A Mini Project Report
On

Pose-Driven Badminton Performance Evaluation

In partial fulfilment of the requirements for the award of

BACHELOR OF TECHNOLOGY

in

Computer Science and Engineering

MAMIDALA HARINI (21E51A0564)

KUPPILI VINAY KUMAR (21E51A0560)

MALLULA MUKESH (21E51A0563)

ABHISHEK JOSEPH (21E51A0558)

Under the guidance of

Mr. DHARMENDRA K ROY

Associate Professor

Department of CSE, HITAM

HYDERABAD INSTITUTE OF TECHNOLOGY AND MANAGEMENT

Autonomous, Afflilated to JNTUH, Approved by AICTE, Accredited by NAAC A+, NBA.

Basuragadi (Village), Medchal (Mandal), Medchal - Malkajgiri (Dist.), Hyderabad, TS- 501401.

2024–2025

	Table of Contents	
S.No	**TITLE**	**Page No**
	Abstract	3
1	Introduction	4
2	Literature Review	7
3	System Requirements	14
4	Problem Statement	17
5	Proposed Algorithm	18
6	UML Diagrams	20
7	Implementation	23
8	Conclusion and Future Scope	34
9	References	36

ABSTRACT

In badminton, fast reflexes and quick thinking are required for various shots, each associated with peculiarities and techniques. It is a matter of learning how to hit the shuttlecock and understanding the elements that lead to successful shots. According to our research, badminton players' performance can be evaluated through posture estimation methods. This research is based on two main objectives: the first one is to develop a method that would, based on the characterization of different kinds of badminton shots, for instance, smashes, drop shots, or net shots, identify images processed using certain algorithms-the CNN algorithm for shot type classification. This should be significant as it lays the foundation for further analysis. The other goal is to determine what makes a great shot from an otherwise bad one. We run machine learning algorithms to extract salient features that appear in the successful shot samples. For instance, such features depict enhanced areas concerning the shots of novice players. This kind of study helps beginners and intermediate players as they learn new things and get opportunities to practice targeted skills.

Keywords: Sports Analytics, Machine Learning, Image Processing, Convolutional Neural Network (CNN), Tensorflow Movenet Algorithm.

1. INTRODUCTION

Badminton is one of the sports in which players use angles and movements to deliver some effective shots. Every angle and movement matters to win matches and challenge other players. Players must fully understand each technique and the respective angles to be used for better performance.

A badminton player delivers shots, but their range depends on how much force his body produces when hitting the shuttlecock. The shot speed also depends on how the shuttlecock is hit and the positioning of the racket. Power is necessary to strike hard and position oneself to deliver a forceful blow to the shuttlecock. The body should also lean slightly to be sure that all the energy generated goes into the racket and the shuttlecock. Further, the elbow and wrist angles play an important role in perfectly hitting the shuttlecock. Even the slightest change in angle can knock the shuttlecock beyond the net or too close to the net on the other side. The posture determines the successful shots.

More specifically, how a player's body is positioned impacts their distribution of weight and the momentum of their shot. When players have proper posture, the movement by the player improves, which then benefits the shots. For instance, a good net shot entails bending the knees and lowering the center of gravity for a shot. Such stances can divert an opponent's attack and see the player back to a neutral position in no time.

This analysis points out that the stress on the muscles and joints in maintaining a fair stance. A proper stance reduces the susceptibility to overstretching or over-bending and thus diminishes the possibility of causing severe injuries. For example, in a long-distance shot, players should position the toes and knees of the leading leg in alignment so that there is no pressure built on the knee. The shot is affected by preceding and subsequent movements. Transitions between stances must flow throughout the game, alternating between shots that are either offensive or defensive.

Readiness depends on a player's posture and positioning of their body to execute the flow into subsequent shots. Regarding movement, elite players will strategize to mislead their opponents by appropriately controlling the angles of their bodies and stances.

To learn the proper body angles and positions, players perform specific warm-up exercises that aid in developing muscle memory. In training routines, repeated drills help practice the correct techniques while rectifying any mistakes that could impair performance. This means that through the use of different angles and postures, participants can determine where they need to improve in terms of shot effectiveness. The body position of a player has an impact on shot efficiency. Therefore, the correct posture, stance, and kinetic movements determine power, stability, and trickery during play. Players must continually practice and invest in high-level technologies to take their skills to the highest level.

The idea here in this project is to apply CNNs, which are advanced deep-learning models designed to work with structured grid data like images. CNNs apply filters in their convolutional layers to learn and identify hierarchical features based on spatial aspects—ranging from edges and textures to shapes and objects. This feature is useful in examining various patterns of images, and therefore, CNNs are a very essential tool in pose estimation and sporting analysis. Another mainstream model used in this project was TensorFlow MoveNet, Google's model. MoveNet is built with the idea of recognizing different poses of a human body precisely and at high frame rates to detect significant body key points. An advantage of the model is its light architecture, making it deployable on multiple devices, including even mobile phones and edge devices. It can thus successfully navigate complex environments and track multiple subjects simultaneously, which is why MoveNet should be quite appropriate for analyzing the movement intricacies of badminton players.

Fig. 1.1 Badminton Player

In this project, the focus is to employ Convolutional Neural Networks which are specifically deep learning models developed to work in structured grid data such as images. CNNs use filters in the convolutional layers to learn and discover hierarchal features from input data based on spatial features from edge and texture to shapes and objects. This capability is quite useful when one aims at analyzing different patterns on the images, making the CNNs as a basic tool when it comes to pose estimations and sports analysis.

Yet another leading model used in this project is TensorFlow MoveNet that is designed by Google. MoveNet is intended to accurately and at a high frame rate recognize different human poses, detecting significant points of people's bodies. One of the most significant features of this model is that it has a light-weighted architecture, hence it may be easily deployed and operate on the different devices, such as mobile phones, edge devices. Logically, therefore, MoveNet is shown to be able to handle complex environments and multiple subjects at a go and thus fully capable of tracking an analysis of the badminton player's complex movements.

2.LITERATURE REVIEW

2.1 Sports Action Detection and Counting Algorithm Based on Pose Estimation and Its Application in Physical Education Teaching

The paper[1] "Sports Action Detection and Counting Algorithm Based on Pose Estimation and Its Application in Physical Education Teaching" by Zhengyuan Song and Zhonghai Chen proposed an algorithm that applies 3D bone keypoint detection and pose estimation to accurately detect and count sports actions. It showed an accuracy of 96% both in playground and outdoor settings and high reliability with different participants. The algorithm is strong and effective for physical education teaching and training.

