INTELLIGENT SYSTEM OF BADMINTON ACTION BASED ON DEEP LEARNING NETWORKS

SYAZANI BIN SYAMIZEY

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

INTELLIGENT SYSTEM OF BADMINTON ACTION BASED ON DEEP LEARNING NETWORKS

SYAZANI BIN SYAMIZEY

This report is submitted in partial fulfilment of the requirements for the degree of Bachelor of Computer Engineering with Honours

Faculty of Electronics and Computer Technology and Engineering
Universiti Teknikal Malaysia Melaka

2024

DECLARATION

I declare that this report entitled “Intelligent System Of Badminton Action Based On Deep Learning Networks” is the result of my own work except for quotes as cited in the references.

Signature : Syazani.....

Author : Syazani Bin Syamizey.....

Date : 24 June 2024.....

APPROVAL

I hereby declare that I have read this thesis and in my opinion this thesis is sufficient in terms of scope and quality for the award of Bachelor of Computer Engineering with Honours.

Signature : .....

Supervisor Name : Prof Madya Dr Abd Majid Bin Darsono

Date : 27 June 2024

DEDICATION

This thesis is dedicated to the memory of my dear mother, Lilis Haryanti. Her tenacity and kindness continue to inspire me every day. To my father, Syamizey bin Shawal, whose unfailing support and advice have served as the cornerstone for my life. To my family, for their unwavering support and believe in me. To my significant other, whose love and patience have supported me throughout this journey. I also like to thank my supervisor, Dr. Abd Majid bin Darsono, for his tremendous guidance and assistance. Thank you for your unwavering support and love. This achievement is as much yours as mine.

ABSTRACT

Computer vision plays a vital role in sports analytics by facilitating the automated identification, monitoring, and examination of players' motions and activities. By combining the strengths of YOLOv7 Pose Estimation and LSTM models, this research project aims to construct an intelligent badminton action detection system. By precisely identifying and classifying badminton movement, such as smash and serve, the goal was to improve the analysis of badminton shot classification in video footage. To do this, a specific badminton match recordings were gathered from YouTube, and individual shot instances were identified. The X-coordinate, Y-coordinate, and confidence ratings of each frame were extracted using YOLOv7 Pose Estimation. These key points were arranged into thirty-frame sequences, resulting in fifty-one features per sequence. Based on this key point data, an LSTM model was then trained to predict badminton shot motions. The study specifically discovered that the combination of pose estimation and LSTM resulted in an impressive accuracy rate of 97%. In addition, alternative algorithms such as GRU and CNN achieved somewhat lower accuracy rates of 93% and 90% respectively. Robust action detection in badminton matches is possible because to the integration of YOLOv7 for posture estimation and LSTM for sequence learning, which produced encouraging results.

ABSTRAK

Dalam analisis sukan, penglihatan komputer membantu dalam pengenalan automatik, pemantauan, dan pemeriksaan pergerakan dan aktiviti pemain. Tujuan projek penyelidikan ini adalah untuk mencipta sistem pengesanan tindakan badminton yang pintar dengan menggabungkan kekuatan model YOLOv7 Pose Estimation dan LSTM. Matlamatnya ialah untuk meningkatkan analisis klasifikasi pukulan badminton dalam video dengan mengklasifikasikan pukulan seperti smash dan serve. Untuk mencapai matlamat, rekod perlawanan badminton dari YouTube dikumpulkan dan tembakan individu telah dikenal pasti. Menggunakan YOLOv7 Pose Estimation, penilaian untuk koordinat X, koordinat Y dan kepercayaan untuk setiap bingkai dikeluarkan. Per urutan terdapat tiga puluh titik utama, yang menghasilkan lima puluh satu ciri. Model LSTM kemudiannya dilatih untuk meramalkan gerakan tembakan badminton berdasarkan data titik utama ini. Secara khusus, kajian ini menunjukkan bahawa gabungan estimasi kedudukan dan LSTM memberikan kadar ketepatan yang mengesankan sebanyak 97%. Tambahan pula, algoritma alternatif seperti CNN dan GRU mencapai kadar ketepatan yang agak rendah sebanyak 90%. Integrasi YOLOv7 dan LSTM membolehkan pengesanan tindakan yang kukuh dalam perlawanan badminton. Keputusan yang menggalakkan telah dihasilkan sebagai hasil daripada ini.

ACKNOWLEDGEMENTS

Above all, I express profound gratitude to my supervisor, Dr. Abd Majid bin Darsono, for his essential mentorship, assistance, and motivation during this project. The completion of this thesis was much facilitated by his skills and insights, and I am deeply grateful for his patience and dedication. I wish to express my sincere gratitude to my family for their steadfast support. My father, Syamizey bin Shawal, has been a profound influence on me, imparting wisdom and instilling strong beliefs, which have formed a solid basis for my development. I am grateful to my extended family as well, whose unwavering belief in my capabilities has propelled my achievements, and whose existence in my life I profoundly value. I am deeply indebted to the teachings and principles of my late mother, Lilis Haryanti, whose unwavering motivation has played a significant role in shaping my character and fostering my determination and commitment. My partner, I greatly appreciate the immeasurable value of your love, patience, and understanding. Finally, I would want to show my appreciation to all those who have made valuable contributions to my academic career. Regardless of the magnitude of your assistance, it has contributed to my achievement.

TABLE OF CONTENTS

Declaration

Approval

Dedication

Abstract **i**

Abstrak **ii**

Acknowledgements **iii**

Table of Contents **iv**

List of Figures **viii**

List of Tables **x**

List of Symbols and Abbreviations **xi**

List of Appendices **xii**

CHAPTER 1: INTRODUCTION **1**

1.1 Background of Project 1

1.2 Problem Statement 3

1.3 Objectives 4

1.4 Scope Of Project 4

1.5 Project Significance 6

1.6 Thesis Outline 6

CHAPTER 2: BACKGROUND STUDY 7

2.1 Badminton – Second Most Popular Sport in The World 7

2.2 Action Recognition 9

2.3 Pose Estimation 10

2.4 LSTM (Long Short-Term Memory) 13

2.5 Badminton Action Recognition 16

2.5.1 Sensor Based Approach 16

2.5.2 Pose Estimation Approach 17

2.5.2.1 Pose Estimation Model 18

2.5.3 Trajectory Analysis Method 19

2.6 Review of Related Work 20

2.7 Action Recognition Using Another Types of Neural Networks 23

2.7.1 Convolutional Neural Networks (CNNs) 23

2.7.2 Gate Recurrent Unit (GRU) 26

CHAPTER 3: METHODOLOGY 28

3.1 Overall Project 29

3.2 Dataset Collection and Preprocessing 30

3.3 Dataset Preprocessing Algorithm Development (KeypointFinder.py). 32

3.4 Dataset Preprocessing by Algorithm 32

3.5 LSTM Model Training 34

3.5.1 LSTM Model Architecture Design 35

3.5.2 Model Training and Evaluation 36

3.6 Development of Simulation Framework Algorithm for Shot Classification. 39

CHAPTER 4: RESULTS AND DISCUSSION 41

4.1 Hyperparameter Tuning 42

4.1.1 Learning Rate, Batch Size and Epochs 42

4.2 Analyzing The Performance of the Proposed LSTM Model 44

4.2.1 Model Accuracy and Loss 45

4.2.2 Confusion Matrix 47

4.3 Comparing Models 50

4.3.1 Gated Recurrent Unit 50

4.3.2 Convolutional Neural Networks (CNNs) 52

4.4 Comparison Between a Proposed Model and Other Models 54

4.5 Real-Time Simulation 56

4.5.1 Benchmarking AI Model Performance Against Human Observation 57

4.5.2 Environment and Sustainability 60

CHAPTER 5: CONCLUSION AND FUTURE WORKS 62

5.1 Conclusion 62

5.2 Future Works 63

5.2.1 Dataset Expansion and Diversity 63

5.2.2 Integration of Shuttlecock Trajectory Data 63

5.2.3 Fine-tuning Hyperparameters and Model Architecture 64

REFERENCES **65**

APPENDICES **77**

LIST OF FIGURES

Figure 2.1 : Classification framework for human action recognition methods [17]....9

Figure 2.2: Configuration of valid human pose estimation [17].....11

Figure 2.3: Multi-person pose estimation. Body parts belonging to the same person are linked, including foot key-points (big toes, small toes, and heels) [22] 11

Figure 2.4 : Output of OpenPose, detecting body, foot, hand, and facial key points in real-time) [22] 11

Figure 2.5 : Block diagram of the LSTM. [65].....14

Figure 2.6: Basic architecture of CNNs.[66] 23

Figure 2.7 : Pose Estimation with graph edge convolution [56].....24

Figure 2.8 : Gated recurrent unit memory block [67].....26

Figure 3.1 : Overall Methodology Flowchart 30

Figure 3.2 : Early Stages Action Video Dataset 31

Figure 3.3 : Selecting Player One ROI. 33

Figure 3.4 : ROI (Region of Interest) of Player One. 33

Figure 3.5 : Extracting key point During Smashing. 33

Figure 3.6 :Key point Dataset after Preprocessing and labelling.....34

Figure 3.7 : LSTM Model Design.....35

Figure 3.8 : LSTM Model Summary 36

Figure 3.9 : Overall Flow of Data Preprocessing and Training.....39

Figure 3.10 : Overall Simulation Framework 40

Figure 4.1 : Training and Validation Loss for Proposed Model 46

Figure 4.2 : Training and Validation Accuracy for Proposed Model 46

Figure 4.3 : Confusion Matrix for the proposed model. 48

Figure 4.4 : Training and Validation Accuracy/Loss for GRU Model 51

Figure 4.5 : Training and Validation Accuracy/Loss for CNN Model 53

Figure 4.6 : Real-Time Simulation User Interface 57

Figure 4.7 : Real-Time Simulation 58

LIST OF TABLES

Table 4.1 : Analyzing the accuracy of proposed LSTM model by tuning the hyperparameter 43

Table 4.2 : Performance Analysis for Proposed LSTM Model. 49

Table 4.3 : Performance Analysis for GRU Model 50

Table 4.4 : Performance Analysis for CNN Model 52

Table 4.5 : Comparison Between Proposed Model with GRU and CNNs 55

Table 4.6 : Prediction of LSTM Model VS Human Manual Observation 58

LIST OF SYMBOLS AND ABBREVIATIONS

YOLO	:	You Only Look Once
LSTM	:	Long Short-Term Memory
GRU	:	Gated Recurrent Unit
CNN	:	Convolutional Neural Network
ReLU	:	Rectified Linear Unit
CONV	:	Convolution
Py	:	Python
ROI	:	Region of Interest
AI	:	Artificial Intelligence
UI	:	User Interface
SDG	:	Sustainable Development Goals

LIST OF APPENDICES

Appendix A: The Code for Dataset Preprocessing and Key Point Extraction 77

Appendix B: Code Highlight for Real-Time Simulation 80

CHAPTER 1:

INTRODUCTION

This chapter presented the general view of the project following with the background and problem statement. Objectives and scopes of the project are also discussed in this chapter. All the details for each section of the project have been commented on in this chapter.

1.1 Background of Project

The integration of deep learning in sports has become increasingly essential due to its potential to revolutionize performance analysis, player development, and audience engagement. Deep learning algorithms have demonstrated remarkable capabilities in processing large volumes of sports-related data, recognizing patterns, and making predictions, making them well-suited for addressing the complexities of sports analytics [1]. Manual analysis of badminton matches is plagued by several critical

issues, making it an inefficient and less reliable method for assessing player performance. The process is both time-consuming and labor-intensive, demanding observers to meticulously review match videos frame-by-frame and record events, leading to a slow and tedious workflow [2][3]. Moreover, the subjectivity and inconsistency among different observers in interpreting and recording match events result in unreliable and inconsistent data [2]. In other context of badminton analysis is the usage of sensor-based approach. The usage of sensors in badminton faces several practical challenges, making their implementation unfeasible in certain contexts. These challenges are rooted in technological, logistical, and practical considerations, as evidenced by the following factors. Wearable sensors, such as accelerometers, often need to be attached to players' bodies to capture motion data accurately. This can lead to discomfort and may interfere with players' movements and performance [4]. Furthermore, purchasing and maintaining sensor equipment, as well as the necessary data processing and analytic infrastructure, can be prohibitively expensive in many badminton training and competitive situations [5]. A more objective and effective solution is provided by the integration of automated systems, which are powered by cutting-edge technologies like computer vision and deep learning networks. This reduces the error margin and gives a thorough study of the dynamics of the game.

The suggested project aims to create an intelligent system based on computer vision with deep learning networks. The goal of the research is to create a system that can accurately and independently identify a variety of badminton moves from video footage, such as smashes, lifts, and serves. The approach used makes use of advanced deep learning models that combine LSTM (Long Short-Term Memory) models for pose estimation algorithms for precise badminton action classification. This tactical

strategy guarantees the accuracy and dependability of the system in identifying a broad range of activities within the badminton environment.

At the end of this project, the anticipated outcome is the creation of a functional system with high recognition accuracy for multiple badminton actions for sports performance analysis, enhancing coaching and training experiences.

1.2 Problem Statement

Rapid evolution of computer vision technology and research and development (R&D) activities has led to the creation of modern and advanced sports analysis techniques based on deep learning. Generally, the use of multiple sensor-based for badminton action recognition may increase the recognition accuracy, but if it is too many sensors can cause inconvenience and discomfort for badminton athletes during training and live matches [6]. Another issue of the use of the sensor based deep learning model and sensor positioning is also prone to the confusions for detecting similar strokes such as smash and clear [7]. Hence, the necessity to simultaneously consider the accuracy of the framework and other factors such as shuttlecock recognition, precise classification of the stroke's recognition and stability of the framework were also crucial. The precise classification of strokes recognition and accurate shuttlecock recognition can be guaranteed by using deep learning. This is because deep learning works excellently in extracting high-level features directly from raw data as its architecture consists of hundreds of hidden layers [8]. Addressing that, the main aim of this project is to develop an intelligent system of badminton action based on computer vision video analysis with deep learning that integrates pose estimation and object detection model.

1.3 Objectives

There are two objectives in this project listed below:

i. To develop a deep learning-based badminton action recognition system that integrates pose estimation and LSTM (Long Short-Term Memory).

ii. To evaluate the proposed model's performance in terms of precision, recall, F1-score and model accuracy.

1.4 Scope Of Project

The goal of this research is to create an advanced badminton action recognition system. The first stage is gathering a variety of high-definition badminton match recordings from YouTube to provide the groundwork for later stages. The main objective is to locate critical points, i.e., smash and lift, when temporal fragments are carefully cut and saved independently. Each video frame is then processed using the YOLOv7 Pose Estimation Model to extract the expected key points (X-coordinate, Y-coordinate, and confidence scores). After grouping these focal points into sequences of thirty frames, 51 features are produced, each of which carefully captures the details of the 17 key points. The next stage contributes to a thorough knowledge of different badminton motions by training an LSTM Model with key point data derived from the YOLOv7 Pose Estimation Model. The LSTM Model's input data dimensions are (n * 30 * 51), which captures the key point data's sequential structure.

The project integrates a practical feature by making a mask with the same shape and presenting an example image from the videos in addition to its fundamental development. The system then uses the YOLOv7 Pose Model to identify key points

during specific badminton motions, such as smash, drop, lift, and serve, and select the Region of Interest (ROI) for Player 1. Note that Player 1, who is closer to the camera, is the subject of this discussion since Player 2, who is positioned farther from the court camera, is subject to limitations relating to missing key point landmarks. In addition, the trained LSTM model is tested on sample videos and produces shot predictions in the output.

The development of badminton action recognition utilizing Google Colab as a simulation platform. Google Colab is the Jupyter notebook environment that allows users to collaborate and access the powerful hardware accelerators such as GPU (Graphics Processing Unit). Google Colab offers GPUs from NVIDIA, such as Tesla T4, K80 and P100, which are used for extensive wide and variety machine learning and data analysis tasks. It offers a Jupyter environment that is interactive and supports several programming languages such as Python that will be used in this project. Python is rich in ecosystem of libraries and frameworks, flexible, easy to use and has strong community and support. The libraries used in this project include TensorFlow, Matplotlib, OpenCV and NumPy as open-sources libraries.

To give coaches and players useful information about gaming dynamics, this systematic project aims to improve the accuracy of badminton action recognition by integrating deep learning. The project provides more than just data processing; it involves training and implementing complex models to guarantee a strong and intelligent system that can precisely predict and classify different badminton strokes.

1.5 Project Significance

The significance of this project lies in its potential to transform badminton gameplay analysis and comprehension by utilizing deep learning technology. In addition to improving sports analytics efficiency, the project offers coaches and athletes insightful information by creating a sophisticated system for accurate action identification. A more sophisticated understanding of player motions and shot dynamics is made possible by the methodical approach that combines YOLOv7 Pose Estimation and LSTM models. This initiative has the potential to completely change the way badminton is researched by providing an invaluable resource for enhancing player performance, fine-tuning coaching techniques, and expanding the field of sports analytics.

1.6 Thesis Outline

This thesis consists of five chapters, starting with an introduction outlining the objectives, scope, and significance of developing an intelligent badminton action recognition system. Chapter two goes into further detail about the background research, highlighting the drawbacks of the current methods and providing support for using a computer vision-based system. The third chapter describes the approach in depth, emphasizing the gathering of various badminton match recordings, the creation of the system utilizing YOLOv7 Pose Estimation and LSTM models, and pragmatic issues such as Region of Interest selection and image display. Results from the trained system are examined and presented in the fourth chapter, offering an understanding of how well it performed. The thesis is ended in the last chapter, which also offers directions for future research and highlights important discoveries. This will help to improve badminton coaching and training experiences while also advancing the field of sports analytics.

CHAPTER 2:

BACKGROUND STUDY

This chapter discusses the related previous research on sports and badminton analysis encompassing various types of approach. It also explains in detail the concept of pose estimation and LSTM (Long Short-Term Memory). Ideas and techniques were obtained from those journals and research papers. Based on the methods that been used in that previous research, the best solution was selected and applied to this project as the methodology.

