ISSN No:-2456-2165

AI-Based Yoga Pose Estimation for Android Application

Girija Gireesh Chiddarwar, Abhishek Ranjane, Mugdha Chindhe, Rachana Deodhar, Palash Gangamwar
Computer Department, SCOE, Vadgaon,
Pune-411041, India

Abstract:- The importance of yoga is renowned worldwide and its health benefits, which were preached by ancient sages, have stood the test of time. Even though yoga is becoming preeminent, there are important challenges faced while doing yoga such as performing it with incorrect form, the classes being expensive and the shortage of time in our busy lives. Computer vision techniques exhibit promising solutions for human pose estimation. However, these techniques are seldom applied in the domain of health or exercise, with no literature or projects cited specifically for yoga. This paper surveys the various technologies that can be used for pose estimation and concludes the best method based on the usability for an android application. The paper then discusses the methodology that will be used to deploy the yoga pose estimation on an android application, how the app is modeled and the working of each component is explained.

Keywords:- *AI; Yoga; Deep Learning; PoseNet; Android.*

I. INTRODUCTION

Deep learning techniques have proved their importance for object detection tasks. The same models can be effectively used to detect the various essential body parts and estimate the pose of the user in real-time. The paper surveys the different options that can be used for pose estimation such as OpenPose, Posenet, DeepPose, and concludes which technique should be used for deploying it for an android application for doing yoga.

There are several methods that can be used for pose estimation. We shall discuss some of the most important ones. In this paper we shall discuss the evolution of human pose estimation over the years and how Posenet is the most suitable for our project (yoga pose estimation on android). The project provides real-time pose estimation for yoga pose estimation and correction on the client-side. This is achieved with Tensorflow. The library allows the inference of models on Android, which makes it faster and more privacy respecting. Posenet is an open-sourced technology that allows us to extract the 17 essential points natively. A skeleton of the human pose is drawn with the help of these points, which is then used to derive angles between these points thus enabling us to effectively correct the user's yoga poses. This methodology is used in our Android application along with Google's text-to-speech and speech-to-text modules for the user to do yoga very effectively.

II. RELATED WORK

- *Deep Pose*

DeepPose was the first major paper[1], published in CVPR 2014 that applied Deep Learning to Human pose estimation. It achieved SOTA performance and beat existing models back in the year 2014. The model has an AlexNet backend and estimated pose in a holistic fashion, i.e certain poses can be estimated even if some joints are hidden when the pose is reasoned holistically. The paper applies Deep Learning (CNN) to pose estimation and kicked off research in this direction. The model used regression for XY locations for certain regions. This added complexity and weakened generalization hence performing poorly.

- *Efficient Object Localization Using Convolutional Networks*

This approach in this paper[2] uses heatmaps by sliding through multiple subsets of the image (windows) in parallel to simultaneously capture features at a variety of scales. A heatmap predicts the probability of one of the essential points. This model is very successful for pose detection and is used in current models as well.

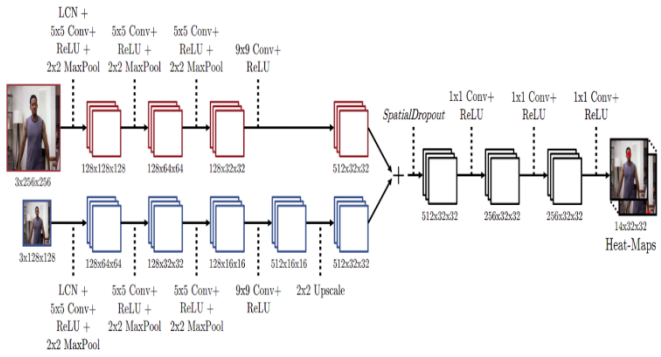


Fig 1:- Working of Sliding window detector

Heatmaps work better than direct joint regression. This model, however, does not include structure modeling, that is, taking into consideration the human body structure, such as body part proportion, symmetries, joint limits, among others.

- *Convolutional Pose Machines*

This interesting paper[3] proposes the serial usage of stages to predict. a CPM (Convolutional Pose Machine) consists of an image feature computation module followed by a prediction module. The multiple stages can be trained

end to end. Stage 1 generates heatmaps and image evidence as input for stage 2. Stages > 2 are just repetitions of the second stage.

