Video Action Recognition for Indoor Sports
(versão corrigida após defesa)

Guilherme Poeta Fernandes

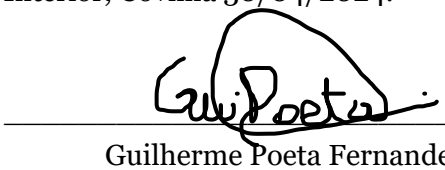
Dissertação para obtenção do Grau de Mestre em
Engenharia Informática
(2º ciclo de estudos)

Orientador: Prof. Doutor Bruno Miguel Correia da Silva
Coorientador: Prof. Doutor Hugo Pedro Martins Carriço Proença

abril de 2024

Declaração de Integridade

Eu, Guilherme Poeta Fernandes, que abaixo assino, estudante com número de inscrição M11101 do curso de 2ºciclo de Engenharia Informática da Faculdade de Engenharia, declaro ter desenvolvido o presente trabalho e elaborado o presente texto em total consonância com o Código de Integridade da Universidade da Beira Interior.
Mais concretamente afirmo não ter incorrido em qualquer das variedades de Fraude Académica, e que aqui declaro conhecer, e que em particular atendi à exigida referenciação de frases, extratos, imagens e outras formas de trabalho intelectual, e assim assumo na íntegra as responsabilidades da autoria.
Universidade of Beira Interior, Covilhã 30/04/2024.



Guilherme Poeta Fernandes

Acknowledgments

"You have to start from the bottom, assemble it piece by piece until you get where you want to be. And many obstacles will have to be overcome for everything to fit together. Sometimes we make mistakes and we have to start from scratch, picking up the pieces and recovering what was lost. Not everyone is prepared and many give up and stop where they are. In the midst of these ups and downs, there are times when we get so low that we can't see where to find the strength to get up. This is the so-called winter of life, which comes from time to time to take away our colours and many times it seems to have no end. It may even last a little longer, but life follows its seasons and after the winter it brings you spring full of flowers and that air of good news. It brings you reasons to go on. And more than that, it gives you chances and opportunities. But it depends on each one's will to see or to continue where you are."

- Andressa Karina Voltolini

I am humbled to begin this Dissertation with a touching message that summarizes the transformational journey I have embarked on since the beginning of this course. During the development, I faced many obstacles that tested my resilience and challenged the progress of this Dissertation. There were times when these obstacles slowed development progress, leaving me stranded not knowing what to do. Nonetheless, during these tough moments I had to live up to my expectations and concentrate on working with everything that I had.

The ability to work under pressure is a definite skill that will be of tremendous importance in the professional world. I have come to learn that even though these obstacles have brought me multiple changes, they have gradually helped me to be mindful and resourceful, improving my ability to handle complexity and finding new ways to solve problems.

The hardships and obstructions that I have dealt with throughout the development of this Dissertation have provided me with valuable teachings that have helped me grow beyond my comfort zone and break out of the limits I had set for myself for years. The act of figuring out the best way to tackle certain issues was an essential learning experience because it shows an understanding of the importance of perseverance and persistence.

These experiences turned out to be very fulfilling because they show me the importance of overcoming adversity, always pursuing my goals no matter how difficult they are, and seeing the bright side when I am faced with difficulties or challenges, so this will for sure be an asset that I will bring with me into the future.

I sincerely thank those who helped me greatly on this journey.

Firstly, I specify that my gratitude goes to **Prof. PhD. Bruno Miguel Correia da Silva**, who has been a guide, a strong supporter, and my encourager, and whose contribution to the completion of this work has been great for the realisation of this Dissertation. His guidance was essential in reaching the right direction for this work.

Secondly, **Ph.D. Prof. Hugo Pedro Martins Carriço Proença** supported and stayed with me over the process until the end of this Dissertation completion, offering his suggestions and several hints.

Thirdly, the work described in this Dissertation was carried out at the Instituto de Telecomunicações and Secure and Intelligent Networked Software Systems Laboratory (SINSLab) - Covilhã Delegation, located at the Universidade da Beira Interior, Covilhã, Portugal. A big thank you to both.

Fourthly, **to all my friends**, our nightly fun discussions and battles were always a source of inspiration and determination. For this, you are the reason why I managed to keep fighting my fears and self-doubt throughout my academic journey.

During this, your sincere friendship kept reminding me to stay strong and be more confident to face the problems ahead. I would like to acknowledge your continuous support in helping me whenever I needed the most, sharing resources, giving me inputs whenever needed, and other awesome activities that we shared during these past years.

And last but not least, I want to make **a special mention of my beloved family** to express my utmost gratitude for their unconditional love, support, and constant inspiration to me in the course of writing my Dissertation. From the start, they have served a constant source of support and encouragement, supporting me through the ups and downs with their unwavering faith in my abilities. Since as long ago as I can remember, I've always been motivated in improving and the prospect of learning and gaining knowledge. Without you beside me, my learning journey would not have been possible, and your dedication has been the stone that has helped me make my dream come true. I cannot put into words how grateful I am for every single sacrifice you made so that I was able to fulfil my academic dream.

Thank You.

