ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 11 Issue 7 July 2024

Integrating AI-Powered Pose Detection for Holistic Fitness Monitoring: Exploring Traditional Yoga Postures and Exercise Recognition

[1] Dr. Mansoor Hussain D, [2] Omkar Prashant Karmarkar, [3] Dhananjay Singh Chauhan, [4] Bitan Mallik, [5] Sohil Agarwal, [6] Abhijay Dhodapkar

[1] [2] [3] [4] [5] [6] Vellore Institute of Technology SCOPE Chennai, India
Corresponding Author Email: [1] mansoorhussain.d@vit.ac.in, [2] omkarprashant.karmarkar2021@vitstudent.ac.in, [3] dhananjaysingh.chauhan2021@vitstudent.ac.in, [4] bitanmallik2002@gmail.com, [5] Sohil3002z@gmail.com, [6] abhijayd.04@outlook.com

Abstract*— This research introduces a pioneering web-based platform that melds ancient wellness practices with cutting- edge technology, transforming the realms of yoga and fitness. Utilizing ml5.js and sophisticated machine learning, the platform refines exercise performance monitoring, focusing on the revital- ization of Surya Namaskar—a foundational yoga sequence—and various exercises such as squats, push-ups, and lunges.Surya Namaskar, or sun salutation, offers significant physical and men- tal benefits, including enhanced strength, flexibility, endurance, cardiovascular health, mindfulness, and overall vitality. Our platform employs machine learning to provide supervised pose recognition, ensuring precision and fostering disciplined, goal- oriented fitness routines.The project encompasses three main sections: Surya Namaskar Posture Detection, Exercise Detection, and Yoga Posture Detection. Users receive real-time feedback and performance metrics, optimizing their fitness journeys. p5.js creates the user interface and captures pose lines, while the pre- trained ml5.js model, based on PoseNet, ensures accurate pose recognition.Acknowledging existing research in deep learning and pose estimation, our project addresses limitations through seamless web integration and customized neural networks for superior detection efficiency. Methodology includes data collec- tion, model training, and real-time pose classification, empha- sizing user interaction, pose correction, and exercise tracking.To synchronize continuous feature extraction and model detection in dynamic sequences like Surya Namaskar, we employ a mutex lock mechanism, ensuring accurate and stable model transitions. Comprehensive user performance reporting includes metrics such as calories expended.In essence, this research offers a holistic approach to wellness, elegantly combining traditional yoga practices with modern technology to enhance physical and mental well-being.

Index Terms*— ai-powered pose detection, surya namaskar, real- time feedback, machine learning, posenet models, physi- cal and mental well-being, yoga pose identification, deep learn- ing, fitness application, technology-assisted health solutions.

I. MOTIVATION AND OBJECTIVE

The motivation arises from the need to bridge the gap between ancient wellness practices and contemporary fitness demands, ensuring that individuals can benefit from time-honored exercises like Surya Namaskar, squats, and push-ups with greater precision and effectiveness. Traditional methods often lack the ability to provide immediate, personalized feedback and detailed performance tracking, which can hinder progress and discourage consistent practice. By addressing these shortcomings, this research aims to optimize individual fitness journeys, making wellness routines more accessible, adaptable, and efficient. This approach seeks to empower users to achieve holistic well-being and vitality in their daily lives, catering to the growing demand for more tailored and insightful fitness solutions.

The objective of this research is to develop a web-based platform that integrates ancient wellness practices with ad- vanced machine learning to enhance the accuracy and effec- tiveness of yoga and fitness routines, specifically focusing on Surya Namaskar and other exercises. The platform aims to provide real-time feedback and performance metrics through sophisticated pose recognition and user interaction.

