VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

J Component report

Programme : B.Tech(CSE)

Course Title : IMAGE PROCESSING

Course Code : CSE4019

Slot : E1 + TE1

Title: Detection of Yoga Poses using CNN and OpenPose

Team Members: Saravanan Kishan | 19BCE1337

Sanjit Kapoor | 19BCE1247

S M Satya Sree Narayanan | 19BCE1172

Praneeth Sethumadhavan | 19BCE1599

Faculty: Dr. S Geetha **Sign:**

Date: 10th May 2022

Detection of Yoga Poses using OpenPose and CNN

Saravanan Kishan, Praneeth Sethumadhavan, Sanjit Kapoor, Satya Sree Narayanan
School of Computer Science & Engineering., Vellore Institute of Technology Chennai, India
Email: kishansara22@gmail.com, praneeth.sethumadhavan@gmail.com, kapoorsanjit@gmail.com, satyasreenarayanan@gmail.com

***Abstract*—There are many benefits associated with doing yoga. People either prefer to learn yoga through a physical class or through nowadays there has been an increase in online yoga classes. As beneficial as yoga is, it is important to perform it in the proper way. If yoga is done without proper guidance or done incorrectly, it could have severe consequences on your health. This gives rise to the need for a yoga pose detection mechanism. This work aims to estimate yoga poses from images. Initially pre-processing steps like smoothing, sharpening of image is done to remove different noises and maintain the quality of the image. OpenPose [1] is used to detect joints from the human body. OpenPose is used to identify the joints in the image and is given as input to the prediction model. Deep learning model Convolutional Neural Network (CNN) is vastly used for classification of images. So, we have decided to use Densenet model, which is a variation of CNN, for classification. Comparative analysis of the outputs of the classification model, with and without OpenPose and with and without pre-processing filters have been recorded and inferences have been made and verified.**

***Index Terms*—OpenPose [1], Convolutional Neural Network(CNN), Image filters**

I. Introduction

Presently, technology has taken over the world, majority of a person's time is spent using a mobile phone or sitting infront of laptop. There is no priority given for his/her health. Humans have started working like machines and working with machines. Everyone must realise health is of utmost importance and maximum care should be given to it. We have seen increasing cases of illness and people complaining of fatigue very easily. The key is to have a work-life balance.

Along with the work done, its essential to include a daily or at least weekly schedule for physical exercise. There is no need for it to be vigorous. It can even be walking at a gentle pace. But the best form of exercise both for the body and the mind is yoga. There have been several instances where performing yoga has put both the body and mind at ease. Involving yourself in yoga around an hour per day has been proven to improve your productivity.

The Coronavirus situation has meant that people fear being in contact with anyone except their family members. They don't know who is healthy or infected and thus this has affected all forms of learning. Both academic and sports related learning has been affected. During this period of the Covid pandemic, learning yoga in a group class or having an instructor to teach yoga may become risky. The person who wants to learn yoga may not want to be in contact with anyone else.

There are online yoga classes provided but it is difficult for the instructor to monitor the student's progress. Learning yoga incorrectly could become harmful for your posture, so it is of utmost importance that the yoga poses that you do is accurate and does not affect your overall posture.

To achieve this, a model that can remotely detect and classify the images of yoga poses must be designed. This will overcome the issues mentioned above and ensure the user can conveniently learn yoga in the right way without any risks in a remote location.

II. Dataset

When performing detection of yoga, it is essential to have a dataset that satisfies criteria such as variety of yoga poses and also variation in the background of the images collected. This will give rise to more insights when the model is implemented. the dataset was obtained from Kaggle. The data was collected in a way so that they represent different yoga poses performed by people of different genders and age groups. The dataset that we will be operating on will contain a total of 107 asanas. This dataset is one of the rare datasets with such a large variety shown in the type of asanas. In total 5994 images are present in the dataset for all the asanas combined. Each asana on average contains over 50 images with a maximum of 90 images. The image of an asana along with the label of the asana is provided so that it can be used for classification. The images individually have different backgrounds and surrounding features which will require different preprocessing tasks.