2.1 Badminton – Second Most Popular Sport in The World

Badminton is a popular racquet sport that has a past that goes back more than 2,000 years. The current form of the game was created around the 1850s [9].The sport can be traced back to old societies in Asia and Europe, where it was first played as a fun competitive activity. Badminton has become an official sport with set rules and sets of tools over time. The name of the sport comes from the Duke of Beaufort's Badminton House in Gloucestershire, England, where it was first played in the middle of the 1800s. In 1893, the Badminton Association of England was formed, and the

first set of rules was written down. This made the sport more official and famous. Badminton has changed over time from a fun activity to a sports game with a rich culture history that is played all over the world.

Badminton is played on a rectangular court with a net in the middle. It can be played singles (one player on each side) or pairs (two players on each side). One player's goal is to hit the shuttlecock over the net and put it inside the other player's court while making it hard for the other player to return it. Tennis is a sport where players must quickly move around the court and use different strokes, like the serve, clear, smash, and drop shot, to get the upper hand on their opponents. Badminton is scored by rally points, which are earned for every serve. The first team to reach 21 points wins the game and has a two-point lead over the other team. Quick reflexes and smart play are important in this sport, which makes it exciting and interesting for both players and viewers.

There are several reasons why badminton has become so popular around the world. First, it's appealing because there aren't many rules on the pitch, and it's easy to learn, so a lot of people can do it [10] A lot of people also like how the sport is structured temporally, with short bursts of high-intensity activity followed by short breaks [11]. Also, the physical needs of badminton, like having to be quick on your feet and getting enough oxygen, have helped make it so famous [12]. Facilities made just for badminton, like badminton rooms, have also helped the sport become more famous. There is also a lot of interest in badminton because it is taught in physical education classes at schools and universities [13].

2.2 Action Recognition

The study of action recognition is an important part of computer vision. It has many uses, including smart video monitoring, virtual reality, human-computer interaction, and more [14]. It involves finding different actions in video clips, and researchers are very interested in it because it has a lot of study worth and could be used in many ways [15]. Recent research has shown that neural networks trained to recognize objects can also learn visual traits that help classify actions. This shows how different computer vision tasks are linked [16] .A lot of recent study has also been focused on improving action recognition methods. This includes progress in hand-designed action features, deep learning-based action feature representation methods, and methods for recognizing human-object interactions [17].

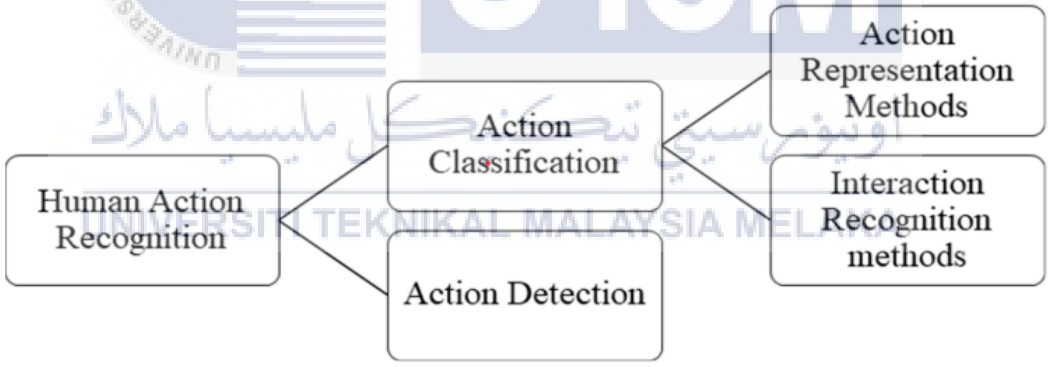

Figure 2.1 : Classification framework for human action recognition methods [17]

A lot of different areas, like sports, dance, and aerobics, use action recognition. It's very important for competition training, national health, and tracking people's movements [18] Action recognition in films could also be used for multimedia indexing, public safety surveillance, and teaching music in a way that involves the

students [19]. Automatically recognizing what people do can make monitoring easier and have effects on public safety and security [20].

To summarize, action recognition is an essential area of study in computer vision research that has a wide range of practical uses. In recent years, there have been notable developments in this subject, especially in the creation of deep learning-based techniques and their use in other domains.

2.3 Pose Estimation

Pose estimation is an important part of computer vision and artificial intelligence. It can be used for many things, from figuring out where a person is to analyzing things that happen in videos, like crashes. In the past few years, a lot of progress has been made in this area. New methods and formulas have been created to solve problems by correctly figuring out the poses of people and things. Deep learning, especially as shown by DeepPose and OpenPose [21], [22], has become a key factor in improving pose prediction. Deep neural networks are used in these methods to figure out the poses of different people. Part affinity fields are used to estimate the 2D poses of multiple people in real time.

Figure 2.3: Multi-person pose estimation. Body parts belonging to the same person are linked, including foot key-points (big toes, small toes, and heels) [22]

Figure 2.2: Configuration of valid human pose estimation [17]

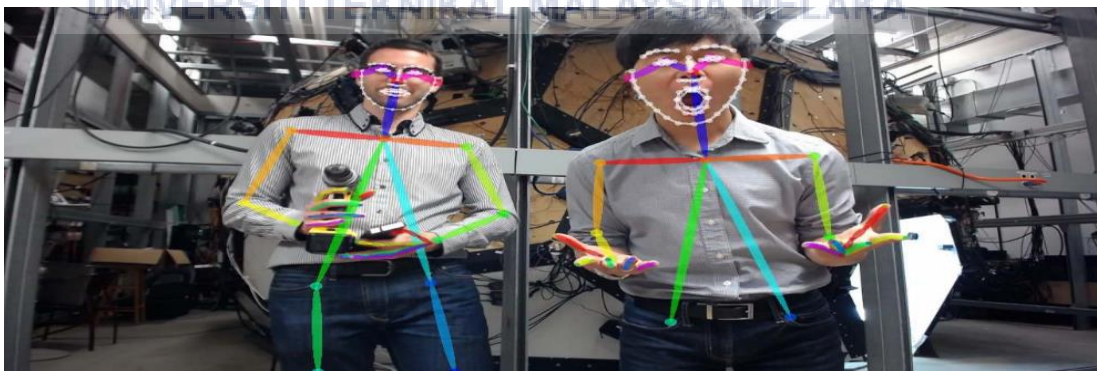

Figure 2.4 : Output of OpenPose, detecting body, foot, hand, and facial key points in real-time) [22]

The study [23] by Munea et al. gives an interesting summary of the progress made in estimating the pose of a person, including a list of the models used in 2D pose estimate. This study fills in gaps in our knowledge and sheds light on the current state of 2D human pose estimation studies. It also helps us understand what other research is being done in this area. Sapp et al. [24] also contribute to the field by solving the problem of articulated human pose estimation by creating a chain of visual structure models that go from coarse to fine. This makes pose estimation models more accurate.

Zhang and Jiang's work [25] shows how important accurate pose prediction is for recognizing visual things and how important it is in many computer vision applications. Ghiass et al. [26] show how RGB-D sensors and 3D morphable models can be used to accurately and fully automatically estimate 3D head pose and eye gaze. They show how depth data can be used to estimate poses.

The work of Eichner and Ferrari [27] shows how useful human pose co-estimation is for whole-body imaging, pattern recognition, and picture analysis. Chen et al.'s plan [28] calls for the creation of multimodal data fusion pose estimate algorithms that are especially good at dealing with problems that come up in complex scenes with targets that aren't well-textured and lighting that isn't ideal. This suggestion adds to the current development of motion estimation techniques.

Pose estimate research uses many different approaches, some based on deep learning and others on multimodal data fusion algorithms. These together move this important area of computer vision and artificial intelligence forward.

2.4 LSTM (Long Short-Term Memory)

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture designed to address the vanishing and exploding gradient problems encountered in conventional RNNs [29]. LSTM has been widely used for sequential data analysis, demonstrating effectiveness in various applications such as speech recognition [29], malware detection [30], rainfall-runoff modeling [31] and energy-efficient inference acceleration [32]. Its capability to handle long-term dependencies has made it suitable for tasks like time series prediction [33], hand gesture recognition [34], and anomaly detection [35].

The Long Short-Term Memory (LSTM) network addresses several critical challenges in sequence learning and time series analysis. LSTM is specifically designed to solve the problems of vanishing and exploding gradients that are commonly encountered in traditional recurrent neural networks (RNNs) [36]. These issues hinder the ability of RNNs to effectively capture long-term dependencies and sequential patterns in data, which is crucial for tasks such as time series prediction, video analysis, and speech recognition [37]. LSTM's architecture enables it to effectively learn from and model sequences with long-range dependencies, making it well-suited for applications that require understanding and predicting complex temporal patterns [36].

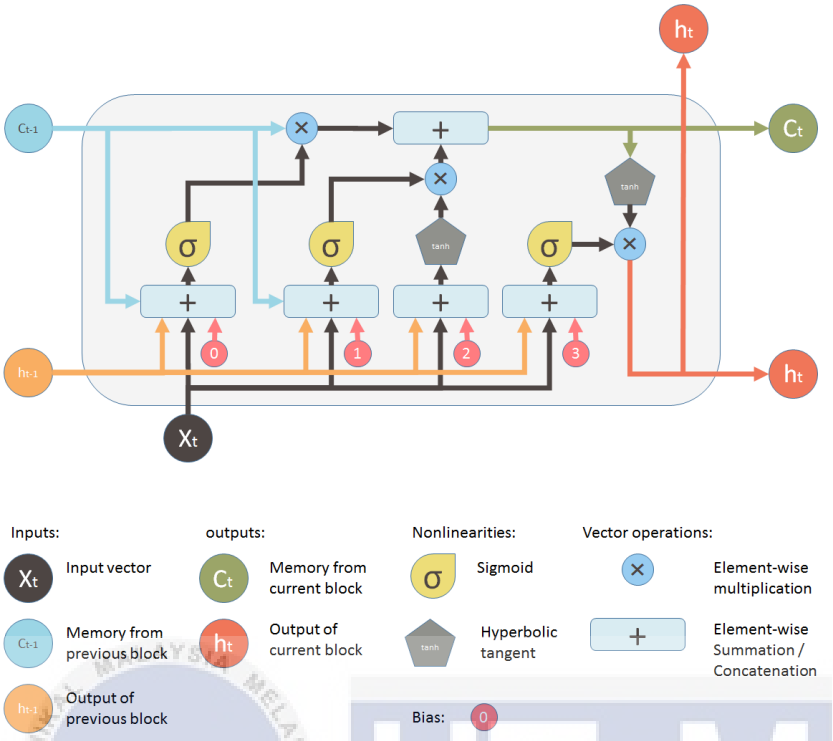

Figure 2.5 : Block diagram of the LSTM. [65]

Block diagram of the LSTM as shown in Figure 2.5 illustrates the internal structure and operation of an LSTM cell, detailing the various components and the flow of data within the cell. It contains inputs, outputs, components, and operations.

Forget Gate (f_t), Input Gate(i_t), Cell State Update (C_t), Output Gate(o_t) and Hidden State Update(h_t) is computes as:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2.1)$$

$$X_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2.2)$$

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2.3)$$

$$ç_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (2.4)$$

$$C_t = f_t * C_{t-1} + X_t * \varsigma_t \quad (2.5)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (2.3)$$

$$h_t = o_t * \tanh(C_t) \quad (2.5)$$

Where;

X_t : Input vector at the current time step.

h_{t-1} : Hidden state from the previous time step.

C_{t-1} : Cell state from the previous time step.

h_t : Hidden state output from the current LSTM cell.

C_t : Cell state output from the current LSTM cell.

Furthermore, LSTM has been applied to various domains to address specific challenges. For instance, in the context of wind power forecasting, LSTM has been utilized to capture the complex non-linear spatiotemporal dynamics of wind power generation, improving the accuracy of forecasting models [38]. Similarly, in the field of traffic prediction, LSTM, in combination with attention mechanisms, has been employed to model the intricate spatiotemporal traffic dynamics in road networks, addressing the challenges associated with predicting travel times accurately [39]. Additionally, LSTM has been used to predict water levels and water quality, leveraging its ability to capture long-term dependencies and sequential patterns in

water-related data, thereby improving the accuracy of water quality and level predictions [40].

In summary, LSTM is designed to address the challenges of capturing long-term dependencies and sequential patterns in data, making it a powerful tool for tasks such as time series prediction, traffic forecasting, wind power prediction, and water quality modeling.

2.5 Badminton Action Recognition

2.5.1 Sensor Based Approach

Past study has paid a lot of attention to action recognition in badminton using sensor-based technology. Accelerometer and gyroscope readings have been used in studies to try to figure out badminton strokes and moves [41]. It has also been suggested that gyroscope sensors and machine learning methods could be used to recognize the different badminton strokes that players use. It has also been investigated how body sensor networks can be used to recognize different badminton moves [42] Also, the use of sensor rackets as a diagnostic and training tool for top badminton players has been investigated, showing the sport's promise for sensor-based technologies [43].

All these studies show that people are becoming more interested in using sensor-based technologies to track and analyze different badminton moves. The use of machine learning methods and networks of accelerometers, gyroscopes, and body sensors shows that badminton strokes and moves can be recognized accurately and quickly. Additionally, the test of sensor stick as a diagnostic tool shows how sensor-based technologies can be used in top badminton training and performance analysis.

When sensor-based technologies are combined with advanced recognition algorithms, it could help us better understand and analyze badminton moves, which could lead to better ways to train and judge player performance.

2.5.2 Pose Estimation Approach

The incorporation of deep learning into pose estimation has been crucial in ensuring precise evaluation of badminton action recognition. The significance of pose estimation algorithms, including OpenPose and HRNet, in badminton action recognition has been emphasized by Su and Feng [44]. By employing these algorithms, the posture of badminton players can be approximated, facilitating the extraction of significant cues and characteristics for action recognition. Furthermore, the integration of deep learning models, including convolutional neural networks (CNN), with pose estimation has proven to be a pivotal factor in attaining precise identification of diverse badminton movements.

The empirical evidence supporting the efficacy of pose estimation algorithms, specifically in the estimation of body poses characterized by many degrees of freedom, has emphasized their criticality in attaining cutting-edge outcomes in action recognition endeavors. Furthermore, it has been demonstrated that pose-based features obtained from estimated poses exhibit enhanced performance in action recognition tasks compared to low or intermediate-level features [45].

In brief, the incorporation of deep learning into pose estimation and the implementation of pose estimation algorithms like OpenPose and HRNet have exhibited exceptional efficacy and hold promise for propelling the comprehension and analysis of badminton movements forward, thereby facilitating enhanced training methodologies and player performance assessment.

2.5.2.1 Pose Estimation Model

OpenPose

OpenPose is an extensively utilized and well-known pose estimation framework that combines deep neural networks. Multiple CNNs and part affinity fields are employed by the architecture to identify and link critical body joints. OpenPose is renowned for its adaptability in managing complex and congested environments and its capability to estimate both 2D and 3D poses.

Pose Net

PoseNet is a Google-developed model for estimating the pose of individual images. Operating on mobile devices, it is optimized for real-time performance and is built upon the MobileNet architecture. The pose estimation capability of PoseNet, which is based on critical body joint positions, is particularly advantageous for tasks that have restricted computational resources.

You Only Look Once Version 7 (YOLOv7) Pose

YOLOv7 is an advancement over the YOLO series, and YOLOv7 Pose is an application designed to estimate the pose of humans. Utilizing a single-shot detection methodology, it is capable of real-time prediction of critical anatomical points and bodily joints. YOLOv7 Pose utilizes the speed and efficacy of the YOLO architecture to perform pose estimation duties.

By utilizing deep learning methodologies, these models autonomously acquire and extract hierarchical features from input images, thereby facilitating precise and effective pose estimation in a wide range of applications. The model selection is

contingent upon various factors, including real-time performance, accuracy prerequisites, and the attributes of the input data.

2.5.3 Trajectory Analysis Method

In badminton, trajectory analysis entails an exhaustive examination of the shuttlecock and the paths traversed by players throughout a match. Through the monitoring of player movements, analysts can discern strategic preferences and playing techniques, including acceleration and speed. Furthermore, an examination of the shuttlecock's trajectory yields valuable information regarding the characteristics of shots, including their velocities, angles, and varieties.

Utilizing shuttlecock trajectory for badminton action recognition necessitates several technological and methodological approaches. The shuttlecock's trajectory is an essential factor in distinguishing various badminton actions. An investigation conducted found that the presentation of the shuttlecock's trajectory in the footage improved the ability of novices to anticipate actions[46]. The importance of shuttlecock trajectory in action recognition is thus highlighted. In addition, the efficacy of a shuttlecock trajectory tracking system was proposed, highlighting the pragmatic implementation of trajectory analysis in the context of badminton action recognition. Furthermore, for action recognition, it is critical to comprehend the trajectory of the badminton shuttlecock, constructed a model to depict this trajectory [47].

Furthermore, significant technological progress has been achieved in this domain. For instance, developed badminton action recognition algorithms utilizing acceleration and angular velocity signals, thereby demonstrating the application of cutting-edge technologies to action recognition [48]. Furthermore, an investigation

examined the application of sensor technology in trajectory prediction by utilizing multiple two-dimensional scanners to detect and predict the trajectory of badminton shuttlecocks [49] .

