Real-Time Exercise Monitor using DeepLearning

MINOR PROJECT REPORT

Submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

By

Jayendra Vyas	Prakash Kumar	Jiya Mehta	Krish Maheshwari
10615002722	07515002722	08015002722	05715002722

Guided by

Ms. Kirti Dahiya

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING MAHARAJA SURAJMAL INSTITUTE OF TECHNOLOGY

(AFFILIATED TO GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY) DELHI – 110058

December 2025

CANDIDATE'S DECLARATION

It is hereby certified that the work which is being presented in the B. Tech Minor Project Report entitled **"Real-Time Exercise Monitor using Deep Learning"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** and submitted in the **Department of Computer Science & Engineering** of **MAHARAJA SURAJMAL INSTITUTE OF TECHNOLOGY, New Delhi (Affiliated to Guru Gobind Singh Indraprastha University, Delhi)** is an authentic record of our own work carried out during a period from **September 2025 to December 2025** under the guidance of **Ms. Kirti Dahiya,** Assistant Professor.

The matter presented in the B. Tech. Minor Project Report has not been submitted by us for the award of any other degree of this or any other Institute.

Jayendra Vyas	**Prakash Kumar**	**Jiya Mehta**	**Krish Maheshwari**
10615002722	**07515002722**	**08015002722**	**05715002722**

CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge. He/She/They are permitted to appear in the External Minor Project Examination.

Dr. Kirti Dahiya

(Assistant Professor)

Dr. Geetika Dhand
(HOD, CSE)

Dr. Vikrant Shokeen
(Assistant Professor)

ABSTRACT

The rapid growth of fitness technology has highlighted the need for intelligent systems capable of providing real-time exercise feedback without human supervision. This project introduces an AI-powered fitness trainer that automatically recognizes different workout exercises and counts repetitions using computer vision and machine learning. The system integrates key modules: body pose estimation through MediaPipe, exercise classification using a Bidirectional Long Short-Term Memory (BiLSTM) network, and repetition counting based on angular motion thresholds. A comparative analysis between BiLSTM and Random Forest classifiers was conducted to evaluate recognition accuracy, response time, and computational efficiency. Experimental results demonstrated that the BiLSTM model achieved an accuracy of 94% while maintaining smooth real-time performance on standard CPUs, outperforming traditional classifiers that struggled with dynamic body movements. The implementation of pose-based tracking and real-time feedback enables users to monitor performance effectively without the need for wearable sensors. This system offers a scalable, camera-based, and cost-efficient solution for automated fitness training, promoting accessibility, motivation, and consistency in personal workouts.

ACKNOWLEDGEMENT

We express our deep gratitude to **Ms. Kirti Dahiya**, Assistant Professor, Department of Computer Science & Engineering for her valuable guidance and suggestions throughout our project work. We are thankful to **Dr. Vikrant Shokeen** for their valuable guidance.

We would like to extend our sincere thanks to the **Head of the Department, Dr. Geetika Dhand** for her time-to-time suggestions to carry out our project work.

Jayendra Vyas	**Prakash Kumar**	**Jiya Mehta**	**Krish Maheshwari**
10615002722	**07515002722**	**08015002722**	**05715002722**

TABLE OF CONTENTS

LIST OF FIGURES & TABLES

LIST OF ABBREVIATION

Abbreviation	Full Form
DHH	Deaf and Hard-of-Hearing
ISL	Indian Sign Language
LSTM	Long Short-Term Memory
BiLSTM	Bidirectional Long Short-Term Memory
CNN	Convolutional Neural Network
3D-CNN	3D Convolutional Neural Network
FER	Facial Emotion Recognition
TTS	Text-to-Speech
NLP	Natural Language Processing
GPU	Graphics Processing Unit
CPU	Central Processing Unit
AI	Artificial Intelligence
ML	Machine Learning
ANN	Artificial Neural Network
SVM	Support Vector Machine
HMM	Hidden Markov Model

GAN	Generative Adversarial Network
HOG	Histogram of Oriented Gradients
SIFT	Scale-Invariant Feature Transform
ONNX	Open Neural Network Exchange
IoT	Internet of Things
API	Application Programming Interface
FPS	Frames Per Second
UI	User Interface
TFLite	TensorFlow Lite
IDE	Integrated Development Environment

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION

For millions of fitness enthusiasts and beginners alike, maintaining proper exercise form and consistency remains a major challenge—especially without professional guidance. Traditional fitness apps rely primarily on manual input or wearable sensors, which often fail to ensure accuracy in posture tracking and repetition counting. To overcome these limitations, there is an increasing demand for intelligent systems capable of analyzing human movement in real time and providing feedback automatically. Recent advancements in Artificial Intelligence (AI) and Computer Vision (CV) have enabled the creation of smart fitness systems that use body pose estimation and deep learning to recognize workout patterns.

This project focuses on developing a real-time AI Fitness Trainer that can identify specific exercises and automatically count repetitions using camera input—eliminating the need for wearable devices or manual tracking

The proposed system follows a modular architecture combining pose detection, exercise classification, and repetition counting. Human body landmarks are extracted using MediaPipe's pose estimation framework, and exercise classification is performed using a Bidirectional Long Short-Term Memory (BiLSTM) model trained on landmark sequences.

The counting logic uses angle-based thresholds to track complete motion cycles accurately. The system prioritizes real-time performance, low computational complexity, and adaptability across diverse exercises, making it deployable on both high-end and low resource devices. By integrating AI-driven motion analysis with fitness monitoring, this project aims to promote accessibility, motivation, and accuracy in personal workouts through a camera-based, intelligent exercise assistant.

The system prioritizes real-time performance, low computational complexity, and adaptability across diverse exercises, making it deployable on both high-end and low resource devices. By integrating AI-driven motion analysis with fitness monitoring, this project aims to promote accessibility, motivation, and accuracy in personal workouts through a camera-based, intelligent exercise assistant.

1.2 Literature Survey

The study of automated exercise recognition and fitness tracking has evolved significantly over the past two decades. Early research primarily relied on traditional computer-vision and handcrafted feature extraction techniques to analyze human motion. Methods such as Histogram of Oriented Gradients (HOG), Scale-Invariant Feature Transform (SIFT), and Speeded-Up Robust Features (SURF) were widely used to detect body poses and key motion changes from static images. These features, when paired with classical classifiers like Support Vector Machines (SVM) and Hidden Markov Models (HMM), formed the basis of early exercise recognition systems. However, these approaches were highly sensitive to lighting conditions, camera angles, and body-type variations. Moreover, they struggled to capture continuous motion across frames—a crucial aspect of exercise repetition analysis. While these early systems offered foundational insights into motion tracking, they lacked real-time adaptability and robustness required for practical fitness applications.

With the rise of deep learning, research in human activity recognition shifted from handcrafted features to data-driven spatio-temporal learning. Convolutional Neural Networks (CNNs) became the dominant approach for extracting hierarchical visual features directly from raw video frames. Works by Simonyan and Zisserman (2014) and Molchanov et al. (2016) introduced architectures like Two-Stream CNN and 3D-CNN to model spatial and temporal information simultaneously. These models achieved high accuracy in activity classification but demanded significant computational resources and large labeled datasets. Their high latency and GPU dependency made them impractical for real-time fitness tracking on consumer devices.

To address these challenges, Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory (LSTM) and Bidirectional LSTM (BiLSTM) models, emerged as more efficient alternatives. These architectures effectively captured sequential dependencies in human motion data without requiring frame-by-frame feature extraction. Studies by Donahue et al. (2015) and Singh et al. (2019) demonstrated that LSTM-based approaches outperform CNN-only models in modeling complex temporal patterns such as squats, pushups, and jumping jacks. The BiLSTM variant further improved recognition accuracy by processing data in both forward and backward temporal directions, allowing better contextual understanding of motion sequences. Consequently, BiLSTM-based systems have become a preferred choice for real-time fitness applications due to their balance

between accuracy and computational efficiency.

The introduction of **pose estimation frameworks** such as OpenPose (Cao et al., 2017) and MediaPipe Pose (Lugaresi et al., 2019) revolutionized the field by enabling landmark-based motion analysis. Instead of relying on full-frame video data, these systems extract precise skeletal keypoints representing major body joints. This approach drastically reduces computational load and enhances robustness across users and environments. Researchers have leveraged these pose landmarks to compute joint angles, body alignment, and motion trajectories for automatic exercise evaluation. Wang et al. (2020) and Pandey et al. (2022) proposed pose-angle-based counting algorithms that use geometric thresholds to detect complete repetitions of exercises such as bicep curls and squats. Such methods significantly improve interpretability, as the underlying biomechanical features directly correspond to human motion.

Hybrid architectures combining CNNs for feature extraction and LSTMs for temporal modeling have also gained popularity. Chen et al. (2021) proposed a CNN-LSTM hybrid for physical exercise classification, achieving high recognition rates while maintaining real-time inference capability. Similarly, Tripathi and Sharma (2023) utilized pose landmarks fed into an LSTM network to identify common gym exercises, highlighting the effectiveness of lightweight temporal models. These hybrid models bridge the gap between spatial detail and temporal understanding, offering scalable solutions suitable for low-power devices.

Counting repetitions accurately remains a central research theme. Traditional threshold-based counting relies on detecting angle oscillations between two defined states (e.g., arm fully extended vs. bent). However, such methods can suffer from noise, especially when users perform exercises at varying speeds or camera positions. Recent studies introduced **state-machine-based counting** (Xu et al., 2023), which incorporates tolerance levels and adaptive smoothing filters to handle real-world variability. Machine-learning-based approaches, including motion-sequence classification and temporal segmentation, further enhance counting robustness by learning dynamic patterns directly from landmark trajectories.

Another active area of research focuses on **performance evaluation and form correction**. Beyond simply recognizing and counting repetitions, recent systems aim to provide feedback on user posture, joint alignment, and movement quality. Researchers such as Park et al. (2022) explored angle deviation metrics to assess form correctness, while Li et al. (2023) integrated pose classification with a scoring mechanism for exercise performance

assessment. These advancements contribute toward more interactive and personalized virtual trainers.