II. INTRODUCTION

Surya Namaskar, or sun salutation, is a dynamic sequence of yoga poses that holds numerous ad- vantages for physical and mental well-being. Firstly, it serves as a comprehensive warm- up exercise, gradually awakening the body and preparing it for more intense physical activity. The sequence en- gages multiple muscle groups, including the core, arms, legs, and back, promoting strength, flexibility, and endurance. Addition- ally, the rhythmic breathing synchronized with each movement enhances cardiovascular health, stimulating circulation and

oxygenation throughout the body. From a mental perspective, the repetitive nature of Surya Namaskar encourages mind- fulness and concentration, fos-tering a sense of calm and focus. Regular practice of Surya Namaskar is known to boost energy levels, improve posture, and promote overall vitality. Moreover, its accessibility and adaptability make it suitable for practitioners of all ages and

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 11 Issue 7 July 2024

fitness levels, offering a versatile and convenient way to incor- porate yoga into daily routines. Overall, Surya Namaskar stands as a holistic practice that harmonizes the body, mind, and spirit, contributing to holistic well-being and vitality.

In this Comprehensive Product, our focus is directed toward the meticulous incorporation of the fusion of machine learning methodologies and time-honoured traditions of yoga, coupled with contem- porary fitness regimes. Our model provides su- pervised recognition of poses. Here ml5js is used only to detect poses that are trained by us. This technology not only aims to enhance the precision of exercise performance monitoring but also to provide users with personalized guidance, performance metrics, and insights to optimize their fitness journey.

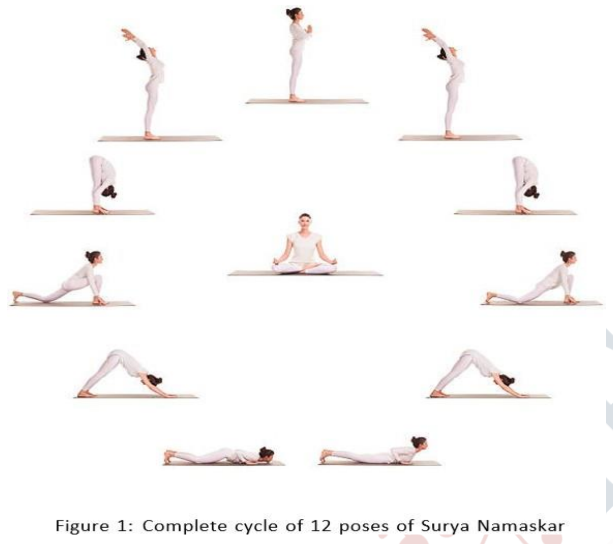


Fig. 1. Complete cycle of 12 poses of Surya Namaskar

The research project spans three distinctive sections:

- Surya Namaskar Posture Detection: Users engage in a sequence of 12 prescribed postures, with each posture requiring a fixed duration preset based on the user's requirement. The application ensures that users progress to the subsequent posture only when the current one is accurately performed. This Builds Discipline and Fitness goal-oriented tasks requiring the user to complete the set 12 postures to complete the cycle.Figure 1
- Exercise Detection: We have trained our models for the following exercises:
  - Lunges
  - Leg-Raise
  - Push-Ups
  - Knee-touch
  - Squats

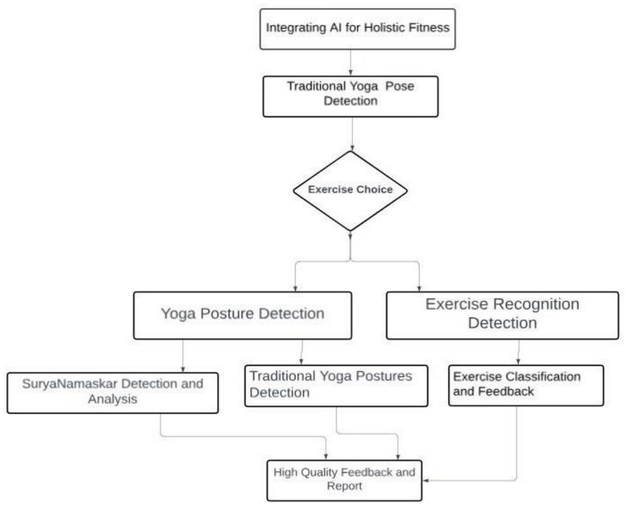

Integrating AI for Holistic Fitness
Traditional Yoga Pose Detection
Exercise Choice
Yoga Posture Detection
Exercise Recognition Detection
SuryaNamaskar Detection and Analysis
Traditional Yoga Postures Detection
Exercise Classification and Feedback
High Quality Feedback and Report