2.6 Review of Related Work

Researchers have been pushing the limits of badminton action recognition in the quickly developing field of sports technology. The research by (Anik M et al.,2016) [50] aims to create a system that uses motion sensors—specifically, accelerometers and gyroscopes—to identify different activities within a badminton game. The study's principal discovery is that the system, which uses the Support Vector Machines (SVM) and K-Nearest Neighbors (K-NN) classifiers, shows considerable promise in identifying actions such as smash, serve, and backhand. (Steels T et al.,2020) [41] revolutionized badminton action recognition by investigating the integration of Convolutional Neural Networks (CNNs), in contrast to (Anik M et al.,2016) [50] outdated sensor-based approach. Their innovative method made use of accelerometers and gyroscopes to provide a covert and affordable way to measure player performance. With a remarkable 99% accuracy rate in classifying strokes, this study represented a revolutionary turning point in sports analysis. (Yip et al.,2020's) [51] contribution, which delves deeper into the field of sensor-driven badminton action recognition focusing on various type of smashes and integrating Convolutional Neural Networks (CNNs), was prompted by this seismic shift. They used accelerometer and gyroscope data instead of the more basic badminton action, offering a low-cost and non-intrusive way to improve badminton player performance. Developing this sensor-based path further, (Qin L et al.,2022) [6] conducted a thorough investigation titled "Optimizing Badminton Action Recognition with Deep Learning and Sensor Fusion." They effectively integrated inertial, electrocardiogram,

and electroencephalogram sensors with deep learning to leverage sensor technology in their pursuit of improving athlete performance through motion analysis. With a focus on the vital role played by the athlete's dominant hand, this study sought to maximize recognition accuracy.

Previous studies have only focused on sensor-based action recognition. Addressing this, (Bo et al., 2023) [52] in their paper “Intelligent System of Badminton Serve Action Based on YOLOv5 and OpenPose”, by utilizing computer vision and artificial intelligence, the researchers aim to optimize badminton coaching and training. Creating an effective system that automatically recognizes and scores badminton movements—with a focus on serves action in particular—is their main goal. The study emphasizes the benefits of deep learning models, highlighting their robustness, accuracy, and speed when compared to conventional methods. In addition, the system provides comprehensive scoring, which improves players' understanding by including both overall actions and specific limb movements. The study highlights artificial intelligence's potential in the sports sector and the seamless switch from vision-based to sensor-driven methods demonstrates a common path that could revolutionize the recognition of badminton actions.

An important research gap in the literature to date is the absence of a comprehensive system that combines LSTM with a wide variety of pose estimations model for a broad range of badminton actions (serve, smash, drop, clear, lift, drive, block, net kill, and net shot) using deep learning models without the need for extra sensors. Even though earlier research has been very helpful, the lack of a comprehensive framework makes it difficult to identify and analyze badminton actions holistically.

Table 2: Review of Related Work

Study	Aim	Method	Accuracy
[50]	The aim of this study is to develop a system for recognizing various activities within a badminton game using motion sensors, specifically accelerometers and gyroscopes	K-Nearest Neighbors (K-NN) and Support Vector Machines (SVM) classifiers.	(KNN):58% recognition accuracy, (SVM):88.89%
[41]	Research goal: Automatically recognize nine distinct badminton strokes	Convolutional Neural Network (CNN)	86% with CNN, improved to 99% with the combination of sensor
[10]	Leverage deep learning, specifically continuous learning for recognizing human hitting badminton action.	LSTM, GCN Network, Sensor Module mounted on Racket,	92%classification accuracy using continuous learning and feature decoupling
[6]	Investigate and optimize recognition of various badminton actions using a sensor-based deep learning approach	optimal number and combination of sensors for the highest recognition accuracy	AlexNet:92% GoogleNet:89%
[52]	Develop an efficient system for detecting and scoring badminton player actions, particularly the serve action.	Combination of YOLOv5 (object detection model) and OpenPose (human pose recognition technology).	Not Stated

2.7 Action Recognition Using Another Types of Neural Networks

2.7.1 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a type of advanced machine learning algorithms specifically designed for the purpose of processing and analysing visual data. CNNs are highly efficient in tasks connected to image recognition, object detection, and other image-related processing due to their ability to acquire spatial hierarchies of information autonomously and flexibly from input images.

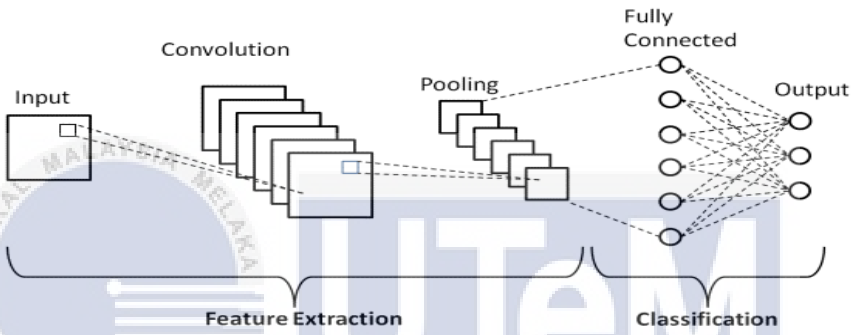

Figure 2.6: Basic architecture of CNNs.[66]

Based on Figure 2.6, A Convolutional Neural Network (CNN) is a type of neural network that processes images by applying filters to detect features such as edges and shapes. The network consists of multiple layers, starting with an input layer that takes the raw image data. Convolutional layers then apply filters to create feature maps, while an activation function (typically ReLU) adds non-linearity to help the network learn complex patterns. Pooling layers are used to reduce the size of the feature maps, making the network more efficient and reducing overfitting. This process of convolution and pooling can be repeated multiple times. The final feature maps are flattened into a one-dimensional vector and passed through fully connected layers for the final classification. The output layer provides the final predictions, often in the

form of class probabilities. This structure enables CNNs to effectively learn and identify features in images.

Action recognition using Convolutional Neural Networks (CNNs) is a significant area of research in computer vision. Researchers have conducted various studies to improve the accuracy and efficiency of action recognition systems. Karpathy et al. [53] demonstrated that retraining the top layers of CNNs on the UCF-101 dataset improved action recognition performance. Tran et al. [54] emphasised the effectiveness of 3D CNNs in achieving strong action recognition results when trained on large-scale datasets. Additionally, Hara et al. [55] found that combining RGB and stacked optical flow frames can enhance action recognition accuracy.

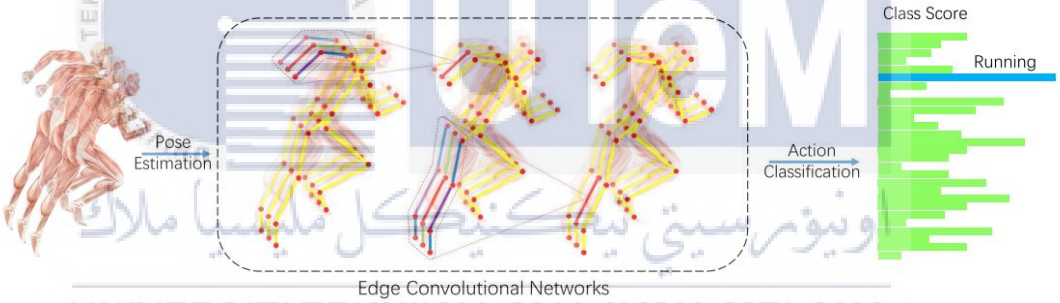

Figure 2.7 : Pose Estimation with graph edge convolution [56]

Furthermore, the integration of several networks has been investigated for the purpose of action recognition. In addition, Zhang et al. [56] presented graph edge convolutional neural networks for the purpose of skeleton-based action recognition, showcasing notable enhancements in performance compared to previous approaches.

Various architectural designs have been suggested to enhance the efficiency of spatio-temporal learning in action recognition. Leong et al. [57] introduced a Semi-CNN architecture that achieved better performance than 3D models while using fewer

parameters, hence eliminating the problem of overfitting. Qi [58] examined the utilisation of CNN models in conjunction with pyramid algorithms for the purpose of recognising aerobics actions. This study demonstrated the possibilities of combining different models in a creative way.

Furthermore, the use of deep learning methods for human action recognition has gained traction. Shi et al. [59] highlighted the success of deep learning-based methods in various applications such as intelligent security, human-computer interaction, and video classification. Zhao et al. [60] discussed end-to-end deep learning approaches for action recognition, categorizing them into different CNN-based and LSTM-based methods based on network structures.

To summarise, the literature review on action recognition using Convolutional Neural Networks (CNNs) demonstrates the progress made in models, fusion techniques, and deep learning methods to improve the precision and effectiveness of action recognition systems. Scientists have investigated different designs, techniques for combining data, and arrangements of networks to enhance the accuracy of Convolutional Neural Networks (CNNs) in identifying human behaviours.

2.7.2 Gate Recurrent Unit (GRU)

GRU, short for Gated Recurrent Unit, is a specific sort of Recurrent Neural Network (RNN) structure that is specifically built to process sequence data. It is related to another RNN architecture called LSTM, which stands for Long Short-Term Memory. GRUs are designed to address the challenges commonly encountered with conventional RNNs, such as the problem of vanishing gradients. They also provide a more streamlined and computationally economical alternative to LSTMs. Several studies have explored the application of GRUs in action recognition tasks to improve accuracy and efficiency.

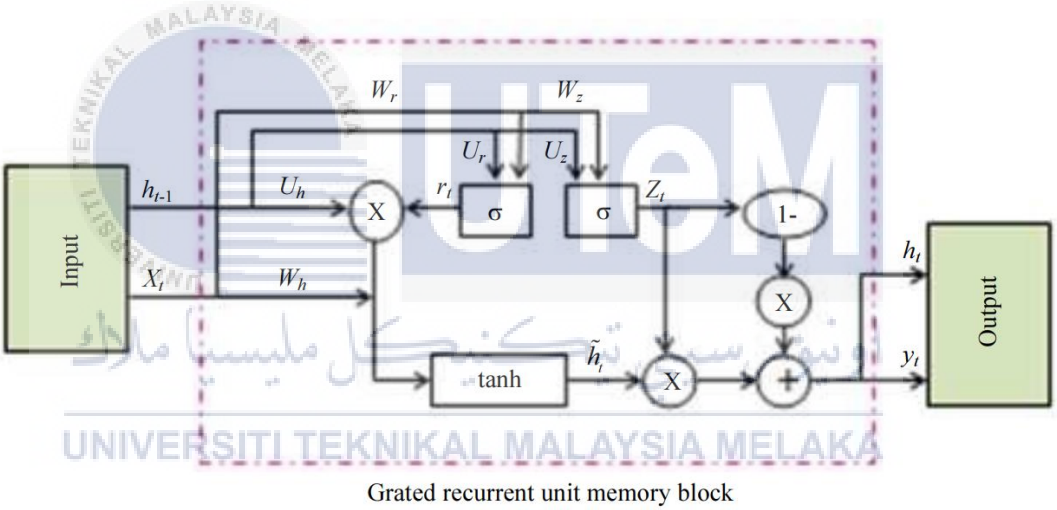

Figure 2.8 : Gated recurrent unit memory block [67]

This Figure 2.8 shows the internal workings of a recurrent neural network (RNN) type called a Gated Recurrent Unit (GRU), which is used to handle sequential data. Reset and update gates are the two main gates used by GRUs. The update gate chooses how much of the historical data should be preserved, while the reset gate chooses how much of the historical data should be erased. This process aids in the upkeep and updating of a hidden state that the GRU uses to extract pertinent data from the sequence. GRUs efficiently learn patterns over time by integrating the input and prior

hidden state through these gates; this makes them ideal for tasks like action identification in films and time series prediction.

In their study, Yin et al. [61] introduced a comprehensive action identification model that incorporates Temporal Convolutional Neural Networks (TCN) to extract local temporal characteristics and a GRU layer to capture global temporal features. This approach significantly improves the accuracy of recognising action fragments. Jaouedi et al. [62] have highlighted the capacity of GRUs to acquire and utilise sequential and temporal input, which is essential for video identification tasks.

Furthermore, Ullah & Munir [63] introduced a framework that employs stacked bi-directional GRUs for long-term temporal modeling and human action recognition, demonstrating the effectiveness of GRUs in capturing complex temporal patterns. Additionally, Lu et al. [64] proposed a Multichannel CNN-GRU model for human activity recognition, highlighting the importance of GRUs in handling variable-length sequences and capturing long-distance dependencies.

To summarise, the literature review on action recognition utilising Gated Recurrent Units (GRUs) highlights the importance of these units in capturing temporal patterns and enhancing the efficiency of action recognition systems. GRUs have demonstrated their adaptability in a wide range of applications, including video identification and voice emotion recognition. This showcases their ability to efficiently process sequential input, underscoring their potential.

CHAPTER 3:

METHODOLOGY

This chapter presents the design technique for the proposed system that aims to recognize badminton actions. The method incorporates YOLOv7 pose to provide accurate pose estimation, leveraging its object identification skills to detect crucial locations on a badminton player's body. In addition, an LSTM model is included for action prediction, utilizing its expertise in processing sequential data, and identifying temporal relationships. The used technologies include PyTorch for YOLOv7, TensorFlow/Keras for the LSTM model, OpenCV for image processing, NumPy for numerical data manipulation, and other libraries for specialized applications. The objective of the system is to improve the accuracy of recognizing badminton actions using deep learning. This will offer coaches and players useful insights into the dynamics of the game. The technique entails analyzing high-definition badminton match records, utilizing YOLOv7 to identify crucial moments, training the LSTM model, and evaluating its accuracy in predicting and categorizing different badminton strokes through example videos.

3.1 Overall Project

The study commences by conducting as shown in Figure 3.1 comprehensive research on deep learning for the recognition of badminton actions, with the aim of gaining a clear understanding of current methodologies, advancements, and challenges in this field. An organised data collection strategy guarantees a thorough dataset encompassing various gameplay scenarios and player proficiency levels. The utilisation of this dataset is crucial for both the training and assessment of the proposed system. Data preparation tackles challenges such as noise, inconsistencies, and fluctuations in lighting, which ultimately allows for accurate posture prediction utilising the YOLOv7 model. The YOLOv7 model precisely detects crucial anatomical landmarks during badminton motions, which are subsequently utilised as input for the Long Short-Term Memory (LSTM) model. The LSTM model, selected for its capacity to handle sequential data, accurately captures the temporal dependencies in posture sequences. The model's ability to predict and classify badminton moves is enhanced by rigorous training, with a particular emphasis on capturing the game's dynamic aspect. Performance evaluation use metrics to gauge the precision, resilience, and applicability of the system. The study culminates with a thorough report that provides a concise summary of the results and insights. It emphasises potential advancements and areas for further investigation.

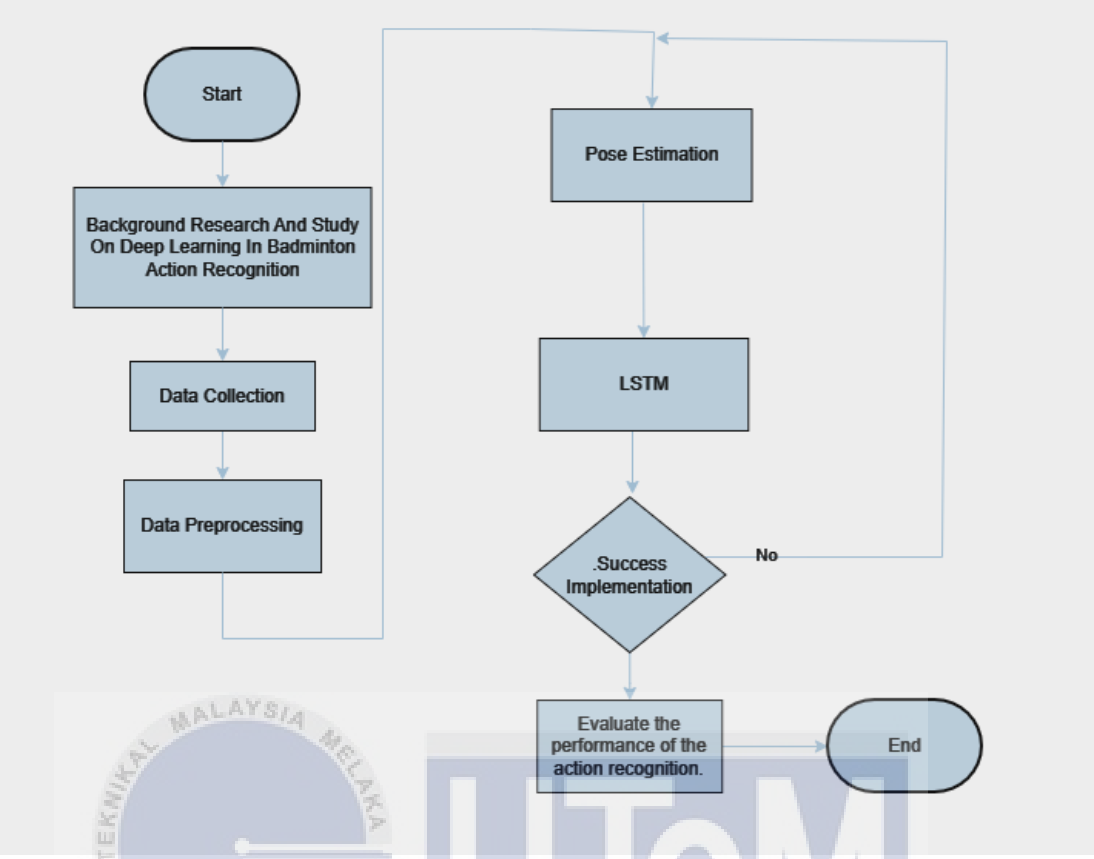

Figure 3.1 : Overall Methodology Flowchart

3.2 Dataset Collection and Preprocessing

The dataset will be obtained from the Badminton World Federation (BWF) YouTube account, which provides high-quality video content. The films from the Victor Denmark Open 2023 competition will be used specifically because of their consistent camera angles, excellent definition, and professional coverage. The study will focus on six key badminton actions shot:

- Smash
- Serve
- Drive

- Net
- Lift
- Block

After obtaining the unprocessed video material, we utilize video editing tools to isolate and remove parts that correlate to badminton motions. Every activity is carefully defined and arranged into distinct folders as in Figure 3.2, guaranteeing convenient access and efficient management during following processing phases. The careful arrangement of data enables effective annotation and labelling, which are essential stages in developing strong action recognition models.