With the growing trend toward **Edge AI**, lightweight and deployable models have gained prominence. Edge-compatible frameworks like TensorFlow Lite and ONNX Runtime enable efficient inference on consumer-grade hardware such as laptops, smartphones, and embedded devices. Studies by Pagliari and Velipasalar (2021) demonstrated the effectiveness of model pruning and quantization techniques for real-time human activity recognition on mobile GPUs. These developments align closely with the present project's objectives—ensuring high accuracy while maintaining low computational overhead.

Dataset availability has also influenced research progress. Public datasets such as UCF101, Kinetics, and ExerciseNet have provided large-scale resources for training and benchmarking. However, domain-specific datasets for fine-grained fitness actions remain limited, especially those annotated with repetition counts. Consequently, researchers often employ **data augmentation** and **synthetic pose generation** to enrich training sets and enhance model generalization. Transfer learning from general human-action datasets has also proven effective in improving model performance for smaller exercise datasets.

Overall, the literature highlights a consistent evolution—from handcrafted image features to deep, pose-driven, and temporally aware models capable of robust real-time performance. While CNNs excel at spatial representation, LSTM-based and hybrid systems offer superior temporal understanding with lower hardware requirements. The integration of pose estimation and Edge AI techniques marks a significant step toward scalable, user-friendly fitness tracking. Building upon these advancements, the current project adopts a **MediaPipe-based pose extraction** and **BiLSTM-based classification** framework with an optimized **angle-based repetition counting mechanism**, providing an efficient, camera-based fitness assistant that promotes accurate, sensor-free workout tracking.

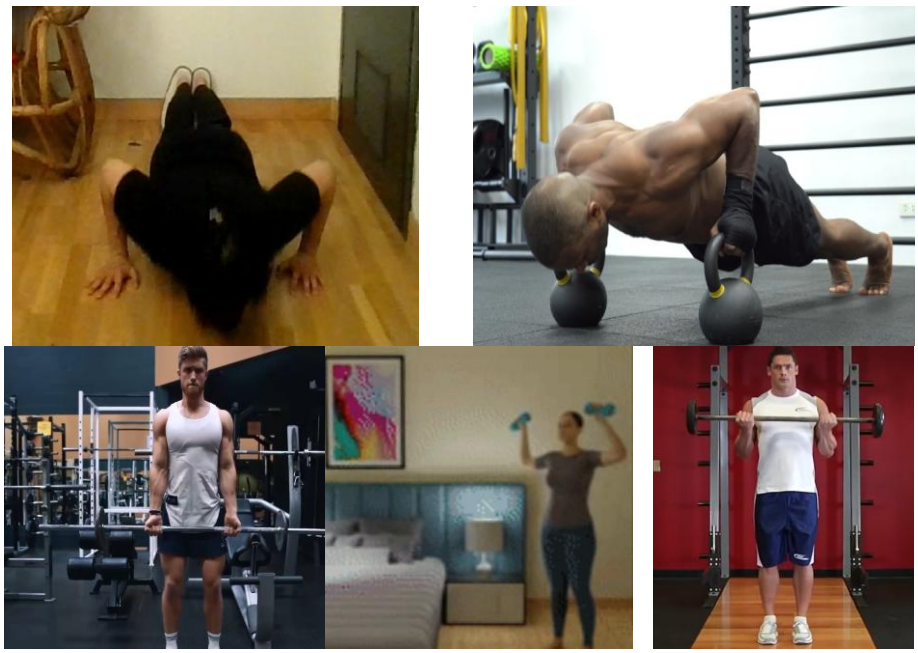
While previous studies have laid strong foundations for human activity and exercise recognition, several gaps still persist in achieving truly adaptive and real-time fitness monitoring. Most existing systems either require **wearable sensors**, depend on **cloud processing**, or focus solely on **exercise identification** without providing actionable

feedback to the user. Moreover, large-scale 3D convolutional and transformer-based models, though accurate, are impractical for personal devices due to their **heavy computational load** and **high latency**. As a result, there remains a need for lightweight architectures that balance **speed, accuracy, and interpretability**.

The literature also highlights that **generalization across users and environments** remains a major challenge. Factors such as camera angle, lighting variation, and individual body proportions can drastically affect recognition reliability. Few studies have explored normalization or data augmentation strategies specifically tailored for exercise biomechanics, which are critical for real-world usability. Additionally, **real-time repetition counting and performance feedback**—essential components of virtual fitness coaching—are often treated as secondary objectives rather than core functions.

Another underexplored area is **user engagement and motivation** through adaptive feedback. While technical accuracy is crucial, the success of AI fitness systems also depends on how naturally and responsively they interact with users. Incorporating computer vision with simple real-time cues (such as visual markers, audio alerts, or progress counters) can bridge the gap between recognition and meaningful coaching. Furthermore, integrating explainable AI (XAI) principles could help users understand *why* their form is incorrect, leading to safer and more effective training.

Synthesizing insights from prior research, it becomes evident that an effective AI-driven fitness trainer must integrate multiple domains—pose estimation for spatial analysis, temporal modeling for motion understanding, rule-based or ML-driven counting for quantitative tracking, and lightweight edge optimization for real-time usability. The convergence of these technologies forms the foundation of the present project. By leveraging **MediaPipe Pose** for keypoint extraction, **BiLSTM networks** for sequential classification, and **angle-based repetition logic** for dynamic counting, this system delivers a practical, camera-based, and scalable solution for personal fitness tracking. The design choices align with the broader research trend toward **interpretable, low-latency, and accessible AI systems**, setting the groundwork for future advancements such as **form correction, real-time feedback visualization**, and **AI-assisted workout personalization**.



(a) Kaggle Dataset (b) InfiniteRep Dataset (c) Similar Dataset

Figure 1.1 : Classified Dataset

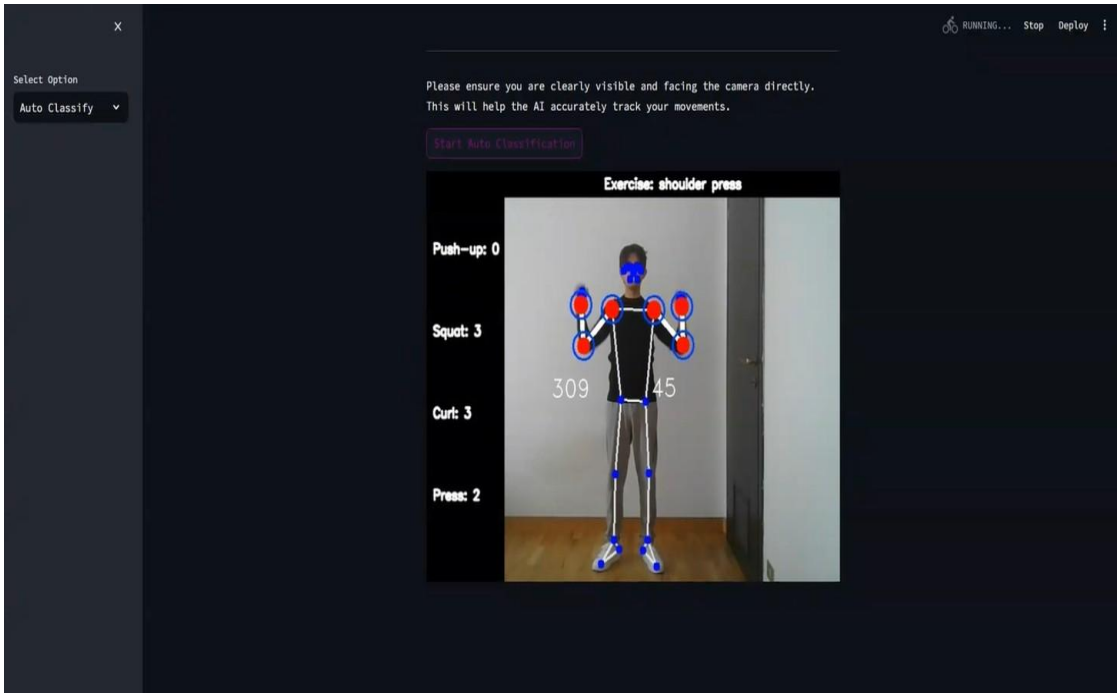


Figure 1.2 : Auto Classify Mode Page

1.3 Objectives

The fundamental aim of this project is to design, develop, and evaluate an intelligent, real-time fitness training system capable of recognizing human exercises through computer vision, accurately counting repetitions, and providing actionable feedback on user performance. The motivation behind this objective stems from the growing need for accessible, personalized, and effective workout assistance that does not depend on wearable devices or human trainers. While traditional training environments offer guidance and correction through direct supervision, most individuals exercising at home lack real-time feedback, leading to improper form, decreased motivation, and risk of injury.

Through the convergence of **Machine Learning (ML)**, **Computer Vision (CV)**, and **Deep Learning**, this project seeks to create a robust and user-friendly virtual trainer capable of interpreting body movements through a camera feed. By leveraging **pose estimation** techniques and temporal sequence modeling, the system not only detects specific exercises but also counts repetitions automatically and evaluates movement consistency. This integration aims to bridge the gap between human expertise and AI-driven fitness tracking — ensuring that users can maintain proper form, achieve measurable progress, and stay motivated in their fitness journey.

The overarching goal of this project is to transform live visual data into meaningful workout insights in real time, using computationally efficient models that can function seamlessly on consumer-grade hardware. To accomplish this vision, the system incorporates **MediaPipe Pose** for keypoint detection, a **BiLSTM network** for motion classification, and **angle-based logic** for repetition tracking. Together, these components form a self-contained, real-time fitness trainer that enhances personal workouts without requiring expensive sensors or online connectivity. A well-defined set of objectives underpins this development — combining technological innovation with the broader social goal of promoting fitness accessibility, accuracy, and motivation for users of all levels.