Resumo

A crescente demanda por tecnologia de Inteligência Artificial na área do desporto surgiu como uma tendência proeminente e notável nos últimos anos. Para além de auxiliar os árbitros em momentos cruciais na tomada de decisões, esta surgiu como um fator de mudança, fornecendo aos treinadores capacidades analíticas avançadas que melhoram exponencialmente os métodos de treino, o desempenho e a capacidade atlética das suas equipas durante os jogos. A integração da Inteligência Artificial e da visão computacional em várias aplicações desportivas, tais como a monitorização da condição física dos jogadores, a deteção de lesões, a transmissão e a tecnologia *wearable*, acelerou ainda mais esta revolução. Embora os sensores e os dispositivos de localização ofereçam dados valiosos, as suas limitações em tempo real são ultrapassadas pelo potencial da análise de vídeo para extrair informações essenciais.

Na indústria do desporto televisivo, a Inteligência Artificial melhorou significativamente a cobertura desportiva, transformando a experiência de visualização para os espectadores. Esta transformação tem sido possível graças à evolução do reconhecimento da ação humana no campo da visão computacional, tornando a classificação de acções e a localização espácio-temporal de acções aspectos fundamentais e extensivamente estudados. Através do reconhecimento de actividades, este estudo visa detetar e classificar eventos ou actividades pré-treinados a partir de gravações de vídeos.

Neste trabalho, pretendemos compreender a eficácia das Redes Neurais Convolucionais (Convolutional Neural Networks (CNN)) em comparação com outros métodos tradicionais e desenvolver um sistema de baixo custo e alto rendimento para o reconhecimento de ações durante um jogo de Basquetebol. Este sistema fornece dados importantes aos treinadores para avaliar e melhorar o desempenho da sua equipa. Através da união entre tecnologia e desporto, esta Dissertação visa melhorar a análise de Basquetebol, explorando soluções fiáveis através da aplicação de métodos aprendizagem profunda (Deep Learning (DL)) ao conjunto de dados.

Palavras-chave

Análise de Vídeos Desportivos, Basquetebol, Inteligência Artificial, Aprendizagem profunda, Reconhecimento de Acções, Redes Neurais Convolucionais, Redes Neurais Profundas, Processamento de Imagem e Vídeo, Análise Desportiva.

Resumo Alargado

A crescente demanda por tecnologia de Inteligência Artificial na área do desporto surgiu como uma tendência proeminente e notável nos últimos anos. Para além de auxiliar os árbitros em momentos cruciais na tomada de decisões, esta surgiu como um fator de mudança, fornecendo aos treinadores capacidades analíticas avançadas que melhoram exponencialmente os métodos de treino, o desempenho e a capacidade atlética das suas equipas durante os jogos. A integração da Inteligência Artificial e da visão computacional em várias aplicações desportivas, tais como a monitorização da condição física dos jogadores, a deteção de lesões, a transmissão e a tecnologia *wearable*, acelerou ainda mais esta revolução. Embora os sensores e os dispositivos de localização ofereçam dados valiosos, as suas limitações em tempo real são ultrapassadas pelo potencial da análise de vídeo para extrair informações essenciais.

Na indústria do desporto televisivo, a Inteligência Artificial melhorou significativamente a cobertura desportiva, transformando a experiência de visualização para os espectadores. Esta transformação tem sido possível graças à evolução do reconhecimento da ação humana no campo da visão computacional, tornando a classificação de acções e a localização espácio-temporal de acções aspectos fundamentais e extensivamente estudados. Através do reconhecimento de actividades, este estudo visa detetar e classificar eventos ou actividades pré-treinados a partir de gravações de vídeos.

Embora a análise de vídeo prometa aplicações inovadoras no desporto, também apresenta desafios em vários domínios, incluindo mudanças de movimento, posicionamento da câmara, ruído de fundo e disposição do campo de jogo durante as gravações. O reconhecimento de acções de forma independente depende de vários factores, como a pose humana e a interação com objectos. Estes desafios podem ser resolvidos através da criação de um conjunto de características que descrevem o movimento na sua forma mais detalhada e da formação de um classificador com base neste conjunto de características seleccionadas.

Uma excelente solução para os problemas de reconhecimento de vídeo reside na utilização de Redes Neurais Profundas (Deep Neural Networks (DNN)). Ao contrário da aprendizagem automática convencional, estas permitem aos sistemas de Inteligência Artificial aprender autonomamente características que representam acções específicas, eliminando a necessidade de uma engenharia manual extensiva. Neste sentido, este trabalho centra-se na investigação e aplicação de abordagens que visam o reconhecimento de actividades para desportos de pavilhão, usando como caso de estudo o jogo de Basquetebol, tirando partido de tecnologias de visão computacional.

Neste trabalho, pretendemos compreender a eficácia das Redes Neurais Convolucionais (CNN) em comparação com outros métodos tradicionais e desenvolver um sistema de baixo custo e alto rendimento para o reconhecimento de ações durante um jogo de Basquetebol. Este sistema fornece dados importantes aos treinadores para avaliar e melhorar o desempenho da sua equipa. Através da união entre tecnologia e desporto, esta Dissertação visa melhorar a análise de Basquetebol, explorando soluções fiáveis através da aplicação de métodos aprendizagem profunda (DL) ao conjunto de dados.

No final desta Dissertação, devemos ter desenvolvido um sistema eficaz no reconhecimento de determinadas ações durante os jogos/treinos, que seja de baixo custo capaz de atender às necessidades dos clubes de basquetebol mais pequenos, e desta forma, permitir que estes possam utilizar o mesmo para melhorar o seu jogo e poderem, possivelmente, equiparar-se a clubes maiores.