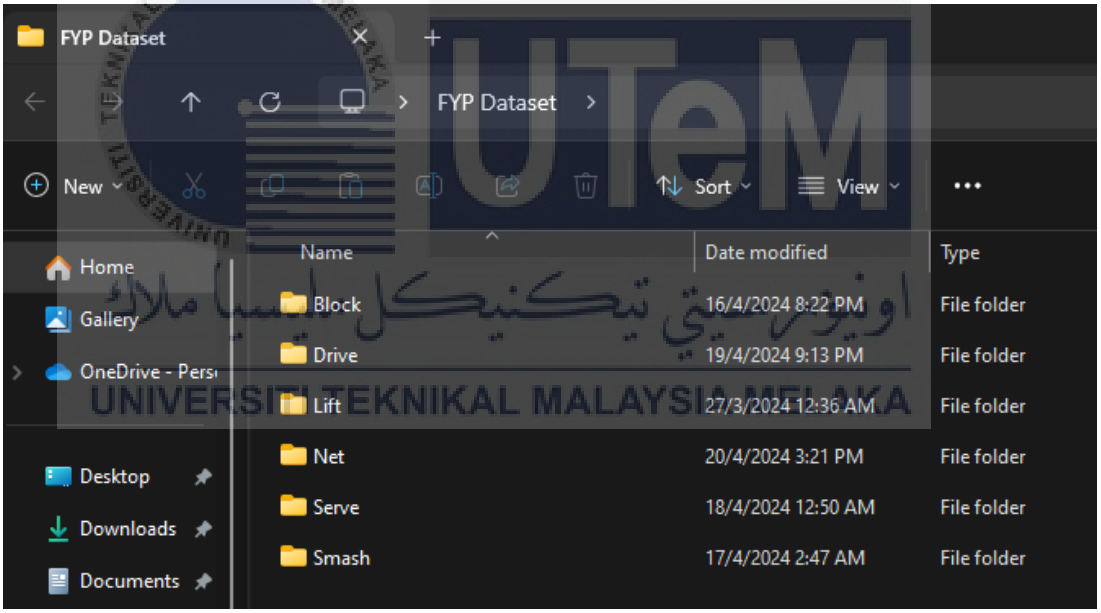

Figure 3.2 : Early Stages Action Video Dataset

Therefore, the final dataset has around 5,000 frames for each action, which is a good number of examples for training and validating the model. In addition, the dataset includes about 30000 frames by adding up frames from all six actions. This

big dataset makes it easier to train a strong model, which lets it recognize a wide range of badminton movements with high accuracy and generalizability.

3.3 Dataset Preprocessing Algorithm Development (KeypointFinder.py).

First, the important libraries needed for model inference and video processing are imported. It then defines a function that makes object classification easier by loading class names from a file. Subsequently, the primary function (run) coordinates the dataset preprocessing pipeline, which consists of multiple crucial stages. Video capture from a specified source—a video file or a live webcam feed—is initialized by the algorithm. It recognizes things in every frame of the video, including badminton players, by using the YOLOv7 model. Important contextual data is obtained during the object detection stage, which is used to estimate the pose afterwards.

3.4 Dataset Preprocessing by Algorithm

The frames are taken out of the video source and used as a starting point for further research. A Region of Interest (ROI) is made to focus on Player 1 more closely, and YOLOv7 pose estimate is used to get key point forecasts. These key points, which are locations shown by (x, y) coordinates, are taken out and stacked as 30 frame sequences, with 51 features in each sequence as shown in Figure 3.3,Figure 3.4,Figure 3.5,and saved into .py file containing all the key points for each action and then concatenated into a single NumPy array file as shown in Figure 3.6. This sequential picture is key to understanding how badminton movements change over time.

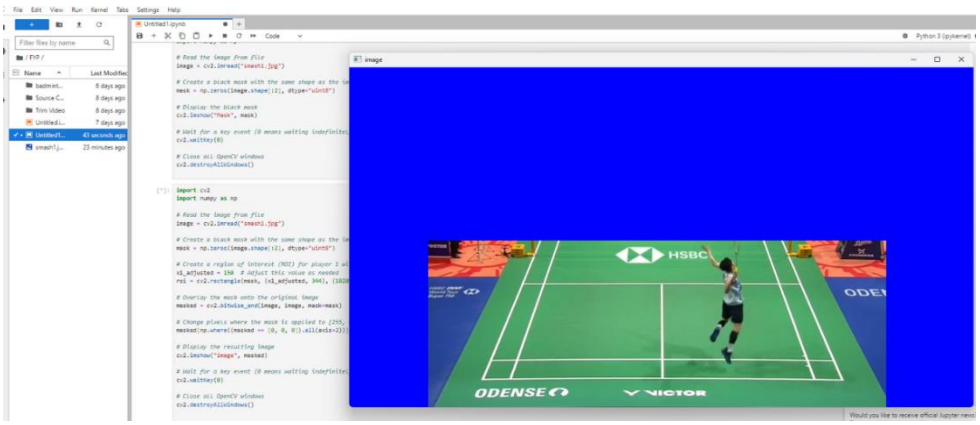

Figure 3.4 : ROI (Region of Interest) of Player One.

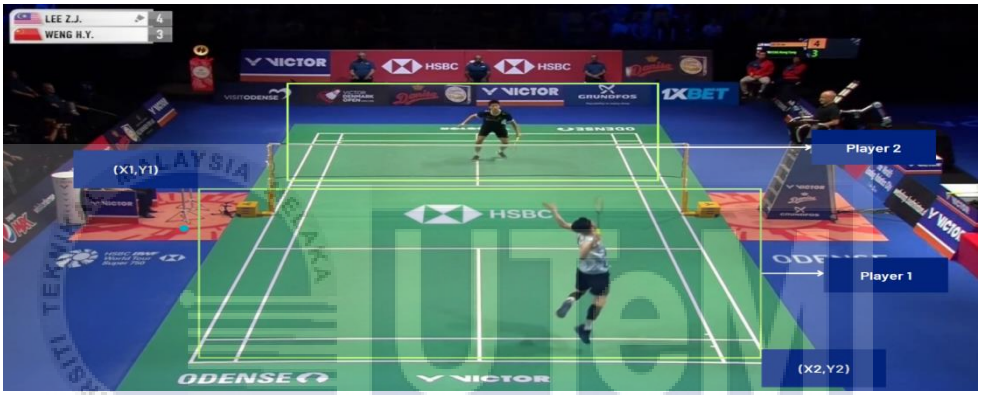

Figure 3.3 : Selecting Player One ROI.

Figure 3.5 : Extracting key point During Smashing.

The next important step is to save the data from the collected keypoints for each action. Basically, the key points of each badminton action are the final dataset that has been processed. This will be used to train the LSTM model. The collected key point data is used to train the LSTM model, which takes advantage of its natural ability to understand how events depend on each other in a certain order. By combining sequential data, the learned LSTM model improves its accuracy and turns into a smarter forecast of badminton moves.

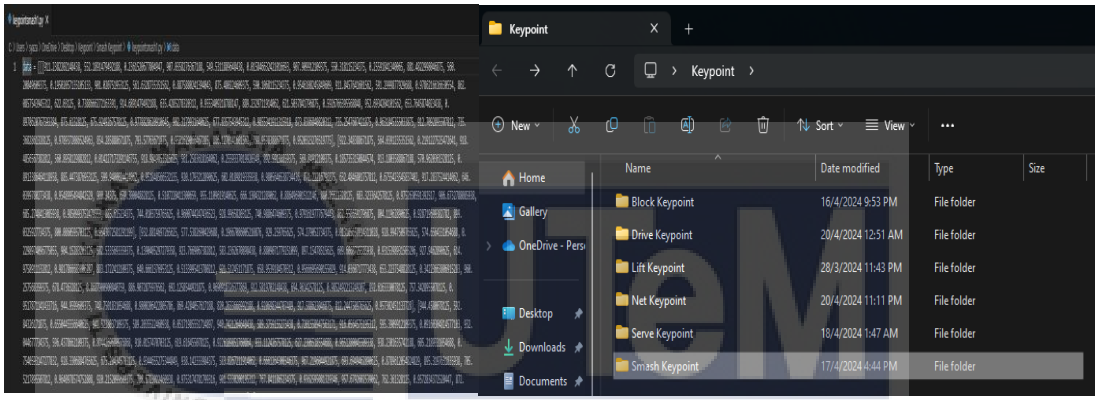

Figure 3.6 :Key point Dataset after Preprocessing and labelling.

3.5 LSTM Model Training

The input data is made up of key point sequences that were taken from badminton videos. Each sequence depicts the players' position dynamics during a 30-frame period. A three-dimensional array of shape (n * 30 * 51) is created by stacking these sequences together. The number 'n' indicates the number of samples, 30 the number of frames per sequence, and 51 the dimensionality of each key point (X-coordinate, Y-coordinate, and confidence score for 17 key points).

The train_test_split function from the scikit-learn library divides the dataset into training and testing sets. This guarantees that performance of the model may be

assessed on unobserved data. To aid in model training, one-hot encoding is also used to encode the target labels, or badminton actions. By converting them into a binary matrix, this procedure makes categorical labels appropriate for use in classification problems. The final trained models will be saved in .h5 forms.

3.5.1 LSTM Model Architecture Design

```
model = Sequential()
model.add(LSTM(64, return_sequences=True, activation='relu', input_shape=(30,51), recurrent_dropout=0.0))
model.add(LayerNormalization(axis=1))
model.add(LSTM(128, return_sequences=True, activation='relu'))
model.add(LSTM(128, return_sequences=True, activation='relu'))
model.add(LayerNormalization(axis=1))
model.add(LSTM(64, return_sequences=False, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(actions.shape[0], activation='softmax'))
```

Figure 3.7 : LSTM Model Design

To efficiently process key point sequences and extract significant features for action recognition, the LSTM model architecture was created such as depicts in Figure 3.7 and Figure 3.8 . For classification, the model consists of several LSTM layers followed by fully linked (Dense) layers. The input layer anticipates 51-dimensional sequences of 30 key points, each of which represents the X and Y coordinates of 17 key points and their confidence ratings.

The input sequences' temporal dependencies and patterns are captured by the LSTM layers. Sequences are processed by the first LSTM layer, which has 64 units, and output sequences are returned. To enhance convergence and stabilize the training process, layer normalization is used. The sequential data is further processed using 128-unit LSTM layers to extract higher-level characteristics.

Fully connected (Dense) layers are added after the LSTM layers to carry out classification using the characteristics that were retrieved. To provide non-linearity

and improve the representational capability of the model, intermediary dense layers with ReLU activation functions are incorporated into the model. Ultimately, the output layer is subjected to a softmax activation function to derive probability distributions for various badminton actions.

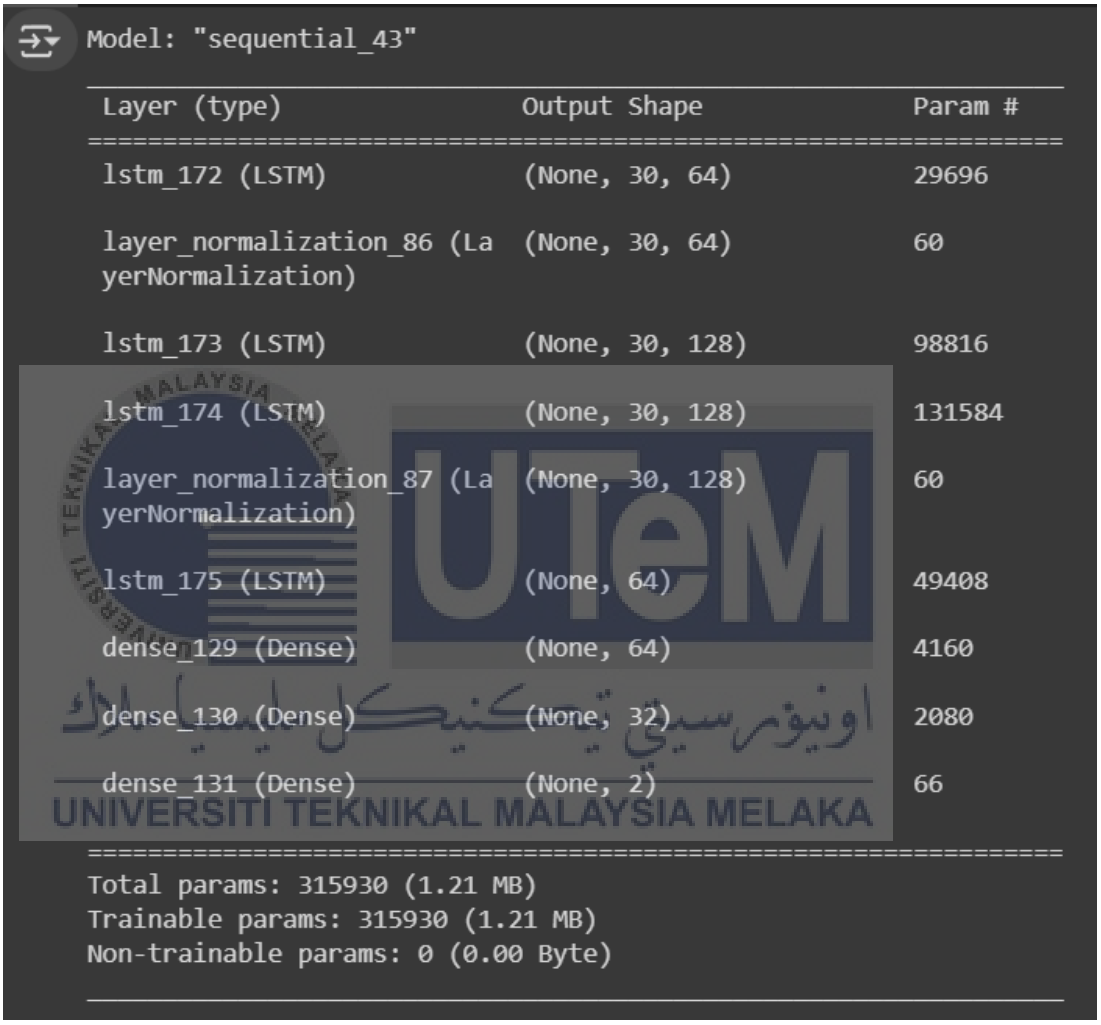

Figure 3.8 : LSTM Model Summary

3.5.2 Model Training and Evaluation

After the model architecture is established, suitable loss functions (‘categorical_crossentropy’) and optimization methods such as Adam are used to compile it. Next, using the fit approach, the model is trained on the training dataset to learn how to map keypoint sequences to matching action labels. Validation data is

used to track the model's performance throughout training, and interfaces like TensorBoard can be used to display training metrics and track convergence.

The model's predictions are broken down into depth in a confusion matrix. It displays the quantity of true positives (actions that were accurately predicted), true negatives (non-actions that were correctly detected), false positives (actions that were mistakenly predicted), and false negatives (activities that were missed). This makes it easier to comprehend how many forecasts came true as well as why others were incorrect.

The model's capacity for generalization is evaluated by analysing its performance on the testing dataset following training. It is possible to calculate metrics derived from confusion matrix like accuracy, precision, recall, and F1-score to assess how well the model can identify various badminton movements. Furthermore, classification reports and confusion matrices shed light on the model's advantages and disadvantages for action recognition. The model underwent training using various combinations of batch size, epoch, and learning rate to determine the optimal configuration for action recognition in badminton :

- Batch Size : 16,32, 64
- Epoch : 50,100,150,200
- Learning Rate: 0.001 (constant)

Each combination was tested to determine how it affects the model's ability to generalize to new, unseen data. The model's ability to generalize is assessed on a different testing dataset following training. For this assessment, several performance metrics computed, such as:

- Accuracy :

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions} \quad (3.1)$$

- Precision :

$$Precision = \frac{TP}{TP+FP} \quad (3.2)$$

- Recall :

$$Recall = \frac{TP}{TP+FN} \quad (3.3)$$

- F1-score :

$$F1\ score = \frac{2\times(Precision\times Recall)}{Precision+Recall} \quad (3.4)$$

Where :

True Positive (TP): The number of samples that were correctly predicted as positive.

True Negative (TN): The number of samples that were correctly predicted as negative.

False Positive (FP): The number of samples that were incorrectly predicted as positive.

False Negative (FN): The number of samples that were incorrectly predicted as negative.