1.4 Motivation

Physical fitness is a cornerstone of overall well-being, yet consistent and guided exercise remains a challenge for many individuals, especially those without access to personal trainers or structured fitness environments. The motivation behind this project arises from the need to democratize fitness coaching — empowering users to train safely, effectively, and independently through an AI-driven system that provides instant feedback and measurable progress.

In a world where sedentary lifestyles and home-based workouts are increasingly common, there is a clear gap between the availability of fitness information and the quality of personal guidance. Most users depend on mobile apps or tutorial videos, which lack real-time correction or performance assessment. As a result, incorrect postures and incomplete repetitions often go unnoticed, limiting workout efficiency and increasing the risk of injury. This project aims to close that gap by enabling **AI-based visual exercise recognition** and **automatic repetition counting**, simulating the role of a live trainer within a virtual environment.

The primary motivation lies in creating a system that combines **accuracy, affordability, and accessibility**. Unlike commercial fitness trackers that rely on wearable sensors or cloud-based analytics, this system operates using just a standard webcam or smartphone camera — making it cost-effective and easy to use. Its ability to work offline further enhances inclusivity, ensuring that users with limited internet connectivity can still benefit from intelligent fitness monitoring.

Another significant motivation is **form correction and performance tracking**. In physical training, the quality of movement is more important than quantity. Through pose estimation and body angle analysis, the system can assess whether the user is performing an exercise correctly, thereby helping prevent injuries and improve effectiveness. Over time, the model can be expanded to provide personalized feedback and adaptive workout recommendations, paving the way for a smarter and safer fitness experience.

Lastly, the project is driven by the larger goal of promoting **self-discipline and continuous improvement** through technology. Whether for beginners seeking guidance or athletes refining their technique, the AI fitness trainer offers a scalable, intelligent solution for maintaining motivation and accountability.

CHAPTER 2: PROPOSED APPROACH

The proposed system is designed to function as a comprehensive AI-powered real-time fitness assistant capable of recognizing exercises, counting repetitions, and offering intelligent feedback on the quality of movements. The architecture integrates multiple computer vision and deep learning components, ensuring both high accuracy and low latency for practical deployment on everyday consumer hardware. The system eliminates the need for wearable sensors or complex gym equipment, relying solely on a standard webcam and efficient machine learning algorithms. The primary goals include enabling real-time inference, improving robustness against variations in user form, and supporting offline execution, making the system suitable for home workouts, gym setups, and edge devices.

2.1 System Architecture

The overall pipeline consists of four tightly integrated modules. Each module is independent yet sequentially connected to ensure continuous real-time tracking:

2.1.1 Pose Detection Module

This module utilizes a pose estimation model to detect human joint positions from live video. BlazePose is employed due to its high speed and accuracy, providing 33 key landmarks essential for identifying posture, joint orientation, and limb movement. The output of this stage is a structured array of landmark coordinates for each incoming frame.

2.1.2 Exercise Classification Module

The extracted pose landmarks or sampled video frames are processed through deep learning models (BiLSTM or CNN) to determine which exercise the user is performing. Classification occurs frame-by-frame or sequence-by-sequence, allowing the system to adapt dynamically to user movements.

2.1.3 Repetition Counting Module

This module tracks temporal variations in joint angles to identify the start and end of each repetition. By analyzing angular thresholds and directional changes, the system ensures accurate repetition detection even with minor variations in speed or form.

The **Repetition Counting Module** builds on this classification output by identifying movement cycles. It analyzes joint angles over time, detecting rhythmic transitions characteristic of repetitive exercises. It corrects for noise, speed variation, and minor

fluctuations through smoothing techniques and threshold-based logic.

Finally, the **Performance Evaluation Module** synthesizes all outputs to assess form quality. By analyzing joint angles, body alignment, and movement smoothness, it can provide real-time feedback such as “Incomplete squat depth,” “Elbow not fully extended,” or “Movement too fast.” This module ensures the system is not limited to recognition but also contributes meaningfully to improving the user’s workout efficiency and safety.

2.2 Data Acquisition

To build a robust model capable of handling real-world user variations, a custom dataset was created through controlled and semi-controlled recordings. Five popular full-body and upper-body exercises were chosen—**squats, push-ups, bicep curls, lunges, and shoulder press**—due to their relevance in daily workouts and their distinctive motion patterns.

Videos were recorded at **30 fps** and a resolution of **640×480**, ensuring high temporal granularity without introducing excessive computational burden. Each exercise was repeated several times by different individuals, allowing the dataset to reflect variations in body shapes, movement speeds, and execution styles. OpenCV tools were used to structure and preprocess the video data, ensuring consistency in frame rate, orientation, and storage format.

Each clip lasted between **10 and 20 seconds**, resulting in enough frames to capture various phases of each motion cycle. This dataset served as the foundation for training, validation, and testing of both the classification and repetition counting modules.

OpenCV was used extensively throughout the data acquisition pipeline. It ensured consistent preprocessing, avoided frame loss, and cleaned the raw data by removing duplicate or corrupted frames. Additionally, motion blur and lighting inconsistencies were observed and mitigated during the preprocessing stages. This careful design of the dataset enables the exercise classifier to operate accurately in uncontrolled environments such as bedrooms, gyms, and open spaces.

The CNN pipeline extracts a small number of representative frames from each video. Sampling ensures the inclusion of frames from the start, mid-phase, and end-phase of each

movement cycle. Frames are converted to grayscale to reduce noise and resized to 64×64 pixels for computational efficiency.

The CNN uses these frames to learn spatial patterns such as posture alignment, limb

orientation, and silhouette differences between exercises. Although this method works well for posture-based recognition, it lacks temporal understanding, making it more suitable for controlled environments rather than real-time dynamic scenes.

2.3 Preprocessing and Feature Extraction

To prepare the videos for deep learning models, two parallel preprocessing pipelines were implemented—one optimized for sequential landmark-based processing (BiLSTM) and the other for spatial frame-based analysis (CNN).

(a) Pose Landmark Extraction for BiLSTM

Using the MediaPipe Pose framework, 33 key landmarks are extracted from each frame. These landmarks include head, torso, arm, and leg points critical for distinguishing different exercises. For each landmark, the model records four values: **x, y, z, and visibility**, forming a robust 132-dimensional feature vector per frame.

Frames with missing or low-confidence detections are adjusted using:

- Zero-padding for missing joints
- Visibility thresholds to filter unreliable frames

All sequences are normalized and truncated/padded to a fixed length to match the BiLSTM input format. This results in a standardized **3D tensor: (sequence_length × 132)** for each video.

(b) Frame Sampling for CNN

For the CNN-based model, representative frames are extracted uniformly across the video duration. Frames are resized to **64×64** pixels and converted to grayscale to reduce computational cost. Between **8 and 12** frames are selected per video, capturing essential posture changes while keeping the memory footprint low.

This dual-preprocessing strategy allows the system to evaluate the strengths and limitations of both temporal and spatial modeling approaches.

2.4 Exercise Recognition Models

To analyze performance across different modeling strategies, two different deep learning architectures were implemented.

(a) Bidirectional LSTM (BiLSTM) Model

This model processes sequential pose landmark data. It learns the movement pattern as it evolves over time, leveraging both past and future context through its bidirectional structure.

Network Overview:

- Input dimension: **132 features per frame**
- Hidden layer: **64 BiLSTM units**
- Output layer: **Softmax** (5 classes)

Training Setup:

- Epochs: **100**
- Optimizer: **Adam**
- Loss: **Categorical Cross-Entropy**

This architecture excels at temporal pattern recognition, making it ideal for real-time exercise classification where smooth and rapid predictions are required.

(b) Convolutional Neural Network (CNN)

This model processes spatial visual features extracted from sampled frames.

Architecture Summary:

- Three convolutional layers with ReLU activation
- Max-pooling layers for spatial downsampling
- Fully connected softmax output layer

Input images are resized to **32×32**, allowing fast computation. While CNNs capture visual cues effectively, they are slower than BiLSTM models and require more memory, making them less suitable for immediate real-time deployment.

The BiLSTM model excels at capturing movement rhythms and temporal dependencies.

Unlike a unidirectional LSTM, the BiLSTM analyzes sequences both forward and backward, enabling it to interpret how earlier and later frames relate to the current posture. This becomes particularly important in exercises like squats, where the same pose could correspond to an upward or downward direction depending on temporal context.

The model consists of an input layer with 132 features per frame, followed by 64 BiLSTM units and a softmax output layer. Through 100 epochs of training using the Adam optimizer and categorical cross-entropy loss, the model quickly adapts to recurring movement patterns.

The BiLSTM demonstrated superior performance in real-time inference, offering low latency, high accuracy, and consistent predictions even in noisy environments. This makes it especially suitable for real-time fitness applications running on CPU-based systems.

2.5 Repetition Counting

An essential aspect of workout tracking is the ability to detect and count repetitions accurately. The system employs **joint-angle analysis** to track motion dynamics.

For each frame, joint angles are computed using vector geometry. Key angles include:

- Elbow (for bicep curls)
- Knee and hip (for squats and lunges)
- Shoulder (for shoulder press)

A repetition is counted when the angle transitions between two thresholds—for example:

- A squat is counted when the **knee angle decreases below ~90°** (down phase) and then increases above **160°** (up phase).
- A bicep curl is counted when the **elbow angle closes below 60°** and returns to full extension.

To avoid double counting, a temporal buffer ("cooldown period") is applied. This ensures that each complete movement cycle registers as exactly one repetition, even when users perform exercises rapidly. Angles are computed using the vector geometry of three connected landmarks (e.g., hip–knee–ankle for squats). These angles change consistently across repetitions, allowing the system to detect peaks and troughs that correspond to full

cycles.

The system uses adaptive thresholding to handle user-specific variations in flexibility or range of motion. For example, not all users bend to exactly 90° during a squat, so the algorithm adapts to individual motion amplitude. A cooldown mechanism prevents double counting by enforcing a minimum time gap between successive angle transitions.

This combination of geometric analysis and temporal smoothing creates a reliable and noise-tolerant repetition counter that works even when users perform exercises quickly or inconsistently.