Abstract

The increasing demand for Artificial Intelligence (AI) technologies in sport has emerged as a major note in recent years. Besides helping referees at crucial decision times, AI has become a game changer, giving coaches advanced analytical capabilities that dramatically improve their teams' training strategies, performance and athleticism during games through AI integration of computer vision into sports applications. Through wearable technology, this transformation has been accelerated by sensors and positioning devices that provide valuable information, but overcome their real-time limitations through the ability of video analytics to extract data needs to be increased.

Television (TV) AI technology in the sports industry has greatly enhanced sports coverage, transforming the viewing experience for the fans at home. This shift was driven by the evolution of human action recognition in computer vision, which made action classification and spatio-temporal action placement an important area that has been extensively studied through motion recognition.

While video analysis promises new applications in sports, it also presents challenges in a variety of settings, including changes in speed, camera placement, background noise, and field conditions during recording. Independent action detection is based on various factors such as human position and interaction with objects. These difficulties can be overcomed by designing segments that adequately describe motion and developing classifications based on these selected features

A good solution to video recognition problems lies in the use of DNN. Unlike traditional machine learning, DNN are able to learn features representing specific behaviors from AI systems without requiring extensive manual processes In this regard, this Dissertation focused on research and techniques in the field of computer vision for indoor sports, namely Basketball.

In this work, we aim to understand the efficiency of CNN compared to other traditional methods and develop an action recognition algorithm for Basketball games. This system provides valuable data for coaches to monitor and improve their team's performance. By bridging the gap between technology and sports, this Dissertation seeks to improve Basketball research, applying deep learning techniques to data sets to find reliable solutions.

Keywords

Sport Video Analysis, Basketball, Artificial Intelligence, Deep Learning, Action Recognition, Convolutional Neural Networks, Deep Neural Networks, Image and Video Processing, Sports Analytics.

Contents

List of Figures

List of Tables

Acronyms

AI	Artificial Intelligence
ANN	Artificial Neural Network
CDC	Clube Desportivo da Covilhã
CNN	Convolutional Neural Networks
CVAT	Computer Vision Annotation Tool
DL	Deep Learning
DNN	Deep Neural Networks
FOV	Field of view
fps	frames per second
GPU	Graphics Processing Unit
GMM	Gaussian Mixture Model
GPS	Global Positioning System
GUI	Graphical User Interface
HAR	Human Action Recognition
HEVC	High Efficiency Video Coding
HOG	Histogram of Oriented Gradients
IoT	Internet of Things
IOU	Intersection over Union
KDE	Kernel Density Estimation
KNN	K-Nearest Neighbour
LBP	Local Binary Pattern
LSTM	Long Term Short Memory
mAP	Mean Average Precision
ML	Machine Learning
NMS	Non-Maximum Suppression

NN	Neural Networks
Ph.D.	Doctor of Philosophy
RFID	Radio Frequency Identification
ROI	Regions of Interest
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Networks
SIFT	Scale Invariant Feature Transform
SSD	Single Shot Multibox Detectors
SVM	Support Vector Machines
TCN	Temporal Convolutional Networks
TV	Television
UBI	Universidade da Beira Interior
UHD	Ultra High Definition
VM	Virtual Machine
VGG	Visual Geometry Group
YOLO	You Only Look Once

Chapter 1

Introduction

In the last few years, the request for the integration of AI technology in sports seems to be growing at a high speed, which has become one of the emerging topics in the technological development world. Moreover, helping referees with decision-making at critical moments as well as supply coaches with analytic ability that is often times greater than anything for their team's development, training, and performance during the game. During the last few years, the usage of AI and computer vision has been highly requested in numerous sports for applications such as monitoring player fitness, player injury detection, broadcasting and streaming, wearable technology, etc. Through the use of sensors and some tracking devices, we're able to extract some valuable data, although being limited and challenging to analyse it in real-time. Placing sensors on the player's body during a practice or game is far from practical in order to collect data in a real-world scenario. These flaws regarding the use of sensors are easily fixable if, through video analysis, we're able to extract information rather than using data sensors.

Sports TV industries have also implemented AI technology in order to enhance sports coverage and spectators' viewing experiences, all thanks to the evolution of human action recognition in computer vision. Being able to assign a video to a set of predefined action classes and identifying an action's spatio-temporal location are two of the most fundamental and massively studied topics in this context. Through activity recognition, we intend to detect and classify pre-trained events or activities from multiple video recordings. Although video analysis can be very useful and revolutionary in the fields of sports, unfortunately, it still encompasses some problems within multiple domains. During a recording, there are several differentiating features within a specified category of images, such as shifts in movement, camera positioning, background noise, and the courts's layout. The recognition of an action on its own is also dependent on multiple factors, for instance, the human pose and the objects with which it interacts. The above-mentioned setbacks can be solved by creating a set of features that can describe motion in its finest form. A classifier is then trained on this selected set of features. Even though this solution sounds quite simple to handle, the time we have to spend in order to create a valid set of features is considerable. An excellent solution that can provide satisfactory results for video recognition problems is the use of DNN. These, in short, allow us to train an AI to predict outputs given a set of inputs. In this case, we'll need to find a set of features for the training data that will classify them into a defined number of classes. DNN are very different from conventional machine learning. As mentioned above, if we wanted to train an AI, we would need to devote a considerable amount of time to engineering a conventional machine learning system to detect the features that represent an action in

Basketball. With DL, we only need to worry about supplying the system with a very large quantity of images of the specific action being taken, and the system will autonomously learn the features that represent it. This Dissertation aims to address the activity recognition capabilities during a Basketball game through the use of AI.