Fig. 2. Working flow diagram

- Experience the freedom to perform these exercises as many times as you want with an integrated counter seamlessly counter assisting you in tracking the number of repetitions. Elevate your exercise routine with this.
- Yoga Posture Detection: We have trained our models for the following traditional yoga:
  - Warrior Pose
  - Dog Posture
  - Tree Posture
  - Mountain Posture
  - Chair Posture

Feel the stillness within as you flow through each posture of the Yoga Session, supported by precise time intervals. Hold each pose with grace and strength, knowing you have the perfect time to deepen your breath, refine your alignment, and tune into your inner being. Let the timer be your companion, not a taskmaster. It allows you to surrender to the present moment, to fully inhabit each pose without the distraction of wondering if it's time to move on

In the context of this research endeavour, we employ p5.js to generate a standardized canvas on the user's interface, capturing pose lines through landmarks that represent the user's current posture. These landmark data are then relayed to the ml5.js model—a meticulously pre-trained algorithm specializing in the recognition of PoseNet Models, performed by an expert in the field.

Where users are encouraged to perform each posture for a user-selected duration before seamlessly tran- sitioning to the next. This approach provides overall physical and mental well-being.

III. BACKGROUND REVIEW

This section acknowledges the substantial contributions made in the yoga and exercise field. The vast expanse of

existing literature stands as a testament to the commendable efforts and profound insights offered by

IFERP Explore Your Research Journey..

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 11 Issue 7 July 2024

researchers, scholars, and practitioners in this domain.

A. Findings from Research Papers

- **Deep learning and Posenet:** It involves two modules: pre-processing and native (real-time) parts. The pre-processing module extracts target values for each yoga pose, while the native part predicts the actual poses done by the user in real time. The system uses TensorFlow- Lite for easy implementation of the PoseNet model on mobile devices and OpenCV for calculating correct pose angles. This offers real-time pose estimation and correction for yoga poses, making it an effective tool for yoga practitioners. Figure 3 [1]
- **TL-MobileNet-DA (Transfer learning)** Transfer learn- ing is a machine learning technique that involves using a pre-trained model as a starting point for a new task, in this case, recognizing yoga postures. The pre-trained model is fine-tuned on the new dataset, and the weights of the model are adjusted to improve performance on the new task. The TL-MobileNet-DA model was chosen as the optimal model for accuracy and sensitivity.[2]
- **Logistic Regression Model along with Mediapipe and OpenCv:** Logistic regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. In the context of the paper, a logistic regression model is used to classify data and detect yoga poses based on the x, y, and z coordinates of joint points. This model is trained and tested to achieve a high accuracy of nearly 100 per cent.[3]
- **Deep learning:** Deep learning-based 2D/3D human pose estimation methods from monocular images or video footage of humans. The key technologies used in the methodology include deep learning techniques such as Convolutional Neural Networks (CNNs) and recurrent neural networks (RNNs) for feature extraction and se- quence modelling. These techniques have been widely adopted in the field of computer vision and have shown remarkable performance in tasks such as image recogni- tion and sequence modeling.[4]

B. Limitations of the above papers and how do we overcome them

Our project successfully detects and classifies postures with efficiency. We've achieved seamless web integration using ml5.js and p5.js, empowering users to customize and build interactive applications. ML5 Js is based on PoseNet and provides 2 models i.e. RestNet50 and MobileNetV1, with the ability to run directly in web browsers or on mobile devices. This simplicity facilitates integration into various applications without the need for extensive development efforts.