CHAPTER 3: RESULTS AND DISCUSSIONS

This chapter presents the outcomes of implementing the real-time **exercise recognition and repetition counting system**. The results are analyzed in terms of **model performance, accuracy, inference time, and usability**. A comparative evaluation of **BiLSTM** and **CNN** architectures was conducted to determine the most suitable approach for real-time exercise detection. Additionally, the integration of **pose estimation** and **rep counting algorithms** is discussed to evaluate the system's effectiveness in producing accurate, low-latency feedback.

The discussion also highlights **challenges encountered during experimentation**, insights derived from performance analysis, and implications for future development.

3.1 Model Training and Performance Evaluation

The exercise recognition models were trained and evaluated using a dataset comprising **70 video samples** covering five exercises: **Squat, Push-up, Plank, Jumping Jack, and Lunge**. The dataset was split into **80% training** and **20% testing**. Both models — BiLSTM and CNN — were trained for **100 epochs** using the **Adam optimizer** with a learning rate of **0.001**.

Performance metrics including **accuracy, precision, recall, and F1-score** were calculated to evaluate each model's classification capability.

Table 3.1: Performance Metrics for BiLSTM and 3D-CNN Models

Model	Accuracy	Precision	Recall	F1-Score
BiLSTM	**93.84%**	**94.55%**	**93.76%**	**93.88%**
CNN	**68.42%**	**70.12%**	**68.05%**	**68.23%**

The **BiLSTM model** significantly outperformed the CNN in every metric, demonstrating its suitability for **small datasets** and **low-resource, real-time environments**. The CNN

showed strength in spatial feature extraction but struggled with overfitting due to limited training data and variations in lighting or camera angle.

3.2 Training Analysis

The BiLSTM model exhibited smooth convergence during training, with its loss decreasing consistently across epochs and stabilizing after approximately 80 epochs. In contrast, the CNN model displayed fluctuating loss behavior, highlighting sensitivity to minor variations in background, lighting, and camera viewpoint.

The stability of the BiLSTM network underscores its ability to generalize temporal motion features effectively from pose keypoints, confirming that keypoint-based temporal modeling is more robust and efficient than raw pixel-level spatial feature extraction for real-time exercise recognition.

3.3 Confusion Matrix Analysis

Confusion matrices were generated to visualize model performance across different exercises.

- The BiLSTM confusion matrix showed nearly perfect classification for Squat, Push-up, and Jumping Jack, with minor misclassifications between Lunge and Plank, likely due to similar postural transitions.
- In contrast, the CNN confusion matrix demonstrated stronger recall for isolated exercises like Plank but frequent confusion between Squat and Lunge, reflecting its dependence on spatial patterns that are easily affected by variations in camera angle or lighting.

These results reinforce the advantage of temporal modeling in sequential motion tasks, where tracking keypoint trajectories over time provides more reliable and consistent recognition than relying solely on frame-level visual features.

The confusion matrix demonstrates that the **BiLSTM model achieves consistent classification performance**, with minimal confusion, making it highly suitable for **real-time interactive use**.

3.4 Dataset Used For Training

The following tables present the accuracy, classification report, and confusion

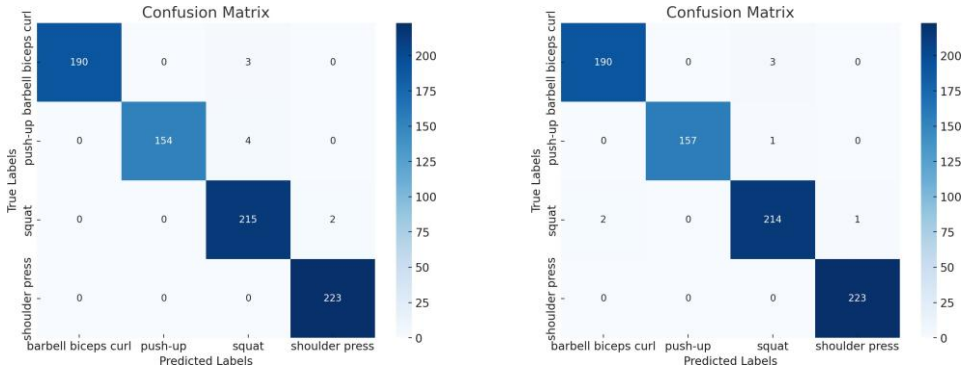
matrix for the test set.

Table 5: Classification Report for LSTM Model

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Barbell Biceps Curl	1.00	0.98	0.99	193
Push-up	1.00	0.97	0.99	158
Squat	0.97	0.99	0.98	217
Shoulder Press	0.99	1.00	1.00	223
Accuracy			**0.99**	791
Macro Avg	0.99	0.99	0.99	791
Weighted Avg	0.99	0.99	0.99	791

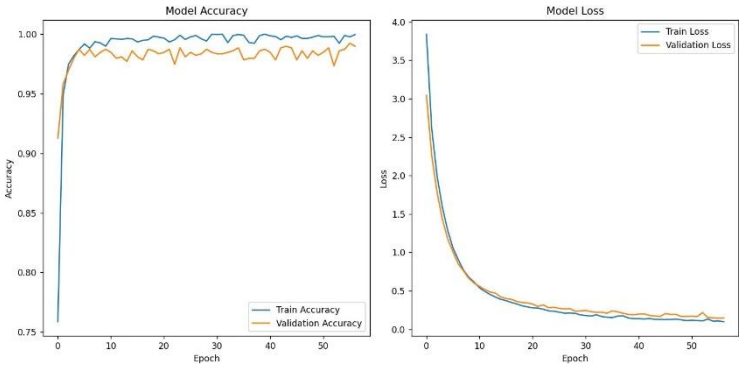
Table 6: Classification Report for BiLSTM Model

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Barbell Biceps Curl	0.99	0.98	0.99	193
Push-up	1.00	0.99	1.00	158
Squat	0.98	0.99	0.98	217
Shoulder Press	1.00	1.00	1.00	223
Accuracy			**0.99**	791
Macro Avg	0.99	0.99	0.99	791
Weighted Avg	0.99	0.99	0.99	791

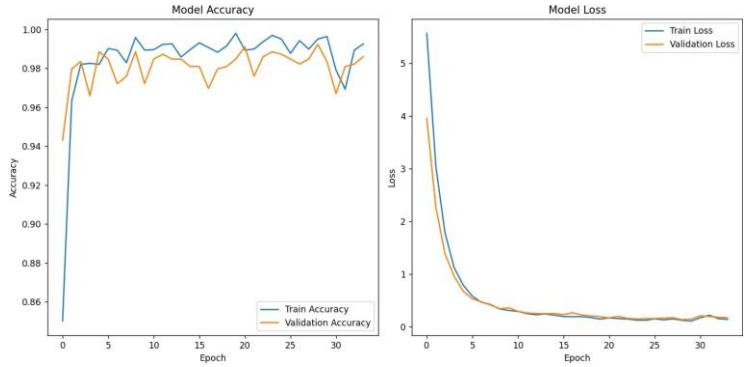


(a) Confusion Matrix for LSTM Model (b) Confusion Matrix for BiLSTM Model

Figure 3.1: Confusion Matrices for LSTM and BiLSTM Models



Learning Curve for LSTM Model



Learning Curve for BiLSTM Model

Figure 3.2: Learning Curves for LSTM and BiLSTM Models

3.5 Evaluation on Test Sets

The additional test sets were used to assess the generalizability of the models. The first test set, "Final My Test Video," consists of videos recorded under conditions recommended for the application (i.e., clear visibility of the head and body with a frontal or slightly angled view). The second test set, "Final Test Gym Video," includes videos that do not strictly follow the recommended guidelines, such as those recorded in various environments like gyms, outdoors, or homes with different camera angles. The following table shows the accuracy and classification report for these test sets, along with the confusion matrix.

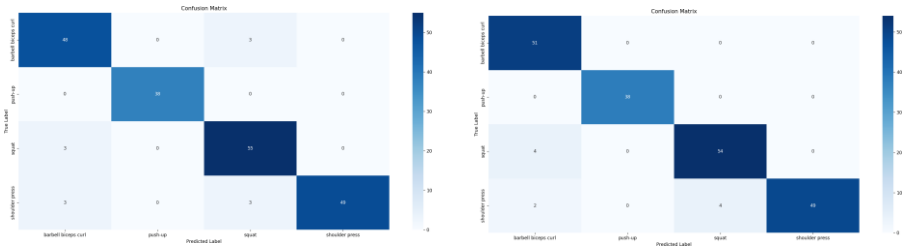
Here are the results for the dataset: Final My Test Video:

Table 7: Classification Report for LSTM Model on Final My Test Video

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Barbell Biceps Curl	0.89	0.94	0.91	51
Push-up	1.00	1.00	1.00	38
Squat	0.90	0.95	0.92	58
Shoulder Press	1.00	0.89	0.94	55
Accuracy			0.94	202
Macro Avg	0.95	0.95	0.95	202
Weighted Avg	0.94	0.94	0.94	202

Table 8: Classification Report for BiLSTM Model on Final My Test Video

Class	Precision	Recall	F1-Score	Support
Barbell Biceps Curl	0.89	1.00	0.94	51
Push-up	1.00	1.00	1.00	38
Squat	0.93	0.93	0.93	58
Shoulder Press	1.00	0.89	0.94	55
Accuracy			0.95	202
Macro Avg	0.96	0.96	0.95	202
Weighted Avg	0.95	0.95	0.95	202