1.1 Motivation and Objectives

This Dissertation was proposed by professor Bruno Miguel Correia da Silva Doctor of Philosophy (Ph.D.) with the co-supervision of professor Hugo Proença Ph.D. within the context of the course module ”Dissertation or Internship Project in Computer Engineering” at Universidade da Beira Interior (UBI). The primary objective of this Dissertation was to discover a proficient fusion of Internet of Things (IoT) and AI that could transform sports training, elevate performance analysis, and encompassing injury prevention in the sport of Basketball.

The videos gathered throughout the entire Dissertation were filmed at Clube Desportivo da Covilhã (CDC) at Covilhã, Portugal, using real game footage. The activities in Basketball videos are primarily divided into three groups: recognising players and differentiating between teams, monitoring the basketball's movement, and assessing successful shots. These groups will be further discussed in **Chapter 4**.

The main objective of this Dissertation is to study the implementation of CNN performance and to develop a system capable of accurate action recognition analysis of a Basketball game that will provide the coach with valuable data that will eventually allow the team to enhance their future performance. To achieve this main objective, the following partial goals were defined:

- **Fundamental Concepts and Related Work** - we explored essential principles and relevant prior research to establish a solid foundation for our study.
- **Proposed Method** - we proposed our approach to address the research issue, outlining the methodology and techniques employed.
- **Results** - we presented and analysed the outcomes of our study, presenting findings and insights.
- **Testing** - we accessed and validated the performance and functionality of our developed system.

For this Dissertation, we focused on **1)** creating a system that uses a simple and common camera for video capture. The goal is to create a system that is simple to use and with low cost to the coach, and **2)** analyse and recognise simple, but very important, game actions, such as a successful made baskets and basic player actions.

1.2 Dissertation Organisation

The organisation of this Dissertation is thoughtfully devised to enable a thorough examination of the topic. The subsequent chapters provide a coherent progression of insights and analyses.

Chapter 1: Presents the topic at hand, action recognition in indoor sports. With a focus on the dynamics, challenges, and opportunities within the field of indoor sports, we explore the fundamental concepts for our research. We then present the motivation for its development and its main objectives to develop a specialised action recognition framework and evaluate its performance in real-world indoor sports scenarios.

Chapter 2: Provides a foundational introduction to AI, along with an exploration of its relevant branches that interconnect with the core subject. The chapter also explores the detailed overview of AI-driven algorithms optimised for action recognition within the sports domain.

Chapter 3: Examines the current state of technology, focusing on technologies currently under study.

Chapter 4: Details the proposed methodology in a creative way, providing a systematic approach to gaining valuable insights from the perspectives captured throughout the Dissertation's trajectory.

Chapter 5: Examines the essence of our research, focusing on Basketball and highlighting the results of our analysis of practice standards for indoor sports. We examine the implications of our findings for athlete performance, coaching methods, and spectator engagement, and we also highlight some limitations. We also provide an insight into the methods.

Chapter 6: Represents the culmination of the progress and efforts in this Dissertation, and combines the main findings and the conclusion of the research with a brief but comprehensive overview of the broader topic, providing valuable insights for future development.

The goal of this thoroughly organised Dissertation is to offer an in-depth analysis of the combination of AI with sports action recognition, leading to a clear comprehension of the topic and its possible implications.

Chapter 2

Fundamental Concepts and Related Work

2.1 Chapter Overview

In this chapter, we review essential concepts linked to AI, its assimilation into contemporary advanced technologies, and the solutions that offer similar features to indoor sports video action recognition systems.

Section 2.2 introduces AI, presenting its subfields and the advances that it has has made since its initial conceptualisation to its current state.

Section 2.3 focuses on the significance of action recognition in sports analysis, exploring the importance of identifying actions within the sports context. We also explore the challenges that need to be overcome for precise recognition, and lastly, we highlight the latest methodologies.

Section 2.4 explores essential aspects like pre-processing techniques, feature extraction, motion analysis, object detection, DL-based trackers, and spatio-temporal representations. Together, these elements collaboratively contribute to interpreting actions within sports videos, enhancing our ability to gather relevant information.

Section 2.5 explores the use of CNN for Action Recognition specifically for Basketball. Our exploration will present how these networks bring about a transformative shift in the interpretation of Basketball actions recorded in videos.

Section 2.6 we explore the complexities and obstacles that emerge when attempting to accurately recognise and analyse actions in the dynamic context of sports scenarios.

Section 2.7 encapsulates the most relevant findings regarding this chapter.

2.2 Artificial Intelligence

AI dates back to the mid-20th century. While American computer scientist John McCarthy [3] coined the term ”Artificial Intelligence” at a Dartmouth workshop in 1956, its foundation and origins can be traced back to the 1940s and 1950s. They have contributed significantly to the theoretical and mathematical roots in this case, but we must ask ourselves, ”What exactly is AI?” AI can be defined as the intelligence displayed by machines

[4] and has become an effective method of learning and reasoning.

After it was used by the Allies during World War II, Alan Turing asked a simple question that would change history forever: **”Can machines think?”**. When his famous book ”Computing Machinery and Intelligence” was released in 1950, followed by the ”Turing Test,” Alan Turing set out what would become the central goal and vision of AI.[5] Therefore it can be considered a field within computer science whose main goal is to develop [6] machines that can exhibit intelligent thought and behavior. In terms of size, AI includes many branches, and Figure 2.1 illustrates some of its well-known branches.

