Yoga Pose Detection using Keras Neural Network

Malavika A
Department of Computer Applications
Amal Jyothi College of Engineering
Kanjirappally, India
malavikaa2023b@mca.ajce.in

Anit James
Department of Computer Applications
Amal Jyothi College of Engineering
Kanjirappally, India
anitjames@amaljyothi.ac.in

Abstract— **Our project aims to create an intricate deep learning system that can accurately detect human poses. We plan on utilizing various technologies such as Mediapipe, OpenCV, TensorFlow, Keras, and NumPy to develop this model. The project focuses on detecting human body poses accurately, particularly for the purpose of yoga. The project begins with the implementation of the Mediapipe pose detection library to capture the human body pose. Next, a simple Dense network model is developed using Keras, which is trained on the data obtained from the Mediapipe pose detection library. The training process involves feeding the model with input data and corresponding output labels to optimize the model for better predictions. Once the model is trained, an inference file is created to use the model for predicting the human body pose.**

Keywords- Pose Detection, Mediapipe, OpenCv, TensorFlow, Keras, Numpy, Deep Learning, Dense Network, Yoga

I. INTRODUCTION

Pose identification is a fundamental computer vision job with wide applications in disciplines including physical therapy, dance, and sports training. Deep learning algorithms have recently made significant strides, enabling the creation of robust posture detection models that can instantly anticipate human body poses with accuracy. In this study, we describe a unique method for pose detection that combines the Mediapipe and OpenCV libraries with the Keras neural network methodology.

The field of posture discovery has experienced momentous advance inside the past few a long time, all much appreciated to the development of modern profound learning techniques. Different libraries and systems have been created to encourage the method of posture location, such as Mediapipe, OpenCV, TensorFlow, Keras, and NumPy. These libraries give capable devices and calculations for analyzing and preparing picture and video information, empowering the improvement of exact and proficient posture discovery models. In this method, the human body pose is extracted from video data using the Mediapipe pose detection library.

A Keras neural network model, trained to precisely anticipate the position of the human body, is then given the collected data. The Keras neural network model is designed using a Dense network architecture, which is the type of feedforward neural network which consists of fully connected layers. Yoga, which involves various poses and movements, is a challenging domain for pose detection. This system demonstrates the effectiveness of our developed pose detection model in accurately predicting yoga poses, which has various applications in fields such as sports training and physical therapy. Our project strives to contribute towards the continuous development of pose identification through the utilization of deep learning techniques. The primary objective is to facilitate the advancement of state-of-the-art technologies in this field.

A. Nature of Keras Neural Network

Keras offers a convenient platform for developers to efficiently test and refine diverse configurations of neural network architectures. The software furnishes a vast selection of predefined layers that can be combined to construct intricate models, encompassing convolutional, recurrent, and dense layers. Keras comprises an array of loss functions, optimizers, and metrics that can be employed for model training and evaluation.

II. KERAS NEURAL NETWORKS

The Keras framework serves as an open-source infrastructure for the deployment of deep learning procedures by relying on the programming language of Python. The software offers a straightforward and intuitive interface that facilitates the construction and refinement of artificial neural networks. The Keras framework exhibits high flexibility and accommodates the integration of diverse backends, comprising prominent platforms such as TensorFlow, Microsoft Cognitive Toolkit, Theano, and PlaidML among others.

By inserting layers to a neural network model made with Keras, you may specify the model's architecture. For various neural network topologies, Keras offers a variety of built-in layers, including dense (completely connected) layers, convolutional layers, recurrent layers, and others. You can also customize the layers by specifying various parameters such as the activation functions, regularization, and dropout. All things considered, Keras is a well-liked option for developing deep learning models because to its simplicity, adaptability, and interoperability with a variety of hardware and software platforms.