Unlike MobileNet which is good for small, low latency power models parameterized to meet the resource constraints. In the above projects, OpenCv and MediaPipe are employed,

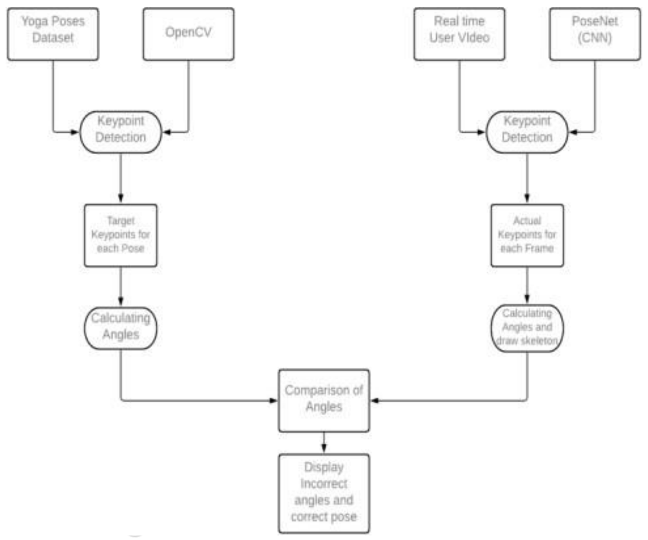

Yoga Poses Dataset
OpenCV
Keypoint Detection
Target Keypoints for each Pose
Calculating Angles
Real time User Video
PoseNet (CNN)
Keypoint Detection
Actual Keypoints for each Frame
Calculating Angles and draw skeleton
Comparison of Angles
Display Incorrect angles and correct pose

Fig. 3. Working of Deep learning and posenet

along with custom angle calculations for posture detection. Our approach involves utilizing a neural network to store custom-trained postures, enhancing the efficiency of detection. This tailored neural network setup provides flexibility and adaptability, making our project versatile for various appli- cations.

IV. METHODOLOGY

This code utilizes the ml5.js library to perform pose classi- fication using the Pose Net model for human pose estimation. The implementation is divided into three parts: data collection, model training, and model deployment. The main focus is on collecting training data for different poses, training a neu- ral network, and deploying the trained model for real-time pose classification through a webcam feed.

In the coding section of this project in the HTML, we have included and imported the necessary files to libraries of ml5 js and p5 .js which we would be using in our js file created which is an important setup. JavaScript file involves the use of a library and its functions for our setup and pose detection.

A. Data Collection

1) Canvas and Video Capture:

Upon the initiation of the web page, the setup() function, an integral component of the p5.js library, is executed to establish the foundational structure. This includes the generation of a canvas with dimensions 640 x 480 within the browser window. Simultaneously, the create- Capture() function is utilized to capture user video input, serving as the primary input for subsequent processes.

2) Model Integration:

The integration of the PoseNet model, a sophisticated machine learning model designed for precise pose esti-

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 11 Issue 7 July 2024

mation, is realized through the implementation of the ml5.poseNet() function. This function seamlessly incorporates the captured video as input, facilitating the subsequent processing of pose- related data.

3) Key Press Event Handling Label Assignment:

User interaction is managed through the implementation of the keyPressed() function. This function serves as the conduit for user key presses, enabling the assignment of labels and the systematic collection of data. Triggered by a designated key (e.g., “a”), a 10-second window is activated, prompting the user to execute a specific posture. Subsequently, the gotPoses() function is invoked to accurately extract landmark data points from the video, which are then meticulously stored in an array. Each set of data is accompanied by its corresponding label for future reference.