2.2 Strategy Analysis of Badminton Players Using Deep Learning from IMU and UWB Wearables

The paper [2] "Strategy Analysis of Badminton Players Using Deep Learning from IMU and UWB Wearables" is inspired by the concept of badminton players' strategies analyzed through deep learning from data gathered by IMU and UWB wearables.. It attempts to make meaningful extraction from the player's movement and position on the ground during a game for better proficiency in analysis. The paper shows how wearable technologies can actually be helpful in understanding player strategies or improving training methods by using deep learning algorithms.

2.3 AI Coach for Badminton

The project [3] "AI Coach for Badminton" by Dhruv Toshniwal, Arpit Patil, and Nancy Vachhani develops a device that improves a player's skill based on the video recording of the playing session and sensor data from the player. All the movements, the techniques of shots, footwork, and stroke angles are taken into account using TrackNet OpenPose, and YOLOv3 with wearable sensors comprising APDM Opal and XiaoYu 2.0. Neural networks test stances, techniques, and muscle orientation as well as give information on how to rectify this while monitoring the dynamics of the shuttlecock. By using the Olympics dataset, it trains models with comparisons of novices against that of professionals, always generating reports

on inefficiencies and suggestions until optimal performance is achieved.

2.4 Exploration of Applying of Pose Estimation Techniques in Table Tennis

The project[4] "Exploration of Applying Pose Estimation Techniques in Table Tennis" by Chih-Hung Wu, Te-Cheng Wu, and Wen-Bin Lin investigates the application of OpenPose in pose estimation for table tennis. It focuses on tactical analysis and enhancement using key points and player gestures retrieved from video footage. A deep learning AI model is utilized to make classifications of poses as well as assess player activities at all times. The study indicates the benefits of GPU processing when it comes to efficiency, eliminating other disadvantages including false recognition of pose through some angles and sometimes even detecting poor footwork.. The system aims to enhance the performance of players through the system by the crucial pose analysis.

2.5 Swin-Pose: Human Pose Estimation Using Swin Transformers.

The paper [5] "Swin-Pose uses Swin Transformers and feature pyramid fusion to improve human pose estimation", provides better accuracy than the CNN models like HRNet-W48 on the COCO dataset, although at a higher computation cost. It makes use of long-range dependencies and multi-scale feature integration for attaining precise keypoint detection.

2.6 Player Pose Analysis in Tennis Video based on Pose Estimation

The study [6]entitled "Player Pose Analysis in Tennis Video based on Pose Estimation" details the approach to determining the form of tennis players using the application of pose estimation along with an unsupervised Bag of Words technique. It computes the probabilities of shot success through the classification of coordinates obtained for joint positions in combination with those from shot position data. Using videos of the University of Tsukuba tennis team, the study effectively relates movements made by players, especially those at their ankles, to the results of shots.

2.7 A Survey on Single Person Pose Estimation

The paper [7] titled "Single Person Pose Estimation: A Survey" by Feng Zhang, Xiatian Zhu, and Chen Wang (2021) reviews the state-of-the-art single-person pose estimation based on

traditional methods and the latest deep learning-based solutions, including solutions to common challenges such as occlusion, multi-view integration, and precision in 2D and 3D pose estimation. The authors evaluate various datasets, metrics, and models and point out studies to improve pose estimation techniques.

2.8 A Comprehensive Review of Monocular Human Pose Estimation: Deep Learning Approaches

The paper [8] "Monocular Human Pose Estimation: A Survey of Deep Learning-based Methods" by Yucheng Chen, Yingli Tian, and Mingyi He reviews advanced deep learning techniques for monocular human pose estimation. This covers the CNN and RNN-based methods, the challenges such as occlusions and pose variations, and a review of the datasets and metrics used so far. The paper also explains all the developments taking place in the field while suggesting the future scope for further work that can enhance the accuracy and efficiency of the monocular pose estimation technique.

2.9 PoseTrack: Joint Multi-Person Pose Estimation and Tracking

The paper [9] "PoseTrack: Joint Multi-Person Pose Estimation and Tracking" by Umar Iqbal, Anton Milan, and Juergen Gall offers a framework for simultaneous multi-person pose estimation and tracking. The method unites human pose estimation with the capabilities of tracking to detect and monitor the poses across multiple at the same time. The idea presented in the paper with the novel approach can easily handle complex scenarios involving large crowds, occlusions, and various pose options, thus presenting possible solutions to problems of interest in real-time applications

2.10 Real-Time Social Distancing Detection System Utilizing Pose Information with Auto-Calibration

The study [10] titled "Real-time Social Distancing Detecting System with Auto-Calibration using Pose Information" by Gaku Nakano and Shoji Nishimura proposes a system using human pose information for real-time social distance detection. The system calculates the 3D geometric parameters, including the position and rotation of the camera, to measure distances between people. Its auto-calibration feature eliminates the interference of manual adjustments thereby making it an efficient method for tracking social distance.

2.11 Sports Analytics: A Comparison of Machine Learning Performance for Profiling Badminton Athletes

The paper[11] "Sports Analytics: A Comparison of Machine Learning Performance for Profiling Badminton Athletes" by Herbie Ewaldo Sinadia and I Made Murwantara compares the performance of different machine learning algorithms for profiling badminton athletes. The study explores various techniques to analyze and evaluate players' performances, aiming to identify key attributes that contribute to success in badminton. By applying machine learning models, the paper examines how these techniques can provide insights into athletes' strengths and weaknesses, offering a data-driven approach to enhance training and performance assessment in the sport.

2.12 3D Human Pose Estimation Through Multi-Scale Graph Convolution and Hierarchical Body Pooling

The Research [12] "3D Human Pose Estimation with Multi-scale Graph Convolution and Hierarchical Body Pooling" by Ke Huang, TianQi Sui, and Hong Wu introduces a new technique of 3D human pose estimation. The technique integrates multi-scale graph convolution along with hierarchical body pooling for significant improvement in the accuracy and robustness of predictions including overcoming the problem of occlusion and diverse configurations of the body.

2.13 Survey on Feature Fusion Techniques in Deep Learning for Generic Object Detection

The paper [13] "Feature Fusion Methods in Deep-Learning Generic Object Detection: A Survey" by Jiang Deng, Sun Bei, Su Shaojing, and Zuo Zhen gives an overview of various methodologies of feature fusion used in deep learning for object detection. The paper discusses advancements and challenges in the field of feature fusion through improving the performance of models for object detection, particularly in intricate environments.