With its ability to facilitate the development and training of neural network models through minimal lines of code, Keras confers upon users an extensive range of pre-existing

DOI: 10.5281/zenodo.7955615
ISBN: 978-93-5906-046-0@2023, Dept. of Computer Applications, Amal Jyothi College of Engineering Kanjirappally, Kottayam

models and instruments to exert control over diverse types and formats of data. The aforementioned system also endorses both central processing unit (CPU) and graphics processing unit (GPU) computations, rendering it adaptable and proficient for the instruction of substantial deep learning models. Keras is currently one of the most prevalent deep learning libraries utilised in the field.

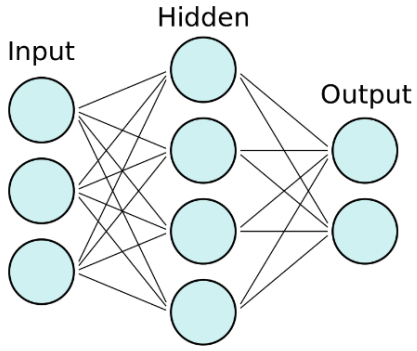

Fig 1

An internal neural network of Keras is commonly made up of three different types of layers, as shown in the previous figure [Fig. 1].

Input Layer:

The input layer of a neural network represents the primary layer which receives input data as its initial step of processing. The utilization of the Input () function in Keras is intended for specifying the shape and data type of the input data during the creation of the input layer. The input layer functions as a conduit for transmitting data to the succeeding layer without undergoing any computational processes on it.

Hidden Layer:

The layer(s) between the input and output layers is/are the hidden layer(s). The incoming data is processed by these layers, which result in a more meaningful representation that may be used to create predictions. Dense, Conv2D, LSTM, and other hidden layer types are among those offered by Keras. Different types of layers run various computations on the input data.

Output Layer:

The output layer in a neural network is the last layer that generates the model's output. To build the layer in the Keras framework, the function Dense() is utilized. Its primary functions consist of specifying the number of output neurons and determining the appropriate activation function to be employed. To exemplify in a scenario of binary classification, the layer of output would comprise exclusively a solitary neuron featuring a sigmoid activation function. On the other hand, for a multiclass classification problem, the output layer would have multiple neurons with a softmax activation function.

Overall, stacking these layers to create a sequential model defines the architecture of a neural network in Keras. The individual problem being solved dictates the number and type of layers that are employed as well as the hyperparameters (such as the quantity of neurons, activation functions, and learning rate).

III. LITERATURE REVIEW

Debabrata Swain, Santosh Satapathy, Pramoda Patro, Aditya Kumar Sahu et al. [1], in their paper titled “Yoga Pose Monitoring System using Deep Learning continuously monitors person’s activity and movement. The acknowledgment of pose has the potential to form a personalized and independent hone, empowering people to ace yoga postures with accuracy and without the required for outside direction. Fifteen people have executed six postures of yoga in a entirety of 85 video recordings, making up the handpicked dataset. The primary step includes the extraction of the user's keypoints through the utilization of the Mediapipe library. The distinguishing proof of Yoga postures has been made conceivable through the application of a profound learning demonstrate that utilizes a combination of long short-term memory (LSTM) and convolutional neural organize (CNN). This cutting-edge innovation is competent of recognizing Yoga stances in real-time checked recordings. The CNN layer is utilized to extricate highlights from keypoints, and it is taken after by LSTM, which perceives when a arrangement of outlines happens so that expectations may be shaped. The postures are at that point categorised as adjust or inappropriate. The system will give text/speech criticism to the client in the event that the proper position is found.

Debabrata Swain, Santosh Satapathy, Biswaranjan Acharya, Madhu Shukla, Vassilis C. Gerogiannis, Andreas Kanavos and Dimitris Giakovis et al. [2], in their paper titled "Deep Learning Models for Yoga Pose Monitoring takes video sequence frames as input in real time. During the keypoints extraction phase, the system detects and extracts important keypoints based on the user's position. The pose prediction phase utilizes a hybrid model that combines Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM). The CNN is a multilayered Artificial Neural Network (ANN) that recognizes objects and classifies images, while the LSTM is useful for understanding the sequence of frames occurring in a specific yoga pose. Furthermore, the final phase of the system is a pose correction. The user receives feedback to correct their pose and determine the similarity percentage compared to the actual pose. A Dense layer is employed as the final layer using Softmax as the activation function which assigns probabilities of different poses based on the current given input. The output obtained from this layer is polled on 45 frames to obtain the final prediction. To optimize the model's performance, the Adam optimizer with a learning rate equal to 0.0001 is utilized. This optimizer makes a difference the show to merge rapidly by including force and scaling terms as demonstrated in Condition. The study gives intriguing bits of knowledge into how profound learning models can be connected to screen yoga postures in real-time, advertising input to clients for more compelling hone sessions.