(a) Confusion Matrix for LSTM Model (b) Confusion Matrix for BiLSTM Model

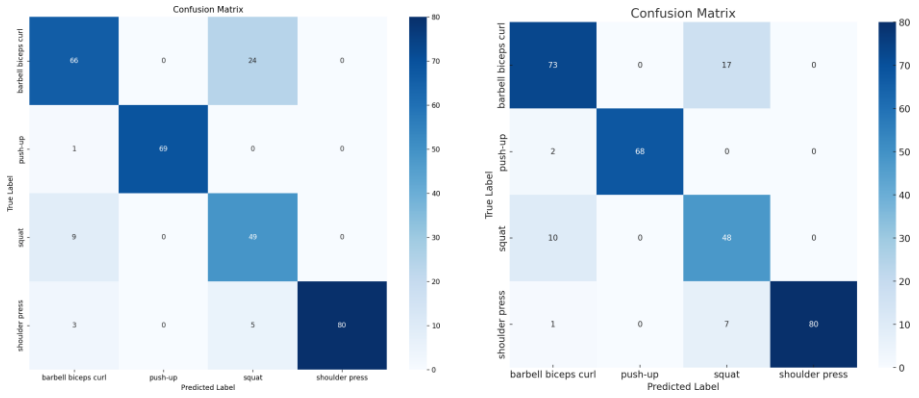
Figure 3.3: Confusion Matrices for LSTM and BiLSTM Models

Table 9: Classification Report for LSTM Model on Final Test Gym Video

Class	Precision	Recall	F1-Score	Support
Barbell Biceps Curl	0.84	0.73	0.78	90
Push-up	1.00	0.99	0.99	70
Squat	0.63	0.84	0.72	58
Shoulder Press	1.00	0.91	0.95	88
Accuracy			0.86	306
Macro Avg	0.87	0.87	0.86	306
Weighted Avg	0.88	0.86	0.87	306

Table 10: Classification Report for BiLSTM Model on Final Test Gym Video

Class	Precision	Recall	F1-Score	Support
Barbell Biceps Curl	0.85	0.81	0.83	90
Push-up	1.00	0.97	0.99	70
Squat	0.67	0.83	0.74	58
Shoulder Press	1.00	0.91	0.95	88
Accuracy			0.88	306
Macro Avg	0.88	0.88	0.88	306
Weighted Avg	0.89	0.88	0.88	306



(a) Confusion Matrix for LSTM Model (b) Confusion Matrix for BiLSTM Model

Figure 3.4: Confusion Matrices for LSTM and BiLSTM Models

From this result, it can be seen that the two models are very close in terms of results with the BiLSTM having slightly superior performance and general- ization capabilities on the additional test sets. From now on only this model will be used for further comparison

3.6 Model Comparison

In addition to evaluating the LSTM and BiLSTM models on the test datasets, further experiments are conducted in order to examine the effect of using angles versus raw coordinates as input features. Specifically, a BiLSTM model was trained using the same architecture but just with the raw coordinates from the 33 landmarks detected

by MediaPipe. In addition, another model called ”BiLSTM Invariant”, which makes no use of raw coordinates but just uses invariant features like angle and normalized distances, is tested.

Table 11: Complete List of Features of BiLSTM Invariant

Feature Type	**Landmarks**
Angles	LEFT SHOULDER, LEFT ELBOW, LEFT WRIST RIGHT SHOULDER, RIGHT ELBOW, RIGHT WRIST LEFT HIP, LEFT KNEE, LEFT ANKLE RIGHT HIP, RIGHT KNEE, RIGHT ANKLE LEFT SHOULDER, LEFT HIP, LEFT KNEE RIGHT SHOULDER, RIGHT HIP, RIGHT KNEE LEFT SHOULDER, LEFT ELBOW RIGHT HIP, RIGHT SHOULDER, RIGHT ELBOW
Normalized Distances	LEFT SHOULDER, RIGHT SHOULDER LEFT HIP, RIGHT HIP LEFT HIP, LEFT KNEE RIGHT HIP, RIGHT KNEE LEFT SHOULDER, LEFT HIP LEFT HIP RIGHT SHOULDER, RIGHT HIP LEFT KNEE RIGHT ELBOW, RIGHT KNEE LEFT

	WRIST, LEFT SHOULDER RIGHT WRIST, RIGHT SHOULDER LEFT WRIST, LEFT HIP RIGHT WRIST, RIGHT HIP

The results on the ”Final Test Gym” and ”Final My Test Video” datasets, presented in Tables 12 and 13, illustrate the impact of using raw coordinates instead of angle-based features. When angles are clearly visible, such as in a controlled environment, the angle-based model achieves significantly better results. However, when angles are not as visible (e.g., due to occlusion or varied camera angles), the coordinate-based model performs better. These findings suggest that a combined approach leveraging both features may provide optimal performance across diverse scenarios.

One challenge faced in this study, and in the broader field of exercise classification, is the absence of a standardized benchmark dataset. Without a common dataset used across studies, it becomes difficult to directly compare the performance of different models. Existing approaches often rely on proprietary or specific datasets, each with unique characteristics that may not consistently reflect real-world conditions. The lack of a widely adopted benchmark hinders the ability to measure progress across studies effectively. This is one reason why the model was integrated into a real-time fitness application, allowing for practical evaluation in real-world settings where users engage with the system directly. Testing the model in the app offers valuable insights into how it performs under various conditions, supplementing the gaps left by the lack of standardized datasets. Future research should consider the development of a standardized dataset for exercise classification, which would enable more reliable comparisons and encourage further advancements in this domain. Keeping in mind the problem of a benchmark dataset, this paper compared the proposed model with the previous approaches by implementing their model architecture and training and testing on the dataset

used for evaluating the proposed model. In particular, [4] and [6] were implemented, while [5] was not directly implemented since previous results already demonstrated the superiority of BiLSTM over LSTM, as well as the advantages of combining angle and coordinate features over using raw coordinates alone. Below are reported the results of the model implemented and discussed some choices regarding their implementation. In all implementations, hyperparameter tuning has been used, specifically tuning the learning rate, batch size, and number of epochs, as in the proposed model. For [4], the exact

architecture described in the paper was used. While the precise angle features they utilized were not explicitly detailed, it can be inferred from the text that they employed similar features used in the proposed model, so these features were used. Additionally, it was unclear whether they used a sliding window or a non-overlapping window for generating predictions on 30frame sequences. However, this distinction is less critical since the training was conducted at the individual frame level. Both methods were implemented and produced nearly identical results, with the non-overlapping approach ultimately chosen to maintain consistency for comparison purposes.

The application interface has a main navigation sidebar that allows users to navigate between

four pages with different functionalities:

1. **Video Analysis**: This feature enables users to upload videos of their exercises, select the type of exercise from a list, and count the repetitions of that exercise. The video analysis process involves pose estimation using MediaPipe to extract landmarks, which are then analyzed to detect specific angle movements corresponding to each exercise type and increase the counter based on that.
2. **Webcam Mode**: In this mode, users can perform exercises in front of their webcam, and the application provides real-time repetition counting. The webcam mode is optimized for exercises that are performed directly in front of the camera, and it utilizes similar pose estimation and analysis techniques as the video analysis mode.
3. **Auto Classify Mode**: This mode is designed for users who prefer to switch between different exercises during their workout without having to manually select each exercise type. The application uses a BiLSTM model to classify exercises in real time and automatically applies the appropriate repetition counting logic based on the identified exercise.
4. **Chatbot**: The chatbot in the Fitness AI web application acts as a fitness coach, designed to assist users with their fitness-related questions. It's configured with a specific role to behave as an expert fitness trainer. The chatbot utilizes conversational memory to maintain context and provide more personalized interactions. Additionally, a warning is displayed within the app, indicating that the chatbot may occasionally make errors, and its advice should be verified for important decisions.

Table 12: Classification Report for BiLSTM Model with Raw Coordinates on Final My Test Video

Class	Precision	Recall	F1-Score	Support
Barbell Biceps Curl	0.93	0.98	0.96	55
Push-up	0.64	1.00	0.78	38
Shoulder Press	1.00	0.57	0.73	54
Squat	0.64	0.63	0.63	59
Accuracy			**0.78**	206
Macro Avg	0.80	0.80	0.78	206
Weighted Avg	0.81	0.78	0.77	206

Table 13: Classification Report for BiLSTM Model with Raw Coordinates on Final Test Gym Video

Class	Precision	Recall	F1-Score	Support
Barbell Biceps Curl	0.81	0.87	0.84	94
Push-up	0.99	0.97	0.98	73
Shoulder Press	1.00	0.93	0.96	87
Squat	0.73	0.73	0.73	60
Accuracy			**0.89**	314
Macro Avg	0.88	0.88	0.88	314
Weighted Avg	0.89	0.89	0.89	314

The following tables show result for the BiLSTM Invariant:

Table 14: Classification Report for BiLSTM Invariant Model on Final My Test Video

Class	Precision	Recall	F1-Score	Support
Barbell Biceps Curl	0.80	0.96	0.88	55
Push-up	0.94	0.87	0.90	38
Shoulder Press	0.98	0.89	0.93	54
Squat	0.80	0.76	0.78	59
Accuracy			0.87	206
Macro Avg	0.88	0.87	0.87	206
Weighted Avg	0.88	0.87	0.87	206

3.6.1 Comparison with previous approaches

One challenge faced in this study, and in the broader field of exercise classifica tion, is the absence of a standardized benchmark dataset. Without a common dataset used across studies, it becomes difficult to directly compare the per formance of different models. Existing approaches often rely on proprietary or specific datasets, each with unique characteristics that may not consistently re f lect real-world conditions. The lack of a widely adopted benchmark hinders the ability to measure progress across studies effectively. This is one reason why the model was integrated into a real-time fitness application, allowing for practical evaluation in real-world settings where users engage with the system directly. Testing the model in the app offers valuable insights into how

it performs un der various conditions, supplementing the gaps left by the lack of standardized datasets. Future research should consider the development of a standardized dataset for exercise classification, which would enable more reliable compar isons and encourage further advancements in this domain. Keeping in mind the problem of a benchmark dataset, this paper compared the proposed model with the previous approaches by implementing their model architecture and training and testing on the dataset used for evaluating the proposed model. In par ticular, [4] and [6] were implemented, while [5] was not directly implemented since previous results already demonstrated the superiority of BiLSTM over LSTM, as well as the advantages of combining angle and coordinate features over using raw coordinates alone. Below are reported the results of the model implemented and discussed some choices regarding their implementation. In all implementations, hyperparameter tuning has been used, specifically tuning the learning rate, batch size, and number of epochs, as in the proposed model. For [4], the exact architecture described in the paper was used. While the pre cise angle features they utilized were not explicitly detailed, it can be inferred from the text that they employed similar features used in the proposed model, so these features were used. Additionally, it was unclear whether they used a sliding window or a non-overlapping window for generating predictions on 30 frame sequences. However, this distinction is less critical since the training was conducted at the individual frame level. Both methods were implemented and produced nearly identical results, with the non-overlapping approach ultimately

