# Aasna: Kinematic Yoga Posture Detection And Correction System Using CNN

*Ranjana* Jadhav, *Vaidehi* Ligde, *Rushikesh* Malpani, *Phinehas* Mane and *Soham* Borkar

Department of Artificial Intelligence and data science, Vishwakarma Institute of Technology, Pune, India, 411037.

**Abstract.** Yoga is a very popular form of exercise that originated in India that has numerous benefits for the mind and body. According to recent statistics, there are over 300 million yoga practitioners worldwide, with the number of yoga instructors increasing annually. However, incorrect yoga postures can lead to injuries and health complications. This highlights the importance of correct yoga posture and the need for a system that can detect and correct improper poses. This abstract presents a yoga posture detection and correction system designed using OpenCV for computer vision, and kinematic representation of the human body considering 17 points mapped on the human body, utilizing the tf-pose estimation algorithm for precise pose estimation. The system also includes a convolutional neural network (CNN) model developed using the Keras API and trained on the TensorFlow platform's MoveNet architecture for handling training of the model. The MoveNet pose estimation module has been used to detect keypoints of the human body which achieved an accuracy of 99.88%. The system works by live capturing of the yoga practitioner using a camera, extracting the key features of the pose, and comparing them with a trained data model of known yoga poses. If the pose is incorrect, the system provides real-time feedback to correct the pose.

**Keywords.** MoveNet, Pose detection, Yoga, kinematic representation, Yoga pose correction, OpenCV, Convolution Neural Network (CNN), TenserFlow, tf-pose estimation

## 1. Introduction

About 39% of adult humans on the earth are overweight. Exercise helps us maintain a healthy weight, a fit body, and a calm mind in addition to helping us lose body weight. Regular exercise also keeps us active and improves blood circulation [1]. Yoga originated in India many years ago as a form of mental and physical training. Sports and yoga have attracted people for countless years, but within the last ten years, a large population has adopted yoga as a way of life. The reason being health care. It's important to perform this exercise properly, especially in the appropriate posture. At some point, people start practicing yoga without the right instruction because they haven't received any help or information and don't know how to do it correctly. As a result of an incorrect position muscle pulls, stiff neck,

ankle sprain, etc. [2] can arise. Yoga should be practiced under the supervision of a trainer, but it is not for everyone. People nowadays use their cellphones to learn how to perform yoga postures and begin practicing them, but they are oblivious of the precision of their poses while they perform them. Numerous works [3] have been presented to address the constraints previously described.

Computer vision experts have difficulty detecting human stance. It has to do with how human joints are arranged so that a skeleton image or video is produced. Deep learning has substantially aided human pose estimation recently, and enormous performance improvements have been made.Deep learning approaches make it easier to map the structure than manually handling the links between structures. [4,5].

The proposed system is a yoga pose detection and correction system. It uses a web-based method to correct faulty yoga postures that people who practice yoga based on their own experience, while viewing yoga videos or using yoga software, utilize when utilizing their devices. Although there are several systems for detecting incorrect yoga postures, there are insufficient systems for correcting them. Moreover, this system consists of providing the user with visual indications in real-time, assisting the user in maintaining a perfect yoga posture throughout the exercise.

## 2. Literature Review

The accuracy of these systems varies depending on the method used and the dataset used for training. Most studies report accuracy rates ranging from 70% to 90%, with some studies reporting higher rates with the use of more advanced techniques. However, accuracy rates can be influenced by several factors, such as variations in lighting and clothing, the complexity of the pose, and individual differences in body shape and movement. The potential applications of yoga pose detection systems are numerous. Such equipment could provide real-time feedback to practitioners, helping them to correct their form and avoid harm. They could also be used in yoga classes to help instructors monitor their students' poses and provide personalized guidance. Additionally, yoga pose detection systems could be used for research purposes, enabling researchers to study the effects of different poses on the body and mind.