*DOI:*10.5281/zenodo.7955615
ISBN: 978-93-5906-046-0@2023, Dept. of Computer Applications, Amal Jyothi College of Engineering Kanjirappally, Kottayam

Sadeka Haque, AKM Shahariar Azad Rabby, Monira Akter Laboni, NAfis Neehal, and Syed Akhter Hossain et.al [3], in their paper titled "ExNET: Deep Neural Network for Exercise Pose Detection" conveys that the way humans interact with machines has completely changed because of deep learning, especially in the area of computer vision. By using this technique, it is now possible to precisely identify human workout stances in photographs. In order to do this objective, a Deep Neural Network model called ExNET was developed. The ExNET model was evaluated on a dataset comprising 2000 images, categorized into 5 different classes. The images were then divided into training and test datasets to train and evaluate the model's performance. The model's accuracy was significantly improved, and several experiments were conducted on the test dataset to assess the model's performance accurately. The ExNET model used a combination of CNNs and other Deep Learning techniques, which allowed it to achieve high accuracy in exercise pose detection. Several techniques were used to improve the model's accuracy, including the use of convolutional neural networks (CNNs), which are particularly effective in image classification tasks because they can learn the features and patterns of images through various layers of filters.

Vivek Anand Thoutam, Anugrah Srivastava, Tapas Badal, Vipul Kumar Mishra, G.R.Sinha, Aditi Sakalle, Harshit Bharadwaj and Manish Raj et.al [4], in their paper titled "Yoga Pose Estimation and Feedback Generation Using Deep Learning" conveys identification of appropriate yoga poses and offer feedback to help the yoga posture improve. For this, a deep learning-based methodology for estimating yoga poses is proposed in this study. Many important points are being detected with varying degrees of confidence when detecting poses for a person. The manner that Keras posture estimation operates allows it to include the first important point discovered without taking into account confidence intervals. In order to take important areas of greatest confidence levels into consideration, a few adjustments were made to the Keras pose estimation in this article. In order to train models to have high accuracy, the study used these x and y coordinates to extract information such angles between body joints and with the ground. These situations are given top priority so that no abnormal data is provided as input.

Rutuja Jagtap, Monali Zanzane, Rutuja Patil et al. [5], in their paper titled "Yoga Pose Detection Using Machine Learning" conveys a yoga pose detection using machine learning to identify and detect yoga pose form. The system works on 8 yoga poses (Vajrasan, Shavasaan, Gomukhasan, Bhadraasan, Dhanuraasan, Shrishasan, Sarvangasan and Chakrasan). It is developed using a GUI-based desktop application using Tkinter library. The input is preprocessed in the form of image, the object is detected, and the core of the human body is identified using the mediapipe and OpenCV library. For training and testing data, csv files are downloaded from Kaggle. Logistic regression models for training and testing are used. The system gets an accuracy score of 100%.

Rutuja Gajbhiye, Snehal Jarag, Pooja Gaikwad, Shweta Koparde et al. [6], in their paper titled "AI Human Pose Estimation: Yoga Pose Detection and Correction" recognizes the yoga poses portrayed by the user. The approach starts by passing an image through a CNN classifier trained to look for people. When the poses are detected, the pose estimation network searches for trained joints and limbs. The computer can then display the image to the user using markers that identify parts of the body. The system utilizes live video feeds from webcams to track the user's movements, with the goal of detecting and correcting yoga poses. To achieve this, a machine learning model called PostNet is employed for constant posture detection. The user first positions themselves in front of the camera and then assumes a yoga pose. The system captures the user's pose and identifies key points, which are visualized on the video canvas. These key points are then used to compare the user's pose with the target yoga pose to determine if any corrections are necessary. If the two poses have a high level of similarity, the user's pose is considered to be perfect.