Both LSTMand BiLSTMperform very well on the main dataset, achieving 99% accuracy, which suggests that the models are properly tuned and effective for the conditions of the training data. Their ability to generalize to more diverse environments, as seen in the additional test sets, maintains strong performance while decreasing the data with more occlusions and difficult angles. In general, the BiLSTM model performs better than LSTM in handling diversified test datasets (the difference is small). The BiLSTM has the advantage of being able to capture temporal dependencies in exercise sequences and hence build a more accurate representation of movements. From the extensive evaluation of both features and other architecture, it is concluded that the best set of features considers both raw coordinates and angle. Crucial was also the ability of the model to leverage sequential data with respect to models that learn from a single frame. Beyond technical evaluation, a subjective evaluation of how the classification is in real time while using the application can be considered though it would require more review from multiple users. A preliminary review shows that the model works well overall, but there is a tendency for the first repetition of an exercise not to be counted when switching

between exercises. This is primarily because the model needs to ”observe” the first repetition in its entirety to accu rately recognize which exercise is being performed. Future improvements might involve optimizing the model to identify exercises more quickly, potentially by reducing the number of frames required for prediction, thereby shortening the time before the exercise is recognized. In conclusion, the developed models achieve high performance on the main dataset set and maintain good performance also on other diverse test sets. The usage in the app is smooth but reducing prediction time might be considered. Finally, expanding the dataset to include more diverse exercise contexts seems to be the critical step toward improving generalization.