A technique to classify the sun salutation yoga positions was proposed by J. Palanimeera et al. [6] utilizing four classifiers i.e., Logistic regression, SVM, Naive Bayes, and KNN. Real-time skeletal drawings of a person's body are created using the pose estimation algorithm. The highest accuracy of 96% is produced by KNN classifiers. Kishore et al. in [7] proposed to make the job of such practitioners easier by employing deep learning-based techniques that can estimate a practitioner's precise posture. The study used five distinct deep learning architectures to achieve this method: EpipolarPose, OpenPose, PoseNet, MoveNet, and MediaPipe. These architectures were each trained separately using the image dataset. This database included five commonly used yoga positions. The MediaPipe architecture has the highest estimation accuracy, according to the study's assessment of each design's estimation accuracy. MoveNet was 12 times slower than OpenPose. MoveNet was the fastest of these five architectures. Rishan and colleagues presented a vision-based system for recognising yoga postures. The Infinity Yoga Tutor programme made advantage of the camera. [8] to monitor user motion. The system consists of two key modules: a pose estimation module that utilizes OpenPose to identify the 25 most essential body parts using the BODY 25 dataset, and a posture recognition module that evaluates and anticipates a user's stance or asana based on a sequence of frames. After training to detect six different asanas and utilizing OpenPose for posture prediction, the selected model had a 99.91% accuracy rate.Thar et al. proposed an open pose yoga assessment method. OpenPose extracts features

and detects poses in real time. To efficiently extract human parts and their relationships, part affinity fields and multi-convolutional neural network (CNN) architecture are utilized. The method makes use of camera footage. Then, it determines a user's body angles based on the specified difference between an instructor's and a student's perspective. If the user's error exceeds the specified threshold, it proposes correction. [9]. Bakshi et al. proposed a Deep Neural Network (DNN)-based technique for predicting human stance. The pose estimations are defined as a body joint-based DNN regression problem.There is a collection of DNN regressions that yield very accurate posture estimations. This technique makes use of the ability to reason about position holistically and includes a simple yet effective formulation that makes use of current breakthroughs in deep learning.[10]. Eichner et al. [11] suggested a novel approach to posture estimation. They address the issue in a novel situation known as Human stance Estimation (PCE), in which many people are in a similar but identifiable stance.PCE's task is to jointly estimate their poses and produce prototypes that approach the common stance. They demonstrated its benefits for analyzing postures in images of synchronized activities and learning prototypes of pose classes totally autonomously using a class-specific Google Images search.Because datasets are scarce and real-time posture detection is challenging, posture identification is a difficult task. To overcome this issue, Agrawal et al. gathered a huge dataset of at least 5500 photos of 10 different yoga poses and used the tf-pose-estimation approach to create the human body's skeleton in real time. The tf-pose skeleton is used to extract joint angles, which are subsequently employed in various machine learning models as features. The Random Forest classifier gave findings with an aftereffect after analyzing the dataset using several Machine Learning classification models. The precision is 99.04%.[12]. Kutalek et.al. presented the idea of a model that can identify yoga positions and show the user a number of frames. The objective of this research is to demonstrate that video frames from a yoga session can be identified and classified using a Convolutional Neural Network (CNN) model that is as basic as possible. A CNN model is then trained using the data, which consists of frames extracted from 162 video clips that were gathered based on the annotations. The OpenCV library is used to record the frames, the Keras API and TensorFlow platform handle the training, and the TensorBoard toolkit is used to display the results. When using the binary cross-entropy loss function and the sigmoid activation function, the Model's multi-class classification accuracy climbs to 91% [13]. Utkarsh and co. [14] created a system that can recognise the various yoga positions that users undertake. The programme makes use of open-source information that includes six distinct yoga positions shown by fifteen different volunteers. First, the system extracts data points from the video footage by using a media pipe pose estimation library. After preprocessing the information gathered, the system analyses and trains the data via classification-based machine learning algorithms. The Support Vector Machine (SVM) classifier, the Naive Bayes classifier, the Random forest classifier, Logistic regression, and the k Nearest Neighbours (KNN) classifier are among the machine learning techniques implemented. The algorithm is 94% accurate almost all of the time.

## 3. Methodology

The methodology for the yoga pose estimation system can be divided into segments: data collection, joint estimation, and pose correction.