4) Neural Network Integration:

A pivotal step involves the instantiation of a ml5.neuralNetwork() object, denoted as “brain.” This object acts as the receptacle for the neural network, seamlessly incorporating the previously collected data, complete with labels.

5) Data Storage:

The culmination of the process is marked by the user’s initiation of data storage by pressing the “S” key. The comprehensive neural network data, encapsulating both input data and network structure, is diligently preserved in JSON format. This stored data serves as a founda- tional resource for subsequent training endeavours.

B. Training

We will utilize ml5’s Neural Network to classify poses obtained from Pose Net detection. Additionally, we’ll employ p5’s library to visualize the epoch-loss graph during neural network training. Initializing a neural network instance with classification options and setting debug to true enables visu- alization of the training process. Before commencing training with the data, normalization is applied to bring the pose Net’s input into the unit plane.

Training parameters include setting epochs to 32 and the batch value to 32. The generated epoch- loss graph allows us to analyze potential under-fitting and over-fitting. Once satisfaction with the model’s performance is achieved, it can be saved for production use. It is saved in three files which are the weight file (contains weights and bias for neural net- work), metafile (contains info about input and output values), and main file (contains info about neural network like layers, activation, etc).

C. Making Predictions

The main core of the project taking in real video of users performing the selected selections of user and performing effective and time-assisted posture detection. This Section provides light on the Working of the Various Sections of the Project.

1) Exercise Section:

In this section, the user has the choice selection of postures Squats, Lunges, Leg Raiser, Push Ups, Bending Toe Touch, Once the User Selects a Required Posture, the Model Corresponding to the Pos- ture is loaded using this method. Example:

const model info = {
model: ‘model2/model.json’, metadata: ’model2/model meta.json’,
weights: ‘model2/model.weights.bin’};

These files are generated by IN THE training setup. The main Working part of the exercise is the vertical or downward movement like In squats, the raise and bend of these two postures are predicted by the model and based on the state machine the count of the current performing pose is stored and rendered on the browser. When the user exits the current exercise the model is removed and the time and steps done are saved in the user’s Reports Sections.

2) Yoga Section:

This is a Timer-Based Section that Enables the User to perform all given postures Tree Posture. Warrior Pose, Dog Posture, Mountain Pose, Chair Pose. Once this section is Chosen by the user a new page loads and a model loads similar to all other sections of the projects. In this there is no model switch involved, one model is capable of performing detections of all the postures and the Js code effectively makes the user perform the current posture for a specific duration and the model ensures the posture correction for this phase.

3) Surya Namaskar Section:

The Surya Namaskar Module lies at the core of the project, strategically designed to optimize the user’s overall fitness level. Building upon the modular archi- tecture employed in other sections, this module de- ploys distinct machine-learning models: one specifically calibrated for standing postures associated with Surya Namaskar and another for sleep-related postures. This targeted division significantly enhances both the perfor- mance and detection accuracy of posture recognition. Upon user initiation through the “Start Button,” the rele- vant model is activated, prompting the user to embark on the Surya Namaskar sequence. Concurrently, the robust model performs continuous pose analysis, employing cutting-edge algorithms to precisely determine whether the user has achieved the desired posture.

D. Challenge:Synchronizing Continuous Feature Extraction and Model Detection

Surya Namaskar, a sequence of twelve yoga postures, presents a dynamic scenario for pose recogni- tion. Landmarks representing key body positions are continuously extracted throughout the sequence. Simultaneously, multiple AI models analyze these landmarks to identify specific postures in real time. The primary challenge lies in ensuring synchronous execution of these processes without introducing delays or errors during model switching.