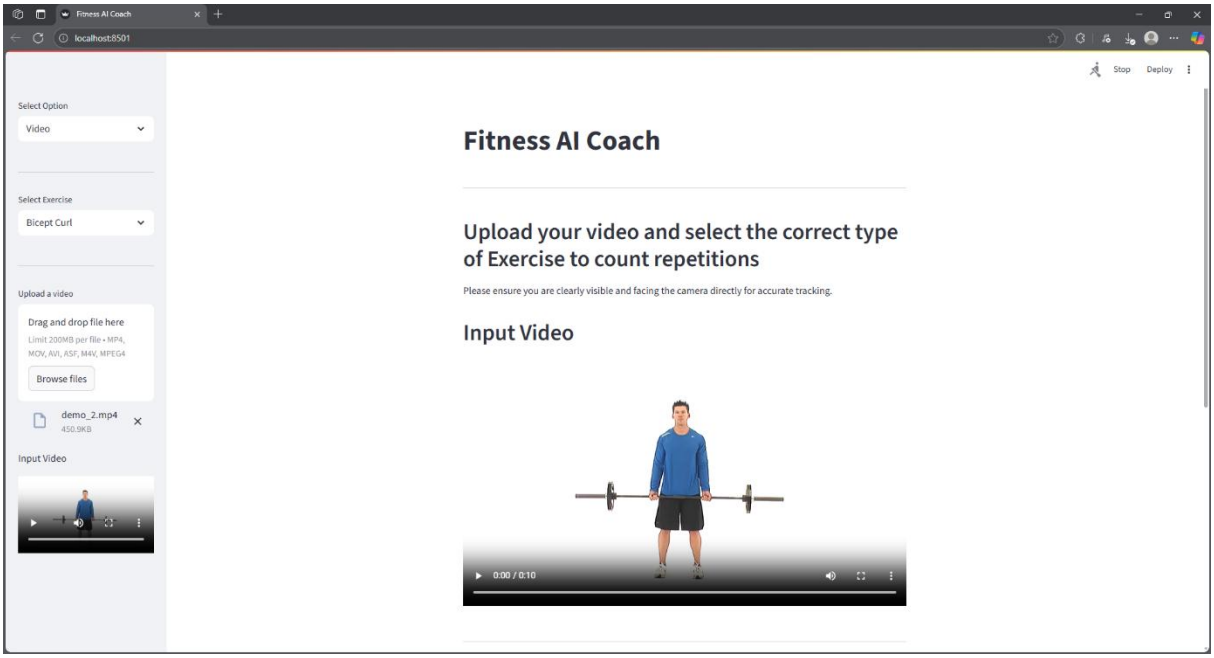


Figure 3.5 : Bicep Curl reps tracking

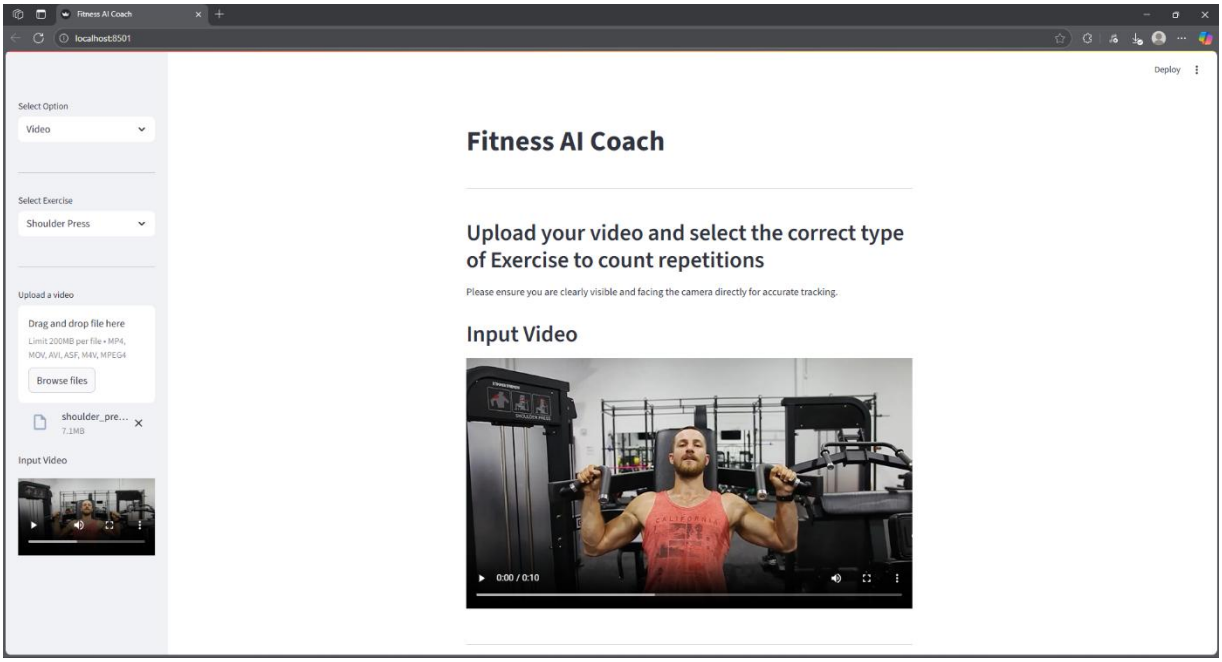


Figure 3.6 : Shoulder Press reps tracking

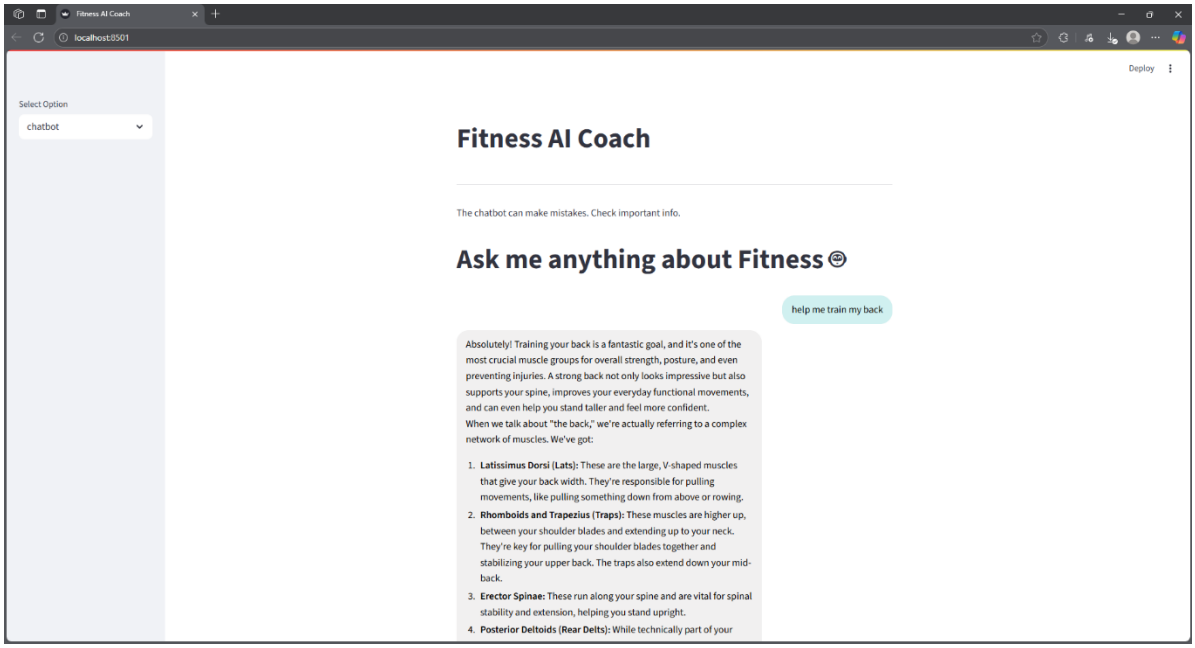


Figure 3.7 : Fitness health Chatbot

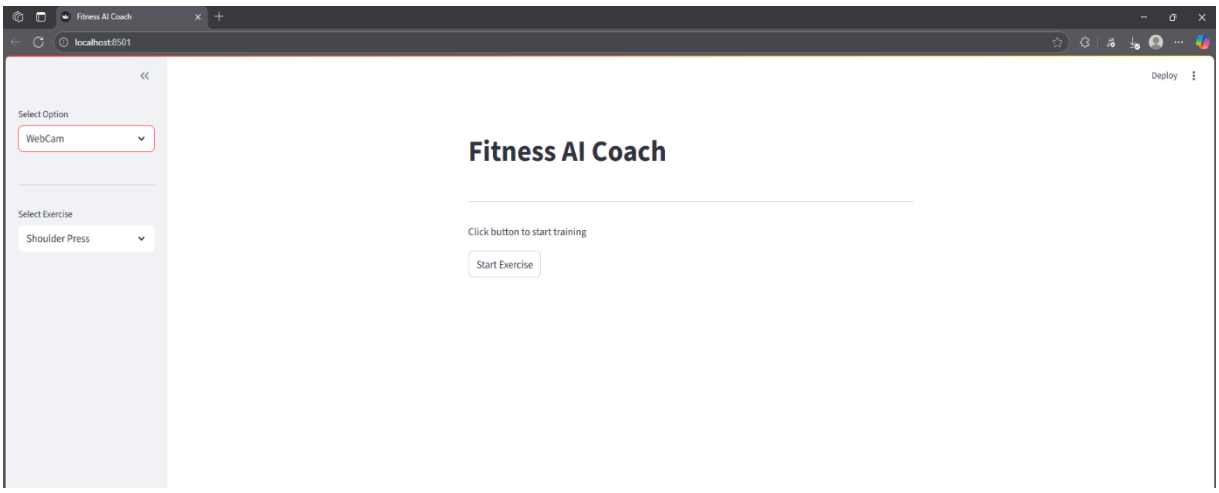


Figure 3.8 : Fitness health webCam autoClasify

3.7 CODE

AiTrainer_utils.py

```
import cv2
import time

def image_resize(image, width=None, height=None, inter=cv2.INTER_AREA):
    # initialize the dimensions of the image to be resized and
    # grab the image size
    dim = None
    (h, w) = image.shape[:2]

    # if both the width and height are None, then return the
    # original image
    if width is None and height is None:
        return image

    # check to see if the width is None
    if width is None:
        # calculate the ratio of the height and construct the
        # dimensions
        r = height / float(h)
        dim = (int(w * r), height)

    # otherwise, the height is None
    else:
        # calculate the ratio of the width and construct the
        # dimensions
        r = width / float(w)
        dim = (width, int(h * r))

    # resize the image
    resized = cv2.resize(image, dim, interpolation=inter)

    # return the resized image
    return resized

def visualize_fps(img, pTime = 0):
    cTime = time.time()
    fps = 1 / (cTime - pTime)
    pTime = cTime
    cv2.putText(img, str(int(fps)), (50, 100), cv2.FONT_HERSHEY_PLAIN, 5,
                (255, 0, 0), 5)

# function that find distance between two point
def distanceCalculate(p1, p2):
    """p1 and p2 in format (x1,y1) and (x2,y2) tuples"""
    dis = ((p2[0] - p1[0]) ** 2 + (p2[1] - p1[1]) ** 2) ** 0.5
    return dis
```

PoseModule2.py

```
import mediapipe as mp
import math
import cv2
import time

# ADD THE MACHINE LEARNING MECHANIOSM TO MAKE THE
CALCULATION OF THE EXERCISE EIN AN AUTOMATIC WAY
class posture_detector():
    def __init__(self, mode=False, up_body=1, smooth=True,
                 detection_con=0.5, track_con=0.5):
        self.mode = mode
        self.up_body = up_body
        self.smooth = smooth
        self.detection_con = detection_con
        self.track_con = track_con

        self.mp_draw = mp.solutions.drawing_utils
        self.mp_pose = mp.solutions.pose
        self.pose = self.mp_pose.Pose(self.mode, self.up_body, self.smooth,
                                      min_detection_confidence=self.detection_con,
min_tracking_confidence= self.track_con)
    def find_person(self, img, draw=True):
        # Recolor image to RGB
        img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        self.results = self.pose.process(img_rgb)

        if self.results.pose_landmarks and draw:
            self.mp_draw.draw_landmarks(
                img, self.results.pose_landmarks, self.mp_pose.POSE_CONNECTIONS)
        return img

    def find_landmarks(self, img, draw=True):
        self.landmark_list = []
```

```
    if self.results.pose_landmarks:
        for id, lm in enumerate(self.results.pose_landmarks.landmark):
            h, w, c = img.shape
            # print(id, lm)
            cx, cy = int(lm.x * w), int(lm.y * h)
            self.landmark_list.append([id, cx, cy])
            if draw:
                cv2.circle(img, (cx, cy), 5, (255, 0, 0), cv2.FILLED)
    return self.landmark_list

# Given any three points/co-ordinates, it gives us an angle(joint)
def find_angle(self, img, p1, p2, p3, draw=True):
    # Get the landmarks
    x1, y1 = self.landmark_list[p1][1:]
    x2, y2 = self.landmark_list[p2][1:]
    x3, y3 = self.landmark_list[p3][1:]
    # Calculate the Angle
    angle = math.degrees(math.atan2(y3 - y2, x3 - x2) -
                         math.atan2(y1 - y2, x1 - x2))
    if angle < 0:
        angle += 360


    # Draw
    if draw:
        cv2.line(img, (x1, y1), (x2, y2), (255, 255, 255), 5)
        cv2.line(img, (x3, y3), (x2, y2), (255, 255, 255), 5)
        cv2.circle(img, (x1, y1), 11, (0, 0, 255), cv2.FILLED)
        cv2.circle(img, (x1, y1), 16, (255, 60, 0), 2)
        cv2.circle(img, (x2, y2), 10, (0, 0, 255), cv2.FILLED)
        cv2.circle(img, (x2, y2), 16, (255, 60, 0), 2)
        cv2.circle(img, (x3, y3), 11, (0, 0, 255), cv2.FILLED)
        cv2.circle(img, (x3, y3), 16, (255, 60, 0), 2)

        cv2.putText(img, str(int(angle)), (x2 - 50, y2 + 60),
                    cv2.FONT_HERSHEY_DUPLEX, 1, (255, 255, 255), 1)
    return angle
```

```
    def find_coordinate(self):
        pass
def main():
    cap = cv2.VideoCapture(0)
    detector = posture_detector()
    while True:
        ret, frame = cap.read()

        pTime = 0

        img = detector.find_person(frame)
        landmark_list = detector.find_landmarks(img, draw=True)
        # angle = detector.find_angle(img, 16, 14, 12)
       # print(landmark_list)
        if len(landmark_list) != 0:

            cv2.circle(
                img, (landmark_list[14][1], landmark_list[14][2]), 15, (0, 0, 255),
cv2.FILLED)

        cTime = time.time()
        fps = 1 / (cTime - pTime)
        pTime = cTime

        cv2.putText(img, str(int(fps)), (70, 50), cv2.FONT_HERSHEY_PLAIN, 3,
                    (255, 0, 0), 3)

        cv2.imshow("Image", img)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break



    cap.release()
```

3.8 DISCUSSIONS

The results obtained from the proposed real-time fitness classification system highlight the significant potential of integrating pose-estimation methods with deep learning architectures for automated exercise recognition and performance tracking. This section expands upon the system's quantitative and qualitative performance, model behavior, limitations, and real-world deployment considerations.

The evaluation demonstrates that a pose-estimation-driven approach is both technically feasible and practically effective for real-world exercise monitoring. By leveraging BlazePose for extracting 3D body landmarks and combining them with derived joint-angle features, the system is able to precisely model human movement patterns. This ability to convert raw visual input into structured biomechanical representations greatly reduces computational overhead compared to processing raw video frames.

The system operated consistently at real-time speeds (20–25 FPS) on consumer-grade hardware, even without GPU acceleration. This validates the suitability of the architecture for deployment in mobile fitness applications, home workout setups, and embedded fitness devices. Real-time inference ensured smooth interaction with users, making the system capable of offering immediate feedback during exercise sessions.
A major component of the evaluation involved comparing two neural network architectures: a standard CNN and a BiLSTM model trained on both landmark coordinates and angle-based features.

The BiLSTM model consistently outperformed the CNN across all exercises and test scenarios. This advantage is primarily attributed to its ability to analyze motion as a temporal sequence, capturing the dynamics of human movement over time rather than treating each frame as an isolated entity. Exercises such as squats, lunges, push-ups, and jumping jacks contain characteristic motion cycles that unfold over several frames.

The CNN achieved reasonable accuracy when postures were static or when exercises had distinct poses, but it struggled when spatial similarities existed across classes.

In contrast, the BiLSTM achieved higher classification accuracy, especially in exercises where the difference lay not in body configuration but in the trajectory and timing of movements.

For instance, squats and lunges may appear similar in a single frame, but their full-body

motion paths are different. The BiLSTM's memory units learned these sequential variations effectively, resulting in significantly fewer misclassifications.

The use of hybrid features—landmark coordinates combined with biomechanically meaningful joint angles—further boosted model performance. While landmarks captured the absolute positions of joints, angles provided invariant representations that were more robust to scale, camera distance, and minor positional shifts. The fusion of these two feature types enabled a richer understanding of both spatial posture and relative joint movement.

One of the core strengths of the system was its ability to generalize across a wide variety of environments, users, and camera conditions. This robustness was achieved through a hybrid training dataset composed of both real-world exercise videos and synthetically generated samples.

3.1 Benefits of Synthetic Data

Synthetic data, created using tools like Blender or Unity-based motion capture simulations, provided:

- Controlled variations in camera angles
- Animated subjects with different body proportions
- Diverse lighting conditions
- Cleanly labeled ground-truth joint positions

This allowed the model to learn generalizable movement patterns without overfitting to the limited variability found in real-world video collections.