Data Collection: The dataset [15] consists of about 2500 images of seven different yoga poses. The poses incorporated into the model were Chair (Utkatasana), Cobra (Bhujangasana), Dog (Adho Mukha Svanasana), Shoulder-stand (Salamba Sarvangasana), Triangle (Trikonasana), Tree (Vrikshasana) and Warrior (Virabhadrasana). All images in the dataset were resized to 300 x 300. We use a dataset of yoga poses to gauge how well our

suggested approach performs. We divided the dataset into training and testing sets at random, using 30% of the data for testing and 70% of the data for training. We give an analysis of the testing set's classification accuracy, which consists of pictures of people doing different yoga positions. We collected a dataset of images and live videos of people performing various yoga poses. We use a high-resolution camera to capture the videos from multiple angles and lighting conditions. We also collect annotations for each image and video, including the names of the yoga poses and the positions of key joints in the body.

Joint Estimation: The model has been created using python programming language. The Jupyter notebook IDE, a web based interactive computing platform has been used for developing the proposed model. We use a deep neural network-based architecture to estimate the joint positions of the user's body. Specifically, we use a modified version of the TensorFlow's pretrained MoveNet model, which is a popular pose estimation network used in various applications. The network takes as input a single RGB image or a video stream and outputs The position of 17 bodily joints, including the head, neck, shoulders, elbows, wrists, hips, knees, and ankles.

Include Health, a digital health firm, created the MoveNet inference model with the assistance of Google.The model is designed to enable remote patient therapy and is available in both web-based and mobile-based versions, which use TensorFlow.js and TensorFlow Lite respectively. MoveNet comes in two variants - Lightning, which prioritizes performance, and Thunder, which focuses on accuracy. MoveNet is a bottom-up model that is very accurate and quick, utilizing MobileNet V2 as a feature extractor and the TensorFlow object detection API. This API supports several detection models for TensorFlow 1 and TensorFlow 2, including MoveNet, which is based on the CenterNet detection model. CenterNet uses regional proposals to detect and classify objects, rather than the traditional anchor-based detection method that uses only the center point as an anchor. In addition, MoveNet takes into account the distance and angle between keypoints, as well as the orientation of keypoints, making it suitable for pose estimation. This feature allows MoveNet to accurately and quickly detect and categorize human poses, making it a valuable tool for digital health applications such as remote therapy.We also evaluate the system's robustness to different lighting conditions, camera angles, and clothing variations. We collect a separate dataset of the same yoga poses under varying lighting conditions, camera angles, and clothing variations. We report the classification accuracy on this dataset to evaluate the system's generalization ability.

Overall, the proposed system's evaluation provides insights into its accuracy, robustness, and generalization ability, which are essential for practical use.

The model was built using the following system configuration: an Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz (8 CPUs), 8192 MB of RAM, and a 64-bit operating system, specifically Windows 11.

### 3.1 Architecture

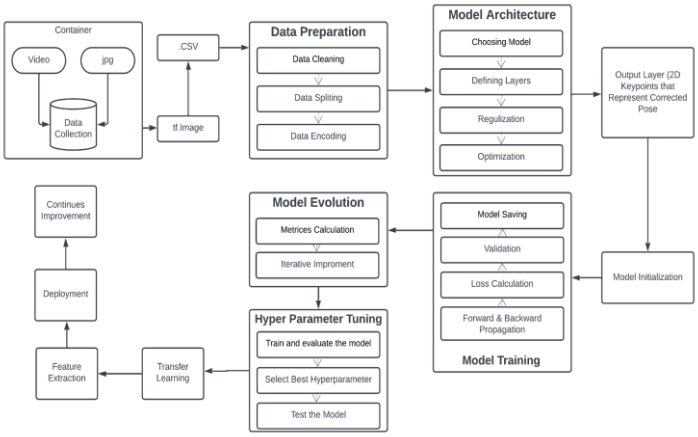


**Fig. 1.** Architecture of Pose Correction Model

*3.1.1 Data Collection:* Collect live video of people executing the yoga positions you want to correct. Check Dataset includes instances of both correct and wrong poses.

*3.1.2 Data Preparation:* Separate the films into individual frames, resize the frames, normalize the pixel values, and divide the dataset into training and testing sets.