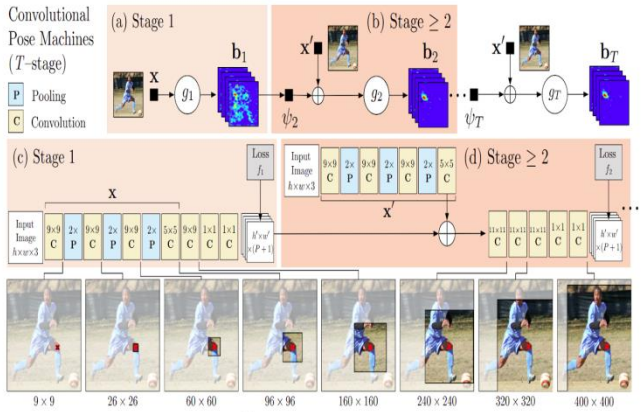


Fig 2:- Effective Receptive Field

➢ *OpenPose*

Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields[4] proposes the detection of multiple people in an image. It uses an approach called non-parametric representation, also known as Part Affinity Fields (PAFs). The architecture encodes a global context, allowing a greedy bottom-up parsing step that maintains high accuracy while achieving real-time performance, irrespective of the number of people in the image. The method got SOTA performance on the MPII Multi-Person benchmark.

The bottom-up approach decouples runtime complexity from the number of people in the image. It uses Part Affinity Fields(PAFs), a set of 2D vector fields that encode the location and orientation of limbs over the image domain.

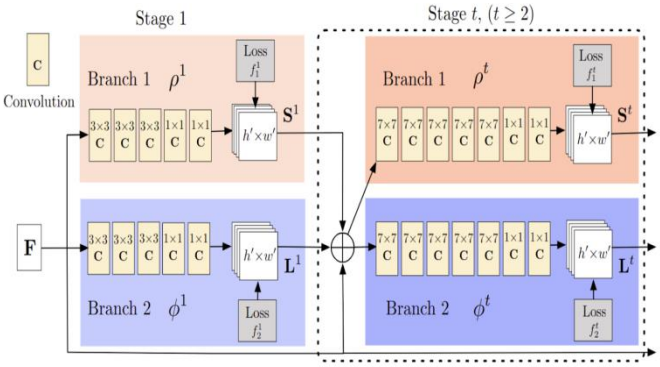


Fig 3:- Architecture of the Two-branch Multi-stage CNN

The problem with OpenPose is that it requires special hardware and can not perform well on mobile devices.

➢ *Deep High-Resolution Representation for Human Pose Estimation.*

The HRNet (High-Resolution Network) model [5] maintains a high-resolution representation throughout the whole process, instead of high → low → high-resolution representation, and this works very well. The architecture starts from a high-resolution subnetwork as the first stage, and gradually adds high-to-low resolution subnetworks one by one to form more stages and connect the multi-resolution subnetworks in parallel.

Repeated multi-scale fusions are conducted by exchanging information across parallel multi-resolution subnetworks over and over through the whole process. Another pro is that this architecture does not use intermediate heatmap supervision. Heatmaps are regressed using an MSE loss.

➢ *Deep High-Resolution Representaion Learning for Human Pose Estimation*

The HRNet (High-Resolution Network) model [5] maintains a high-resolution representation throughout the whole process, instead of high → low → high-resolution representation, and this works very well. The architecture starts from a high-resolution subnetwork as the first stage, and gradually adds high-to-low resolution subnetworks one by one to form more stages and connect the multi-resolution subnetworks in parallel.

Repeated multi-scale fusions are conducted by exchanging information across parallel multi-resolution subnetworks over and over through the whole process. Another pro is that this architecture does not use intermediate heatmap supervision. Heatmaps are regressed using an MSE loss.

➢ *Deep High-Resolution Representaion Learning for Human Pose Estimation*

The paper [6] by George Papandreou et al. presented a box-free bottom-up approach that is used in PoseNet. PoseNet is an open-sourced application that can be deployed using Tensorflow. PoseNet gives an output of poses, pose confidence scores, keypoint positions, and keypoint confidence scores from the model outputs. It uses the best practices from some of the previous papers and gives very high accuracy for the minimal cost it requires.

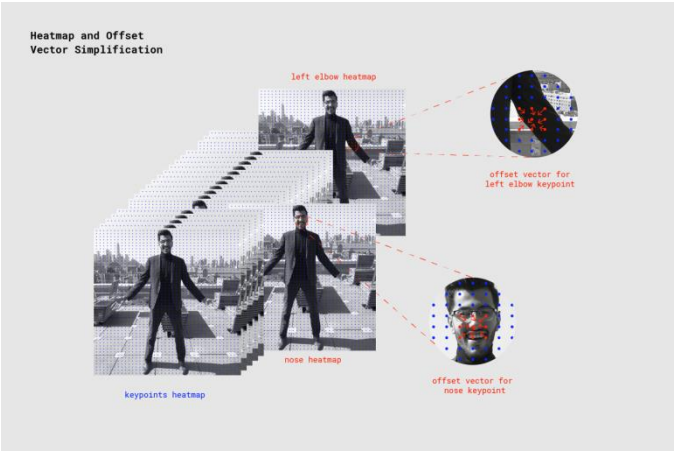


Fig 4:- Architecture of the Two-branch Multi-stage CNN

PoseNet can be configured with the *output stride*. PoseNet is invariant of the size of the image presented, by downscaling the image by a factor of this output stride. The higher the output stride the lower the accuracy, but the faster the speed, which is key in the android implementation. Altrous convolution [7] is then used to enable the

convolution filters in the subsequent layers to have a wider field of view. PoseNet reapplies the concept of Heatmaps from Efficient Object Localization Using Convolutional Networks [2]. It also uses *offset vectors* that correspond in location to the heatmaps points, and are used to predict the exact locations of the keypoints by traveling along the vector from the corresponding heatmap point.