III. Literature Review

The authors of [2] has the objective of ensuring a person learning yoga is doing it correctly. This is a rare work that takes similarity measurements into account by comparing the posture of the learner and the instructor. The input is taken from a webcam. It initially uses OpenPose to detect the key points that are finding the body parts of a person. A CNN model is then trained to estimate the pose. The similarity measurement is based on the angles of the specified body parts of the learner and instructor. These specified body parts are known as coordinated points. This is to estimate how much a learner can follow the instructor and most importantly it must be regardless of body size or age. A key feature of this work is that it can indicate which part of the yoga pose is incorrect. If the angle of deviation is larger than 45 degrees, then that point is displayed red and otherwise it is in green. This helps to ensure that the learner can correct his

mistakes. The output of this model is based on the average angle deviation.

Similarly in paper [3] the authors aim to promote seeks to create a support system for novices to encourage regular squat training, which is home-based, by employing an achievement score and video annotation as improvement inputs. Here OpenPose was used to detect joints and the squat positions were extracted from it. OpenPose was used to detect squat positions at different angles. It was able to detect calculate scores of the squat position with specific criteria.

In [4],Fazil et al. have developed an application which detects the yoga pose of the user and suggesting corrections to the user if it is an improper pose. The authors have taken up two different approaches for identifying the pose of the user: One involves using OpenPose and the other involves using Mask RCNN. Once the pose has been identified, they then move onto to detecting the pose of the user using a combination of CNN and LSTM models. The application is used for continuous real time monitoring of the yoga pose and subsequent detection.

[5] has presented an unique approach to detection yoga asanas using deep learning. It is unique as the work used end-to-end deep learning pipeline architecture. One limitation is that the dataset consists of a limited variety of only 6 asanas. The deep learning model is hyrbid in nature as it combines Convolutional Neural Network with Long Short-term Memory algorthms to detect yoga poses in real time videos. The system incorporates OpenPose before passing into the deep learning model. The accuracy obtained is approximately 99% which is excellent considering that is it performed on videos.

The key feature of [6] is that it has developed a Mobile application for yoga pose detection. It incorporated OpenPose to detect the keypoints and then CNN to remove the redundant segments of the input. The angles between the keypoints are calculated and then an overall score of the user is computed. Voice service is also a feature in this work which makes it more iteractive for the user, The feedback is given immediately to the user taking very less time so that user can correct his position if he/she performing the yoga pose incorrectly. One future improvement that can be made to this system is to incorporate Natural language processing techniques to the voice service.

In paper [7] the authors create a system that gets the image of a person doing a yoga position and compares it to the right way by comparing the angles the bodies create and tells correction to be made to improve the pose. Keypoints in the body of the image can be estimated using different methods OpenPose, Posenet and PIFPAF. The OpenPose method is used, and the pose is extracted with it indicating 18 keypoints (ear, nose and etc.) on the body. For classification of pose the image is fed to SVM model then plain CNN and then finally a CNN and LSTM combined model which is then passed through a Softmax layer to make prediction. The best prediction model turned out to be the CNN and LSTM combined model which gave an accuracy of 0.99 for classification of 6 yoga asanas.

The authors in [8] have compared the predictive capabilities of several machine learning algorithms for yoga pose detection. The dataset used by the authors was self-created and contained around 400-900 images for each of the 10 yoga poses. The dataset contained a total of 5459 images. The first step in identifying the yoga pose involved increasing brightness of the images and resizing them to a uniform size of 500x500 resolution. Then the pose-estimation algorithm provided by Tensorflow was used to mark each joint of the body and connect them with a stick diagram. The coordinates of the joints were used to calculate 12 different angles that were used as features to detect the yoga poses. The features were extracted and stored in a CSV file with labels. The dataset was then split in 80:20 ratio for training and testing of the machine learning models that were used for prediction, namely Logistic Regression, Random Forest, SVM, Decision Tree, Naive Bayes and KNN. The authors reported that an accuracy of 94.28% was obtained from all of the machine learning algorithms.

In [9], the authors make use of four machine learning paradigms, namely KNN, Support Vector Machines, Nave Bayes and Logistic regression for the classifications of the yoga poses. The dataset used by the authors was created by them and consists of the 12 poses that are part of the sun salutation asana. The images from the dataset were resized and a stick diagram was created that marked each joint of the body and connected them. Feature extraction is performed on top of it and then predictions are carried out. It was observed that the best model for prediction was KNN with an accuracy of 96

The authors in paper [10] have used CNN to detect hand gestures using palm fingertips and joints. The system was able to detect depth from the image for identifying the hand gesture. A 2D heatmap and picture characteristics were produced using the suggested CNN model. The normalised 3D hand pose was then obtained using convolution. For 2D and 3D estimations of heatmap and vector representation poses, we trained and assessed the suggested technique. For normalised 3D hand poses, the keypoints error of the model performance was around 3%, according to the findings.