*3.1.3 Model Architecture*: Select the best model architecture for your task. You might utilize Movenet, a lightweight and efficient deep learning network for human pose prediction. Movenet employs a single-person pose estimation architecture, with the purpose of estimating a person's 2D key points in a video frame.

*3.1.4 Model Training*: Using the training set, train the Movenet model. To avoid overfitting, employ approaches like data augmentation and regularization.

*3.1.5 Feature Extraction:* Fine-tune the Movenet model by layering a classification layer on top of the posture estimate output. Based on the position of the keypoints, the classification layer may predict whether the pose is correct or incorrect. Train the new model using the training set and evaluate its performance with the testing set.

*3.1.6 Hyperparameters:* Optimize the model's performance by adjusting hyperparameters such as learning rate, batch size, and number of layers.

*3.1.7 Deployment:* Use a user-friendly interface, such as a mobile app or website, to deploy the Movement-based paradigm. Users can upload videos of themselves performing yoga positions, and the model can provide criticism on their posture.

*3.1.8 Continuous Improvement:* Monitor the model's performance and gather user feedback to improve the model's accuracy and usability over time. You might also gather more data and retrain the model on a regular basis to enhance its accuracy over time.

Classification layer for Pose correction: The joint estimation process involves the following steps:

1. Input live video: The input to the system is a live video stream of the user performing a yoga pose.
2. Preprocessing: The input live video is preprocessed to convert it to image and resize it to a fixed resolution and normalize the pixel values.

3. Joint Estimation: The preprocessed live video is passed through the modified TensorFlow Pose network to estimate the positions of the body joints using kinematic representation. Scikit-learn (Sklearn) has been used toData should be divided into training and testing.

Pose Correction: TensorFlow Movenet is a deep learning model for yoga stance identification that uses a convolutional neural network (CNN) architecture. CNN takes an input image in .jpg format and produces a set of 17 keypoint coordinates that correspond to the location of body parts in the image in the .csv file format. These keypoint coordinates are then used to determine the yoga pose being performed.

### 3.2 Mathematical Equation

The mathematical equation for the TensorFlow Movenet model can be represented as:

$$Y = f(Wx + b) \quad (1)$$

Where Y is the model's output, which represents the keypoint coordinates of the body parts, x is the input picture, W and b are the CNN's weights and biases, and f is the activation function applied to the CNN's output.

The video input is initially processed through a succession of convolutional and pooling layers to extract information such as edge and shape detection from the video. These characteristics are subsequently sent through a sequence of MoveNet layers, which yield the output Y. The keypoint coordinates in Y are normalized between 0 and 1, where (0,0) represents the image's top left corner and (1,1) represents the image's bottom right corner, both of which will be 300 X 300 pixels. These coordinates are then used to determine the yoga pose correctness.

The proposed system uses mathematical equations that calculate the difference between the actual and correct pose and provide feedback on how to adjust the form to achieve the correct pose. Let X be the sequence of image frames of the practitioner's pose, and let Y_actual be the set of keypoints predicted by the pose detection algorithm for that pose. Let Y_correct be the set of key points for the correct version of the pose. Mathematically, the correction system can be represented as a function g that takes as input the predicted key points Y_actual and the correct key points Y_correct, and outputs a set of correction values C:

$$C = g(Y\_actual, Y\_correct) \quad (2)$$

The correction values C represent the difference between the predicted key points and the correct key points and can be used to provide feedback to the practitioner on how to adjust their form. The function g can be implemented using a mathematical operation such as subtraction or Euclidean distance:

$$C = ||Y\_correct - Y\_actual|| \quad (3)$$

where || || denotes the Euclidean distance.

The correction values are then used to provide feedback to the practitioner, such as by highlighting the areas that need to be adjusted in white color and highlighting the areas in green color which are correct. This feedback can be delivered in real-time using a web-based platform application.

Confidence Score: We calculated confidence score in our system which represents the level of certainty that the system has in its prediction of the practitioner's pose. It is a numerical value that indicates how closely the predicted key points match the key points of the correct

pose. We are using this confidence score to determine the correctness of practitioners. The confidence score for each of these key points indicates the level of certainty that the system has in its prediction of the location of that keypoint. The confidence scores can be obtained from the output of the model, which provides a set of heatmaps corresponding to each keypoint.