PoseNet offers both single and multiple pose estimations. Both of these techniques used different approaches. Single pose estimation is much faster and efficient, thus making it suitable to use on an android device. PoseNet is a 2D pose estimation technique, which only does the job of giving the XY coordinates as opposed to XYZ coordinates present in 3D methods. PoseNet also does not require special hardware, unlike OpenPose[4]. Thus, we can conclude that PoseNet is the most suitable method that can be used for pose estimation in a mobile application.

III. METHODOLOGY

The methodology will be discussed in 2 modules namely the pre-processed part if the system and the Native (Real-Time) part of the system. The purpose of preprocessing is to get target values for the poses that need to be performed. The Native part deals with the programs that run on the android device, without any external interaction. The task of the Native part is to predict the actual poses done by the user, in real-time, and provide the user with interfaces to do yoga efficiently.

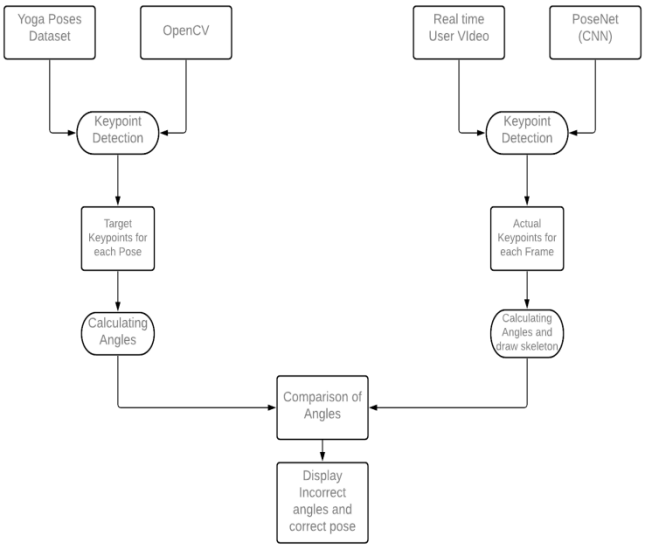


Fig 5:- methodology of the System

Finally, by using techniques, later discussed in the paper, the keypoints given by Pre-processing part and the Native part are compared to get the measure of correctness of the yoga pose performed by the user.

➢ *Preprocessing Part of the system*

The objective of this module is to get the target values, that is the ideal values, of each yoga pose to use as a metric to determine the correctness of actual yoga pose later.

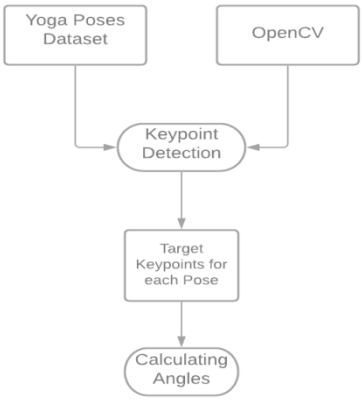


Fig 6:- Pre-processing part of the system.

In this module, the target values are extracted in the following steps:

- We collect a single ideal image for getting the keypoints. We store these locally in our machine.
- We use OpenCV to predict the 17 essential keypoints using a pretrained model, OpenPose [4], that uses the 2-branch multi-stage CNN.
- The keypoints are noted to be later hard-coded as the target values in the next phase.

➢ *Native (Real-Time) application*

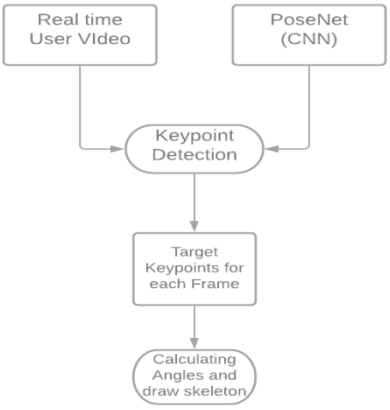
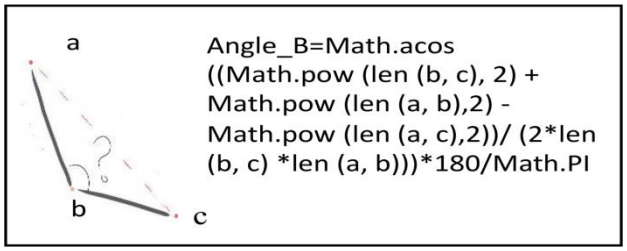


