Department of Information Science and Engineering, B.N.M Institute of Technology, INDIA
[4]Student, Department of Information Science and Engineering, B.N.M Institute of Technology, INDIA
[5]Student, Department of Information Science and Engineering, B.N.M Institute of Technology, INDIA

[1]Corresponding Author: madhuraprakash5@gmail.com

ABSTRACT

There has been over the past few years, a very increased popularity for yoga. A lot of literatures have been published that claim yoga to be beneficial in improving the overall lifestyle and health especially in rehabilitation, mental health and more. Considering the fast-paced lives that individuals live, people usually prefer to exercise or work-out from the comfort of their homes and with that a need for an instructor arises. Hence why, we have developed a self-assisted system which can be used to detect and classify yoga asanas, which is discussed in-depth in this paper. Especially now when the pandemic has taken over the world, it is not feasible to attend physical classes or have an instructor over. Using the technology of Computer Vision, a computer-assisted system such as the one discussed, comes in very handy. The technologies such as ml5.js, PoseNet and Neural Networks are made use for the human pose estimation and classification. The proposed system uses the above-mentioned technologies to take in a real-time video input and analyze the pose of an individual, and classifies the poses into yoga asanas. It also displays the name of the yoga asana that is detected along with the confidence score.

Keywords— ml5.js, Neural Network, PoseNet

I. INTRODUCTION

Human pose recognition is considered as a well-established computer vision method has introduced several challenges over the years. Its key function is to locate key-points of a person's body in many fields including video-surveillance, assisted living, human–computer interaction, biometrics, sports, in-home health monitoring, etc. When we consider status of the health of a particular individual, it can be evaluated and predicted with the help of monitoring and recognizing their activities.

The application of Yoga posture recognition is more or less new. Yoga, nowadays is gaining increasing importance in the field of medical research community, and numerous literatures have been proposed for various medical applications such as cardiac rehabilitation, positive body image intervention, and mental illnesses. Yoga exercises boost physical fitness, it also helps in cleansing the body, mind, and soul. It contains many asanas and each of them shows the physical postures. For people who haven't been exercising in a while, Yoga is very helpful. It's helpful for people who have conditions such as arthritis or osteoporosis.

Considering the fast-paced lives, exercising at home is mostly preferred by people these days, as some of the resources are not publicly available, human pose recognition can be utilized to develop a self-assist exercise system that allows individuals to understand, learn and practice exercises correctly by themselves. The performance of the users by a computer-assisted self-training systems especially in sports to prevent injuries. These kinds of computer-assisted systems come into use when groupwise activities and other interactions may not be feasible when there is a pandemic. Similar works have been proposed which includes automated and semi-automated systems for analyzing the sports and exercise activities which can be accessed according to one's convenience.

II. METHODOLOGY

CNN is used internally by PoseNet for recognizing human poses and giving key-points as output. The neural network is trained on key points of joint locations of the human skeleton or can be trained directly on the video using CNN. The model was used on the PoseNet key points to classify yoga asanas and achieved an average accuracy of 98%, hence there was no need for Convolution Neural Network.

To detect parts such as elbow hips, wrists, knees and ankles, a deep learning TensorFlow model called PoseNet which allows one to estimate and track human poses is used. It is more user-friendly and requires minimum 4GB RAM, 2GB GPU do detect poses in real-time with good response time The model can estimate the position of a person's joints in an image. The aim is to identify the positions of feature points in order to track human body movement. The confidence score and an array of key points listed by Part ID, each with score and position for all 17 points are returned. By getting (x, y) coordinates of each joint, one can identify pose of the person. In such a manner, continuously data is

This Work is under Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

collected for a certain Asana and stored in the Json file. The process is repeated for all the Asanas. To make ML approachable for a wide audience of artists, creative coders, and students ml5.js is used. The access to ML algorithms and models within the browser is provided by ml5.js library, which is made on top of TensorFlow.js with no other external dependencies.