The confidence score is determined using the mean squared error (MSE) and/or the root mean squared error (RMSE) between the anticipated and correct keypoints. The MSE is defined as the average of the squared discrepancies between the expected and correct keypoints:

$$MSE = 1/N * \Sigma_i(Y\_correct,i - Y\_predicted,i)^2 \quad (4)$$

where N is the number of keypoints, Y_correct, i is the correct value of the ith keypoint, and Y_predicted,i is the predicted value of the ith keypoint.

$$RMSE = \sqrt{(MSE)} \quad (5)$$

A lower RMSE indicates a higher confidence score and a more accurate prediction of the practitioner's pose. The confidence score can be used to adjust the system's feedback based on the level of certainty in its prediction. Proposed system will give correct results if the confidence score we get is above 0.8.

### 3.3 Kinematic Representation

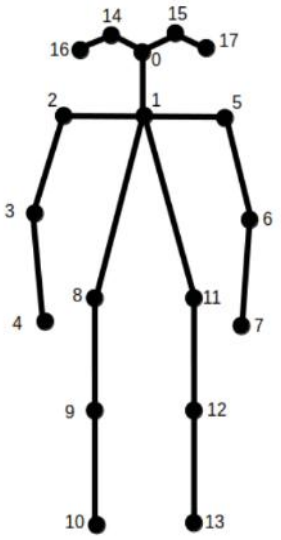


**Fig. 2.** Kinematic Representation of MoveNet Model

The MoveNet model uses 17 kinematic key points because these are the key points that are necessary to accurately estimate the pose of a human body. Kinematic representation is a way of representing the motion of objects using mathematical equations and diagrams.Kinematic representation is useful for predicting the motion of objects based on their initial conditions and the forces acting on them. It can also be used to analyze and optimize the motion of complex systems, such as robots or vehicles. Kinematic representation is often used in conjunction with other branches of physics, such as dynamics and control theory, to design and control systems that move with precision and accuracy.These kinematic keypoints are selected based on their importance in determining the orientation and position of the body.

These keypoints are used to estimate the orientation and position of the head, torso, arms, and legs, which are the main components of the human body. By focusing on these key points, MoveNet can accurately estimate the pose of a human body while keeping the model lightweight and efficient, making it suitable for real-time applications such as remote therapy.

An interface is necessary to make this system user-friendly. Thus, a website has been created using ReactJS. A popular open source front end javascript toolkit called ReactJS [21] is used to create user interfaces specifically for single page applications. The most crucial component of the application that minimizes page reloading and improves overall efficiency is the virtual DOM. The website uses JavaScript, which also gives access to NPM, a package management that makes it simpler to install external dependencies [22]. The model has been deployed on a react application using tensorflow.js library.

A help page is designed in order to assist the user in using the website. The website asks for the pose to be performed. The user needs to select a desired pose before starting the exercise. After selecting the pose, the website asks for permission to access the camera. Yoga poses are usually difficult to perform. In order to help the user in doing so, an image displaying the pose and a comprehensive text of instruction is displayed. The benefits of doing the selected yoga pose are also displayed in order to make the user aware of its benefits.

## 4. Results

As a result the user visits the website, the user is directed to the homepage. The instructions are shown to use the web application. The user can click on the Start button on the homepage to go to the pose description page. The user can choose a pose from the given list. After selection of a pose, the user can read the steps and click on the Start Pose button. The camera will turn on and start detecting the critical areas of the user's body as shown in Fig. 3.