Josvin Jose, Shailesh S et.al [7], in their paper titled "Yoga Asana Identification: A Deep Learning Approach" introduces a system that employs deep learning techniques, including convolutional neural networks (CNN) and transfer learning, to recognize yoga postures from images or frames of videos. The model was trained and evaluated using images of 10 different asanas, and the transfer learning-based prediction model demonstrated an impressive 85% accuracy rate. This system represents an early step towards developing an automated tool for analyzing yoga images and videos. To activate the output of all dense layers, the ReLU activation function is applied. In the final dense layer, the Softmax function is utilized to normalize the network output into a probability distribution for the predicted output classes. The network is compiled using the ADAM optimizer and the classification cross-entropy as the loss function.

Utkarsh Bahukhandi, Dr. Shikha Gupta et.al [8], in their paper titled "Yoga Pose Detection and Classification using Machine Learning techniques" conveys a pose detection methodology by firstly creating an environment and installing necessary libraries such as NumPy, pandas, seaborn, sklearn, OpenCV, and Mediapipe, open-source video data from Kaggle featuring 6 yoga poses performed by 15 volunteers (both male and female) is collected. The frames of these videos are then processed by Mediapipe, which identifies 33 key points on each frame and provides their corresponding 3D coordinates and visibility values, forming a new dataset. Data preprocessing techniques, such as normalization and feature engineering, are applied to the dataset to prepare it for machine learning models. This includes converting key points to vectors and calculating joint angles. Finally, the processed data is fed into classification-based machine learning algorithms for training.

IV. PROPOSED METHODOLOGY

We implemented a system that captures the user's pose using a webcam and processes the data in real-time. We

*DOI:*10.5281/zenodo.7955615
ISBN: 978-93-5906-046-0@2023, Dept. of Computer Applications, Amal Jyothi College of Engineering Kanjirappally, Kottayam

utilized the OpenCV library for image processing and the MediaPipe Pose estimation API for pose detection. The seminar implements a system for capturing yoga poses and classifying them using machine learning techniques. The system comprises of three major functions like the "capture_pose","train_model", and "inf". The "capture_pose" function captures a yoga pose from a user and saves it to the database. The "train_model" function uses the saved data to train a deep learning model, while the "inf" function uses the trained model to classify a new yoga pose captured by the user. The name of the specific asana and the recording of how that pose has to be done is stored into the database [Fig 2.1].

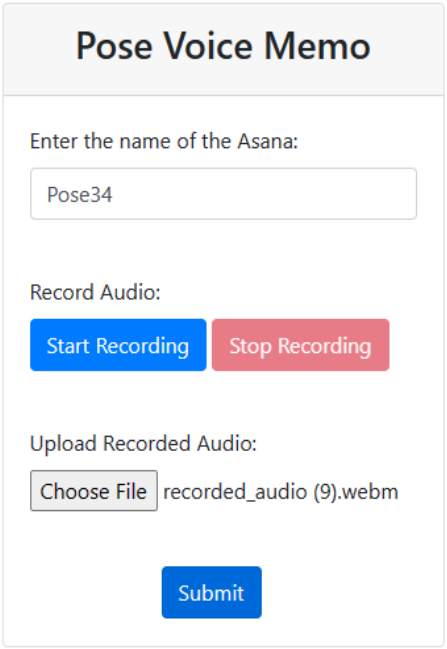

Fig 2

	id [PK] bigint	name character varying (100)	data jsonb	instructor_id integer	audio_file character varying (100)
1	1	Pose 21	[[0.0, 0.0, ...	2	audio/recorded_audio_...
2	2	pose 44	[[0.0, 0.0, ...	2	audio/recorded_audio_...
3	3	Pose 55	[[0.0, 0.0, ...	2	audio/recorded_audio_...
4	4	Pose 34	[[0.0, 0.0, ...	2	audio/recorded_audio_...

