e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

INTELLIGENT YOGA POSE RECOGNITION SYSTEM USING DEEP LEARNING FOR ENHANCED PRECISION

Praphullakumar Vilasrao Lokhande*1, Prof. P.R. Gadekar*2

*1.2Smt. Kashibai Navale College Of Engineering, Vadgaon, Pune, India.

Affiliated By Savitribai Phule Pune University, India.

DOI : https://www.doi.org/10.56726/IRJMETS63941

ABSTRACT

An intelligent system is being developed for the real-time recognition and assessment of yoga poses, utilizing advanced deep learning algorithms. This system will integrate pose detection with a scoring mechanism to offer valuable feedback on alignment and posture accuracy, enhancing the effectiveness of yoga practice. By employing a hybrid machine learning approach that combines linear regression for feature extraction and real-time video processing, the system will provide robust and adaptable pose recognition across a variety of yoga poses and environments. The application aims to assist both yoga practitioners and instructors by delivering precise, accessible pose analysis, supporting self-learning and continuous improvement in practice.

Keywords: Future Yoga Recognition System, Pose Alignment Feedback, Hybrid Machine Learning, Yoga Posture Analysis, Deep Learning In Yoga.

I. INTRODUCTION

In recent years, yoga has become increasingly popular worldwide as a comprehensive practice promoting both physical and mental health. However, assessing one's yoga postures accurately can be difficult without professional guidance. To solve this issue, we propose a deep learning-based system that offers real-time feedback on yoga poses through pose recognition and evaluation. The system is designed to be user-friendly, operating with standard cameras while providing insights into posture alignment and accuracy. This project utilizes advanced algorithms and pose estimation techniques to capture and analyze yoga poses. With its high level of accuracy in pose detection, this system can significantly enhance practitioners' self-guided practice and help prevent injuries caused by misaligned postures.

II. METHODOLOGY

In health and fitness applications, yoga pose identification is essential since it gives real-time feedback on how perfect a stance is. It is becoming more possible to detect and classify yoga poses from live camera feeds using computer vision techniques and machine learning models. This research provides a method for yoga stance recognition based on a straightforward machine learning algorithm called logistic regression. Real-time camera input is processed by the system, which then compares it to a pre-labeled CSV data-set of yoga positions. Real-time prediction and classification of the executed yoga pose is the goal in order to provide feedback for correcting posture. The system can be represented as S = I, P, O, where: I represents the input, P represents the procedure, and O represents the output.

Input (I):

The input to the system is a live camera feed, where a CSV data-set is used to train and guide the pose detection process.

Procedure (P):

The procedure involves using the live camera feed (I) to capture data, upon which the system performs operations and calculates predictions based on the data-set. The system processes the input and performs pose recognition using the trained model.

Output (O):

The output of the system is the detection of yoga poses through the live camera feed, providing real-time feedback on the practitioner's pose alignment and accuracy.

Input The device receives its input from a live video feed that shows the subject doing yoga positions. Steps that take place during input acquisition are as follows:

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

Live Camera Setup: The subject's full body is captured by the camera while they perform different yoga poses. It must guarantee that vital bodily parts, such limbs and joints, are visible. Frame extraction involves separating the continuously streaming live video feed into separate frames, each of which is then processed to determine a person's posture.

Keypoint Detection: To identify the body keypoints from the frames, pose estimation methods like OpenPose, MediaPipe, or HRNet are used. The coordinates of important joints, including the knees, ankles, elbows, and shoulders, are represented by these keypoints.

Collection (CSV Format): The machine learning model is based on a pre-labeled CSV data-set. The collection includes: Pose Name: The name given to the class or label in yoga.

Body Key-points: The major body joints' (x, y) coordinates that were taken from the dataset's reference poses. Originated Characteristics: Characteristics that can aid in differentiating between various poses include joint angles and key point distances. The Logistic Regression model, which will be used to categorize the yoga positions, needs to be trained on this data-set.

Method: The method, which entails feature extraction, data Preprocessing, and classification using logistic regression, is the central component of the system.

Pre-processing: To guarantee uniformity, the live camera feed's frames are prepossessed. The data is cleaned to eliminate noise and false positives, and the keypoints coordinates are scaled to match a standard frame size.

Feature extraction: involves creating a set of features based 4 Authorized licensed use limited to: AISSMS's Institute of Info Technology - PUNE. Downloaded on June 10,2024 at 08:13:08 UTC from IEEE Xplore. Restrictions apply. on the keypoints found in the live feed. The (x, y) coordinates of the major body joints are among these properties, and they may also contain derived metrics like joint angles or distances. The body's posture in each frame is quantitatively described by these properties.

Model of Logistic Regression: Based on the variables that were retrieved, a linear classifier called logistic regression is employed to predict the yoga posture. The model presupposes a link between the likelihood of each pose class and the input features (joint positions and derived metrics). The following steps are involved in the process:

Training: The CSV dataset, which contains labeled poses, is used to train the model. A sigmoid function is used in logistic regression to describe the likelihood that a given pose falls into a particular class.