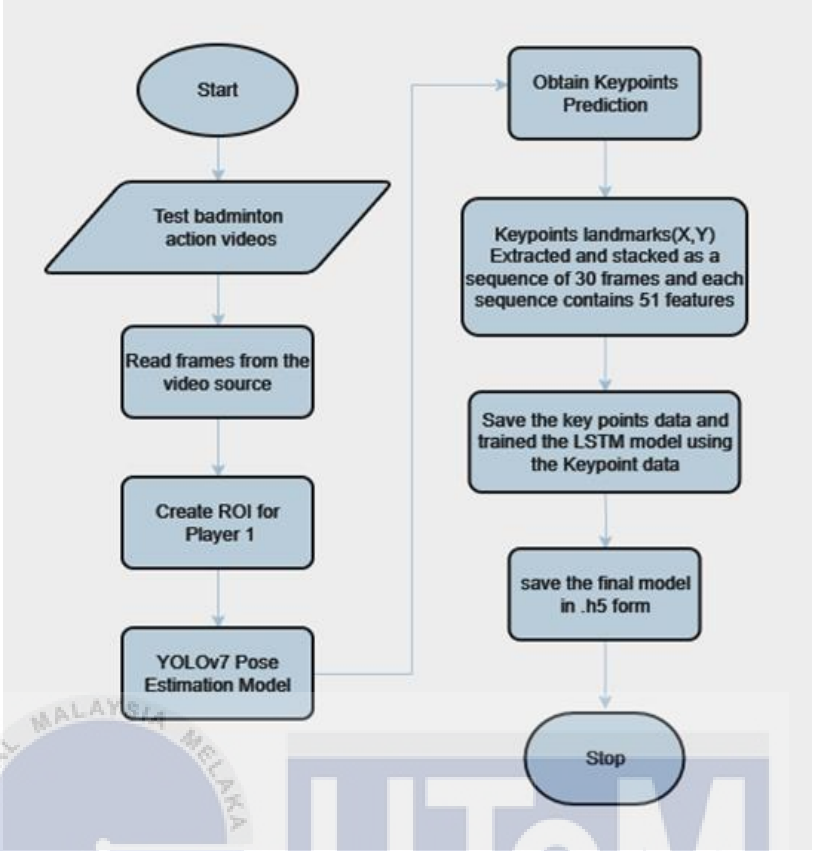

Figure 3.9 : Overall Flow of Data Preprocessing and Training

3.6 Development of Simulation Framework Algorithm for Shot Classification.

The badminton action test video clip serves as the primary means of assessing the performance of the simulation system. To improve the accuracy of the analysis, video frames are processed using a mask to highlight areas of interest. Key point predictions are extracted via YOLOv7 posture estimation, which offers a comprehensive perspective on player movements. These important points are refined for greater precision by non-maximum suppression, and frames are graphically indicated with bounding boxes and key points. The framework employs a pre-trained LSTM model from a.h5 file to anticipate badminton moves by extracting 30-frame key point patterns for both players. Together, these enable the system to recognise intricate patterns and reliably classify photos. With a user-friendly user interface (UI),

coaches, players, and enthusiasts may easily navigate the system and see the anticipated shot type, shot count, and likelihood indicator shown in real time. This integration demonstrates how the simulation system can improve computer vision applications used for sports analysis and training. The full procedure of the simulation test framework is shown in Figure 3.10.

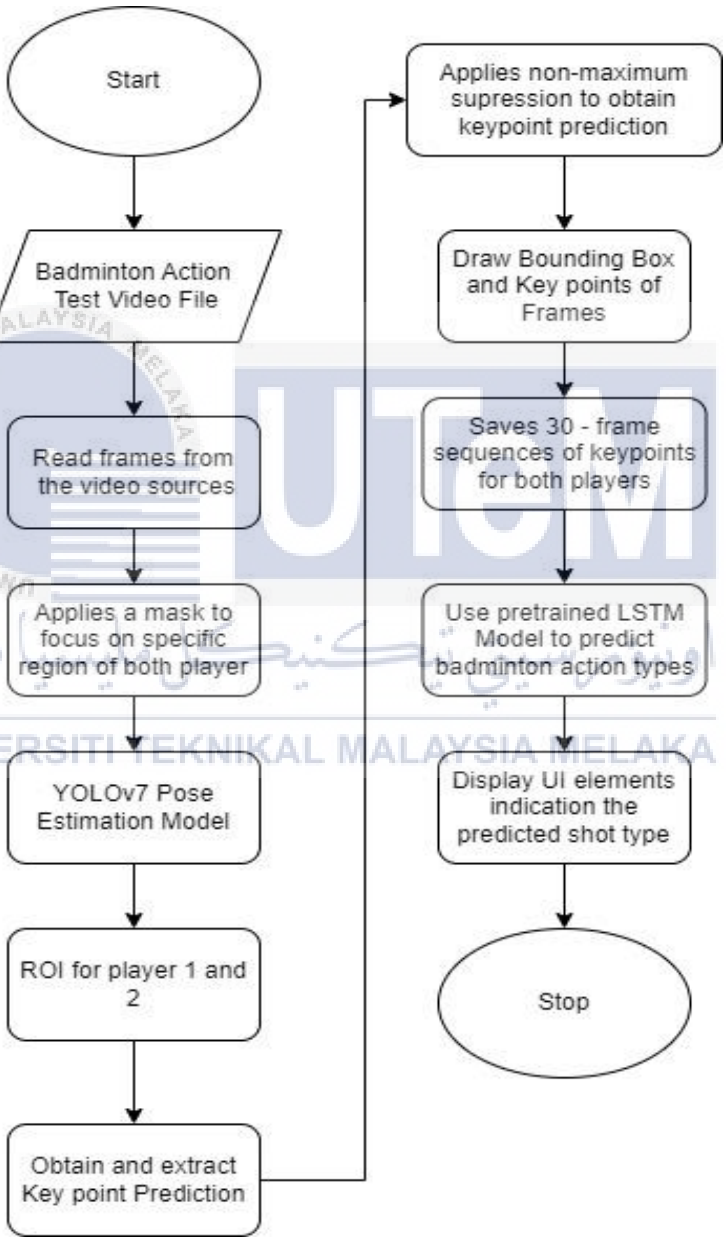

Figure 3.10 : Overall Simulation Framework

CHAPTER 4:

RESULTS AND DISCUSSION

This chapter introduces the Long Short-Term Memory (LSTM) networks and YOLO v7 position estimation-based intelligent badminton action recognition system. The main objectives are to show the outcomes, verify the process, and assess how well it performs in comparison to Convolutional Neural Networks (CNN) and Gated Recurrent Units (GRU). The procedure of developing the system included gathering datasets, preparing the data, extracting critical points, and using the LSTM model. Precise tracking of badminton movements was guaranteed by training and categorization in a simulation environment. The chapter describes how to optimise accuracy using hyperparameter tweaking, which considers learning rate, batch size, and epochs. For the bespoke LSTM model, performance evaluation with confusion matrices and additional metrics is presented. Model accuracy, precision, recall, and F1 score are compared with CNN and GRU models.

4.1 Hyperparameter Tuning

When creating an intelligent badminton action recognition system with LSTM networks and YOLO v7 posture estimation, hyperparameter tuning is essential. This entails deciding on the ideal values for learning rate, batch size, and number of epochs, among other factors that regulate the model's architecture and training procedure. By guaranteeing that the model learns from training data effectively, proper tuning maximises model performance. Training stability is impacted by batch size, and convergence speed is influenced by learning rate. By keeping these variables in check, the model is better able to adapt to new data and avoid problems like under- or overfitting.

4.1.1 Learning Rate, Batch Size and Epochs

To create an intelligent badminton action recognition system, it is essential to adjust the learning rate, batch size, and number of epochs. This is because these hyperparameters have a direct impact on how well and quickly the model learns from the data. An ideal learning rate guarantees that the model converges to a good solution without overshooting or becoming stuck. The learning rate regulates how rapidly the model modifies its weights in response to the loss gradient. A well-set batch size balances the computational effort and the accuracy of gradient estimates. The batch size affects the stability and speed of the learning process. How many times the model views the complete dataset during training is determined by the number of epochs; adjusting this makes sure the model has enough exposure to the data to identify significant patterns without overfitting. When combined, these hyperparameters improve the system's overall performance by enabling the highest precision and dependability in the recognition of badminton actions.

Table 4.1 : Analyzing the accuracy of proposed LSTM model by tuning the hyperparameter

Batch Size	Epoch	Learning Rate	Precision	Recall	F1-Score	Test Loss	Test Accuracy
16	50	0.001	0.80	0.62	0.57	0.49	0.65
16	100	0.001	0.75	0.76	0.75	0.59	0.76
16	150	0.001	0.22	0.5	0.31	0.72	0.46
16	**200**	**0.001**	**0.98**	**0.97**	**0.97**	**0.11**	**0.97**
32	50	0.001	0.90	0.85	0.86	0.42	0.86
32	100	0.001	0.94	0.92	0.93	0.36	0.93
32	150	0.001	0.95	0.95	0.95	0.17	0.95
32	200	0.001	0.97	0.96	0.96	0.12	0.96
64	50	0.001	0.77	0.72	0.72	0.49	0.73
64	100	0.001	0.91	0.88	0.88	0.38	0.90
64	150	0.001	0.85	0.85	0.84	0.31	0.85
64	200	0.001	0.81	0.79	0.79	0.46	0.80

The Table 4.1 shows the outcomes of hyperparameter tweaking with different batch sizes, epochs, and a fixed learning rate of 0.001 for the intelligent badminton action recognition system. Test loss, test accuracy, F1-score, precision, and recall are among the evaluation criteria. The results show significant variation with the number of epochs for a batch size of 16. The model produced an F1-score of 0.57 and a test accuracy of 0.65 at 50 epochs, with a reasonable precision of 0.80 and a somewhat low recall of 0.62. The recall significantly improved to 0.76 as the epochs grew to 100, balancing with the precision to produce a better accuracy of 0.76 and a higher F1-score of 0.75. But when the epochs were increased even more to 150, the precision dropped precipitously to 0.22 and the F1-score to 0.31, indicating overfitting. The model performed remarkably well at 200 epochs, with the lowest test loss of 0.11 and

the highest precision (0.98), recall (0.97), F1-score (0.97), and test accuracy (0.97) among all configurations.

Compared to a batch size of 16, the model performed better overall with a batch size of 32 across various epoch settings. The model demonstrated strong performance at 50 epochs, with recall of 0.85, F1-score of 0.86, precision of 0.90, and test accuracy of 0.86. These metrics showed even greater improvement when the epochs were increased to 100, with precision of 0.94, recall of 0.92, F1-score of 0.93, and test accuracy of 0.93. The model demonstrated strong performance at 150 epochs, with 0.95 precision and recall, 0.95 F1-score, and 0.95 test accuracy. With one of the lowest test losses (0.12) and a high-test accuracy of 0.96, the precision increased slightly to 0.97 at 200 epochs, along with a recall of 0.96 and an F1-score of 0.96.

The model's performance demonstrated greater stability when the batch size was 64. The test accuracy was 0.73 at 50 epochs, with precision of 0.77, recall of 0.72, and F1-score of 0.72. The precision, recall, and F1-score all greatly increased to 0.91, 0.88, and 0.90 with an increase in epochs to 100. The measures showed a modest decline at 150 epochs: 0.85 for precision, 0.85 for recall, 0.84 for F1-score, and 0.85 for test accuracy. With a precision of 0.81, recall of 0.79, F1-score of 0.79, and test accuracy of 0.80 at 200 epochs, the model's performance showed a modest fall, indicating the possibility of overfitting or diminishing returns from further epochs.

4.2 Analyzing The Performance of the Proposed LSTM Model

To guarantee the effectiveness and dependability of the intelligent badminton action recognition system, it is imperative to do an analysis of the suggested model. Additionally, by evaluating the model's strengths and weaknesses using a variety of metrics, including test loss, test accuracy, precision, recall, and F1-score, performance

optimisation is accomplished. This makes it possible to make specific improvements, like modifying the model architecture or fine-tuning hyperparameters.

4.2.1 Model Accuracy and Loss

To evaluate the model's effectiveness and dependability in precisely identifying badminton motions, it is essential to examine its accuracy and loss. Model correctness provides information on the percentage of accurate predictions, which helps to assess the system's overall efficacy. Furthermore, by keeping an eye on the loss function, we can gain insight into the training dynamics and generalisation capacity of the model; lower loss values correspond to better alignment between anticipated and actual values. Examining accuracy and loss closely allows us to see possible problems like under- or overfitting, which helps direct the system's refinement process and guarantee dependable outcomes in practical situations. Based on Figure 4.2 , The graph illustrates the progression of model accuracy for 200 epochs, with the x-axis representing the epochs ranging from 0 to 200, and the y-axis representing the accuracy ranging from 0.65 to 1.00. The training accuracy is represented by the blue line, while the validation accuracy is represented by the orange line. Both are clearly labelled in the plot's legend. At the beginning, the accuracies of both models increase rapidly from 0.65 to 0.85 within the first 50 epochs. After that, they continue to improve gradually and reach approximately 0.95 by epoch 100. During epochs 100 to 200, the accuracies vary about 0.95, occasionally approaching 1.00.The training and validation accuracies exhibit a consistent proximity, suggesting a strong ability to generalize and no occurrence of overfitting. The model has largely converged after epoch 100, as indicated by minor fluctuations. These variations are likely a result of the stochastic training process.

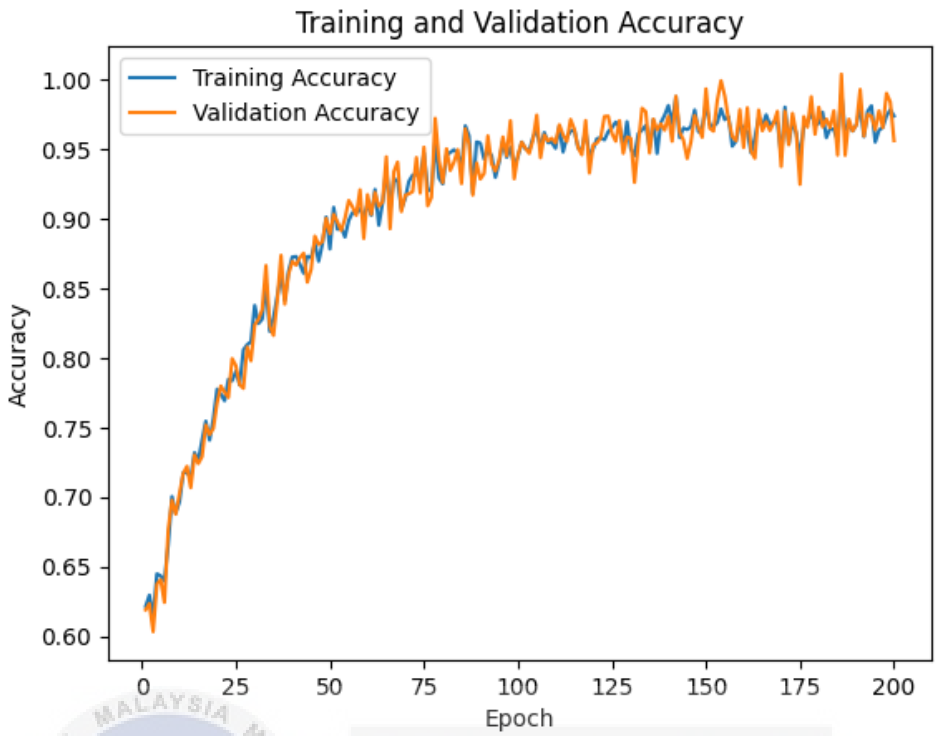

Figure 4.2 : Training and Validation Accuracy for Proposed Model

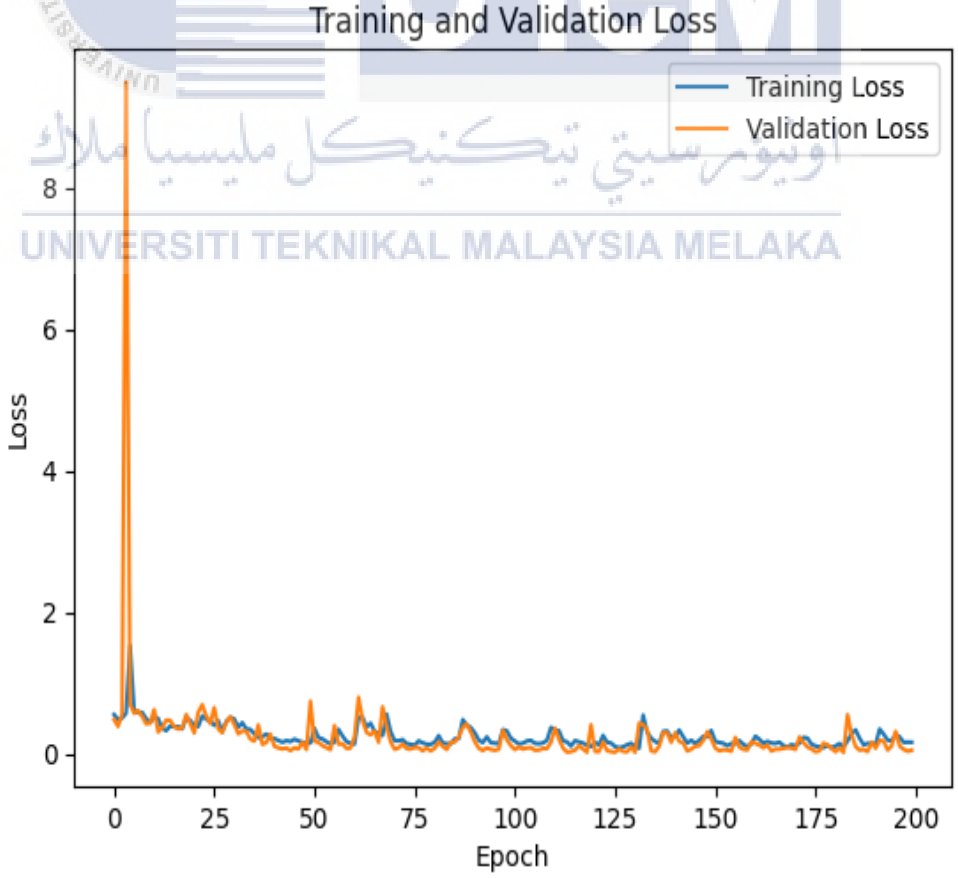

Figure 4.1 : Training and Validation Loss for Proposed Model

Then based on Figure 4.1,the training loss, represented by the blue line, initially begins at a somewhat high value but experiences a significant drop within the first few epochs. The drop observed during training indicates successful acquisition of knowledge from the training data. On the other hand, the validation loss, indicated by the orange line, shows a comparable trend, beginning at a high level and then reducing substantially in the initial epochs. However, it exhibits greater variability after reaching its minimum value at epoch 50. The declining loss numbers reflect ongoing enhancement in the model's performance during the training process. However, the changes found in the validation loss give rise to worries regarding either overfitting or inconsistencies in the validation data. Gaining a comprehensive understanding of these patterns offers useful insights for improving the model and resolving any potential problems that could impact its capacity to generalise.