E. Proposed Solution: Model Switching with Mutex Lock

To address this challenge, we propose utilizing a mutex lock, a synchronization mechanism that ensures exclusive access to a shared resource by multiple threads or processes. In our case, the mutex lock controls access to the currently active AI model during switching procedures. The implementation involves the following steps:

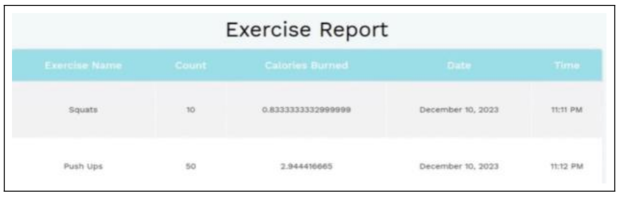
1) **Initialization:**During system setup, all AI models and the mutex lock are initialized.
2) **Current Model Maintenance:**A global variable keeps track of the currently active model for seamless refer- encing.
3) **Switching Logic:**
   - **Pre-Switch Lock Acquisition:**Before transitioning to a new model, the mutex lock is acquired to prevent concurrent access to the previous model during its final classification.
   - **Post-Switch Lock Acquisition:**Once the previous model completes its final classifica- tion, the mutex lock is again acquired to prepare for the switch.
4) **Model Switch:**The active model reference is updated within the protected space provided by the lock, reflect- ing the switch to the new model.
5) **Release Lock:**Once the switch is complete, the mutex lock is released, allowing the classification loop to resume with the newly activated model.

F.Analysis and Validation

The implementation of the mutex lock mechanism demon- strably improves the stability and efficiency of model switch- ing within the Surya Namaskar pose recognition system. By preventing concurrent access and data race conditions, this approach ensures accurate pose detection throughout the dynamic sequence. Further analysis and testing could focus on optimizing the lock acquisition and release timings for even greater performance enhancements.

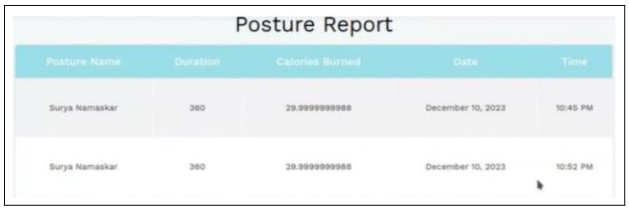
V. ANALYSIS AND RESULTS

Comprehensive Reporting: The culminating facet of the project delivers an exhaustive report for each section. This includes intricacies such as calories expended and detailed performance metrics, offering users a thorough understanding of their participation and progress in the research study.



Exercise Report

Exercise Name	Count	Calories Burned	Date	Time
Squats	10	0.8333333332999999	December 10, 2023	11:11 PM
Push Ups	50	2.944416665	December 10, 2023	11:12 PM



Posture Report

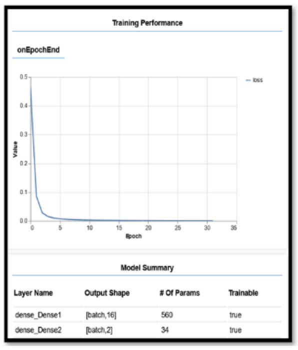
Posture Name	Duration	Calories Burned	Date	Time
Surya Namaskar	360	29.9999999988	December 10, 2023	10:45 PM
Surya Namaskar	360	29.9999999988	December 10, 2023	10:52 PM

The Below graph gives epoch diagrams that hold signifi- cance in our project, An epoch graph is a vital tool in machine learning, tracking a model's training process, applicable in frameworks like ML5.js for prediction tasks. An epoch rep- resents a full pass through the dataset during training, and the graph visually depicts the model's performance evolution, offering insights:

1) **Convergence:** Indicates if the model is converging to- wards an optimal solution, seen through stabilized or decreased error metrics.
2) **Overfitting:** Reveals when the model relies too much on training data, losing generalization ability, often shown by a gap between training and validation performance.
3) **Underfitting:** Indicates the model's failure to capture underlying patterns, reflected in consistently high error rates across training and validation sets.
4) **Learning Rate:** Helps adjust this hyperparameter, con- trolling the pace of model updates, crucial for accelera- tion, and avoiding oscillations or divergence.