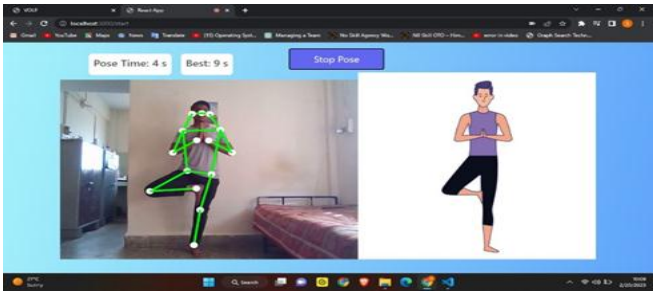


**Fig. 3.** Real-time Yoga Activity Page

The system detects the 17 key points over the user’s body, performing Tree pose as shown in Fig. 7. If the user holds the correct posture the skeleton displayed over the user's body turns into green as shown in fig 4(a) and if the user is performing inappropriately the skeleton turns into white as shown in fig 4(b). The system beeps and keeps track of the time for which the user holds the correct position.

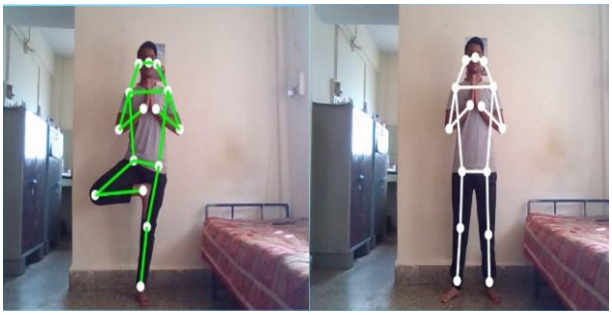

**Fig. 4 (a)** Correct posture **Fig 4. (b)** Incorrect posture

The training and testing accuracy of the proposed model comes out to be 99.29% and 99.88% respectively. The training and testing loss is 0.017 and 0.0066 respectively using 200 epochs and 16 batch size while training the model.

| Model | Year | Dataset | Accuracy | Speed | Resource Consumption | Application |
|---|---|---|---|---|---|---|
| TenserFlow Movenet | 2020 | COCOMPII | 97.2% | Real-time | Low | Fitness tracking,human-computer Interaction |
| Openpose | 2019 | COCOMPII | 91% | Real-time | High | Human pose estimation in images and video |
| AlphaPose | 2018 | COCOMPII | Not Specified | Real-time | High | Human pose estimation in images and video |
| Efficient Pose | 2021 | COCOMPII | 33% | Real-time | Low | Human pose estimation in images and video |

**Fig. 5.** Comparative analysis of models

A lower loss indicates a better model. The model's performance for training and validation sets is determined using the training and validation data. Loss is not a percentage, in contrast to accuracy. It represents the total number of mistakes committed for every example in the training or validation sets.

TensorFlow Movenet is a lightweight and popular model that outperforms other models for real-time yoga position analysis on mobile devices. Its high keypoint recognition accuracy allows it to recognise the intricacies of yoga positions even with differences in body form and clothes. Movenet is designed for real-time performance, giving practitioners quick feedback throughout sessions to help them prevent injuries and improve their form. Movenet, being an open-source model, allows developers to freely modify and utilize it, resulting in the production of numerous applications and tools for yoga position analysis.

## 5. Conclusion

In Conclusion, the yoga pose correction system developed with the TF-position estimation model and the CNN algorithm shows excellent results for correcting yoga practitioners' posture and reducing harm. The ability of the model to recognise and correct incorrect postures in real time has important implications for improving the effectiveness of yoga posture. The integration of the TF-position estimation model with the CNN algorithm allows

for effective posture correction, allowing the system to deliver real time feedback to the user. In the future, the diversity of the dataset can be expanded by introducing many other yoga poses. A mobile application can be developed for greater usability. A database can be maintained to keep track of the daily yoga activity of the user. This can be used to provide insights to the user.

## 6. References

[1] Anilkumar, Ardra, Athulya KT, Sarath Sajan, and Sreeja KA. "Pose Estimated Yoga Monitoring System." *Available at SSRN 3882498* (2021).

[2] Jose, Josvin, and S. Shailesh. "Yoga asana identification: a deep learning approach." In *IOP Conference Series: Materials Science and Engineering*, vol. 1110, no. **1**, p. 012002. IOP Publishing,( 2021).

[3] Nagalakshmi Vallabhaneni, Dr P. Prabhavathy. "The analysis of the impact of yoga on healthcare and conventional strategies for human pose recognition." *Turkish Journal of Computer and Mathematics Education (TURNCOAT)* 12, no. **6** (2021): 1772-1783.