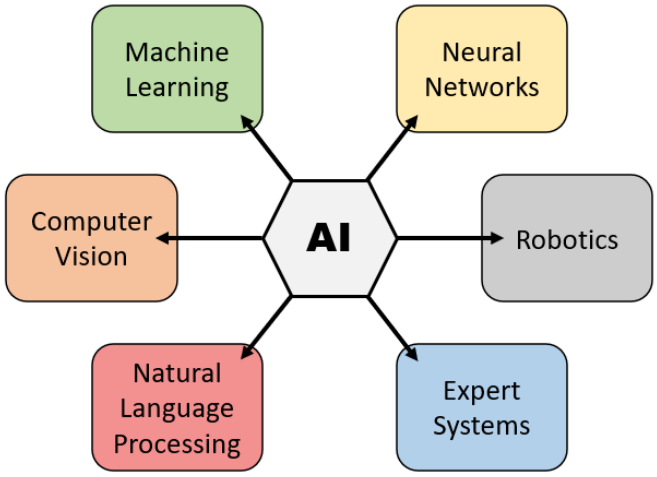


Figure 2.1: AI well-known branches.

As shown in the previous figure, there exist several subdomains such as Machine Learning (ML) and DL, these play an important role in processing large amounts of unstructured data, e.g., text, video, images, and many more [7].

2.2.1 Types of Artificial Intelligence

AI can be divided into these two separate categories, **weak AI** and **strong AI**:

- **Weak Artificial Intelligence**

  Weak AI refers to a learning algorithm that is only capable of performing a determined task. The AI in question can only complete a specific task without any human assistance. However, many AI systems are still regarded as weak AI. These systems have been implemented in areas like robotics and driving assistance. They include ”Alexa,” ”Siri,” and chatbots, among other devices. Incidentally, AI may have the capacity to perform complex problems better than most humans, yet still, it is tightly limited to what it can do based on its programming alone [8].

- **Strong Artificial Intelligence**

  AI's outstanding mental attributes and abilities very closely resemble the ones that we humans possess. In the case of a human equivalent AI, we would no longer be able to use the simple process of observation to distinguish the software solution from the human brain because AI attempts to emulate the exact functions of the living human brain [9]. The second view of this perspective is that a computer can cleverly be programmed to purely stimulate our complex human "intelligence," displaying the ability to recognise feelings, emotions, and other emotional conditions typical of humans.

Therefore, as previously stated, weak AI refers to AI systems that are designed and trained to perform a specific task or a very limited range of tasks. Regarding the system under development, we will attempt to identify and detect particular elements within the playing field (basketballs, players, and made baskets) of a Basketball match. Such tasks are assigned to Narrow AI due to the fact that they are mostly designed towards solving a specific problem with a targeted resolution. Using computer vision dynamics and the latest DL system, the system will be able to process video frames from a Basketball game and find and monitor not only the basketball but also its players. The CNN, which is one of the DL models, showed high efficiency and effectiveness for Internet of Things (IoT) image and object recognition tasks.

2.3 Action Recognition in Indoor Sports: A Focus on Basketball

2.3.1 Overview on Action Recognition

Usually, when the term "artificial intelligence" is mentioned, most people immediately think of robots. This association comes from the depiction of humanoid machines in movies and TV shows that often leads, in general, to rebellion against humans; however, there is a complete distance between this perception and reality. As mentioned earlier, AI's ultimate goal is to develop machines that can mimic basic human cognitive functions such as learning, reasoning, and perception.

Some of the AI systems that were deemed to belong to the present technology have now become obsolete because of recent technological advancements. Say, for example, that today's machines that were able to do basic mathematical calculations or recognise text through the process of optical character recognition do not already qualify as AI since these are already built-in computer features. Also, as AI technology continues to progress, some sectors already have this technology as an advantage. As part of the given Dissertation, we will be taking a detailed look at one of those sectors, the use of AI in Basketball. The unique environment of the game makes it an ideal setting for utilising AI technologies.

2.3.2 Importance of Action Recognition in Sports Analysis

AI has begun to revolutionise Basketball as a whole; many of the existing Basketball elements are currently undergoing a deep restructuring. Having a vast range of variables comprised of many factors, including the analysis of individual player performance, a real-time coaching and control system, player care using the latest techniques to avoid injuries, as well as fan involvement and immersion. Eventually, we will illustrate AI in Basketball in more detail and, of course, how AI has fundamentally changed Basketball through the implementation of technology. We will discuss the multiple AI applications and show how they contribute to both the business and the operation.

1. **Performance Analysis and Player Development**

   The technology of AI has been of great help to coaches and players when studying the performance of Basketball players. Moreover, coaches and players can now have the opportunity to get the relevant details of the game mechanics. Within it, AI coaches are able to run analysis algorithms through data, including tracking players, shot charts, and performance metrics, to identify and break down useful trends and players' strengths and weaknesses.

   Nowadays, there are various tracking technologies, such as optical tracking systems, wearable sensors, and so on. These technologies can gather very useful information while the teams are physically active. Algorithms for AI can go through the database and perform an analysis that will give metrics related to player movement, positioning, speed, acceleration, and so on. Training and coaching staff can also take advantage of these resources to plan training better, make decisions based on data, and create individualised development plans for each player [10].

   AI algorithms also possess the ability to analyse the spaces taken by shots and the selections made with regard to shots to see when making the most critical shots, raising effectiveness in this regard. Through the examination of shot charts, AI systems can offer suggestions to players on possible modifications to their techniques, release points, and target areas for their shot attempts. Rong Ji, Shenyang Sport University [11], explored an interesting method to recognise Basketball shooting gestures using image feature extraction and ML, while also employing feature selection and Gaussian hidden variables for accurate classification and recognition of Basketball shooting gestures.