Classification of Multiple Classes: While logistic regression is by nature binary, techniques such as One-vsRest (OvR) allow it to be applied to multiclass issues, including pose categorization. In OvR, the model compares each class to every other class in order to train a distinct classifier for each class. The class most likely to is selected as the expected

Position Forecast: The system gathers information from the live video feed and feeds them into the Logistic Regression model in real-time. The detected yoga position is determined by taking the class with the highest probability out of all the pose classes that the model predicts.

Mechanism of Feedback: The user receives immediate feedback from the system after a pose is anticipated. The feedback comprises the anticipated pose name and maybe a graphical overlay displaying the keypoints that have been recognized on the live video feed. Pose Classification Using Logistic Regression The rationale behind selecting Logistic Regression for realtime applications is its computing efficiency, interpretability, and simplicity. Its application in pose detection is highlighted by the following points:

Label-to-Feature Mapping: By maximizing the probability of accurately identifying the input posture, Logistic Regression determines the ideal weights for each feature (keypoint coordinates, angles, etc.).

Probabilistic Output: By converting the linear combination of data into a probability value using the sigmoid function, which is employed in logistic regression, the model is able to forecast the probability that the input will belong to a specific pose class.

Multi-class Extension: Logistic Regression can be extended to multiclass classification, in which each yoga posture is regarded as a distinct class, using the One-vs-Rest (OvR) technique. The prediction is chosen from the poses with the highest likelihood score.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

Model Training: To minimize the discrepancy between expected probability and actual posture labels, the weights are optimized using a loss function, such as cross-entropy, during the training phase. This methodology describes a system that uses Logistic Regression to recognize yoga positions. It takes in live video stream input, processes it to extract characteristics, and then uses a trained Logistic Regression model to classify poses. The system offers real-time pose detection and feedback by utilizing keypoints and features obtained from a pre-labeled CSV dataset. Logistic regression's ease of use and effectiveness make it a good fit for real-time applications where quick classification is essential. To enhance pose accuracy and the user experience overall, more enhancements can be made by increasing the dataset, improving the feature extraction procedure, and including sophisticated feedback systems.

III. SOFTWARE AND HARDWARE REQUIREMENTS

Software Requirements:

Operating System: The system will run on Windows 10, which provides a stable and user-friendly platform for the application.

Front End: The graphical user interface (GUI) will be built using Tkinter, a standard Python library for creating desktop applications.

Database: The system will use SQLite as the database, a lightweight, server-less database engine that stores data in a single file, making it easy to manage and query.

Tool: The application will be developed using Python 3.8, a popular programming language known for its simplicity and efficiency in building diverse applications.

IDE: The Spyder integrated development environment (IDE) will be used for coding, offering features like code completion, debugging, and testing capabilities tailored for Python development.

Hardware Requirements:

System: The application will run on a Pentium IV processor with a speed of 2.4 GHz, which provides adequate performance for running the system without significant delays.

Hard Disk: A 40 GB hard drive will be sufficient for storing the software, data, and logs, ensuring the system can function without space limitations.

Monitor: A 15-inch monitor will provide enough screen real estate to interact with the application and view details clearly.

RAM: With 16 GB of RAM, the system will handle memory-intensive operations and large datasets efficiently, ensuring smooth performance and responsiveness.

IV. SYSTEM ARCHITECTURE

The design of the system for classifying yoga poses consists of multiple stages, starting from real-time posture predictions to live input capture. The subsequent elements delineate the interplay between every stage to guarantee precise categorization of yoga poses.

1. Source: Real time video-feed The first input that the system receives is from a live camera that broadcasts video frames continually. These frames serve as the Pre-processing raw data source. To process the input without latency and guarantee that the system can predict poses dynamically, a real-time capture pipeline is required. Every frame that is taken acts as a snapshot of an image that is subsequently sent for additional processing.

2. Getting Ready In order to standardize the raw video frames for the following phases, several crucial processes are applied To them at this stage:

3. Grayscale Conversion: An RGB or color image is converted to grayscale for each frame. This conversion concentrates on the structural properties of the posture instead of color, which is not important for pose classification, and lowers the computational cost.

4. Resizing: Each frame's dimensions are changed to a predetermined resolution. In addition to increasing computing efficiency without sacrificing important information, this guarantees consistency throughout the dataset. 224x224 pixels or less are common scaling settings, depending on the input resolution. Normalization: To provide uniformity throughout all input frames, the pixel values are standardized. Normalization prevents

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

feature extraction from being skewed by maintaining consistency in illumination and contrast variations. Pixel values may be scaled, for example, from 0 to 1, which would help the model train and predict more quickly.