III. PRIOR APPROACH

There are generally three approaches for pose recognition: OpenPose, Posenet and PifPaf. OpenPose is a multi-person key point detection technique that has brought a dramatic change in the domain of pose estimation. It uses convolutional neural network architecture that recognises facial, foot and hand key points of an individual from an image. The Human body joints are recognized using the RGB camera. Here the keypoints are nose, ears, shoulders, wrists, ankles, hips, knees, elbows, neck and eyes. The result obtained is represented by processing the inputs from a camera in static images, pre-recorded videos or real time video as keypoints. The approach proposed in [1] uses Long short-term memory and Convolutional neural network on data received from OpenPose that detects and extracts the key points which are passed to the model where convolutional neural network detects patterns and long short-term memory analyses the change over time. The model recognizes the six asanas from recorded videos and in real time for twelve individuals with an accuracy of 98.92%.

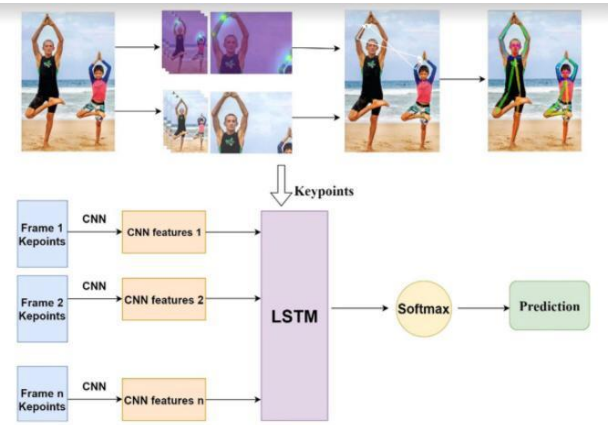

Figure 1: Existing Mechanism

The approach [2] proposes an application where a variety of images - cat and dog are used for classification. On CPU System, four classifier combinations and activation functions are compared with four different structures of convolutional neural network. Here Keras, Tensorflow, Deep Learning, Relu, Sigmoid, CNN, Tanh, SoftMax, Image Classification are used. Similarly in the approach [3] uses deep learning to classify yoga images. CMU'S open source OpenPose is used as preprocessing module and the model uses RGB skeleton image dataset which runs through CNN classifier. The drawback was that the accuracy achieved is 78% with a smaller dataset. An approach using 3DCNN method is also proposed [4] where 3DCNN method is used in order to recognize the yoga pose which is then introduced to supplementary layers like batch normalization and average pooling that improves computational efficiency. In approach [5] using Part Affinity Fields proposes PAF to learn on how to associate body parts with individuals from the image as proposed in [6]. High accuracy and real time performance is gained from the bottom-up system regardless the number of persons within the image. The only drawback is that the implementation is with respect to images only. Similarly Joint Regressors [7] are used to address the issue of estimating two- dimensional pose from still images. As Joint Regressors two layered random forests are employed where the first layer of random forest acts as independent, discriminative part classifier and the second layer of random forest considers the estimated class distributions of the first one into account and thereby is able to predict joint locations. PoseNet considers input as a processed camera image and outputs data about the keypoints. The keypoints detected are listed by a part ID, with a confidence score in the range of 0.0 and 1.0. The confidence score shows the probability that a key point can be used to estimate a single or multiple poses, which means the version of algorithm can detect only single individual from an image or video. The approach [8] proposes that PoseNet grasps single 224x224 RGB image and regresses the camera's six degrees of freedom pose. The system instructs a CNN to regress the six degrees of freedom cameras pose from a RGB image until the end. The algorithm works in indoors and in open-air in real time considering 5ms per frame to determine. It also aims to trace the usage of multi-view geometry as a training data source for deep pose regressors and survey the probabilistic extensions to the algorithm in future. PifPaf is the latest approach for two dimensional multi person human pose estimation. It is a bottom-up approach and uses part intensity field to localize the body part and part association field to associate body parts in order to structure into human poses. With respect to lower resolution and better performance in jam-packed places it is considered as a better model due to the encoding of the information in a newer composite part association field and integration of selected Laplace Loss. The approach [9] proposes a bottom-up method for multi person 2D human pose estimation that is applicable for urban mobility. The goal of the method is to estimate human poses in packed images.