2. **Tactical Decision-Making and Game Planning**

   Tactical decision-making in Basketball has improved over the years, all thanks to advances in AI technology, providing valuable insights for coaches to plan their games and develop tactical adjustments. AI algorithms that are effective in detecting opponents' moves, lineup optimisation, historical data analysis, player performance data, and real-time game data can also help coaches to develop offensive and defensive strategies.

AI is able to look at the videos from archived games to spot the weak points of the opposing team and come up with their strengths. Additionally, this play analysis process will benefit offensive coaches by providing defensive game plans and exploiting opponents' weaknesses.

Furthermore, this real-time data can be processed during the game, for example, the positions and movements of players, the basketball, and shooting analysis. They would offer more tips and advice on the spot to the coaches, and by means of this real-time analysis, coaches can make instant decisions about whether a substitution is needed, an adjustment of the defence, or a different offensive play call.

3. **Injury Prevention and Player Health**

   Integrated AI technology has played an important role in avoiding player injuries as well as preserving player health in Basketball. This enabled the detection of injury risk indicators in athlete movements, providing early alerts for coaches and health specialist personnel [12].

   Wearable sensors equipped with AI may be used to monitor and examine, right on the spot, a player's movements, force to joints, and other physical indices. Whether a player shows a sign of abnormal movement, fatigue, etc., these can be indicated as a potential symptom of a possible injury. Furthermore, programmes designed to optimise the recovery and rehabilitation of injured players also benefit from the use of AI. By analysing information about the players' recovery, physical behaviour, and past injuries, AI algorithms can deliver personalised injury recovery plans and monitor player progress, ensuring a safe return to the court.

In recent years, several sports companies and organisations have been exploring and attempting to implement action recognition systems for various purposes, like athlete performance analysis, coaching, and fan engagement, as mentioned earlier. Some of these companies and organisations that have been known to use or explore action recognition systems in sports are as follows:

- **Catapult Sports** [13]: Leading provider of wearable technology for sports tracking and performance analysis. They offer a variety of features, including Global Positioning System (GPS) trackers and inertial sensors, which give players valuable data on acceleration, distance covered, acceleration, and more;
- **STATSports**[14]: Company specialising in athlete tracking systems. They also provide wearable GPS devices that track and analyse various metrics during training and games to help teams perform better and reduce the risk of injury;
- **Playsight Interactive**[15]: Uses the SmartCourt system, which uses a combination of cameras and AI to provide real-time video analysis of sports such as Tennis, Basketball and Soccer. This allows coaches and players to review performance, explore what options they have at their disposal, and make data-driven decisions;

- **Kinexon**[16]: Provides a wearable sensor that captures real-time data of player movement and interaction during team play. Their framework helps teams explore strategy, player placement, and other key business metrics;
- **Hawk-Eye Innovations**[17]: Widely known for ball control technology used in Tennis and Cricket. It provides real-time data for oficials, broadcasters and fans.

The implementation of AI in Basketball has brought significant benefits to player performance analysis, as discussed earlier. With this technology, Basketball can benefit from these new innovations that can shape the future of the game. Outlined below is an overview of some technologies that proved invaluable when designing this Basketball-focused Dissertation.

- **Second Spectrum**[18]: Second Spectrum is a technology company that provides advanced analytics and video analysis for Basketball. Their technology uses cameras to capture player movement and basketball tracking, providing insight into player positioning and style of play. The data is used by teams, and researchers to conduct in-depth performance analysis;
- **ShotTracker**[19]: Provides a system that tracks player movement and shot attempts on the court in real time. The system includes wearable sensors for players and sensors in Basketball, allowing teams to collect data on shot accuracy, shot location, and player position;
- **Nex Team**[20]: Expert in AI-driven analytics for Basketball. Their technology tracks player movement, Basketball elements, and interactions, providing detailed insights into player performance, organisation, and game performance.

There are a lot of corporations and technologies that have been used in Basketball through action recognition and analytics. Real-time examination, imaginative graphics, and fascinating experiences are some of the technologies that have illustrated the Basketball match in a deep way for fans, transforming the channel of communication between the viewer and the events from a spectative to an interactive one. Basketball is a sport that is still in its developing stage, and that is where these companies find an opportunity to play a role in moulding the future of the game by pushing what can be achieved both on and off the court.

2.3.3 Challenges and Limitations in Action Recognition

In the action recognition field, with its applications in various areas applicable to sport analysis, surveillance, and human-computer interaction, is an interesting subject. However, it is subject to sets of problems and restrictions, which deserve a deep and comprehensive thinking before the issue can be creatively solved. Actions happening in such dynamic contexts usually possess a very complex nature, and it is difficult to understand

them properly. However, these challenges act as reference points that stimulate continuous research and improvements in strong and accurate action recognition systems.