2.14 Sports Analytics Review: Artificial Intelligence Applications, Emerging Technologies, and Algorithmic Perspective

The paper[14] "Sports Analytics Review: Artificial Intelligence Applications, Emerging Technologies, and Algorithmic Perspective" by Indrajeet Ghosh, Sreenivasan Ramasamy

Ramamurthy, Avijoy Chakma, and Nirmalya Roy offers a comprehensive review of the applications of artificial intelligence in sports analytics. It discusses how AI techniques are being used to enhance performance analysis, player monitoring, injury prediction, and strategy optimization in sports. The paper discusses new technologies in sports analytics like wearable sensors, computer vision, and deep learning; it also discusses several algorithms that have been used for extracting valuable insights from sports data. The review is keen on the increasing role of AI in transforming sports science and changing decision-making processes among coaches, athletes, and analysts.

2.15 Deep High-Resolution Representation Learning for Visual Recognition

The paper [15] "Deep High-Resolution Representation Learning for Visual Recognition" by Jingdong Wang, et al., reveals a deep learning framework aimed at visual recognition tasks, focusing on the acquisition of high-resolution representations. This methodology enhances the efficacy of visual models by utilizing high-resolution feature maps and deep neural networks for improved accuracy in visual recognition.

2.16 Player Pose Analysis in Tennis Videos Using Pose Estimation

The paper titled [16]" Player Pose Analysis in Tennis Video based on Pose Estimation" conceives a methodology to analyze the techniques of tennis players by proposing pose estimation to determine joint locations. These locations are then tagged using the Bag of Words (BoW) approach and merged with shot location for shot success prediction. The system focuses on the movements of the players, especially on ankle dynamics, linking posture to shot success, and delivers precise predictions using data from tennis matches at the University of Tsukuba.

2.17 AlphaPose: Real-Time Whole-Body Multi-Person Pose Estimation and Tracking

The paper [17]"AlphaPose: Whole-Body Regional Multi-Person Pose Estimation and Tracking in Real-Time" is a real-time system, engineered for accurate whole-body pose estimation and tracking of the face, body, hands, and feet. It employs SIKR techniques for rapid localization and Aware Identity Embedding for precise tracking. AlphaPose surpasses existing methodologies in speed and accuracy while being publicly available.

2.18 A Statistical Approach for Analyzing Technical Data from Badminton Matches Utilizing 2-D Seriate Images

The Research [18] "A Statistical Method for Analysis of Technical Data of a Badminton Match Based on 2-D Seriate Images" describes the statistical method used to analyze badminton match performance using 2-D images, in the process focusing on technical data extraction that helps to evaluate the movement and shot execution of players. The technique combines statistical models with image analysis to make performance evaluation more accurate.

2.19 Skeleton-Based Keyframe Detection Framework for Sports Action Analysis: Badminton Smash Case

The paper [19] "Skeleton-Based Keyframe Detection Framework for Sports Action Analysis: Badminton Smash Case" describes a skeleton data-based keyframe detection framework for identifying critical keyframes in badminton smash action. Such an approach helps in analyzing the performance of sports through the important detection of instances in the action flow, and thus it can be very useful for the coaches and analysts.

2.20 AI-powered Badminton Video Detection: Enhancing Gameplay Analysis and Training

The paper[20] "AI-powered Badminton Video Detection: Enhancing Gameplay Analysis and Training" by Jay Kim and Kevin Cheng explores the use of AI techniques, particularly computer vision, to enhance badminton gameplay analysis and training. It discusses how AI-driven video detection systems can track and analyze player movements, shots, and strategies in real-time from video footage. The paper highlights the benefits of automating gameplay analysis for coaches and players, such as improving performance through detailed feedback, identifying weaknesses, and enhancing training efficiency. It also addresses challenges like the accuracy of detection and the need for large datasets for training AI models, while emphasizing the potential of AI in revolutionizing badminton training and coaching methods.

2.21 The Analysis of Technical Characteristics of Badminton for Sports With Neurorobotics Under Machine Learning

The paper[21] "The Analysis of Technical Characteristics of Badminton for Sports With Neurorobotics Under Machine Learning" by Fan Wu and Huan Chen explores the use of

neurorobotics and machine learning to analyze badminton's technical characteristics. The authors focus on how technology can improve the analysis of sports techniques, specifically badminton, by combining machine learning models with neurorobotic systems.

2.22 The effect of real-time pose recognition on badminton learning performance

The paper[22] "The effect of real-time pose recognition on badminton learning performance" by Kuo-Chin Lin, Cheng-Wen Ko, Hui-Chun Hung, and Nian-Shing Chen examines the impact of real-time pose recognition technology on improving badminton learning performance.

3.SYSTEM REQUIREMENTS

a. Software Requirements

Operating System: Windows, macOS, or Linux. We have chosen to work on Windows Operating System because of its Graphic User Interface (GUI). It is easy to use and provides easy installation methods to work on the simulator.

Programming Language: Python has been used for scripting and integration of autonomous driving algorithms with the simulator. Python is an easy-to-use language with syntax which is near to natural human language, hence can also be understood by the uninitiated. It also provides various libraries and packages which have pre-defined code for Machine Learning applications, which takes away the pressure of hard-coding algorithms.

Integrated Development Environment (IDE): A Python IDE such as PyCharm, Visual Studio Code, or Jupyter Notebook can be used for developing and testing algorithms. We have chosen to work with Visual Studio Code as it is very light-in-weight, has multiple plugins which make coding easier.

Version Control: Git has been used for version control and collaboration. This software enables easy collaboration between teammates so that work can be tracked and assigned to a common account. Since this is a version control software, the code can be brought to its previous versions to reverse any implementations.

b. Hardware Requirements

Based on the software being used the hardware requirements may vary.

Processor: Intel Core i5 or AMD Ryzen 5 processor or higher is advised for smooth simulation and algorithm processing.

RAM: Minimum 8GB RAM to handle simulation software, algorithm processing, and data manipulation efficiently.

Graphics Card: NVIDIA GeForce GTX 1050 or AMD Radeon RX 560 or equivalent for rendering simulation graphics smoothly. Integrated Graphics card like IRIS xe can be used.

Storage: Solid State Drive (SSD) with at least 256GB of storage space for storing simulation data, algorithms, and development tools.

Display: Monitor with a resolution of at least 1920x1080 pixels for optimal visualization of simulation graphics and development environment.

c. Dataset

The dataset for this project has been created by deriving the images from google images. The dataset consists three main folders: train, test, and validation which are used for the model creation process. Each primary folder has subfolders named as: net shot, drive shot and smash shot folders containing the respective badminton shot images. The whole dataset consists a total of 519 images.