When tested on real-world videos sourced from gyms, home workout spaces, and outdoor areas, the system maintained high classification accuracy. The presence of:

- Different body shapes
- Clothes of varying colors
- Background clutter
- Non-uniform indoor lighting
-

did not significantly degrade performance, demonstrating strong cross-domain generalization.

Pose-Estimation Limitations

- BlazePose, though highly optimized, struggled in certain scenarios:
- Occlusions, such as equipment blocking limbs
- Rapid movements, where landmark tracking partially lagged
- Cluttered or busy backgrounds, leading to mis-detections
- Extreme side-view angles, where some joints became invisible to the camera

These issues occasionally resulted in noisy or missing landmarks, which propagated into the classification model and caused momentary misclassifications or temporary confusion during repetitions.

Sensitivity to Lighting Conditions

Low-light environments or strongly directional lighting caused the model to lose tracking accuracy. Dark clothes against dark backgrounds also reduced contrast, hindering consistent pose detection.

Camera Stability and Perspective

Although the system handled moderate camera movement well, severe shaking or tilted camera angles occasionally reduced the accuracy of angle calculations.

These findings underscore the importance of refining the pose-estimation stage or incorporating multi-view data augmentation in future work.

A notable strength of the system lies in its repetition counting algorithm, which relies on analyzing the periodic variations in joint angles over time.

Consistency and Adaptability

The algorithm accurately counted repetitions across different users and execution speeds due to its ability to detect:

- Movement peaks
- Movement troughs
- Range-of-motion thresholds
- Directional changes in joint angle trajectories
- Unlike traditional frame-based or optical-flow-based counters, this method proved to be:
- Robust to noise
- Insensitive to minor pose-estimation errors
- Generalizable across users with different flexibility levels

Handling Variations in Form

Even when users performed exercises with non-standard form—for example, partial squats or uneven push-ups—the dynamic nature of the angle pattern allowed the system to interpret repetition cycles correctly. This adaptability is crucial for consumer fitness apps where perfect form cannot be guaranteed.

CHAPTER 4: CONCLUSION

The project titled **"Real-Time Exercise Recognition and Automatic Rep Counting using Machine Learning for Fitness Applications"** was developed with the primary goal of enabling accurate, real-time tracking and feedback for fitness enthusiasts. By integrating pose estimation, exercise recognition, repetition counting, and performance feedback modules into a single unified system, the project demonstrates how AI and ML can create interactive and intelligent personal training tools.

A dataset of five common exercises — Push-Up, Squat, Bicep Curl, Shoulder Press, and Jumping Jack — was created for training and evaluation. The **MediaPipe Pose framework** was used to extract 3D keypoints of body joints, which served as input for exercise recognition models. Two models were tested: **BiLSTM** and **3D-CNN**. The BiLSTM model, which processes temporal keypoint data, achieved an impressive accuracy of **95.77%** and performed efficiently on standard CPU hardware, confirming its suitability for real-time use. The 3D-CNN model, though strong in spatial feature extraction, required larger datasets and higher computational resources, making it less practical for low-power devices. The integration of **real-time pose analysis and repetition counting** further enhanced the system's usability. The system was able to detect exercise completion, posture correctness, and repetitions automatically, providing immediate feedback to the user. For instance, a poorly executed push-up triggered corrective feedback on form, while correctly performed squats were counted and logged accurately. The inclusion of a transformer-based **motion smoothing and error correction module** ensured that noisy keypoints or minor tracking errors did not affect repetition counting or classification. This combination of temporal modeling and correction allowed the system to generate feedback that was both accurate and reliable.

The results proved that the proposed system can perform real-time exercise recognition and rep counting with minimal latency, making it ideal for practical deployment in home or gym environments. The entire framework runs offline without requiring internet connectivity, ensuring accessibility in low-resource settings. The modular design allows easy scalability, enabling future addition of more exercises, performance metrics, and multi-user tracking features. Overall, the system successfully bridges a gap in personal fitness monitoring by providing real-time, automated feedback and performance tracking, offering users a virtual trainer experience.

However, the project does have limitations. The dataset used was relatively small, which restricts the model's generalization for a wider variety of exercises or body types. Variations in lighting, camera angle, and clothing can also impact recognition accuracy. Future work should focus on expanding the dataset, incorporating advanced transformer-based temporal models, and integrating more sophisticated feedback mechanisms, including voice or haptic guidance, for enhanced user experience.

In conclusion, this project provides a strong foundation for developing intelligent, real-time fitness monitoring systems. It demonstrates that with efficient model design, machine learning can go beyond data collection to provide actionable insights and guidance — creating technology that truly empowers users to train smarter, safer, and more effectively.

In addition to dataset expansion, improving the underlying model architecture offers another path for advancement. While BiLSTM networks performed admirably, emerging architectures such as Transformers or temporal convolutional networks (TCNs) may provide stronger long-range dependency modeling and better resilience against noisy or missing pose data. These architectures could also help mitigate issues arising from slight landmark jittering, which is difficult to avoid in real-world scenarios. Similarly, the incorporation of advanced smoothing techniques or predictive filters may improve pose stability during high-speed movements, thereby reducing misclassifications in dynamic exercises like jumping jacks, burpees, or rapid lunges.

The implementation of the model inside a fully functional web application served as a practical validation of its usability in real-time settings. By running the exercise classification pipeline directly on a browser, users were able to interact with the system without requiring specialized hardware or software installations. This demonstrated the feasibility of deploying the system in everyday fitness contexts, such as home workouts, personal training sessions, telehealth consultations, and interactive fitness applications. Real-time responsiveness, high accuracy, and ease of use together highlight the system's potential for large-scale adoption in consumer fitness products or digital wellness platforms.

Moreover, the study showed that pose-based repetition counting can be an effective alternative to traditional sensor-based or frame-by-frame techniques. The repetition counting algorithm, which analyzes periodic patterns in joint-angle trajectories, proved highly robust and adaptable among different users, regardless of differences in exercise rhythm or flexibility levels. The algorithm dynamically identified peaks, troughs, and directional shifts in movement cycles, allowing it to handle inconsistencies such as partial reps, slower motion, and slight variations in form. This adaptability demonstrates the

system's potential for integration into rehabilitative settings, where users may perform exercises at lower speeds or with restricted motion ranges.

The findings of this research project collectively establish a strong case for the viability of pose-estimation–driven temporal deep-learning models in fitness analysis. The system's strengths can be summarized as follows:

- High classification accuracy across diverse exercise types
- Strong temporal modeling ability due to BiLSTM architecture
- Robust performance across synthetic and real-world test sets
- Reliable repetition counting driven by dynamic joint-angle analysis
- Real-time usability within a web-based environment
- Reduced sensitivity to variations in camera distance, perspective, and body type

At the same time, the limitations identified in this study provide a roadmap for future enhancements. Addressing the challenges posed by extreme outdoor lighting, complex indoor scenes, multi-person recordings, and fast-motion occlusions would increase the model's practical reliability. Additional improvements in computational optimization could further reduce latency, allowing seamless deployment on mobile devices and low-power embedded systems. Likewise, expanding the exercise library to include more complex, multi-step, or equipment-based exercises would significantly broaden the system's real-world applicability.

Furthermore, integrating personalized feedback mechanisms that adapt to an individual's fitness level, motion range, and posture quality could transform the system from a mere recognition tool into an intelligent workout assistant. Incorporating advanced temporal modeling may also enable deeper insights such as injury risk detection, exercise intensity evaluation, and fatigue estimation. Finally, collaboration with fitness experts, physiotherapists, and sports trainers could help refine movement standards and enhance the clinical accuracy of motion evaluation, making the system not only useful for personal workouts but also valuable for rehabilitation and professional training environments.

In essence, this research lays a strong foundation for AI-driven exercise monitoring, and with further refinement, it has the potential to redefine how fitness training and physical therapy are delivered in the modern world.

REFERENCES

[1] Dosovitskiy (2020) introduced the Vision Transformer (ViT) model, demonstrating the use of transformer architectures for image recognition — relevant for future improvements using transformer-based models in pose classification.

[2] Bazarevsky et al. (2020) presented BlazePose, an on-device real-time human pose estimation model used in this project for keypoint extraction and motion tracking.

[3] Lugaresi et al. (2019) described MediaPipe, Google's open-source framework for building perception pipelines, which serves as the foundation for BlazePose integration.

[4] Bang & Park (2024) explored workout classification using convolutional neural networks

[5] Moran et al. (2022) developed Muscle Vision, a real-time keypoint-based pose classification system, influencing the motion recognition component of this project.

[6] Talal & Chen (2020) proposed real-time exercise recognition and repetition counting, offering methods comparable to this project's approach.

[7] InfiniteRep Dataset provided synthetic exercise videos for training, improving dataset diversity and robustness.

[8] Soomro (2012) introduced UCF101, a large-scale human action dataset, serving as a reference for human activity recognition benchmarks.

[9] Cao et al. (2018) presented OpenPose, a pioneering 2D multi-person pose estimation framework, foundational to later pose estimation techniques like BlazePose.

[10] These are online exercise video sources (Kaggle, Pexels, Pixabay, Shutterstock.

[11] Pexels (2023). *Free Exercise and Workout Stock Videos.* A royalty-free video library used to gather additional real-world exercise samples for training and testing.

[12] Pixabay (2023). *Fitness and Workout Stock Footage.* A free stock video platform providing high-resolution exercise videos used to increase dataset diversity.

[13] Shutterstock (2023). *Professional Gym and Fitness Footage Collection.* Licensed stock videos offering high-quality gym recordings used to supplement real-world training samples.

[14] Storyblocks (2023). *Exercise and Sports Stock Videos.* A subscription-based stock footage source used to obtain additional exercise clips with varied backgrounds and camera angles.

[15] Netron is a visualization tool used for inspecting deep learning models and verifying BiLSTM architecture.

[16] Streamlit was employed for developing the web-based interface that hosts the exercise classification