[4] Kothari, Shruti. "Yoga Pose Classification Using Deep Learning." (2020).

[5] Chiddarwar, Girija Gireesh, Abhishek Ranjane, Mugdha Chindhe, Rachana Deodhar, and Palash Gangamwar. "AI-based yoga pose estimation for android application." *Int J Inn Scien Res Tech* 5 (2020): 1070-1073.

[6] Palanimeera, J., and K. Ponmozhi. "Classification of yoga pose using machine learning techniques." *Materials Today: Proceedings* 37 (2021): 2930-2933.

[7] Kishore, D. Mohan, S. Bindu, and Nandi Krishnamurthy Manjunath. "Estimation of yoga postures using machine learning techniques." *International Journal of Yoga* 15, no**. 2** (2022): 137.

[8] Rishan, Fazil, Binali De Silva, Sasmini Alawathugoda, Shakeel Nijabdeen, Lakmal Rupasinghe, and Chethana Liyanapathirana. "Infinity yoga tutor: Yoga posture detection and correction system." In *2020 5th International Conference on Information Technology Research (ICITR)*, pp. **1-6.** IEEE,( 2020).

[9] Thar, Maybel Chan, Khine Zar Ne Winn, and Nobuo 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. IEEE,( 2019).

[10] Bakshi, Aarti, Danish Sheikh, Yasmin Ansari, Chetan Sharma, and Harishchandra Naik. "Pose estimate based yoga instructor." International Journal of Recent Advances in Multidisciplinary Topics 2, no**. 2** (2021): 70-73.

[11] Eichner, Marcin, and Vittorio Ferrari. "Human pose co-estimation and applications." IEEE transactions on pattern analysis and machine intelligence 34, no. **11** (2012): 2282-2288.

[12] Agrawal, Yash, Yash Shah, and Abhishek Sharma. "Implementation of machine learning techniques for identification of yoga poses." In *2020 IEEE 9th international conference on communication systems and network technologies (CSNT)*, pp. 40-43. IEEE, (2020).

[13] Kutálek, Jirı, and K. Kutálek. "Detection of Yoga Poses in Image and Video." *Brno Faculty University of Information and Technology* (2021).

[14] Bahukhandi, Utkarsh, and Shikha Gupta. "Yoga pose detection and classification using machine learning techniques." *Int Res J Mod Eng Technol Sci* 3, no. **12** (2021).

[15] https://www.kaggle.com/datasets/amanupadhyay/yoga-poses

[16] Next-Generation Pose Detection with MoveNet and TensorFlow.js. https://blog.tensorflow.org/2021/05/next-generation-pose-detection-with-movenet-and-tensorflowjs.html, accessed on( Nov. 30, 2022).

[17] Pose estimation and classification on edge devices with MoveNet and TensorFlow Lite. https://blog.tensorflow.org/2021/08/pose-estimation-and-classification-on-edge-devices-with-MoveNet-and-TensorFlow-Lite.html, accessed on( Nov. 30, 2022).
[18] Jo, BeomJun, and SeongKi Kim. "Comparative Analysis of OpenPose, PoseNet, and MoveNet Models for Pose Estimation in Mobile Devices." *Traitement du Signal* 39, no. **1** (2022): 119-124.
[19] Zhou, Xingyi, Dequan Wang, and Philipp Krähenbühl. "Objects as points." *arXiv preprint arXiv:1904.07850* (2019).
[20] Human Pose Classification with MoveNet and TensorFlow Lite. https://www.tensorflow.org/lite/tutorials/pose_classification, accessed on (Dec. 01, 2022).
[21] Rawat, Prateek, and Archana N. Mahajan. "ReactJS: A Modern Web Development Framework." *International Journal of Innovative Science and Research Technology* 5, no**. 11** (2020).
[22] Bhupati Venkat Sai Indla | Yogeshchandra Puranik "Review on React JS" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470,Volume**-5** | (Issue-4,June 2021), pp.1137-1139,URL:www.ijtsrd.com/papers/ijtsrd42490.pdf