Fig. 3.1 Dataset

4. PROBLEM STATEMENT

In badminton, the effectiveness of a player's performance relies on their ability to execute a variety of shots with precision and efficiency. However, assessing and improving shot performance can be challenging for beginners and intermediate players. Traditional coaching methods often lack detailed and objective feedback on specific aspects of shot execution. This study aims to address this issue by developing a system that uses posture estimation and image processing techniques to classify different types of shots—such as smashes, drop shots, and net shots—using Convolutional Neural Networks (CNN). Additionally, the system will analyze key features that distinguish successful shots from unsuccessful ones through machine learning algorithms, providing actionable insights for players to enhance their technique and overall performance.

5. PROPOSED ALGORITHM

The proposed algorithm streamlines badminton training by integrating advanced technologies like Convolutional Neural Networks (CNNs) and TensorFlow MoveNet into a modular system. It begins with acquiring video or image data of players and extracting real-time pose information using MoveNet. The system preprocesses this input and employs CNNs to classify shots into categories such as smashes, drives, and net shots, ensuring accurate identification by addressing potential misclassifications.

Subsequently, MoveNet performs precise pose estimation, identifying key body landmarks such as joint angles and limb positions. These poses are compared against a database of professional player benchmarks to detect discrepancies in stance, movement, and stroke execution. The system evaluates player performance by correlating these differences with shot classification.

Finally, the algorithm generates real-time feedback and personalized recommendations, pinpointing specific areas of improvement, such as arm positioning, body alignment, or stroke timing. This data-driven, modular approach ensures targeted training and helps players enhance their skills to reach professional standards efficiently.

Modules:

1. Input Data Acquisition Module

i. **Description**: Captures video or image data of players performing badminton shots.
ii. **Key Technology**: Real-time pose data is collected using TensorFlow MoveNet for precise limb and body position estimation.

2. Shot Classification Module

i. **Description**: Processes the video or image frames using a CNN to classify shots into categories such as smashes, drives, and net shots.
ii. **Key Technology**: CNNs' ability to learn spatial hierarchies ensures accurate identification of shot types.

iii. **Misclassification Handling**: Corrects any classification errors to maintain the reliability of the system.

3. Pose Estimation Module

i. **Description**: Analyzes the player's pose during each shot, extracting key body landmarks like limb angles, torso alignment, and joint movements.
ii. **Key Technology**: TensorFlow MoveNet provides high-precision, real-time pose estimation.

4. Pose Comparison and Analysis Module

i. **Description**: Compares the beginner's pose with professional players' poses stored in a reference database.
ii. **Key Focus Areas**: Stance, arm tilt, body rotation, stroke timing, and movement execution.
iii. **Output**: Highlights the differences and identifies specific areas for improvement.

5. Performance Evaluation Module

i. **Description**: Evaluates the player's performance by correlating the classified shot type with pose analysis.
ii. **Key Insights**: Provides detailed assessments of shot power, accuracy, and technique.

6. Feedback and Recommendation Module

i. **Description**: Generates real-time, actionable feedback based on the analysis.
ii. **Use Case**: Offers targeted advice to players and coaches, focusing on specific aspects like arm positioning, body alignment, or stroke timing.

6.UML Diagrams

The process of software engineering utilizes the Unified Modelling Language (UML), which is a graphical language. In UML each symbol has well specified semantics. This is to ensure that if one developer create a model in UML, then other developer or other tools can interpret the models unambiguously. With UML, developers can model a system's behavioral, structural, and interactional aspects. There are many different types of diagrams offered, such as class diagrams, use case diagrams, sequence diagrams, activity diagrams, and more. Every figure serves a distinct purpose and shows several aspects of the system's operation, architecture, and interactions. Stakeholders with varying levels of technical proficiency can easily comprehend visual models produced with UML.

i. Use Case Diagram

This part of UML system is all about the provision of necessary attributes for the system, they are specifically focused on behavior modeling. Use case diagrams provide an overall view of how a system interacts with its environment. They also incorporate interaction between the business stakeholders and the system. While use case diagram and use cases identify actors in the system as well as what they 'tell' actors do to the system, they do not show how it works from inside. Use-case diagrams describe and dictate either scope or interaction entailed by the whole system or major components of this very system. Hence, one can describe a complex system simply through single use case diagram or beneath it they may explain different fields for one specific systems.