4.2.2 Confusion Matrix

In this experiment, this project also analyzes the confusion matrix shown in Figure 4.2 which is important to evaluate the performance of the classification model. The categorization performance of the model across many categories of badminton movements is visually represented by the confusion matrix. The number of cases where the true class and the anticipated class match is indicated in each cell. The diagonal cells show the correct predictions, but the off-diagonal cells show the misclassifications. Examining the matrix reveals that most classes have successful predictions, with a preponderance of high counts along the diagonal. Notably, there are 32, 29, 22, 19, 27, and 31 correct predictions for the "Smash," "Serve," "Drive," "Net," "Block," and "Lift" classes, respectively. Still, there are clearly some cases of misclassification, even though they are few. Examples of such misclassifications

include "Serve" being mistaken for "Lift," "Drive" for "Lift," "Net" for "Serve" or "Block," and "Block" for "Lift."

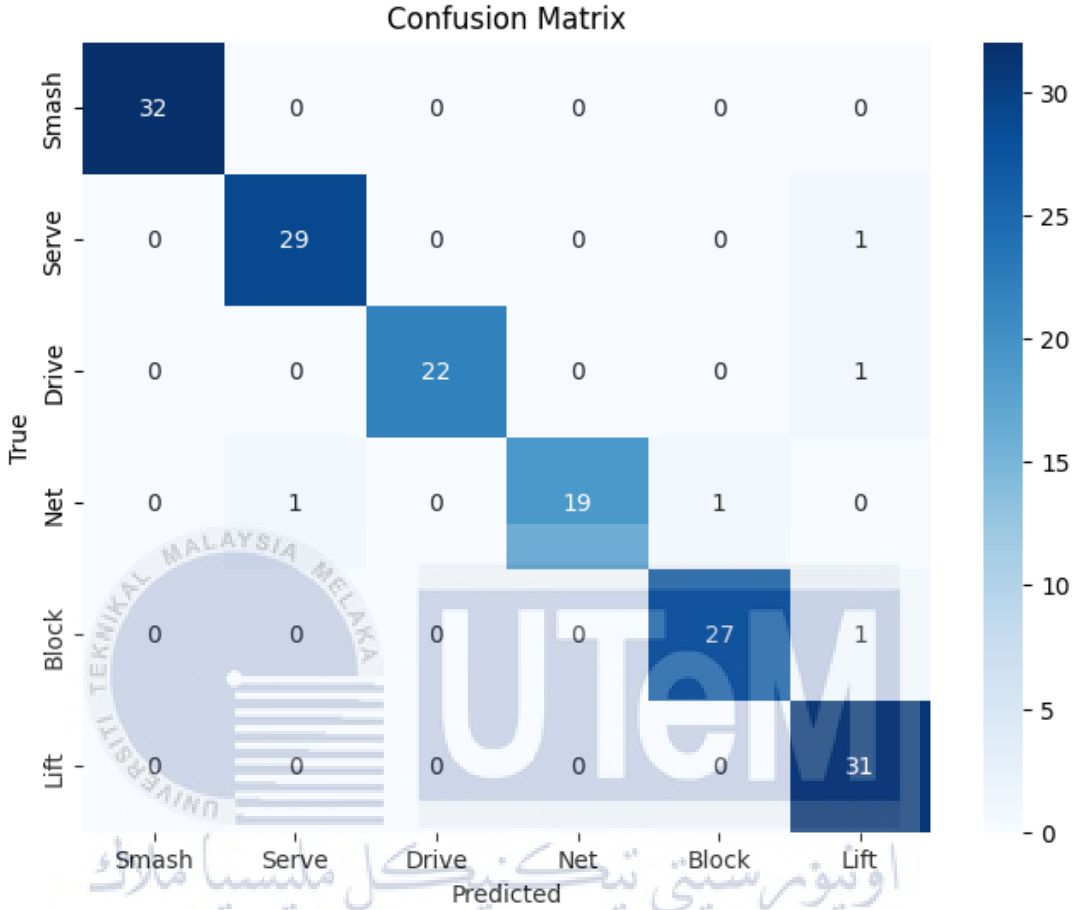

Figure 4.3 : Confusion Matrix for the proposed model.

To achieve one of the outcomes of this project, an experiment to analyze the proposed model's performance was conducted by analyzing the parameters of precision, recall, F1-score and model accuracy. It aims to determine whether the proposed model can classify the six actions well or poorly.

Table 4.2 : Performance Analysis for Proposed LSTM Model.

Parameter	**Action Class**					
	Smash	**Serve**	**Drive**	**Net**	**Block**	**Lift**
Precision	1.00	0.97	1.00	1.00	0.96	0.91
Recall	1.00	0.97	0.96	0.90	0.96	1.00
F1-score	1.00	0.97	0.98	0.95	0.96	0.95
Overall Accuracy (%)	97%					

The Table 4.2 provided presents a comprehensive analysis of the precision, recall, and F1-score metrics for each action class in the badminton recognition system. Precision is a measure that indicates the accuracy of predicting instances belonging to a specific action class. It is calculated by dividing the number of accurately predicted instances by the total number of examples categorised as that action class. Remarkably, classifications such as "Smash," "Drive," and "Net" attained flawless precision scores of 1.00, signifying that every instance classified as these actions were accurately identified. Recall, in a similar vein, quantifies the ratio of accurately anticipated occurrences within the total instances that truly pertain to a specific action class. It is worth mentioning that most classes exhibit strong recall rates. Specifically, the actions "Smash," "Serve," and "Lift" had perfect scores of 1.00, suggesting that the model accurately detected all occurrences of these activities. Moreover, the F1-score, which integrates accuracy and recall into a unified measure, demonstrates the overall efficacy of the model in accurately categorising events across various action

classes. The model's solid performance in successfully recognising badminton motions is further supported by an overall accuracy of 97%. These indicators together offer vital insights into the model's precision, recall, and overall accuracy, showcasing its reliability and usefulness in real-world scenarios.

4.3 Comparing Models

Several models have been trained using the same dataset and using the same hyperparameter as in the proposed model. This is because this experiment compares our proposed model and other models such as GRU (Gated Recurrent Unit), and CNNs (Convolutional Neural Networks) models in terms of graph model accuracy and loss and some parameters to evaluate model performance namely precision, recall, F1-score and model accuracy.

4.3.1 Gated Recurrent Unit

We analyze and compare the Long-Short Term Memory (LSTM) with Gated Recurrent Unit, to assess the performance in recognizing the badminton action. The results of GRU are summarized in Table 4.3 as shown below.

Table 4.3 : Performance Analysis for GRU Model

Batch Size	Epoch	Learning Rate	Precision	Recall	F1-Score	Test Loss	Test Accuracy
16	50	0.001	0.80	0.74	0.70	0.54	0.71
16	100	0.001	0.87	0.86	0.84	0.43	0.84
16	150	0.001	0.91	0.90	0.91	0.23	0.91
16	200	0.001	0.94	0.93	0.93	0.16	0.93
32	50	0.001	0.92	0.90	0.91	0.41	0.91
32	**100**	**0.001**	**0.94**	**0.93**	**0.93**	**0.07**	**0.93**
32	150	0.001	0.78	0.75	0.73	0.45	0.73
32	200	0.001	0.91	0.88	0.88	0.22	0.89
64	50	0.001	0.91	0.88	0.88	0.25	0.89
64	100	0.001	0.89	0.83	0.83	0.41	0.84
64	150	0.001	0.94	0.93	0.93	0.17	0.93
64	200	0.001	0.90	0.85	0.86	0.36	0.86

Upon examining the performance metrics of the GRU model under various batch sizes, epochs, and learning rates, it is evident that the model consistently attains excellent levels of precision, recall, and F1-scores. The test accuracy of the model can reach as high as 93%. The optimal configuration for achieving the greatest results is to use a batch size of 32 and train the model for 100 epochs. This setup produces a precision of 0.94, recall of 0.93, and an F1-score of 0.93. Additionally, the test loss is minimised to 0.07. The GRU model demonstrates consistent and strong performance in recognising badminton actions, especially when using batch sizes of 32 and 64 and training for 200 epochs. Nevertheless, the performance experiences a little decrease when the number of epochs is reduced, indicating the significance of adequate training length for achieving optimal accuracy and minimal loss.

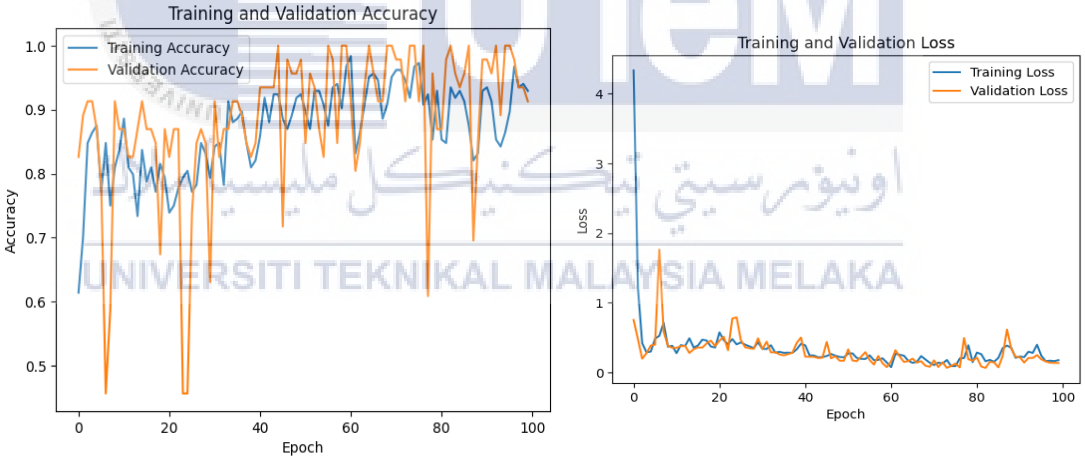

Figure 4.4 : Training and Validation Accuracy/Loss for GRU Model

Based on Figure 4.4 depict the accuracy and loss of a GRU model during training and validation over a span of 100 epochs. The accuracy graph demonstrates that the training accuracy reaches a stable value of approximately 0.9 after some initial volatility, suggesting successful learning. The validation accuracy likewise stabilises, but with more noticeable fluctuations between 0.85 and slightly over 0.9. This indicates that there is unpredictability in the model's performance on unknown data.

The loss graph demonstrates an initial rapid decrease in both training and validation losses, followed by convergence. This suggests successful learning and strong generalisation. The little variations in validation loss and occasional decreases in validation accuracy suggest the possibility of overfitting or inconsistency in the validation set.

4.3.2 Convolutional Neural Networks (CNNs)

We analyze and compare Long-Short Term Memory (LSTM) with Convolutional Neural Networks (CNNs). The results of CNN are summarized in Table 4.4 as shown below.

Table 4.4 : Performance Analysis for CNN Model

Batch Size	Epoch	Learning Rate	Precision	Recall	F1-Score	Test Loss	Test Accuracy
16	50	0.001	0.22	0.5	0.31	0.52	0.45
16	100	0.001	0.22	0.5	0.31	0.65	0.45
16	150	0.001	0.50	0.49	0.50	0.33	0.50
16	200	0.001	0.22	0.5	0.32	0.71	0.46
32	50	0.001	0.9	0.85	0.86	0.36	0.86
32	100	0.001	0.86	0.78	0.78	0.44	0.80
32	150	0.001	0.89	0.83	0.83	0.43	0.84
32	200	0.001	0.86	0.83	0.84	0.39	0.85
64	50	0.001	0.90	0.91	0.90	0.32	0.90
64	100	0.001	0.91	0.90	0.90	0.19	0.90
64	**150**	**0.001**	**0.90**	**0.90**	**0.90**	**0.15**	**0.90**
64	200	0.001	0.86	0.78	0.78	0.41	0.80

Important results are obtained when various parameters for batch sizes, epochs, and a constant learning rate of 0.001 are examined. The performance of the model is poor when the batch size is 16. The test accuracy is between 0.45 and 0.50, and the precision, recall, and F1-scores are poor. A 32-piece batch size increases performance noticeably. With a test accuracy of 0.86, precision, recall, and F1-score attain high levels of around 0.9, 0.85, and 0.86 after 50 epochs. But after 50 epochs, performance starts to fluctuate, which could be a sign of overfitting or a sign that there isn't much more training advantage. A batch size of 64 yields the best results. Across all epochs, precision, recall, and F1-score are continuously high, averaging 0.90. Test accuracy remains approximately 0.90, and after 150 epochs, the lowest test loss of 0.15 is attained, indicating successful generalisation and learning. However, performance starts to slightly decline after 200 epochs, indicating that more training could cause overfitting. With a batch size of 64, the model performs exceptionally well overall, particularly in the range of 100-150 epochs, retaining high test accuracy, low test loss, recall, precision, and F1-scores, demonstrating significant generalisation skills.

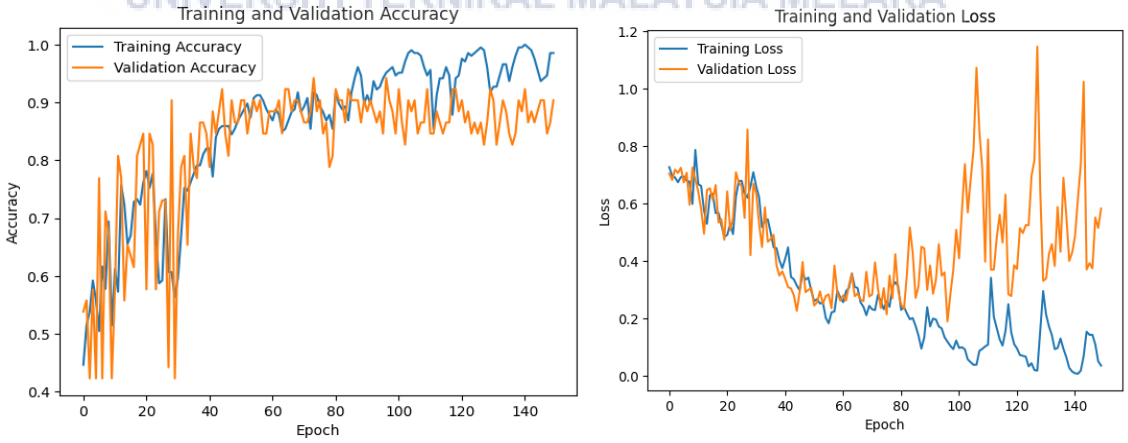

Figure 4.5 : Training and Validation Accuracy/Loss for CNN Model

Figure 4.5 displays the training and validation accuracy graph. Validation accuracy varies but eventually stabilises at 0.85 to 0.9, whereas training accuracy increases gradually to a steady range of 0.9 to 1.0. This indicates that the model is maturing and learning, but it may eventually overfit, resulting in training performance that is higher than reliable validation. Training loss steadily drops to low values, typically less than 0.1, indicating effective learning. But after roughly 80 epochs, validation loss begins to fluctuate and increase, suggesting overfitting. The model's unpredictable behaviour on validation data is emphasised by this difference between training and validation loss.

4.4 Comparison Between a Proposed Model and Other Models

One of the main aspects focused on this project is the comparison of model performance between the proposed model and the other models which are Gated Recurrent Unit (GRU) and Convolutional Neural Networks (CNNs) using the same hyperparameters as in the proposed model and the comparison is selected based on the best overall accuracy acquired from the hyperparameters for each model.

Table 4.5 : Comparison Between Proposed Model with GRU and CNNs

Model	**Parameter**	**Action Class**					
		Smash	**Serve**	**Drive**	**Net**	**Block**	**Lift**
LSTM	Precision	**1.00**	**0.97**	**1.00**	**1.00**	**0.96**	**0.91**
	Recall	**1.00**	**0.97**	**0.96**	**0.90**	**0.96**	**1.00**
	F1-score	**1.00**	**0.97**	**0.98**	**0.95**	**0.96**	**0.95**
	Overall Accuracy	**97%**					
GRU	Precision	1.00	0.87	0.95	0.91	0.96	0.88
	Recall	0.94	0.87	0.91	0.95	0.93	0.97
	F1-Score	0.97	0.87	0.93	0.95	0.93	0.92
	Overall Accuracy	93%					
CNNs	Precision	0.97	0.81	0.91	0.90	0.96	0.88
	Recall	0.94	0.83	0.87	0.90	0.89	0.97
	F1-Score	0.95	0.82	0.89	0.90	0.93	0.92
	Overall Accuracy	90%					

Using precision, recall, F1-score, and total accuracy as measures, the Table 4.5 compares the performance of three models—LSTM, GRU, and CNNs—across six action classes: Smash, Serve, Drive, Net, Block, and Lift. For all action classes, the LSTM model performs exceptionally well with respect to precision, recall, and F1-scores. F1-scores are consistently high, with the lowest being 0.95 (Net, Lift) and the highest 1.00 (Smash, Serve). Precision ranges from 0.91 (Lift) to 1.00 (Smash, Drive, Net), recall from 0.90 (Net) to 1.00 (Smash, Serve, Lift). Out of the three models, this

one has the highest overall accuracy at 97%. With precision ranging from 0.87 (Serve) to 1.00 (Smash), recall from 0.87 (Serve) to 0.97 (Lift), and F1-scores from 0.87 (Serve) to 0.97 (Smash), the GRU model performs similarly but with somewhat lower metrics. As a result, its overall accuracy is 93%. The lowest performing model among the three is the CNN model, which has an overall accuracy of 90%. Its precision ranges from 0.81 (Serve) to 0.97 (Smash), recall from 0.83 (Serve) to 0.97 (Lift), and F1-scores from 0.82 (Serve) to 0.95 (Smash). Consequently, it is evident that the proposed LSTM model outperforms the other two, exhibiting greater accuracy and consistent performance over all assessed action classes.