IV. System Architecture

The diagram figure-1 below illustrates the step by step methodology behind this work.

V. Proposed Method

A. *Preprocessing*

1) Gaussian Blurring

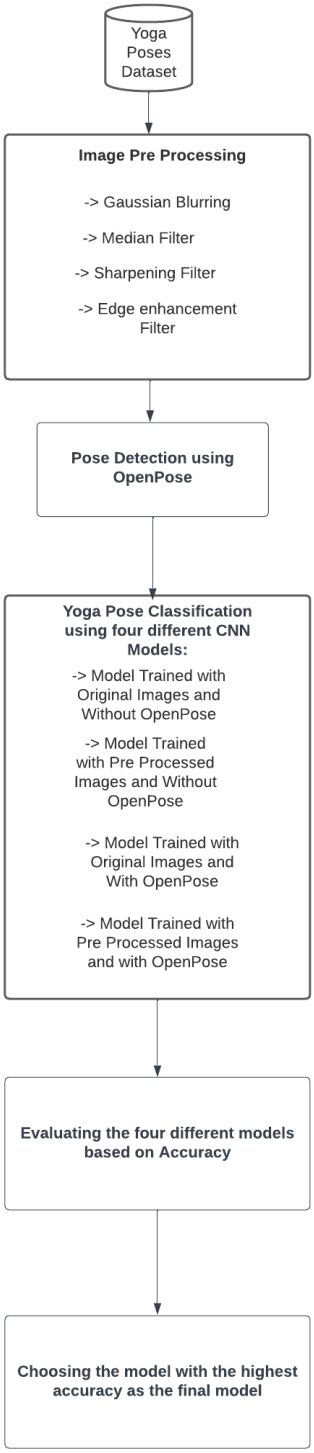

Fig. 1. Proposed System Architecture

- Gaussian kernel is applied on image to perform smoothing of the image.
- A gaussian kernel is approximation of the gaussian function.
- It is done with the function, cv2.GaussianBlur(). We should specify the width and height of the kernel which should be positive and odd.

2) Median filter
   - Here, the function cv2.medianBlur() computes the median of all the pixels under the kernel window and the central pixel is replaced with this median value.
   - This is highly effective in removing salt-and-pepper noise.
   - The central element is always replaced by some pixel value in the image.
3) Sharpening filter
   - Digital unsharp masking is a flexible and powerful way to increase sharpness.
   - The typical blending formula for unsharp masking is:
     sharpened = original + (original blurred) * amount.
4) Edge enhancement filter
   - An Edge Enhancement Filter works by increasing the contrast of the pixels around the specific edges, so that the edges are visible prominently after applying the filter.

```
-1, -1, -1,
-1, 10, -1,
-1, -1, -1
```

Fig. 2. Kernel used for ImageFilter.EDGE_ENHANCE

B. OpenPose

OpenPose is a real time multi person system to identify and detect human body, hand, leg and facial key points. Developed in Carnegie Mellon University, it is availabe in C++ and python [11]. OpenPose [1] makes use of Multi Stage Convolution Neural Network to identify the keypoints. It is capable of detecting upto 135 keypoints of the human body. For this reason, OpenPose has been used to detect the keypoints of the user while he/she is doing Yoga. Once the keypoints are detected, the positions of the various parts of the body can be identified and thereby the corresponding yoga asana can be detected. Every image in the dataset is passed through the OpenPose to detect the keypoints and later on used to train the CNN model.

C. Classification

For the classification of the yoga poses, DenseNet (Dense Convolution Network) is used. DenseNet is a type of Convolution Neural Network that allows for large increase in the depth of convolution network without having adverse effects