Fig 2.1

The first section [Fig 3], "capture_pose", captures the yoga pose by using a webcam [4] to record the user's body movements [3]. The function uses the MediaPipe library for pose estimation, which provides the coordinates of 33 key points on the user's body [1]. The function then checks if the user's entire body is visible in the frame [6] by checking the visibility scores of four key points: the two ankles, the center of the chest, and the center of the hips. If the user is fully visible, the function saves the pose data to a numpy file and also saves it to the database [Fig 2.1] along with the instructor associated with the current user.

Fig 3

The next section, "train_model" [Fig 4], trains a deep learning model using the saved pose data. The function collects the pose data and corresponding labels from the database and converts the labels to categorical format. The function then splits the data into training and validation sets, creates a neural network model using the Keras library [4], and trains the model on the training data. The model uses the RMSprop optimizer and categorical cross-entropy loss function.

Once training is complete, the function saves the model and the labels to disk. RMSprop, short for Root Mean Square Propagation, is an optimization algorithm commonly used in neural networks to adapt the learning rate during training. When it comes to tackling multi-class classification problems, the go-to loss function is none other than "categorical_crossentropy". This particular function has earned a reputation for being the standard choice in the field. It measures the difference between predicted and actual probabilities of each class and tries to minimize the difference between them.

In the realm of neural networks, the softmax function [7] is often employed in the output layer. This particular function functions by taking in a vector of actual numbers and subsequently generating a probability distribution across these numbers. The tanh activation function is used twice, in two different Dense layers of the neural network model. The first Dense layer with 128 nodes contains where the Tanh activation function is used. The input to this layer is the data stored in the ip variable, which is passed through the tanh function before being fed to the next layer. The second use of the tanh activation function is in the second Dense layer with 64 nodes.

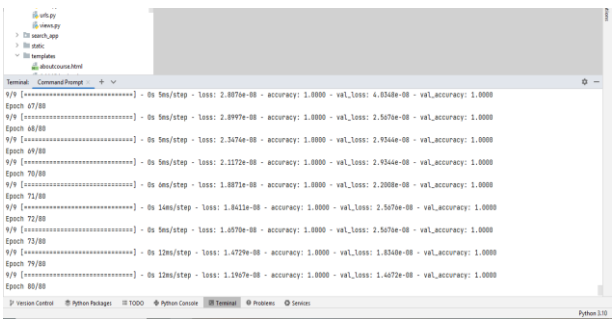

Fig 4

In the Figure [Fig 5], the pose that was previously collected and trained using Keras is shown in the dashboard and the user can try out the poses in the database [Fig 2.1].

*DOI:*10.5281/zenodo.7955615
ISBN: 978-93-5906-046-0@2023, Dept. of Computer Applications, Amal Jyothi College of Engineering Kanjirappally, Kottayam