4.5 Real-Time Simulation

During the real-time simulation phase of the research on action recognition in badminton, a system will be created to analyse and predict actions in real-time as they occur in video footage such as shown in Figure 4.6 **: Real-Time Simulation User Interface** . A few critical components will be incorporated into the video to improve its functionality and offer a clear understanding of the model's performance. A shot count will be displayed in the upper left corner, which will show the running tally of detected shots and the predicted action for each shot, updated in real-time. An indicator of probability will be displayed in the upper right quadrant, which will represent the model's confidence in its predictions as a percentage. In addition, the video will feature key points that represent critical components of the player's body, such as joints, to facilitate the visualisation of the player's movements and their alignment with the anticipated actions. A pre-trained action recognition model will be employed by this system to analyse frames, identify actions, increment the shot count, and update predictions with their associated probabilities. In parallel, a pose estimation model will identify and visualise key points on the participant. A comprehensive real-

time display that demonstrates the capabilities and accuracy of the action recognition model will be generated by combining all this information into the video feed, rendering it a valuable instrument for sports analytics.

4.5.1 Benchmarking AI Model Performance Against Human Observation

Figure 4.6 : Real-Time Simulation User Interface

A structured table will be used to capture essential metrics for evaluation in the comparative analysis between the AI model's performance and manual observation. This table will include columns that represent different badminton actions such as smash, serve, drive, net shot, block, and lift. The "Manual Observation" column will record the observations made by individuals regarding the actions seen in the video clips, which will be used as a benchmark for comparison. In a similar vein, the column labelled "AI Model Prediction" will showcase the predictions produced by the AI model for each action, revealing its identification through video frame analysis. The "Error Rate" column quantifies discrepancies between manual observations and AI predictions by calculating the percentage of incorrect predictions relative to the total actions observed. Finally, the "Accuracy" column will indicate the AI model's

performance by calculating the percentage of correct predictions in relation to the total actions observed. With a structured presentation, you can easily evaluate the accuracy and effectiveness of the AI model in different action types. This will give you valuable insights to improve the model and enhance its applications in sports analytics. The analysis derived from the Table 4.6 and Figure 4.7 is as follows:

Table 4.6 : Prediction of LSTM Model VS Human Manual Observation

Action	Manual Observation	AI Model	Error Rate (%)	Accuracy (%)
Smash	13	13	0	100
Serve	3	2	33.3	66.67
Net	1	0	100	0
Drive	1	0	100	0
Block	3	5	66.67	33.33
Lift	14	15	7.14	92.86

Figure 4.7 : Real-Time Simulation

After examining the AI model's ability to recognise badminton actions compared to human observation, clear patterns may be observed for different actions. These patterns provide insights into both the strengths and weaknesses of the model. The model demonstrates exceptional accuracy in identifying smash motions, obtaining a flawless accuracy rate of 100%. This indicates that the model successfully captures the distinctive characteristics of a smash, allowing for accurate identification. Nevertheless, the model's performance in activities such as the serve, net, drive, and block is noticeably less reliable. For example, although the model correctly predicts two out of three serve actions, it encounters difficulties with the net and drive actions, misclassifying all cases. The difference in performance may arise from the intricacy and variety inherent in these motions, resulting in difficulties in identifying and distinguishing features. Furthermore, the misclassification of block actions underscores the model's challenge in differentiating nuanced differences in technique or distinguishing them from other activities. In contrast, the lift action exhibits a relatively robust performance, with just one misclassification out of fourteen times, showing a high degree of accuracy. However, the occurrence of a single misclassification highlights the necessity for additional improvement, namely in distinguishing lift actions from comparable motions. Overall, the AI model demonstrates competence in identifying specific acts, but it encounters difficulties in accurately categorising others. This emphasises the need for ongoing refining and adjustment to enhance performance in all sorts of actions. To tackle these problems, it may be necessary to improve feature identification algorithms, broaden the range of training data, and enhance the model's capacity to distinguish subtle differences in technique. These efforts will ultimately enhance the model's performance in real-world applications of action recognition in badminton.

4.5.2 Environment and Sustainability

The project greatly contributes to sustainability by aligning with two crucial Sustainable Development Goals (SDGs): SDG 9 - Industry, Innovation, and Infrastructure, and SDG 13 - Climate Action.

Our utilization of cutting-edge technology in sports, namely in the precise categorization of various badminton shot styles, showcases our dedication to progressing the field and its infrastructure. Our project utilizes state-of-the-art computer vision techniques and machine learning algorithms to analyze and improve sports training and performance. Not only does this stimulate innovation in the sports industry, but it also fosters the growth of sophisticated technology infrastructure to facilitate athletic training and competition. In addition, our project places great importance on sustainability, specifically addressing its environmental consequences, in line with SDG 13 - Climate Action. The system's low hardware needs and ability to be used remotely provide substantial benefits in terms of decreasing the environmental impact associated with sports training and analysis. Our project enhances resource efficiency and environmental conservation by reducing the need on complex technology and physical infrastructure. In addition, the capacity to carry out training and analysis from a distance decreases the necessity for travel, thereby further reducing carbon emissions and promoting environmentally friendly practices in sports and athletics.

To summarize, our project's novel application of technology in sports not only enhances the industry and infrastructure, but also encourages sustainability by minimizing environmental harm and cultivating climate-conscious practices. Our project demonstrates the interdependence of industry, innovation, infrastructure, and

climate action by embracing technology advancements and environmental responsibility. It contributes positively to sustainable development endeavors.

CHAPTER 5:

CONCLUSION AND FUTURE WORKS

5.1 Conclusion

Overall, the creation of a sophisticated badminton action identification system with YOLOv7 Pose and LSTM signifies a notable progress in the realm of sports analytics. During this project, we have effectively shown the efficiency of our suggested approach, which included training and assessing the system's performance. Through the utilization of advanced deep learning approaches, specifically the Long Short-Term Memory (LSTM) model, we have successfully attained precise monitoring and dependable identification of diverse badminton movements. The success of our system is based on its capacity to efficiently analyse and comprehend intricate sequences of body key points, allowing it to precisely categorize movements such as Smash, Serve, Drive, Net, Block, and Lift. Chapter 4 delved into multiple facets of our research, encompassing dataset acquisition, data preprocessing, key point extraction, LSTM model structure, training methodologies, and classification

methods. In addition, we performed thorough hyperparameter tweaking to enhance the system's performance, resulting in the attainment of the utmost precision. The evaluation findings, as shown in Table 4.4, unequivocally establish the superiority of the proposed LSTM model compared to alternative methods like CNNs and GRUs. The LSTM model regularly demonstrates superior performance in precision, recall, and F1-score compared to other models across all action classes. The LSTM model has exceptional accuracy of 97% in accurately identifying badminton actions, outperforming both the GRU and CNN models.

5.2 Future Works

There are a few strategies that could be investigated in future research to enhance this project's performance. These strategies aim to address shortcomings or areas that require improvement that have been found throughout the current project.

5.2.1 Dataset Expansion and Diversity

To improve the performance and ability of the intelligent badminton action recognition system to apply to a wide range of situations, it is necessary to focus on increasing the size and variety of the dataset. Gathering supplementary samples from a range of players, skill levels, playing styles, and environments can yield a more extensive and varied set of training data. The augmentation of this dataset can enhance the model's ability to accurately and reliably capture the diverse range of badminton motions, resulting in greater precision and resilience.

5.2.2 Integration of Shuttlecock Trajectory Data

By integrating shuttlecock trajectory data with the existing pose data, we can enhance the available information for action recognition. Through the examination of the shuttlecock's path and the tracking of player motions using pose estimation, the

system can acquire a more thorough comprehension of the game's dynamics and context. By integrating this collective data, it is possible to achieve enhanced precision and contextual understanding in identifying badminton movements, particularly in high-speed rallies or intricate gameplay situations.

5.2.3 Fine-tuning Hyperparameters and Model Architecture

Continuously fine-tuning the hyperparameters and architecture of the LSTM model can greatly enhance its performance and efficiency. By conducting trials with various setups, such as altering the number of LSTM layers, hidden units, learning rates, and dropout rates, it is possible to enhance both the accuracy and convergence speed. In addition, investigating sophisticated optimization approaches, such as learning rate scheduling, gradient clipping, and batch normalizing, can further improve the stability and generalization abilities of the model.

REFERENCES

[1] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artif Intell Rev*, vol. 53, no. 8, pp. 5455–5516, Dec. 2020, doi: 10.1007/s10462-020-09825-6.

[2] G. Torres-Luque, J. C. Blanca-Torres, J. M. Giménez-Egido, D. Cabello-Manrique, and E. Ortega-Toro, "Design, Validation, and Reliability of an Observational Instrument for Technical and Tactical Actions in Singles Badminton," *Front Psychol*, vol. 11, Dec. 2020, doi: 10.3389/fpsyg.2020.582693.

[3] T. Lin, Y. Yang, J. Beyer, and H. Pfister, "Labeling Out-of-View Objects in Immersive Analytics to Support Situated Visual Searching," Dec. 2021, doi: 10.1109/TVCG.2021.3133511.

[4] Y. L. Hsu, H. C. Chang, and Y. J. Chiu, "Wearable Sport Activity Classification Based on Deep Convolutional Neural Network," *IEEE Access*, vol. 7, pp. 170199–170212, 2019, doi: 10.1109/ACCESS.2019.2955545.

[5] H. Ma and X. Pang, “Research and analysis of sport medical data processing algorithms based on deep learning and internet of things,” *IEEE Access*, vol. 7, pp. 118839–118849, 2019, doi: 10.1109/ACCESS.2019.2936945.

[6] L. Qin, D. Wang, M. Guo, X. Sun, and X. Chen, “Optimizing Badminton Action Recognition with Deep Learning and Sensor Fusion: A Study of Sensor Numbers and Combinations,” in *Proceedings of 13th IEEE International Conference on CYBER Technology in Automation, Control, and Intelligent Systems, CYBER 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 273–278. doi: 10.1109/CYBER59472.2023.10256460.

[7] T. Steels, B. Van Herbruggen, J. Fontaine, T. De Pessemier, D. Plets, and E. De Poorter, “Badminton activity recognition using accelerometer data,” *Sensors (Switzerland)*, vol. 20, no. 17, pp. 1–16, Sep. 2020, doi: 10.3390/s20174685.

[8] N. A. Rahmad, M. A. As’Ari, and M. A. As’ari, “The new Convolutional Neural Network (CNN) local feature extractor for automated badminton action recognition on vision based data,” in *Journal of Physics: Conference Series*, Institute of Physics Publishing, Jun. 2020. doi: 10.1088/1742-6596/1529/2/022021.

[9] L. Yu and N. I. Mohamad, “Development of Badminton-specific Footwork Training from Traditional Physical Exercise to Novel Intervention Approaches,” *Physical Activity and Health*, vol. 6, no. 1, pp. 219–225, 2022, doi: 10.5334/paah.207.

[10] J. Liu and B. Liang, “An Action Recognition Technology for Badminton Players Using Deep Learning,” *Mobile Information Systems*, vol. 2022, 2022, doi: 10.1155/2022/3413584.

[11] A. Cejudo, “Risk Factors for, and Prediction of, Shoulder Pain in Young Badminton Players: A Prospective Cohort Study,” *Int J Environ Res Public Health*, vol. 19, no. 20, Oct. 2022, doi: 10.3390/ijerph192013095.

[12] N. T. Ayuningtyas, H. Susanto, and S. Suroto, “Relationship between Somatotype and Physical Fitness: Study on Badminton Athletes of PB Djarum Kudus,” *Jurnal Keolahragaan*, vol. 9, no. 1, May 2021, doi: 10.21831/jk.v9i1.38147.

[13] Z. Ming, Z. Bo, X. Dan, and 衰对比研究 张明社会资本视角下保龄球的迅速衰败和羽毛球的长盛不, “A Comparative Study of the Rapid Decline of Bowling and the Enduring Prosperity of Badminton from the Perspective of Social Capital,” 2021.

[14] S. Xuan, K. Wang, L. Liu, C. Liu, and J. Li, “Algebra Based Human Skeleton Sequence Reduction and Action Recognition,” in *Frontiers in Artificial Intelligence and Applications*, IOS Press BV, Dec. 2021, pp. 467–476. doi: 10.3233/FAIA210435.

[15] S. Ni and D. Yao, “Sports Dance Action Recognition System Oriented to Human Motion Monitoring and Sensing,” *Wirel Commun Mob Comput*, vol. 2021, 2021, doi: 10.1155/2021/5515352.

[16] P. Agrawal, J. Carreira, and J. Malik, "Learning to See by Moving," in *2015 IEEE International Conference on Computer Vision (ICCV)*, IEEE, Dec. 2015, pp. 37–45. doi: 10.1109/ICCV.2015.13.

[17] H. B. Zhang *et al.*, "A comprehensive survey of vision-based human action recognition methods," *Sensors (Switzerland)*, vol. 19, no. 5. MDPI AG, Mar. 01, 2019. doi: 10.3390/s19051005.

[18] H. Jiang and S. B. Tsai, "An Empirical Study on Sports Combination Training Action Recognition Based on SMO Algorithm Optimization Model and Artificial Intelligence," *Math Probl Eng*, vol. 2021, 2021, doi: 10.1155/2021/7217383.

[19] Q. Liang, "Application of Convolution Neural Network (CNN) Model Combined with Pyramid Algorithm in Aerobics Action Recognition," *Comput Intell Neurosci*, vol. 2021, 2021, doi: 10.1155/2021/6170070.

[20] X. Chang and L. Peng, "Visual Sensing Human Motion Detection System for Interactive Music Teaching," *J Sens*, vol. 2021, 2021, doi: 10.1155/2021/2311594.

[21] A. Toshev and C. Szegedy, "DeepPose: Human pose estimation via deep neural networks," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, Sep. 2014, pp. 1653–1660. doi: 10.1109/CVPR.2014.214.

[22] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields," Dec. 2018, [Online]. Available: http://arxiv.org/abs/1812.08008

[23] T. L. Munea, Y. Z. Jembre, H. T. Weldegebriel, L. Chen, C. Huang, and C. Yang, "The Progress of Human Pose Estimation: A Survey and Taxonomy of Models Applied in 2D Human Pose Estimation," *IEEE Access*, vol. 8, pp. 133330–133348, 2020, doi: 10.1109/ACCESS.2020.3010248.

[24] B. Sapp, A. Toshev, and B. Taskar, "LNCS 6312 - Cascaded Models for Articulated Pose Estimation," 2010.

[25] H. Zhang and Z. Jiang, "Agreement function model for pose estimation," *Electron Lett*, vol. 52, no. 20, pp. 1677–1679, Sep. 2016, doi: 10.1049/el.2016.2338.

[26] R. S. Ghiass and D. Laurendeau, "Highly accurate and fully automatic 3D head pose estimation and eye gaze estimation using RGB-3D sensors and 3D morphable models," *Sensors (Switzerland)*, vol. 18, no. 12, Dec. 2018, doi: 10.3390/s18124280.

[27] M. Eichner and V. Ferrari, "Human pose co-estimation and applications," *IEEE Trans Pattern Anal Mach Intell*, vol. 34, no. 11, pp. 2282–2288, 2012, doi: 10.1109/TPAMI.2012.85.

[28] N. Chen, S. Wu, Y. Chen, Z. Wang, and Z. Zhang, "A Pose Estimation Algorithm for Multimodal Data Fusion," *Traitement du Signal*, vol. 39, no. 6, pp. 1971–1979, Dec. 2022, doi: 10.18280/ts.390609.

[29] H. Sak, A. Senior, and F. Beaufays, “Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition,” Feb. 2014, [Online]. Available: http://arxiv.org/abs/1402.1128

[30] C. Avci, B. Tekinerdogan, and C. Catal, “Analyzing the performance of long short-term memory architectures for malware detection models,” *Concurr Comput*, vol. 35, no. 6, p. 1, Mar. 2023, doi: 10.1002/cpe.7581.

[31] F. Kratzert, D. Klotz, C. Brenner, K. Schulz, and M. Herrnegger, “Rainfall-runoff modelling using Long Short-Term Memory (LSTM) networks,” *Hydrol Earth Syst Sci*, vol. 22, no. 11, pp. 6005–6022, Nov. 2018, doi: 10.5194/hess-22-6005-2018.

[32] Z. Chen, H. T. Blair, and J. Cong, “Energy-Efficient LSTM Inference Accelerator for Real-Time Causal Prediction,” *ACM Transact Des Autom Electron Syst*, vol. 27, no. 5, Jun. 2022, doi: 10.1145/3495006.

[33] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, “Deep Learning with Long Short-Term Memory for Time Series Prediction,” *IEEE Communications Magazine*, vol. 57, no. 6, pp. 114–119, Jun. 2019, doi: 10.1109/MCOM.2019.1800155.

[34] T. M. Tai, Y. J. Jhang, Z. W. Liao, K. C. Teng, and W. J. Hwang, “Sensor-Based Continuous Hand Gesture Recognition by Long Short-Term Memory,” *IEEE Sens Lett*, vol. 2, no. 3, Sep. 2018, doi: 10.1109/LSENS.2018.2864963.

[35] T. Ergen and S. S. Kozat, “Unsupervised anomaly detection with LSTM neural networks,” *IEEE Trans Neural Netw Learn Syst*, vol. 31, no. 8, pp. 3127–3141, Aug. 2020, doi: 10.1109/TNNLS.2019.2935975.

[36] S. Hochreiter and J. ¨ Urgen Schmidhuber, “Long Short-Term Memory.”

[37] Z. C. Lipton, J. Berkowitz, and C. Elkan, “A Critical Review of Recurrent Neural Networks for Sequence Learning,” 2015.

[38] E. López, C. Valle, H. Allende, E. Gil, and H. Madsen, “Wind power forecasting based on Echo State Networks and Long Short-Term Memory,” *Energies (Basel)*, vol. 11, no. 3, Feb. 2018, doi: 10.3390/en11030526.

[39] X. Ran, Z. Shan, Y. Fang, and C. Lin, “An LSTM-based method with attention mechanism for travel time prediction,” *Sensors (Switzerland)*, vol. 19, no. 4, Feb. 2019, doi: 10.3390/s19040861.

[40] S. S. Baek, J. Pyo, and J. A. Chun, “Prediction of water level and water quality using a cnn-lstm combined deep learning approach,” *Water (Switzerland)*, vol. 12, no. 12, Dec. 2020, doi: 10.3390/w12123399.