This Work is under Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

IV. OUR APPROACH

Browser friendly applications can be used by anyone with a browser like Google Chrome or Microsoft Edge. The aim is to make the application accessible to everyone, even who may not have good system requirements. For this purpose, we have used ml5.js in our application. It is a JavaScript library built on top of TensorFlow.js. It is a library for handling operations that are GPU accelerated and helps in managing memory for machine learning algorithms. Ml5.js gives direct access to pre-trained models for detecting human poses within the browser. The public understanding of machine learning and fostering deep engagement with ethical computing is given by ml5.js. Ml5.js provides approachable easy to use APIs as one can focus on the logic and flow rather than in-depth functions and implementations.

As previously mentioned, the machine learning model that allows for Real-time Human Pose Estimation is PoseNet. There are different versions of the PoseNet algorithm which can detect for a single pose or multiple poses in an image or a video. For the scope of this paper, we have considered single pose only. The approach is divided into three parts namely, data collection, training the model, and classification.

4.1 Data Collection

We use p5.js, a JavaScript library to capture the video in real-time. Each video frame is loaded into the ml5.posenet model. Ml5's implementation of PoseNet, uses one of the two versions, "MobileNetV1" and "ResNet50". These models are types of CNNs. CNN finds pattern in the pixels of images and through successive layers of computation finds sets of patterns to identify more complex patterns. The model is initially loaded before starting the data collection. On detecting a pose, we add the collected keypoints from the pose detected to the input array corresponding to the label specified. The keypoints are as shown in Table 1.

Table 1: Key Points Identified By Posenet Model

Id	Part
0	Nose
1	Left Eye
2	Right Eye
3	Left Ear
4	Right 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

The keypoints are also displayed on the video frame. In this case, the labels indicate different asanas of our system. The data collected is added to the neural network post the collection and finally saved locally as a JSON file.

4.2 Training the Model

The ml5.neuralNetwork can do classification or regression tasks. The generalized steps for using the ml5.neural network is as follows:

- Step 1: load data or create some data
- Step 2: set the neural network options & initialize the neural network
- Step 3: add data to the neural network
- Step 4: normalize the data
- Step 5: train the neural network
- Step 6: use the trained model to make a classification or regression

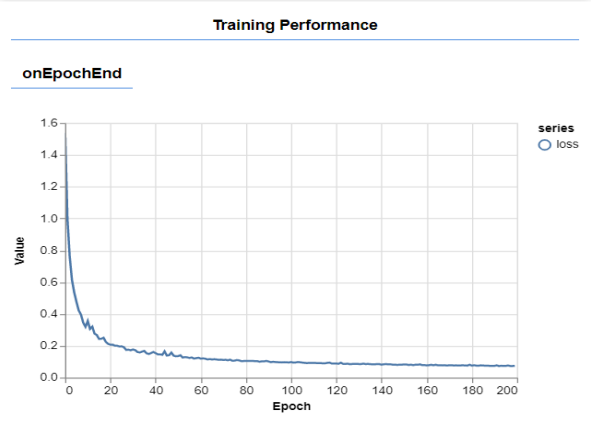

Figure 2: Epoch value set to 200

After loading the data, i.e., the previously saved Json file with keypoints, ml5.neuralnetwork is set with the options that indicate inputs, outputs, task(classification). Custom layers can also be set along with the options. The data is then normalized using normalizeData() which normalizes the data on a scale from 0 to 1. and the loaded data is trained with an epoch value as shown in the Fig 1. The model obtained is saved using save() and consists of three files, the model i.e., model.json, the metadata i.e., model_meta.json, and the weight file i.e., model.weights.bin.

4.3 The Classification

Using the load() function of ml5.js, the three model files are loaded into the neutral network. The PoseNet model is loaded initially as the program starts and the real time video input is taken using p5.js canvas and as the PoseNet detects a pose, the pose key points are given as input for the neural network to be classified into one of the asanas. The pose is classified into one of the eight asanas as shown in the table 2. There is one default pose i.e.,

This Work is under Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

standing, when the user is idle, this is the result displayed.

Table 2: The List of Asanas

No.	Asana
1	Siddhasana
2	Trikonasana
3	Vajrasana
4	Veerabhadrasana
5	Vrukshasana
6	Anjaneyasana
7	Dandasana
8	Default – Standing

The confidence is obtained from the results of classify() function. A threshold of 0.75 is set for the confidence score and the classified asana is displayed on the screen.