Fig 7:- Native part of the system

The android app natively provides a GUI, takes the user video, predicts the angles using PoseNet, calculates the angles and displays the results after comparison. These processes happen in the following steps:

- In the camera activity, the user is asked for permission to access the phones camera. The video is taken and every fifth frame is used by converting it's resolution to 224 x 224.
- The frames are passed into the Posenet Class, where various hyper-parameters are set and the frames are passed through the PoseNet model. The output consists of the coordinates of the 17 essential keypoints of the body.
- The PoseNet model can be easily deployed on an android app by using TensorFlow-Lite. The trained model is stored in a ".tflite" format and can be inferred to get the 17 essential points.

- The skeleton of the user, based on the keypoints is plotted. This is done to give the user a more intuitive experience.
- The distance between each body part is calculated, using Euclidean Distance. Since, the distance might vary from person to person or depending on the phone-user distance, we calculate the angles for comparison.
- The angles are calculated using the cosine law of triangles for both- the target values and the actual values. The mathematical functions are performed using the Math module in android.



- For the GUI, we use a navigation drawer for quickly navigating the app. We include an Home page, Instruction page, and About Us page.
- We use the ImageView Layout along with drawables to display the frames, skeleton, the angles and correctness.
- We have also added Google's text-to-speech and speech-to-text modules for the user to be able to navigate the app with interruption. This Voice-controlled Yoga activity is enabled by KontinousSpeechRecognizer module that allows us to get user sound input and process it continuously.
- Thus, by using these modules, the user can say, "start second pose" to start doing the second pose and will be prompted by "Pose correctly performed" when they hit the target values for the pose.

IV. FUTURE DIRECTION

- Gesture recognition: This feature has not been implemented yet. However, since PoseNet already gives the coordinates of various body parts, it will be fairly easy to implement a gesture recognition system to navigate the app. The app would allow the user to navigate and control the app by using gesture performed by moving their head or hands.
- Progress tracking: This feature will store the measures of the correctness of the user's yoga poses. Giving users the ability to monitor their progress will not only be informative to them of their improvement but it will also keep the user motivated to keep on doing yoga.

V. CONCLUSION

Deep Learning methods have proved to be extremely useful for pose estimation, compared to any other methods. The diverse applications of pose estimation have provoked many advancements in this field, both in terms of speed and accuracy. We have concluded that PoseNet is, by the current standards, the best technique for implementing mobile applications, specifically for yoga. This method can be easily implemented by using the TensorFlow-Lite framework. The pre-trained model can be inferred to get the 17 essential points. We use these points to calculate the angles and compare them to the correct pose angles calculated by OpenCV. The skeleton and wrong angles are displayed. Thus, This paper substantially provided a glimpse of how the methods of pose estimation have evolved over the years, and how they can be effectively used in many applications, such as yoga pose correction.

ACKNOWLEDGMENT

We would also like to take this opportunity to thank out guide, Mrs. G. G. Chiddarwar, for helping us throughout. Also, to our review committee member Mrs. J. B. Kulkarni for invaluable advice and critique. Furthermore, we would like to express gratitude and earnestly acknowledge out parents for supporting us and friends for encouraging us. To Prof. M. P. Wankhade, our Head of Department and our principal, Dr. S. D. Lokhande for backing us.

REFERENCES

[1]. Alexander Toshev and Christian Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 1653-1660
[2]. Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, Christoph Bregler New York University,"Efficient Object Localization Using Convolutional Networks," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 648-656
[3]. Shih-En Wei, Varun Ramakrishna, Takeo Kanade, Yaser Sheikh, "Convolutional Pose Machines," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 4724-4732.
[4]. Zhe Cao, Tomas Simon, Shih-En Wei, Yaser Sheikh, "Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 7291-7299.
[5]. Ke Sun, Bin Xiao, Dong Liu, Jingdong Wang, "Deep High-Resolution Representation Learning for Human Pose Estimation," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 5693-5703
[6]. George Papandreou, Tyler Zhu, Liang-Chieh Chen, Spyros Gidaris, Jonathan Tompson, Kevin Murphy, "PersonLab: Person Pose Estimation and Instance Segmentation with a Bottom-Up, Part-Based, Geometric Embedding Model," The European Conference on Computer Vision (ECCV), 2018, pp. 269-286
[7]. Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, Hartwig Adam,"Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation" The European Conference on Computer Vision (ECCV), 2018, pp. 801-818.