Fig. 6.1 Use Case Diagram

ii. Architecture Diagram

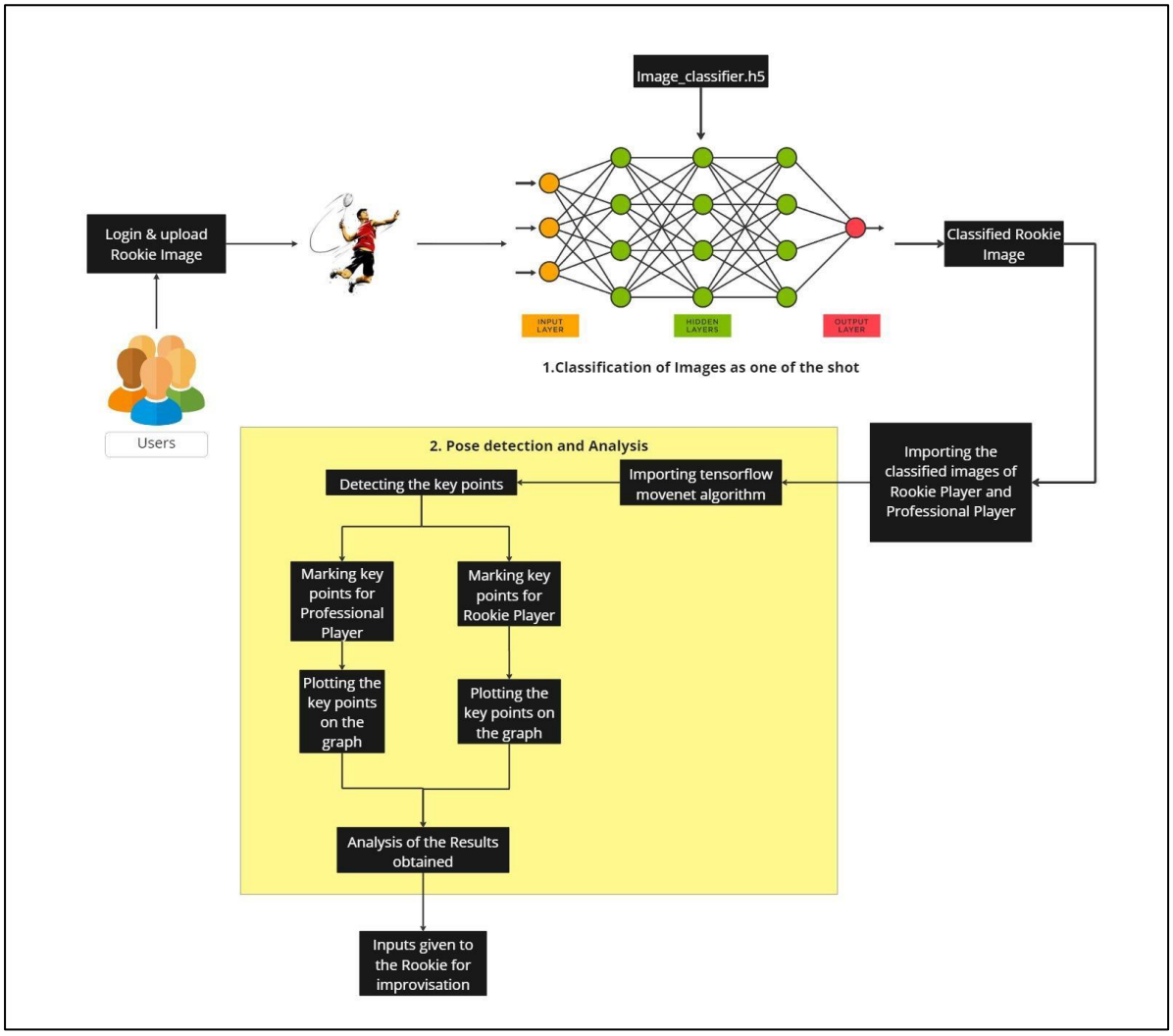

Fig. 6.2 Architecture Diagram

7.IMPLEMENTATIONS, EXPERIMENTAL RESULTS & TEST CASES

app.py

```
import tensorflow as tf
from tensorflow import keras
from keras.models import load_model
import matplotlib as plt
import streamlit as st
import numpy as np
import random
import os
from pathlib import Path
import test

st.title("Badminton Player analysis")
st.header("Classification of shot: ")
st.write("Upload image of the beginner player: ")

model =
load_model('D:\\CODING\\ML\\A10_MINI_PROJECT\\Image_classification\\Image_classify.h5')
data_cat = ['Drive', 'Net shot', 'Smash']
img_height = 180
img_width = 180

# Get the current working directory
cwd = os.getcwd()

# Create a folder to store the uploaded images (if it doesn't exist)
upload_folder = Path(cwd, "uploaded_images")
upload_folder.mkdir(exist_ok=True)

image = st.file_uploader("Upload the image")

if image is not None:
    # Save the uploaded file to the "uploaded_images" folder
    file_path = Path(upload_folder, image.name)
    with open(file_path, "wb") as file:
        file.write(image.getbuffer())
    st.success(f"File '{image.name}' uploaded successfully!")

    # Load the saved image
    image_load = tf.keras.utils.load_img(str(file_path), target_size=(img_height, img_width))
    img_arr = tf.keras.utils.array_to_img(image_load)
    img_bat = tf.expand_dims(img_arr, 0)

    predict = model.predict(img_bat)
```

```
    score = tf.nn.softmax(predict)
    st.image(image, width=200)
    st.success('The shot is of ' + data_cat[np.argmax(score)] + ' type')
    st.write('With accuracy of ' + str(np.max(score) * 100))

    # Pose estimation
    st.header("Pose estimation: ")
    # Selection of professional player drop down list
    prof_player = st.selectbox("Select the Professional Player for comparision : ", ['Select Option',
'Lee Chong Wei', 'Lin Dan', 'PV Sindhu', 'Taufik Hidayat', 'Saina Nehwal'])
    if prof_player != 'Select Option':
        type = data_cat[np.argmax(score)]

        player_folder = os.path.join('D:\\CODING\\ML\\A10_MINI_PROJECT\\Professional\\',
prof_player, type)

        # Get a list of all files in the player's folder
        file_list = os.listdir(player_folder)

        # Filter the list to include only image files (you can modify the extensions as needed)
        image_files = [f for f in file_list if f.endswith(('.jpg', '.png', '.jpeg'))]

        if image_files:
            # Choose a random image file from the list
            random_image = random.choice(image_files)

            # Construct the full path to the random image file
            prof = os.path.join(player_folder, random_image)

        # navigating to the folder of the player
        # prof = os.path.join('D:\\CODING\\ML\\A10_MINI_PROJECT\\Professional\\', prof_player,
type, '1.jpg')

        pose_graph, angles_graph, comparision_statements = test.main(str(file_path), prof)
        st.pyplot(pose_graph)
        st.pyplot(angles_graph)

        for statement in comparision_statements:
            st.success(statement)

    else:
        st.warning("Please select a professional player.")



else:
    st.warning("Please upload an image.")
```

application.py

```
import tensorflow as tf
from tensorflow import keras
from keras.models import  load_model
import streamlit as st
import numpy as np
import os
import test

st.title("Badminton Player analysis")
st.header("Classification of shot: ")
st.write("Upload image of the beginner player: ")

model =
load_model('D:\\CODING\\ML\\A10_MINI_PROJECT\\Image_classification\\Image_classify.h5')
data_cat = ['Drive',
 'Net shot',
 'Smash']
img_height = 180
img_width = 180
# image =st.text_input('Enter Image
name','D:\\CODING\\ML\\A10_MINI_PROJECT\\Images\\test1.jpg')

image = st.file_uploader("Upload the image")

image_load = tf.keras.utils.load_img(image, target_size=(img_height,img_width))
img_arr = tf.keras.utils.array_to_img(image_load)
img_bat=tf.expand_dims(img_arr,0)

predict = model.predict(img_bat)

score = tf.nn.softmax(predict)
st.image(image, width=200)
st.success('The shot is of ' + data_cat[np.argmax(score)] + ' type')
st.write('With accuracy of ' + str(np.max(score)*100))

type = data_cat[np.argmax(score)] + ".jpg"

prof = os.path.join('D:\\CODING\\ML\\A10_MINI_PROJECT\\Professional\\', type)

test.main(image, prof)
```