- **Intra-Class Variability**

  Human actions have many spatial, temporal, and contextual parameters such as speed, angle, posture, etc., which are difficult for robots to replicate. Especially in the Basketball arena, such uniformity relies on non-identical shooting forms, dribbling techniques, and defensive plays made by the players. Every individual player introduces a specific playing style, which represents a significant challenge to the recognition systems that have to be reliable and consistent to discriminate between one and the same player action [21, 22];

- **Occlusion and Temporal Dependencies**

  Problems in the real world are usually content with occlusions, which means uncovering objects and people that are blocking the action's visibility. In this context, not only can players temporarily obstruct each other, but the basketball can also become occluded, introducing complexities to precise recognition [23]. Moreover, the potential for one action to either influence or follow another introduces additional layers of complexity to the recognition process;

- **Environmental Factors**

  Lighting variations, dynamic backgrounds, and the subtleties of playing surfaces all contribute to the inconstancy of captured video data. These environmental factors introduce noise and variations, highly affecting the extraction of certain features for accurate action recognition [24];

- **Limited and Imbalanced Datasets**

  Training complex models will always depend on comprehensive and diverse datasets. Acquiring a well-annotated dataset that is able to cover the various potential actions is a difficult task. Inconsistency in distribution of data, when some cases are underrepresented, ruins the viable performance of the models;

- **Computational Demands**

  The use of advanced models, e.g., CNN, for real-time action recognition consumes significant computer resources. The computational effort needed to introduce complex models into a real-time system is one of the most difficult tasks to solve;

- **Adaptation and Generalisation**

  Developing action recognition models from controlled contexts to the real world is mainly based on an effort of adaptability and generalisation. The capability of models to find new actions or adapt to changes in game procedures dynamically, for example, if there are rule changes, is still an ongoing issue;

- **Ethical Considerations**

The integration of action recognition technology in public areas and sports arenas raises ethical concerns related to privacy, consent, and the potential for improper use. Achieving a balance between technological advancement and ethical precautions holds immense importance;

- **Integration Complexity**

  The action recognition systems, combining with current sports infrastructures, broadcasting platforms, or training programmes, need well-designed and possibly redesigned technology. This aspect, which posed one of the most significant challenges to address, will be comprehensively explored in **Chapter 5**;

- **Interpretability and Explainability**

  Understanding the logic behind decision-making in complex action recognition models continues to be a challenge. Making sure that these models are capable of providing understandable insights and explanations is especially important in sports, such as Basketball, where practical insights can help in improving strategic choices.

While exploring action recognition, we are presented with the task of overcoming these challenges through innovative approaches, accurate algorithms, and ethics considerations, proving that the integration of technology and sports insight can bring the desired productivity.

2.3.4 Advances in Action Recognition Techniques

The past several years have witnessed significant advancements in action recognition thanks to vision computers's technological growth, ML, and DL. Such innovations have in turn been the basis of the new approach to the monitoring of events and the analysis of the content of video data, which has resulted in increased knowledge and scope in different fields [25].

The application of state-of-the-art algorithms is primarily the turning point that showcase these models' great performance. DL techniques, including CNN and Recurrent Neural Networks (RNN), are the decisive ones. When it comes to basketball and the fact that players do many things, including dribbling, shooting, passing, and defensive movements, CNN would have sealed their accuracy in this action by identifying where they perform each one of them.

Additionally, there has been a great deal of progress in temporal modelling; developments like Long Term Short Memory (LSTM) networks and Temporal Convolutional Networks (TCN) have shown to be highly successful in comprehending complex temporal connections [26]. In the Basketball scenario, these techniques are particularly good at identifying specific patterns of movement, such as a player dribbling quickly and then passing the basketball quickly to a teammate.

As a fundamental element of present-day action recognition, attention mechanisms can help the models to focus on selecting the relevant portions of video frames, increasing their capacity to recognise the most vital activities. On a Basketball court, it means that we notice and define key scenes such as dunks and assists using the accuracy that modern technologies are capable providing.

The combination of temporal and spatial streams by means of a two-stream architecture has provided another platform from which we are able achieve further advancements. It's noted that the movements in Basketball are very dynamic and include many complicated moves, such as dribbling, jumping, passing, shooting, and rebounding. However, computational networks are generally better at understanding the complexities of the fast-paced actions happening in these plays. This is due to their ability to handle the combination of both spatial context and motion information [27].

Transfer learning and pre-training, which make use of information from large datasets, allow the training to be completed more quickly and better. This is mainly based on how the two techniques are inevitably intertwined. This means that a model can quickly adjust to recognise a wide range of actions, from free throws to fouls.
Real-time deployments have also become a reality, thanks to advancements in hardware and algorithmic efficiency. This is especially useful in the context of live broadcasting, as immediate highlights output based on important plays can deliver an immersive spectator experience [28].

Nonetheless, such transitions are not without issues. The main issues that must be investigated range from robustness to changing environmental conditions to ethical considerations related to the privacy of the players. The revolution in action recognition systems marked the emergence of technologies that can interpret and understand human activities from video data. Basketball, characterised by its fast pace and complexity, shows off how these enhancements allowed for a better grasp of the aspects for sports execution, which in turn helps the coaches to be more precise, player development, and fan engagement. Technological advancements, extending innovations, and improvements in action recognition are limitless, implying a general revolution in such sports as Basketball.

2.4 Image and Video Processing for Action Recognition

2.4.1 Pre-processing Techniques for Sports Videos

The methods of pre-production of sports videos are an important factor in knowing action, providing a framework for successful and successful research in sport-related contexts. In the case of sports videos, pre-processing methods play an important part in obtaining advanced data and valuable information, leading to successful results. The same is true with video stabilisation, which is preferred on most sporting occasions where the camera shots can be fast and wildly unpredictable. The stabilised footage with a side view camera

during a Basketball game makes it possible to see the players clearly, especially in their movements and actions, as shown in the image [25].