Fig. 3. Identifying the pose using OpenPose

on the training of the model. It is also an improvement over convention CNN because it tackles the issue of vanishing gradient that can occur while training a CNN model. Vanishing gradient problem leads to decrease in the accuracy of prediction because of improper model training as the weights updation is inaccurate. In the proposed DenseNet model, five different layers are present. The first layer is the Dropout layer with a parameter equal to 0.25. This means that the Dropout layer randomly drops out 25% of the data while training of the model. This is done to prevent overfitting of the model. In the next two layers, the Dense layer is used with the parameters 512 and 128 respectively. The Dense layer is deeply connected with the previous layer, which means that every neuron of the Dense layer is connected to every neuron in the preceding layer. The parameters 512 and 128 signify the number of neurons in each layer. Both layers use Rectified Linear Unit (ReLU) as their activation function. The ReLU function returns zero if the input is negative and the input itself if it is positive. It can be defined as max(0,z).The Dropout layer is once again introduced with its parameter equal to 0.2, meaning that 20% of the data is once again dropped to prevent overfitting. The final layer is the Dense layer. The parameter of the final Dense layer is 20, meaning that it contains 20 neurons that are densely connected to the previous layer. The activation function used in the layer is softmax. The softmax activation function gives an output of N values, where N is the number of classification classes in the problem. The N outputs are converted into probabilities for each class present in the classification problem. The class with the highest probability is then assigned as the correct output of our model, thus completing the prediction process. The aforementioned model is run for 20 epochs for thorough training on the dataset for all four combinations of presence and absence of OpenPose and filters and the results are then noted.

VI. Results

The dataset was used to train four different models with each model working on a different version of the dataset. Each of the model was run for 20 epochs and for the 60:40 train test split on the same system and the accuracies were obtained by testing the model on the test set. The first CNN model was trained on records which weren't passed through any filters and OpenPose module. The accuracy obtained in this case was 82.68%. Secondly, the images which were passed only through the OpenPose module were used to train the next Model which yielded an accuracy of 84.84%. For training the third and forth models, the images were preprocessed by passing through the filters in order to improve the quality of the images. The third module was trained using the images that were passed only through the filters and not through OpenPose [1].