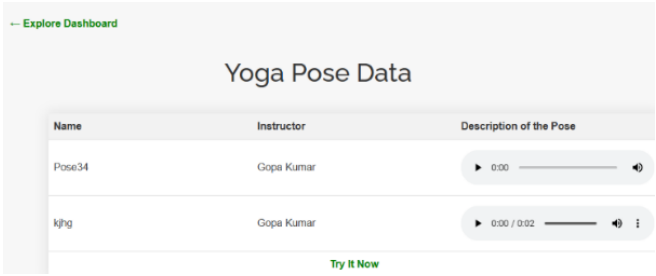

Fig 5

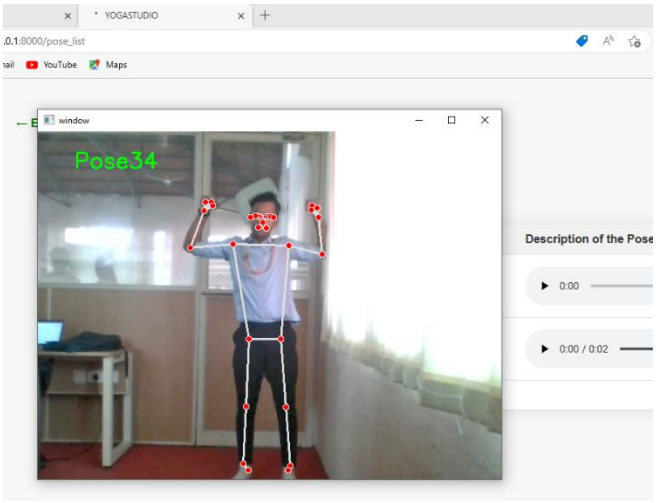

Fig 6

The final section, "inf" [Fig 6], uses the trained model to classify a new yoga pose captured by the user. The function uses the same pose estimation process as the "capture_pose" function to obtain the key points on the user's body. It then normalizes the key point [8] data and feeds it into the trained model to obtain a predicted label. The function also displays the predicted label on the webcam feed for the user to see.

In general, the present system offers a facile and efficacious means for users to record and categorize yoga postures by utilizing machine learning methodology. The present system has the capability to be expanded in order to incorporate supplementary functionalities, such as providing live feedback to the user concerning their form, personalized suggestions for enhancing their poses, and integrating with other apps and services related to yoga. Subsequent studies could further consider the potential utilization of alternate sensor modalities and input mechanisms to achieve more precise and comprehensive pose estimation.

V. CONCLUSION

In conclusion, we proposed a computer vision-based approach to recognize yoga poses using pose estimation techniques. We implemented a system that captures the user's pose using a webcam, processes the data, and recognizes the pose in real-time. We utilized the OpenCV library for image processing and the Mediapipe Pose estimation API for pose detection. We trained a deep learning model using the collected pose data to classify the yoga poses accurately. Our experimental results demonstrate the feasibility and accuracy of the proposed approach for recognizing yoga poses. The proposed system can be used by yoga instructors to monitor their students and provide feedback on their poses, thereby improving their overall health and well-being.

VI. REFERENCES

[1] Debabrata Swain, Santosh Satapathy, Pramoda Patro, Aditya Kumar Sahu Yoga Pose Monitoring System using Deep Learning Published:June 27th, 2022

[2] Debabrata Swain, Santosh Satapathy, Biswarajan Acharya, Madhu Shukla, Vassilis C. Gerogiannis, Andreas Kanavos and Dimitris Giakovis Deep Learning Models for Yoga Pose Monitoring Published: 31 October 2022

[3] Sadeka Haque, AKM Shahariar Azad Rabby, Monira Akter Laboni, NAfis Neehal, and Syed Akhter Hossain ExNET: Deep Neural Network for Exercise Pose Detection Published: 21 July, 2019

[4] Vivek Anand Thoutam, Anugrah Srivastava, Tapas Badal, Vipul Kumar Mishra, G.R.Sinha, Aditi Sakalle, Harshit Bharadwaj and Manish Raj Yoga Pose Estimation and Feedback Generation Using Deep Learning Published: 24 March, 2022

[5] Rutuja Jagtap, Monali Zanzane, Rutuja Patil Yoga Pose Detection Using Machine Learning Published: 05 May, 2022

[6] Rutuja Gajbhiye, Snehal Jarag, Pooja Gaikwad, Shweta Koparde AI Human Pose Estimation: Yoga Pose Detection and Correction Published: 05 May, 2022

[7] Josvin Jose, Shailesh S Yoga Asana Identification: A Deep Learning Approach Pulished: 16 February, 2021

[8] Utkarsh Bahukhandi, Dr. Shikha Gupta Yoga Pose Detection and Classification using Machine Learning techniques Published: 12 December, 2021.

*DOI:*10.5281/zenodo.7955615
ISBN: 978-93-5906-046-0@2023, Dept. of Computer Applications, Amal Jyothi College of Engineering Kanjirappally, Kottayam