[41] T. Steels, B. Van Herbruggen, J. Fontaine, T. De Pessemier, D. Plets, and E. De Poorter, “Badminton activity recognition using accelerometer data,” *Sensors (Switzerland)*, vol. 20, no. 17, pp. 1–16, Sep. 2020, doi: 10.3390/s20174685.

[42] Z. Wang, M. Guo, and C. Zhao, “Badminton Stroke Recognition Based on Body Sensor Networks,” *IEEE Trans Hum Mach Syst*, vol. 46, no. 5, pp. 769–775, Oct. 2016, doi: 10.1109/THMS.2016.2571265.

[43] W. Gawin, A. Herbstreit, U. Fries, and C. Maiwald, “Evaluation of a Freely Available Sensor Racket as a Diagnostic and Training Tool in Elite Badminton,” *International Journal of Racket Sports Science*, vol. 4, no. 1, pp. x–x, 2022, doi: 10.30827/xx.

[44] W. Su and J. Feng, “Research on Methods of Physical Aided Education Based on Deep Learning,” *Sci Program*, vol. 2022, 2022, doi: 10.1155/2022/6447471.

[45] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black, “Towards understanding action recognition,” in *Proceedings of the IEEE International Conference on Computer Vision*, Institute of Electrical and Electronics Engineers Inc., 2013, pp. 3192–3199. doi: 10.1109/ICCV.2013.396.

[46] H. Xu *et al.*, “The role of medial frontal cortex in action anticipation in professional badminton players,” *Front Psychol*, vol. 7, no. NOV, Nov. 2016, doi: 10.3389/fpsyg.2016.01817.

[47] J. Rasmussen and M. de Zee, “A simulation of the effects of badminton serve release height,” *Applied Sciences (Switzerland)*, vol. 11, no. 7, Apr. 2021, doi: 10.3390/app11072903.

[48] J. Liu and B. Liang, “An Action Recognition Technology for Badminton Players Using Deep Learning,” *Mobile Information Systems*, vol. 2022, 2022, doi: 10.1155/2022/3413584.

[49] IEEE Control Systems Society, Institute of Electrical and Electronics Engineers, IEEE Instrumentation and Measurement Society, Institute of Electrical and Electronics Engineers. Kolkata Section. Joint CSS-IMS Chapter,

and Jadavpur University, *CMI 2016 : 2016 IEEE First International Conference on Control, Measurement and Instrumentation (CMI) : January 8-10, 2016, Kolkata, India*.

[50] M. Anik, M. Hassan, H. Mahmud, and K. Hasan, "Activity Recognition of a Badminton Game Through Accelerometer and Gyroscope," in *19th International Conference on Computer and Information Technology*, M. Anik, M. Hassan, H. Mahmud, and K. Hasan, Eds., Dhaka,Bangladesh: Islamic University of Technology (IUT), Gazipur, Bangladesh, Dec. 2016.

[51] Z. Y. Yip, I. Mohd Khairuddin, W. H. Mohd Isa, A. P. P. Abdul Majeed, M. A. Abdullah, and M. A. Mohd Razman, "Badminton Smashing Recognition through Video Performance by using Deep Learning," *MEKATRONIKA*, vol. 4, no. 1, pp. 70–79, Jun. 2022, doi: 10.15282/mekatronika.v4i1.8607.

[52] Z. Bo, Y. Peng, Q. Kaiyi, and L. Fengshuo, "Intelligent System of Badminton Serve Action Based on YOLOv5 and OpenPose," in *2023 4th International Conference on Computer Engineering and Application, ICCEA 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 341–346. doi: 10.1109/ICCEA58433.2023.10135542.

[53] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. F. Li, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, Sep. 2014, pp. 1725–1732. doi: 10.1109/CVPR.2014.223.

[54] D. Tran, H. Wang, L. Torresani, J. Ray, Y. Lecun, and M. Paluri, "A Closer Look at Spatiotemporal Convolutions for Action Recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, Dec. 2018, pp. 6450–6459. doi: 10.1109/CVPR.2018.00675.

[55] K. Hara, H. Kataoka, and Y. Satoh, "Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, Dec. 2018, pp. 6546–6555. doi: 10.1109/CVPR.2018.00685.

[56] X. Zhang, C. Xu, X. Tian, and D. Tao, "Graph edge convolutional neural networks for skeleton-based action recognition," *IEEE Trans Neural Netw Learn Syst*, vol. 31, no. 8, pp. 3047–3060, Aug. 2020, doi: 10.1109/TNNLS.2019.2935173.

[57] M. C. Leong, D. K. Prasad, Y. T. Lee, and F. Lin, "Semi-CNN architecture for effective spatio-temporal learning in action recognition," *Applied Sciences (Switzerland)*, vol. 10, no. 2, Jan. 2020, doi: 10.3390/app10020557.

[58] Q. Liang, "Application of Convolution Neural Network (CNN) Model Combined with Pyramid Algorithm in Aerobics Action Recognition," *Comput Intell Neurosci*, vol. 2021, 2021, doi: 10.1155/2021/6170070.

[59] M. Shi, Y. Meng, and W. Yang, "Skeleton Action Recognition Based on Double Residual Convolutional Neural Network," in *Journal of Physics:*

Conference Series, IOP Publishing Ltd, Feb. 2022. doi: 10.1088/1742-6596/2170/1/012004.

[60] H. Zhao, A. Wang, and Y. Fan, "Research on the Identification and Evaluation of Aerobics Movements Based on Deep Learning," *Sci Program*, vol. 2021, 2021, doi: 10.1155/2021/6433260.

[61] Y. Yin, J. Lin, Q. Zhu, S. Zhang, Y. Zhang, and M. Liu, "Method for Detection of Unsafe Actions in Power Field Based on Edge Computing Architecture," 2021, doi: 10.21203/rs.3.rs-181932/v1.

[62] N. Jaouedi, N. Boujnah, and M. S. Bouhlel, "Deep learning approach for human action recognition using Gated Recurrent Unit neural networks and motion analysis," *Journal of Computer Science*, vol. 15, no. 7. Science Publications, pp. 1040–1049, 2019. doi: 10.3844/jcssp.2019.1040.1049.

[63] H. Ullah, A. Munir, S. Member, and S. Member, "Human Activity Recognition Using Cascaded Dual Attention CNN and Bi-Directional GRU Framework," 2023, doi: 10.36227/techrxiv.20304450.v1.

[64] L. Lu, C. Zhang, K. Cao, T. Deng, and Q. Yang, "A Multichannel CNN-GRU Model for Human Activity Recognition," *IEEE Access*, vol. 10, pp. 66797–66810, 2022, doi: 10.1109/ACCESS.2022.3185112.

[65] D. P. Sari, L. Karlitasari, and F. D. Wihartiko, "Clean Water Demand Prediction Model Using The Long Short Term Memory (LSTM) Method," *Komputasi: Jurnal Ilmiah Ilmu Komputer dan Matematika*, vol. 20, no. 2, pp. 160–168, Jul. 2023, doi: 10.33751/komputasi.v20i2.8060.

[66] C. Saraswathy, K. Nandhini, and R. Kalaivani, “Traffic sign detection and recognition based on convolutional neural network,” *Int J Health Sci (Qassim)*, pp. 1224–1238, Jun. 2022, doi: 10.53730/ijhs.v6ns8.9884.

[67] I. Bibi, A. Akhunzada, J. Malik, J. Iqbal, A. Mussaddiq, and S. Kim, “A Dynamic DL-Driven Architecture to Combat Sophisticated Android Malware,” *IEEE Access*, vol. 8, pp. 129600–129612, 2020, doi: 10.1109/ACCESS.2020.3009819.

APPENDICES

Appendix A

The code for dataset preprocessing and key point extraction (Keypointfinder.Py):

```
from utils.torch_utils import select_device
from models.experimental import attempt_load
from utils.plots import output_to_keypoint, plot_skeleton_kpts, plot_one_box_kpt, colors
from utils.general import non_max_suppression_kpt, strip_optimizer
from torchvision import transforms
import tensorflow
from PIL import ImageFont, ImageDraw, Image


# Creating a load_classes function so we can load the coco.names file and after reading each of
#the name in coco.names file we can return the names in the form of a list
def load_classes(path):
    # Loads *.names file at 'path'
    with open(path, 'r') as f:
        names = f.read().split('\n')
    return list(filter(None, names))

# This is our Main Function
@torch.no_grad()
def run(poseweights='yolov7-w6-pose.pt', source='pose.mp4', device='cpu', names = 'utils/coco.names', line_thickness = 2):

    path = source
    ext = path.split('/')[-1].split('.')[-1].strip().lower()
    if ext in ["mp4", "webm", "avi"] or ext not in ["mp4", "webm", "avi"] and ext.isnumeric():
        input_path = int(path) if path.isnumeric() else path
        device = select_device(opt.device)
        names = load_classes(names)
        half = device.type != 'cpu'
        model = attempt_load(poseweights, map_location=device)
        _ = model.eval()

        cap = cv2.VideoCapture(input_path)
        webcam = False

        if (cap.isOpened() == False):
            print('Error while trying to read video. Please check path again')

        fw, fh = int(cap.get(3)), int(cap.get(4))
        if ext.isnumeric():
            webcam = True
            fw, fh = 1280, 768
        vid_write_image = letterbox(
            cap.read()[1], (fw), stride=64, auto=True)[0]

        resize_height, resize_width = vid_write_image.shape[:2]
        out_video_name = "output" if path.isnumeric(
        ) else f"{input_path.split('/')[-1].split('.')[0]}"
```

```
        out = cv2.VideoWriter(f"{out_video_name}_wed.mp4", cv2.VideoWriter_fourcc(
            *'mp4v'), 30, (resize_width, resize_height))
        if webcam:
            out = cv2.VideoWriter(f"{out_video_name}_mon.mp4", cv2.VideoWriter_fourcc(
                *'mp4v'), 30, (fw, fh))

        frame_count, total_fps = 0, 0

        # ====Loading the  custom font ===========
        fontpath = "sfpro.ttf"
        font = ImageFont.truetype(fontpath, 32)
        # =================================
        sequence = []
        keypoints = []
        j = 1
        seq = 30
        # ========================================
        while cap.isOpened:

            print(f"Frame {frame_count} Processing")
            ret, frame = cap.read()
            if ret:
                origimage = frame
                #Creating a Black Mask
                mask=np.zeros(frame.shape[:2] , dtype="uint8")
                #Defining the ROI for the Player 2 only
                roi = cv2.rectangle(mask, (433, 607), (1471, 1001),(255,255,255), -1)
                #Overlapping the original image on the black mask
                masked=cv2.bitwise_and(frame,frame,mask=mask)
                masked[np.where((masked==[0,0,0]).all(axis=2))]=[255,0,0]
                orig_image = masked

                # preprocess image
                image_ = cv2.cvtColor(origimage, cv2.COLOR_BGR2RGB)
                image = cv2.cvtColor(orig_image, cv2.COLOR_BGR2RGB)
                image_ = letterbox(image_, (resize_width), stride=64, auto=True)[0]

                image = letterbox(image, (resize_width), stride=64, auto=True)[0]
                image__ = image_.copy()
                image___ = image.copy()
                image_ = transforms.ToTensor()(image_)
                image = transforms.ToTensor()(image)
                image_ = torch.tensor(np.array([image_.numpy()]))
                image = torch.tensor(np.array([image.numpy()]))
                image_ = image_.to(device)
```

```
            gn = torch.tensor(img.shape)[[1, 0, 1, 0]]
            for i, pose in enumerate(output_data):

                if len(output_data):
                    for c in pose[:, 5].unique():
                        n = (pose[:, 5] == c).sum()
                        print("No of Objects in Current Frame : {}".format(n))

                    for det_index, (*xyxy, conf, cls) in enumerate(reversed(pose[:,:6])):
                        c = int(cls)
                        kpts = pose[det_index, 6:]
                        label = names[c]
                        plot_one_box_kpt(xyxy, img_img, label=label, color=colors(c, True),
                                    line_thickness=opt.line_thickness, kpts=kpts, steps=3,
                                    orig_shape=img.shape[:2])
                        # preprocess model input data and return the keypoints of the body =======
                        if j <= seq:
                            for idx in range(output.shape[0]):
                                kpts = output[idx, 7:].T
                                plot_skeleton_kpts(img_img, kpts, 3)
                                sequence.append(kpts.tolist())
                        #The keypoints predictions from the pose estimation model is stacked as
                        #a sequence of 30 frames and each sequence contain 51 features
                        # (X-coordinate, y-coordinate and the confidence score of 17 key points)
                        if len(sequence) == 30:
                            keypoints.append(sequence)
                            print(sequence)
                            print(keypoints)
                        if j == seq:
                            sequence = []
                            j = 0
                        j += 1
            if webcam:
                cv2.imshow("Detection", img_img)
                key = cv2.waitKey(1)
                if key == ord('c'):
                    break
            else:
                img_ = img.copy()
                img_ = cv2.resize(
                    img_, (960, 540), interpolation=cv2.INTER_LINEAR)
                cv2.imshow("Detection", img_img)
                cv2.waitKey(1)
```

Appendix B

The essential code highlight for Real Time Simulation (same flow as Keypointfinder.py but adding the pretrained model to predict the action and UI elements.

```
object_counter = {'Smash-Shot': 0, 'Serve-Shot': 0, 'Drive-Shot': 0, 'Net-Shot': 0, 'Lift-Shot': 0, 'Block-Shot': 0}

#Creating a Function  by the name load_classes,using this function we will load the coco.names file and read each
#of the object name, in the coco.names file and this function will return all the object names in the coco.names
#file in the form of a list
def load_classes(path):
    # Loads *.names file at 'path'
    with open(path, 'r') as f:
        names = f.read().split('\n')
    return list(filter(None, names))  # filter removes empty strings (such as last line)

#This is the main function, here we are passing the yolov7 pose weights, by default we are setting the deivce as CPU
#setting the line thick ness of the skeleton lines as well

@torch.no_grad()
def run(poseweights='yolov7-w6-pose.pt', source='pose.mp4', device='cpu', names = 'utils/coco.names', line_thickness = 2):

    path = source
    ext = path.split('/')[-1].split('.')[-1].strip().lower()
    if ext in ["mp4", "webm", "avi"] or ext not in ["mp4", "webm", "avi"] and ext.isnumeric():
        input_path = int(path) if path.isnumeric() else path
        device = select_device(opt.device)
        names = load_classes(names)
        half = device.type != 'cpu'
        model = attempt_load(poseweights, map_location=device)
        _ = model.eval()

        cap = cv2.VideoCapture(input_path)
        webcam = False

        if (cap.isOpened() == False):
            print('Error while trying to read video. Please check path again')
```

```
        # == 5.0 == variable declaration===========
        sequence = []
        keypoints = []
        pose_name = ''
        posename_list = []
        actions = np.array(['Smash-Shot', 'Serve-Shot', 'Drive-Shot', 'Net-Shot', 'Lift-Shot', 'Block-Shot'])
        label_map = {label: num for num, label in enumerate(actions)}
        j = 1
        seq = 30
        # ==========================================
```

```
253             if pose_name == "Smash-Shot":
254                 cv2.line(img_img2, ((width - (width-50)),25), ((width - (width-200)),25), [85,45,255], 40)
255                 cv2.putText(img_img2, "Shot Type", ((width - (width-50)),35), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
256                 cv2.line(img_img2, ((width - (width-50)),75), ((width - (width-500)),75), [85,45,255], 40)
257                 cv2.putText(img_img2, pose_name, ((width - (width-50)),85), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
258
259             elif pose_name == "Serve-Shot":
260                 cv2.line(img_img2, ((width - (width-50)),25), ((width - (width-200)),25), [85,45,255], 40)
261                 cv2.putText(img_img2, "Shot Type", ((width - (width-50)),35), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
262                 cv2.line(img_img2, ((width - (width-50)),75), ((width - (width-500)),75), [85,45,255], 40)
263                 cv2.putText(img_img2, pose_name, ((width - (width-50)),85), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
264
265             elif pose_name == "Drive-Shot":
266                 cv2.line(img_img2, ((width - (width-50)),25), ((width - (width-200)),25), [85,45,255], 40)
267                 cv2.putText(img_img2, "Shot Type", ((width - (width-50)),35), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
268                 cv2.line(img_img2, ((width - (width-50)),75), ((width - (width-500)),75), [85,45,255], 40)
269                 cv2.putText(img_img2, pose_name, ((width - (width-50)),85), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
270
271             elif pose_name == "Net-Shot":
272                 cv2.line(img_img2, ((width - (width-50)),25), ((width - (width-200)),25), [85,45,255], 40)
273                 cv2.putText(img_img2, "Shot Type", ((width - (width-50)),35), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
274                 cv2.line(img_img2, ((width - (width-50)),75), ((width - (width-500)),75), [85,45,255], 40)
275                 cv2.putText(img_img2, pose_name, ((width - (width-50)),85), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
276
277             elif pose_name == "Lift-Shot":
278                 cv2.line(img_img2, ((width - (width-50)),25), ((width - (width-200)),25), [85,45,255], 40)
279                 cv2.putText(img_img2, "Shot Type", ((width - (width-50)),35), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
280                 cv2.line(img_img2, ((width - (width-50)),75), ((width - (width-500)),75), [85,45,255], 40)
281                 cv2.putText(img_img2, pose_name, ((width - (width-50)),85), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
282
283             elif pose_name == "Block-Shot":
284                 cv2.line(img_img2, ((width - (width-50)),25), ((width - (width-200)),25), [85,45,255], 40)
285                 cv2.putText(img_img2, "Shot Type", ((width - (width-50)),35), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
286                 cv2.line(img_img2, ((width - (width-50)),75), ((width - (width-500)),75), [85,45,255], 40)
287                 cv2.putText(img_img2, pose_name, ((width - (width-50)),85), 0, 1, [225, 255, 255], thickness=2, lineType=cv2.LINE_AA)
288
```