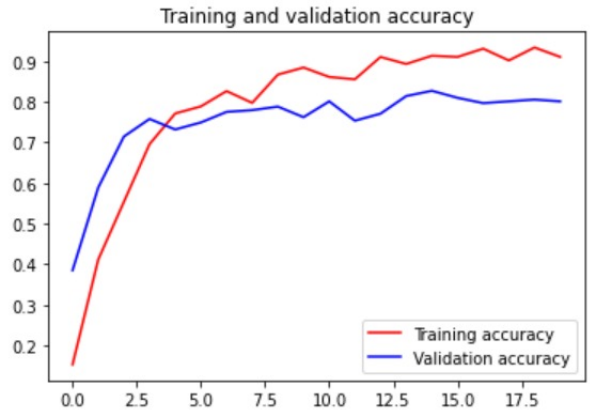

Fig. 4. Without openpose and filters

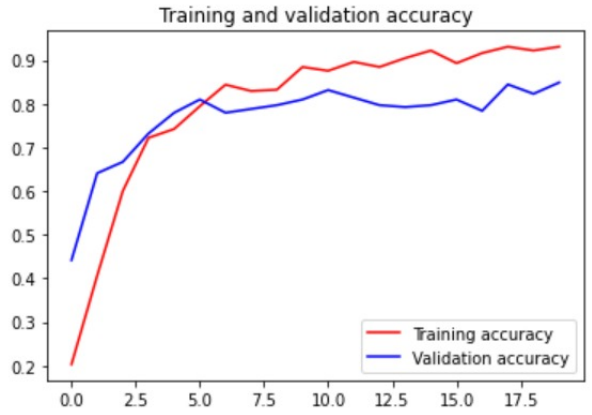

Fig. 5. with openpose and without filters

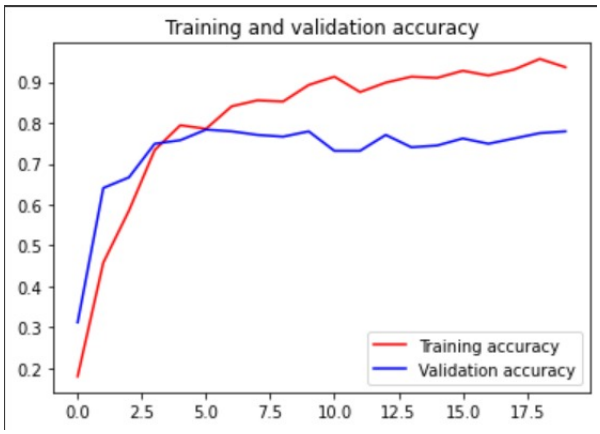

Fig. 6. without openpose but with filters

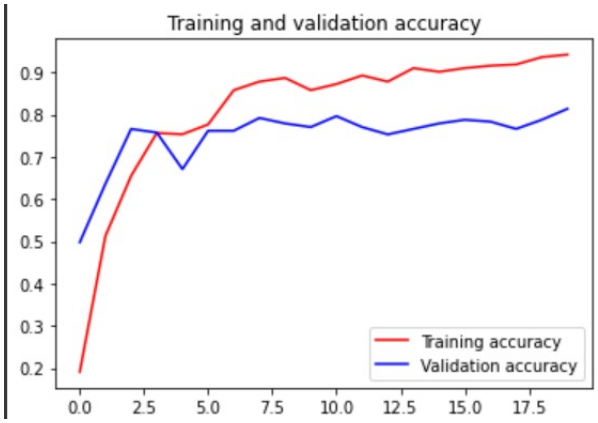

Fig. 7. with openpose and filters

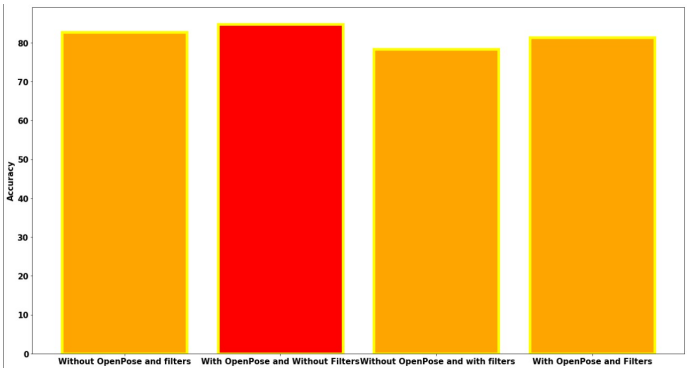

Fig. 8. Model Comparison Bar graph

Types	Without Filters	With Filters
Without OpenPose	82.68	78.35
With OpenPose	84.84	81.38

This model gave the lowest accuracy of 78.35% which was the lowest among the all models trained. The Final model was trained with images that were passed through both the filters and OpenPose module. This model that produced an accuracy of 81.38%. The Results obtained are tabulated in Table-1.

VII. Conclusion

Looking at the results, it can be concluded that the use of OpenPose along with the lack of filters yields the best prediction results with an accuracy of 84.84% whereas an accuracy of 82.68% is obtained when both OpenPose and filters are not used. When image enhancement filters are used, the prediction accuracy with and without OpenPose is calculated to be 81.38% and 78.35% respectively.

References

[1] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," 12 2018.

[2] M. C. Thar, K. Z. N. Winn, and N. Funabiki, "A proposal of yoga pose assessment method using pose detection for self-learning," in *2019 International Conference on Advanced Information Technologies (ICAIT)*, pp. 137–142, 2019.

[3] Y. Hirasawa, N. Gotoda, R. Kanda, K. Hirata, and R. Akagi, "Promotion system for home-based squat training using openpose," in *2020 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, pp. 984–986, 2020.

[4] F. Rishan, B. De Silva, S. Alawathugoda, S. Nijabdeen, L. Rupasinghe, and C. Liyanapathirana, "Infinity yoga tutor: Yoga posture detection and correction system," in *2020 5th International Conference on Information Technology Research (ICITR)*, pp. 1–6, 2020.

[5] S. Yadav, A. Singh, A. Gupta, and J. Raheja, "Real-time yoga recognition using deep learning," *Neural Computing and Applications*, vol. 31, pp. https://link.springer.com/article/10.1007/s00521–019, 12 2019.

[6] R. Huang, J. Wang, H. Lou, H. Lu, and B. Wang, "Miss yoga: A yoga assistant mobile application based on keypoint detection," in *2020 Digital Image Computing: Techniques and Applications (DICTA)*, pp. 1–3, 2020.

[7] D. Kumar and A. Sinha, "Yoga pose detection and classification using deep learning," *International Journal of Scientific Research in Computer Science Engineering and Information Technology*, 11 2020.

[8] Y. Agrawal, Y. Shah, and A. Sharma, "Implementation of machine learning technique for identification of yoga poses," in *2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT)*, pp. 40–43, 2020.

[9] J. Palanimeera and K. Ponmozhi, "Classification of yoga pose using machine learning techniques," *Materials Today: Proceedings*, vol. 37, 10 2020.

[10] J. Shin, M. A. Rahim, O. Yuichi, and Y. Tomioka, "Deep learning-based hand pose estimation from 2d image," in *2020 3rd IEEE International Conference on Knowledge Innovation and Invention (ICKII)*, pp. 108–110, 2020.

[11] G. Van Rossum and F. L. Drake Jr, *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.