V. ANALYSIS

Table 3: The List Of Asanas with Confidence Average

No.	Asana	Classifier	Regressor
1	Siddhasana	99.66743	90.00125
2	Trikonasana	97.87977	87.16578
3	Vajrasana	99.38124	89.95621
4	Veerabhadrasana	99.25908	88.83210
5	Vrukshasana	98.42687	90.63447
6	Anjaneyasana	97.10453	88.45204
7	Dandasana	98.44310	87.86524
8	Default	99.56122	89.96125

Ml5.neuralNetwork provides classification and regression tasks. Once the neural network is trained, it can perform either of the tasks. Usually, classification is used when the prediction is done for discrete values, in this case, the different asanas are the discrete values. Whereas, regression is performed for continuous values. The system was tried for both the tasks and the results have been tabulated in Table 3. The average confidence achieved by classification was approximately 10% more than regression.

VI. CONCLUSION

In this paper, we discuss about the Computer Vision based system that classifies the yoga asanas along with its confidence score, which is the measure of how confident the machine learning model is in classifying and identifying the asanas. The dataset is assembled using the webcam for 3-4 persons. The keypoints of the individuals are detected by PoseNet. The use of Neural Networks and ml5.js on the PoseNet data is observed to be effective and classifies all the 7+1 yoga asanas excellently. Performance of the Convolution Neural Networks (PoseNet) proves that Machine Learning can also be implemented for pose estimation or activity recognition problems. The proposed model currently classifies only 7+1 yoga asanas. There are a number of asanas, and hence creating a pose estimation model that can be successful for all the asanas is a challenging problem. The dataset can be extended by increasing the number of yoga asanas performed by individuals and multiple individuals as well, which works not only in an indoor setting but also outdoors.

REFERENCES

[1] Yadav, Santosh Singh, Amitojdeep Gupta, & Abhishek Raheja, Jagdish. (2019). *Real- time Yoga recognition using deep learning*. Available at: https://link.springer.com/article/10.1007/s00521-019-04232-7.

[2] LI, Ang LI, Yi-xiang LI, & Xue-hui. (2017). TensorFlow and keras-based convolutional neural network in CAT image recognition. *DEStech Transactions on Computer Science and Engineering*. DOI: 10.12783/dtcse/cmsam2017/16428.

[3] A. Lai, B. Reddy, & B. Vlijmen. (2019). *Yog.ai: deep learning for yoga*. Available at: http://cs230.stanford.edu/projects_winter_2019/reports/15813480.pdf.

[4] Jain, Shrajal, Rustagi, Aditya, Saurav, Sumeet, Saini, Ravi, & Singh, Sanjay. (2020). Three- dimensional CNN-inspired deep learning architecture for Yoga pose recognition in the real-world environment. *Neural Computing and Applications*. DOI: 10.1007/s00521-020-05405-5.

[5] Z. Cao, G. Hidalgo Martinez, T. Simon, S. -E. Wei, & Y. A. Sheikh. (2019). OpenPose: Realtime multi-person 2D pose estimation using part affinity fields. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*. DOI: 10.1109/TPAMI.2019.2929257.

[6] Z. Cao, T. Simon, S. Wei, & Y. Sheikh. (2017). Realtime multi-person 2D pose estimation using part affinity fields. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI*, pp. 1302-1310. DOI: 10.1109/CVPR.2017.143.

[7] M. Dantone, J. Gall, C. Leistner and L., & Van Gool. (2013). Human pose estimation using body parts dependent joint regressors. *IEEE Conference on Computer Vision and Pattern Recognition, Portland*, pp. 3041-3048. DOI: 10.1109/CVPR.2013.391.

[8] A. Kendall, M. Grimes, & R. Cipolla. (2015). PoseNet: A convolutional network for real- time 6-DOF camera relocalization. *IEEE International Conference on Computer Vision (ICCV), Santiago*, pp. 2938-2946. DOI: 10.1109/ICCV.2015.336.

[9] S. Kreiss, L. Bertoni, & A. Alahi. (2019). PifPaf: Composite fields for human pose estimation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA*, pp. 11969-11978. DOI: 10.1109/CVPR.2019.01225.

This Work is under Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.