5. Extraction of Features: The next stage is to extract discriminative features from the preprocessed frames, which are essential for identifying yoga Authorized licensed use limited to: AISSMS's Institute of Info Technology - PUNE. Downloaded on June 10,2024 at 08:13:08 UTC from IEEE Xplore. Restrictions apply. positions. Finding landmarks or important locations that symbolize the human skeleton or joints—such as the head, shoulders, elbows, and knees—is required for this. These landmarks give the position a succinct and effective representation, enabling the model to distinguish between various stances with accuracy. Convolutional neural networks (CNNs) can automate feature extraction for deep learning-based architectures; however, in this scenario, a more lightweight feature extractor might be selected to maintain system efficiency.

6. Model of Logistic Regression: We use a logistic regression model for the categorization task. When paired with robust feature extraction, logistic regression—while a less complex model than deep networks—can be incredibly efficient for binary or multi-class classification. The most likely yoga stance is predicted by the logistic regression model in this architecture once it receives the collected features from each frame.

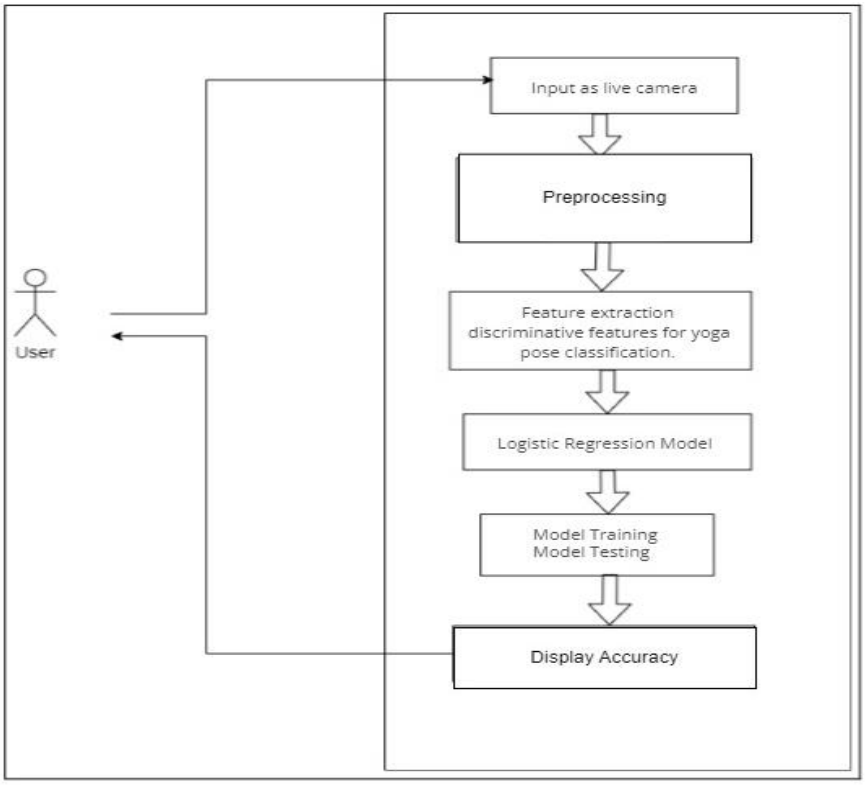

Figure 1: Architecture Diagram

V. SYSTEM DESIGN

Real-time video frames are captured by a live camera input in the system design for classifying yoga poses. Preprocessing is applied to these frames, which includes scaling, normalization, and grayscale conversion. Next, distinguishing characteristics are taken out of every shot, emphasizing important anatomical traits. For pose classification, a logistic regression model that was trained on a labeled data-set of yoga poses is utilized. The algorithm anticipates the current yoga stance after testing and shows it on the live broadcast. The user is guaranteed to receive precise and immediate input on their pose in a dynamic environment because of the design's emphasis on efficiency and real-time performance.

VI. CONCLUSION

The proposed system for yoga posture recognition and correction works by:

1) Detecting the learner's pose,

2) Calculating the angle differences between the learner's and instructor's body positions,

3) Pinpointing the errors in the learner's stance, and

4) Classifying the pose into four levels based on the average angle discrepancies. The system utilizes a time-distributed linear regression algorithm to identify patterns between key points in a single pose.

VII. REFERENCES

[1] Thoutam, V. A., & Srivastava, A. (2022). Yoga pose estimation and feedback generation using deep learning. Computational Intelligence and Neuroscience, 2022, Article ID 4311350. https://doi.org/10.1155/2022/4311350

[2] Swain, D., & Satapathy, S. (2022). Deep learning models for yoga pose monitoring. Algorithms, 15(11), 403. https://doi.org/10.3390/a15110403

[3] Anand, K., & Verma, K. (2023). Yoga pose detection using machine learning libraries. International Research Journal of Engineering and Technology, 10(01), e-ISSN: 2395-0056.

[4] Swain, D., & Satapathy, S. (2023). Yoga pose monitoring system using deep learning. Research Square, Preprint. https://doi.org/10.21203/rs.3.rs-1774107/v1

[5] Swain, D., & Satapathy, S. (2023). Yoga pose monitoring system using deep learning. Research Square, Preprint. https://doi.org/10.21203/rs.3.rs-1774107/v1