test.py

```
import tensorflow as tf
import numpy as np
import cv2
import math
import matplotlib.pyplot as plt

# Define EDGES dictionary with color mappings for each edge
EDGES = {
    (0, 1): (255, 182, 193),  # Nose to Left eye (Light Pink)
    (0, 2): (255, 182, 193),  # Nose to Right eye (Light Pink)
    (1, 3): (255, 105, 180),  # Left eye to Left ear (Hot Pink)
    (2, 4): (255, 105, 180),  # Right eye to Right ear (Hot Pink)
    (0, 5): (135, 206, 250),  # Nose to Left shoulder (Light Sky Blue)
    (0, 6): (135, 206, 250),  # Nose to Right shoulder (Light Sky Blue)
    (5, 7): (255, 165, 0),    # Left shoulder to Left elbow (Orange)
    (7, 9): (255, 69, 0),     # Left elbow to Left wrist (Red Orange)
    (6, 8): (255, 165, 0),    # Right shoulder to Right elbow (Orange)
    (8, 10): (255, 69, 0),    # Right elbow to Right wrist (Red Orange)
    (5, 6): (173, 216, 230),  # Left shoulder to Right shoulder (Light Blue)
    (5, 11): (144, 238, 144), # Left shoulder to Left hip (Light Green)
    (6, 12): (144, 238, 144), # Right shoulder to Right hip (Light Green)
    (11, 13): (0, 128, 0),    # Left hip to Left knee (Green)
    (13, 15): (34, 139, 34),  # Left knee to Left ankle (Forest Green)
    (12, 14): (0, 128, 0),    # Right hip to Right knee (Green)
    (14, 16): (34, 139, 34)   # Right knee to Right ankle (Forest Green)
}

# Define angles to calculate (edges with three points)
ANGLE_EDGES = {
    (5, 7, 9): 'Left Shoulder - Left Elbow - Left Wrist',
    (6, 8, 10): 'Right Shoulder - Right Elbow - Right Wrist',
    (11, 13, 15): 'Left Hip - Left Knee - Left Ankle',
    (12, 14, 16): 'Right Hip - Right Knee - Right Ankle'
}

# Define the desired dimensions for resizing
target_width = 256
target_height = 256

def preprocess_image(image_path, target_width, target_height):
    # Load the image
    image = cv2.imread(image_path)

    # Resize the image
    resized_image = cv2.resize(image, (target_width, target_height))

    # Reshape image for model input
```

```
    img = tf.image.resize_with_pad(np.expand_dims(resized_image, axis=0), target_width,
target_height)
    input_image = tf.cast(img, dtype=tf.float32)

    return input_image, resized_image

# Load the TensorFlow Lite interpreter
interpreter =
tf.lite.Interpreter(model_path="D:\\CODING\ML\\A10_MINI_PROJECT\\PoseEstimation\\t_3.tflite
")
interpreter.allocate_tensors()

def get_keypoints(input_image, input_details, output_details):
    interpreter.set_tensor(input_details[0]['index'], np.array(input_image))
    interpreter.invoke()
    keypoints_with_scores = interpreter.get_tensor(output_details[0]['index'])
    return keypoints_with_scores

# print(keypoints_with_scores1)
# print(keypoints_with_scores2)

# Rendering functions
def draw_connections(frame, keypoints, edges, confidence_threshold):
    y, x, c = frame.shape
    shaped = np.squeeze(np.multiply(keypoints, [y, x, 1]))

    for edge, edge_color in edges.items():
        p1, p2 = edge
        y1, x1, c1 = shaped[p1]
        y2, x2, c2 = shaped[p2]

        if (c1 > confidence_threshold) & (c2 > confidence_threshold):
            # print(c1,c2)
            cv2.line(frame, (int(x1), int(y1)), (int(x2), int(y2)), edge_color, 4)  # Increase line thickness
to 4

def draw_keypoints(frame, keypoints, confidence_threshold):
    y, x, c = frame.shape
    shaped = np.squeeze(np.multiply(keypoints, [y, x, 1]))

    for kp in shaped:
        ky, kx, kp_conf = kp
        if kp_conf > confidence_threshold:
            cv2.circle(frame, (int(kx), int(ky)), 4, (0, 0, 0), -1)

def calculate_angle(p1, p2, p3):
    a = np.array(p1[:2])
    b = np.array(p2[:2])
    c = np.array(p3[:2])
```

```
    ab = a - b
    cb = c - b

    cosine_angle = np.dot(ab, cb) / (np.linalg.norm(ab) * np.linalg.norm(cb))
    angle = np.arccos(cosine_angle)

    return np.degrees(angle)

# Extract x, y coordinates of keypoints
def get_keypoints_xy(keypoints_with_scores, target_height, target_width):
    return np.squeeze(np.multiply(keypoints_with_scores[..., :2], [target_height, target_width]))

# print(keypoints_xy1)
# print(keypoints_xy2)

def get_angle_dict(keypoints, edges):
    angles = {}
    for edge in edges:
        if len(edge) == 3:  # Check if edge definition includes three points for angle calculation
            p1, p2, p3 = edge
            angles[edge] = calculate_angle(keypoints[p1], keypoints[p2], keypoints[p3])
    return angles

def plot_pose_with_angles(keypoints1, keypoints2, edges):
    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 6))

    # Find the maximum x and y coordinates among all keypoints
    max_x = max(np.max(keypoints1[:, 1]), np.max(keypoints2[:, 1]))
    max_y = max(np.max(keypoints1[:, 0]), np.max(keypoints2[:, 0]))
    max_value = max(max_x, max_y)

    # Set the same limits for both x and y axes
    ax1.set_xlim(0, max_value)
    ax1.set_ylim(0, max_value)
    ax2.set_xlim(0, max_value)
    ax2.set_ylim(0, max_value)

    # Draw grid lines with higher resolution
    ax1.set_xticks(np.arange(0, max_value, 16))
    ax1.set_yticks(np.arange(0, max_value, 16))
    ax1.grid(True, linestyle='--', color='gray', alpha=0.5)
    ax2.set_xticks(np.arange(0, max_value, 16))
    ax2.set_yticks(np.arange(0, max_value, 16))
    ax2.grid(True, linestyle='--', color='gray', alpha=0.5)