A. Graphical analysis of model training

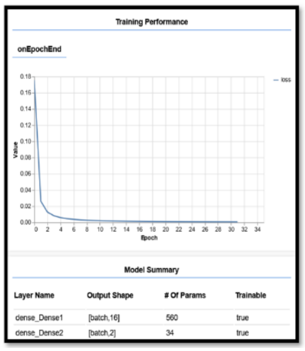
Knee Touch



Model Summary

Layer Name	Output Shape	# Of Params	Trainable
dense_Dense1	[batch,16]	560	true
dense_Dense2	[batch,2]	34	true

Leg Raiser



Model Summary

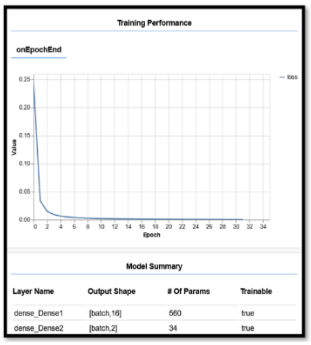
Layer Name	Output Shape	# Of Params	Trainable
dense_Dense1	[batch,16]	560	true
dense_Dense2	[batch,2]	34	true

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 11 Issue 7 July 2024

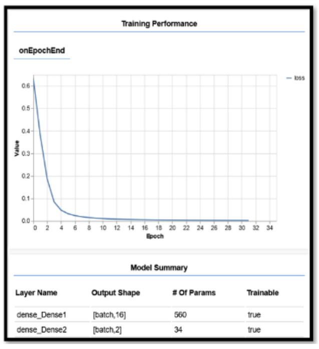
Lunges



Model Summary

Layer Name	Output Shape	# Of Params	Trainable
dense_Dense1	[batch,16]	560	true
dense_Dense2	[batch,2]	34	true

Squats

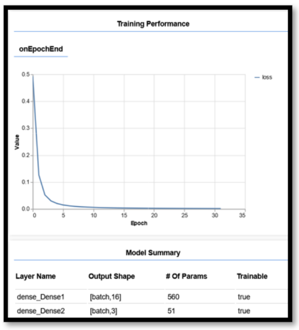


Model Summary

Layer Name	Output Shape	# Of Params	Trainable
dense_Dense1	[batch,16]	560	true
dense_Dense2	[batch,2]	34	true

Fig. 4. Epoch-loss graphs of exercise positions

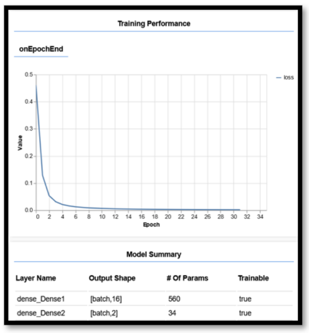
Posture 1, 2, 3



Model Summary

Layer Name	Output Shape	# Of Params	Trainable
dense_Dense1	[batch,16]	560	true
dense_Dense2	[batch,3]	51	true

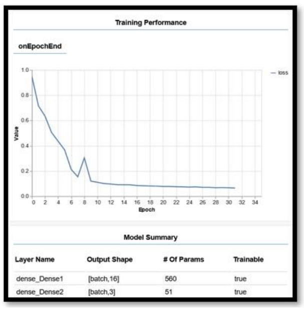
Posture 4, 5



Model Summary

Layer Name	Output Shape	# Of Params	Trainable
dense_Dense1	[batch,16]	560	true
dense_Dense2	[batch,2]	34	true

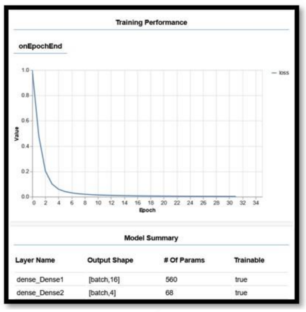
Posture 6, 7, 8



Model Summary

Layer Name	Output Shape	# Of Params	Trainable
dense_Dense1	[batch,16]	560	true
dense_Dense2	[batch,3]	51	true