    # Draw pose connections for(BEGINNER)
    ax1.plot([], [], 'b-', linewidth=2, label='BEGINNER')

    # Draw pose connections for (PROFESSIONAL)
    ax2.plot([], [], 'r-', linewidth=2, label='PROFESSIONAL')
```

```
    # print(edges)
    # print(keypoints2)
    # print(EDGES)

    # Draw pose connections for input.jpg
    # for edge in edges:
    #     edge1 = edge[:2]
    #     edge2 = edge[1:]

    #     if len(edge1) == 2:  # If edge definition includes two points for line drawing
    #         p1, p2 = edge1
    #         # print(p1,p2)
    #         ax1.plot([keypoints1[p1][1], keypoints1[p2][1]], [keypoints1[p1][0], keypoints1[p2][0]], 'b-
', linewidth=2)
    #     if len(edge2) == 2:  # If edge definition includes two points for line drawing
    #         p1, p2 = edge2
    #         # print(p1,p2)
    #         ax1.plot([keypoints1[p1][1], keypoints1[p2][1]], [keypoints1[p1][0], keypoints1[p2][0]], 'b-
', linewidth=2)

    # Draw pose connections for image2.jpg
    # for edge in edges:
    #     if len(edge) == 2:  # If edge definition includes two points for line drawing
    #         p1, p2 = edge
    #         ax2.plot([keypoints2[p1][1], keypoints2[p2][1]], [keypoints2[p1][0], keypoints2[p2][0]], 'r-
', linewidth=2)


    for edge in EDGES:
        p1, p2 = edge
        ax1.plot([keypoints1[p1][1], keypoints1[p2][1]], [keypoints1[p1][0], keypoints1[p2][0]], 'b-',
linewidth=2)
        ax2.plot([keypoints2[p1][1], keypoints2[p2][1]], [keypoints2[p1][0], keypoints2[p2][0]], 'r-',
linewidth=2)

    # Plot the angles at keypoints for input.jpg
    angle_dict1 = get_angle_dict(keypoints1, edges)
    for edge, angle in angle_dict1.items():
        p1, p2, p3 = edge
        ax1.text(keypoints1[p2][1], keypoints1[p2][0], f'{angle:.1f}', fontsize=12, color='red')

    # Plot the angles at keypoints for image2.jpg
    angle_dict2 = get_angle_dict(keypoints2, edges)
    for edge, angle in angle_dict2.items():
        p1, p2, p3 = edge
        ax2.text(keypoints2[p2][1], keypoints2[p2][0], f'{angle:.1f}', fontsize=12, color='blue')

    # Add legend
```

```
    ax1.legend()
    ax2.legend()

    # Invert y-axis
    ax1.invert_yaxis()
    ax2.invert_yaxis()

    # Add titles
    ax2.set_title('PROFESSIONAL')
    ax1.set_title('BEGINNER')

    plt.tight_layout()
    fig = plt.gcf()
    return fig, angle_dict1, angle_dict2

# Draw keypoints and connections on the original images
def draw_pose_on_image(image, keypoints_with_scores, edges, confidence_threshold):
    image_copy = image.copy()
    draw_connections(image_copy, keypoints_with_scores, edges, confidence_threshold)
    draw_keypoints(image_copy, keypoints_with_scores, confidence_threshold)
    return image_copy


# Plot pose with angles

def angles_plot(keypoints_xy1, keypoints_xy2):
    fig, angle_dict1, angle_dict2 = plot_pose_with_angles(keypoints_xy1, keypoints_xy2,
list(ANGLE_EDGES.keys()))

    comparision_statements = ["\nComparison of angles between poses:"]
    # Compare angles
    print("\nComparison of angles between poses:")
    for edge, description in ANGLE_EDGES.items():
        angle1 = angle_dict1.get(edge, None)
        angle2 = angle_dict2.get(edge, None)
        if angle1 and angle2:
            statement = (
                f'{description}:\n\n'
                f'  BEGINNER: {angle1:.1f} degrees\n\n'
                f'  PROFESSIONAL: {angle2:.1f} degrees\n\n'
                f'  Difference: {(angle1 - angle2):.1f} degrees\n'
            )
        comparision_statements.append(statement)

    return fig, comparision_statements

def poseEstimation(image1, image2, keypoints_with_scores1, keypoints_with_scores2):
    image1_with_pose = draw_pose_on_image(image1, keypoints_with_scores1, EDGES, 0.4)
    image2_with_pose = draw_pose_on_image(image2, keypoints_with_scores2, EDGES, 0.4)
```

```
    # Display the images with poses
    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(24, 12))

    ax1.imshow(cv2.cvtColor(image1_with_pose, cv2.COLOR_BGR2RGB))
    ax1.set_title('BEGINNER')

    ax2.imshow(cv2.cvtColor(image2_with_pose, cv2.COLOR_BGR2RGB))
    ax2.set_title('PROFESSIONAL')

    fig = plt.gcf()
    return fig

def main (img1, img2):
    # Load and preprocess the images
    input_image1, image1 = preprocess_image(img1, target_width, target_height)
    input_image2, image2 = preprocess_image(img2, target_width, target_height)

    # Get input and output details
    input_details = interpreter.get_input_details()
    output_details = interpreter.get_output_details()

    keypoints_with_scores1 = get_keypoints(input_image1, input_details, output_details)
    keypoints_with_scores2 = get_keypoints(input_image2, input_details, output_details)

    keypoints_xy1 = get_keypoints_xy(keypoints_with_scores1, target_height, target_width)
    keypoints_xy2 = get_keypoints_xy(keypoints_with_scores2, target_height, target_width)

    # Calling pose estimation function to display images
    pose_graph = poseEstimation(image1, image2, keypoints_with_scores1, keypoints_with_scores2)

    # Angle plotting
    angles_graph, comparision_statements = angles_plot(keypoints_xy1, keypoints_xy2)
    return pose_graph, angles_graph, comparision_statements

# image1 = input("Enter the address of image1: ")
# image2 = input("Enter the address of image2: ")
# main(image1, image2)
```

Pose Estimation outputs

Test-Observations

Test - Insights

8.CONCLUSION AND FUTURE SCOPE

b. Conclusion:

This project demonstrated the great effectiveness of Convolutional Neural Networks (CNNs) and TensorFlow MoveNet in enhancing badminton performance analysis. By employing modern technology, the system effectively classifies badminton shots such as smashes, drives, and net shots using CNNs and accurately identifies the players' poses through TensorFlow MoveNet. This dual functionality not only categorizes shots but also provides a detailed understanding of the biomechanics behind them, helping players pinpoint areas for improvement.

The dataset used, derived from Google Images, was meticulously structured and split into training, testing, and validation sets, which played a critical role in training the CNN. This robust dataset ensured the model's ability to generalize well with new data. Real-time pose estimation, enabled by TensorFlow MoveNet, allows players to receive immediate feedback and make rapid adjustments, expediting the learning process.

The project successfully highlighted the potential of integrating deep learning and pose estimation for parameterizing and analyzing the kinetic characteristics of badminton players' performance. It effectively identified areas of concern by comparing poses and movements of amateur and professional players, offering an invaluable advantage in reforming sports training methodologies.

Additionally, the inclusion of advanced methods like real-time pose estimation facilitated the recognition of different postures and movements, which are essential for delivering actionable feedback. This approach demonstrates the potential of AI in athletic training, further bridging the gap between amateurs and professionals. The system's potential can be further amplified by incorporating a larger dataset, additional shot categories, or even 3D pose estimation, paving the way for even more comprehensive and impactful training solutions.

c. Future Scope:

The current system establishes a strong foundation for badminton player analysis, focusing on shot classification and pose analysis. However, several enhancements can be implemented to further improve its effectiveness:

- **Larger Dataset Utilization**: Incorporating a more extensive dataset would enable the model to train more effectively, leading to greater robustness and accuracy.
- **Expanded Shot Classification**: Beyond categorizing shots into drives, net shots, and smashes, the system can be enhanced to include additional shot types. This would allow for a more thorough and detailed analysis of player performance.
- **3D Pose Estimation**: Introducing 3D analysis would provide deeper insights into the player's performance by delivering a more comprehensive understanding of biomechanics and strategic skills.
- **User-Friendly Interface**: Developing a mobile application for Android and iOS platforms can make the system more accessible and scalable. Features such as gameplay video uploads, professional strategies, and a dashboard to track player progress and performance would enhance usability.
- **Automated Performance Reporting**: Automating the generation of performance reports would simplify gameplay analysis for users. This feature could highlight skill development, pinpoint areas for improvement, and provide actionable insights.

9.REFERENCES

[1] Zhengyuan Song, Zhonghai Chen, "Sports Action Detection and Counting Algorithm Based on Pose Estimation and Its Application in Physical Education Teaching," Informatics, vol. 48, no. 10, DOI: 10.31449/inf.v48i10.5918.

[2] Ben Van Herbruggen, Jaron Fontaine, Jonas Simoen, Lennert De Mey, Daniel Peralta, Adnan Shahid, Eli De Poorter, "Strategy Analysis of Badminton Players Using Deep Learning from IMU and UWB Wearables," Internet of Things, 2024, DOI: 10.1016/j.iot.2024.101260.

[3] D. Toshniwal, A. Patil, and N. Vachhani, "AI Coach for Badminton," 2022 3rd International Conference on Emerging Technology (INCET), Belgaum, India, 2022, pp. 1–7.

[4] C.-H. Wu, T.-C. Wu, and W.-B. Lin, "Investigation of Pose Estimation Techniques in Table Tennis," Appl. Sci., 2023.

[5] Z. Xiong, C. Wang, Y. Li, and Y. Cao, "Swin-Pose: Human Pose Estimation Using Swin Transformers," arXiv preprint, arXiv:2201.07384, 2022. Available: https://arxiv.org/abs/2201.07384

[6] R. Kurose, M. Hayashi, T. Ishii, and Y. Aoki, "Analysis of Player Poses in Tennis Videos Through Pose Estimation," 2018 International Workshop on Advanced Image Technology (IWAIT).

[7] F. Zhang, X. Zhu, and C. Wang, "A Survey on Single Person Pose Estimation," arXiv preprint, arXiv:2109.10056, 2021.

[8] Y. Chen, Y. Tian, and M. He, "A Comprehensive Review of Monocular Human Pose Estimation: Deep Learning Approaches," Computer Vision and Image Understanding, vol. 192, pp. 102-897, 2020.

[9] U. Iqbal, A. Milan, and J. Gall, "Posetrack: Integrated Approach for Multi-Person Pose Estimation and Tracking," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2011–2020.

[10] G. Nakano and S. Nishimura, "Real-Time Social Distancing Detection System Utilizing Pose Information with Auto-Calibration," The First International Conference on AI-ML-Systems, pp. 1–3, 2021.

[11] Herbie Ewaldo Sinadia, I Made Murwantara, "Sports Analytics: A Comparison of Machine Learning Performance for Profiling Badminton Athlete," Proceedings of the 2022 1st International Conference on Technology Innovation and Its Applications (ICTIIA), IEEE, DOI: 10.1109/ICTIIA54654.2022.9935852.

[12] K. Huang, T. Sui, and H. Wu, "3D Human Pose Estimation Through Multi-Scale Graph Convolution and Hierarchical Body Pooling," Multimed. Syst., vol. 28, pp. 403–412, 2022.

[13] J. Deng, S. Bei, S. Shaojing, and Z. Zhen, "A Survey on Feature Fusion Techniques in Deep Learning for Generic Object Detection," 2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), vol. 9, pp. 431–437, 2020.

[14] Indrajeet Ghosh, Sreenivasan Ramasamy Ramamurthy, Avijoy Chakma, Nirmalya Roy, "Sports Analytics Review: Artificial Intelligence Applications, Emerging Technologies, and Algorithmic Perspective," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, First published: 21 March 2023, DOI: 10.1002/widm.1496.J

[15] R. Kurose, M. Hayashi, T. Ishii, and Y. Aoki, "Player Pose Analysis in Tennis Videos Using Pose Estimation," Proceedings of the IEEE 14th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery, 2018.

[16] H. S. Fang, J. Li, H. Tang, C. Xu, H. Zhu, Y. Xiu, Y. L. Li, and C. Lu, "AlphaPose: Real-Time Whole-Body Multi-Person Pose Estimation and Tracking," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022.

[17] B. Chen and Z. Wang, "A Statistical Approach for Analyzing Technical Data from Badminton Matches Utilizing 2-D Seriate Images," TUP, [Online].

[18] M. A. Sarwar, Y.-C. Lin, Y. A. Daraghmi, T.-U. İK, and Y.-L. Li, "Framework for Skeleton-Based Keyframe Detection in Sports Action Analysis: Badminton Smash Case," IEEE.

[19] N. Ding, K. Takeda, and K. Fujii, "Utilizing Deep Reinforcement Learning in Racket Sports for Evaluating Players within Technical and Tactical Contexts," IEEE Access, vol. 10, pp. 54764–54772, 2022.

[20] Jay Kim, Kevin Cheng, "AI-powered Badminton Video Detection: Enhancing Gameplay Analysis and Training," TechRxiv, July 21, 2023, DOI: 10.36227/techrxiv.23708325.v1.

[21] Fan Wu, Huan Chen, "The Analysis of Technical Characteristics of Badminton for Sports With Neurorobotics Under Machine Learning," IEEE Access, vol. 11, pp. 144337-144348, December 21, 2023. DOI: 10.1109/ACCESS.2023.101260.

[22] Kuo-Chin Lin, Cheng-Wen Ko, Hui-Chun Hung, Nian-Shing Chen, "The effect of real-time pose recognition on badminton learning performance," Educational Technology Research and Development, vol. 69, pp. 4772–4786, DOI: 10.1080/10494820.2021.1981396.