Posture 9, 10, 11, 12



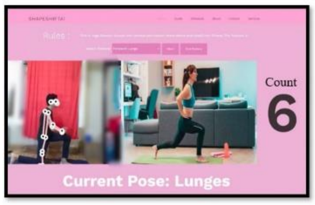
Model Summary

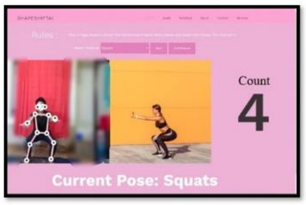
Layer Name	Output Shape	# Of Params	Trainable
dense_Dense1	[batch,16]	560	true
dense_Dense2	[batch,4]	68	true

Fig. 5. Epoch-loss graphs of Surya Namaskar

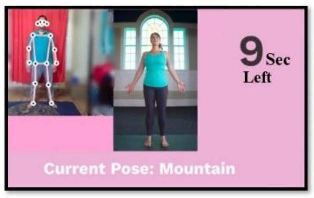
B. Final view

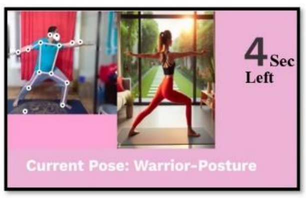
Exercises



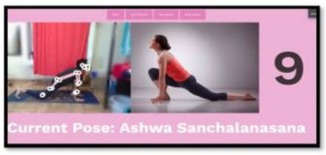


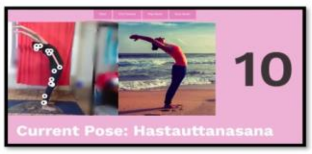
Yoga





Surya Namaskar





Correction message Notified.

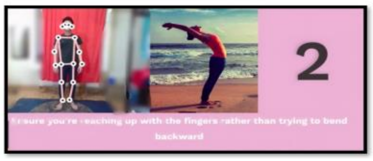

2
Ensure you're reaching up with the fingers rather than trying to bend backward

Fig. 6. Working of Web application

VI. CONCLUSION

This research successfully integrates ancient wellness prac- tices with advanced machine learning, enhancing exercise routines like Surya Namaskar. By utilizing ml5.js and PoseNet for real-time pose recognition and feedback, the platform pro- motes precise, goal-oriented fitness, improving both physical and mental well-being through seamless technology integra- tion.

REFERENCES

[1] Girija Gireesh Chiddarwar, Abhishek Ranjane, Mugdha Chindhe, Rachana Deodhar, Palash Gangamwar: AI-Based Yoga Pose Estimation for Android Application, ISSN No:-2456-2165

[2] Chhaihuoy Long, Eunhye Jo, Yunyoung Nam: Development of a yoga posture coaching system using an interactive display based on transfer learning, The Journal of Supercomputing (2022) 78:5269–5284 https://doi.org/10.1007/s11227-021-04076-wLink

[3] Rutuja Jagtap, Monali Zanzane, Rutuja Patil: Yoga pose detection using machine learning e-ISSN: 2582-5208

[4] Yucheng Chena, Yingli Tianb, Mingyi Hea: Monocular Human Pose Estimation: A Survey of Deep Learning-based Methods: https://doi.org/10.1016/j.cviu.2019.102897Link

[5] Ml5jslibrary documentation https://learn.ml5js.org// reference/indexLink

[6] P5js library documentation https://p5js.org/Link

[7] Machine learning in Javascript guide https://thecodingtrain. com/tracks/ml5js-beginners-guide/Link

[8] Coding Train Ml-5 js Youtube Playlist https://youtu.be/26uAB exmOX4?si=oMn3w3Uj4mYrky98Link

[9] Yoga-AI by cris-maillo https://github.com/cris- maillo/yogAI Link