One of the other important things that we should address is a process of noise cancellation; these are essential for enhancing data quality [29]. Examples of those aspects that may be obscured by the issues brought about by insufficiency/excessive lighting, and other possible environmental aspects. Through the use of noise reduction techniques, it becomes easier to identify distinct movements like dribbling the basketball or taking a shot.

With temporal synchronisation, we are able to align independent video streams on a single timeline, which is essential if we wish to setup multiple cameras. During basketball broadcasts, where various viewpoints capture different aspects of the game at the exact same time, temporal synchronisation ensures a precise alignment of what's happening across multiple viewpoints, which allows for a more detailed analysis in the end. For this particular scenario, only one camera was used to gather the important footage.

Techniques for improving resolution, such as super-resolution, improve video image quality. Higher-definition videos unquestionably provide better details, which are important in recognising specific actions on the court, such as, basketball exchanges by players. Background subtraction and segmentation algorithms separate items in motion from the background, resulting in a foreground mask. This step allows for an improved, focused study of player actions and interactions without interference from static components such as the court, substitutes, coaches, referees, or spectators. Yannick Benezeth et al. (2010) [30], conducted an analysis of various state-of-the-art background subtraction methods, including approaches that range from basic background subtraction with global thresholding, like Basic and MinMax, to more advanced statistical methods like Gaussian Mixture Model (GMM), Kernel Density Estimation (KDE), and Eigen.

Normalisation techniques establish uniform lighting conditions and colour distributions, guaranteeing consistency in the videos. Mona M. Moussa, et al.,2010 [31], suggested a method that they founded on the detection of interest points by using Scale Invariant Feature Transform (SIFT) technique from each video frame. The interest points are selected at the first stage, and then these relevant points are reduced in number; finally, the multi-classified linear Support Vector Machines (SVM) method is used for the classification process. Such factors play a more important role than others, especially when exercising management in these closed games that happen in different arenas that have diverse conditions. Concretely, determining the Regions of Interest (ROI), abstracting the real asset (court or players) areas as the video segments, reducing computational load, and improving overall efficiency [32]

These pre-processing techniques are essential for effective action detection in sports videos and improve the quality, accuracy, and relevance of the raw data by adjusting to the dynamic nature of Basketball, characterised by the diversity of player actions, which con-

tributes greatly to improvement and better performance optimisation.

2.4.2 Feature Extraction in Action Recognition

One of the most challenging but significant tasks in object recognition by computers and AI is the action recognition of images and videos. Its main objective is to create models and systems that are specifically classified for representing and classifying human actions from images and videos. It is necessary to develop algorithms and models that have the ability to automatically interpret human actions and activities. The basic procedures used to help with this ML procedure are the following:

1. **Data Collection** - Access to labelled data such as images or videos of human actions or specific activities of interest. These data sets are important for training and testing action recognition patterns;

2. **Feature Extraction** - The extraction of relevant information from photographs or video images that capture the intrinsic and temporal characteristics of an action. These are then assigned to the event found in the model;

3. **Model Creation** - Action recognition can be modelled and trained using classic ML methods or DL algorithms such as CNN, LSTM or RNN;

4. **Classification** - Recognition of actions in previously unseen images or video using the trained model. The model predicts an action label or class for each frame or sequence;

5. **Evaluation** - Use of appropriate analytical parameters to quantify accuracy, precision, recall, or F1-score when evaluating the performance of a functional recognition model.

The learning and identification of actions from raw images and videos are still an active area of research, and more recently, advances in computer vision and DL came up with a lot of promising results in the topic. Correct action recognition systems are capable of delivering numerous benefits, such as performing video content-based analysis, facilitating human-robot interaction, and determining different types of human behavior.

Identifying specific activities as well as actions performed can be a useful application of the action recognition concept for sports, either during the game or during the session. The method affects the scope of the research and the assessment of the success and training of sports.
Accurate pattern recognition has a wide variety of applications, from improving video content analysis to understanding human behaviour in different situations. On the one hand, the expansion of computer vision, DL, and large labelled datasets continues to improve the accuracy and reliability of these systems, making them a valuable tool in the sports industry.

2.4.3 Motion Analysis in Sports Action Recognition

The extraction and analysis of motion data from sports videos allows for a better understanding of athletic performance. In Basketball, motion analysis plays a primordial role since it gives the team the ability to improve at a whole new level by being able to perceive split-second movements and coordinated plays. Optical flow techniques meticulously trace pixel movement between frames, displaying the object's motion tracks. This detailed evaluation is essential, especially for fast-moving plays like the fast dribbles we see in many Basketball games. Furthermore, this type of analysis also encompasses trajectories while factoring in the complex paths of players and objects over time. Direction, lift, and vector changes are important, which show the position and coordinated play; the latter is particularly important in sports where the accuracy of passing, team dynamics, and the result of the game play a significant role. Laura Sevilla-Lara, et al., 2010 [31] performed an interesting study on this aspect concerning the fusion of optical flow and action recognition, examining the utility of optical flow and identifying the key attributes that render a flow method suitable for action recognition.

Discovering certain basic postural positions is a prerequisite for adept movement depiction. Along with the features that characterise kinematic features, such as velocity and acceleration, they can help detail actions such as walking and running, which, in turn, helps to analyse them on the court. Qili Zeng, et al., 2021 [33], proposed a new technique where they would generate the features of the human motion posture by the combination of topic models and CNN features. Through Basketball gesture recognition as a case study, the research, emphasising human limb performances, scrutinised the motion of the limbs in diverse Basketball actions, including walking, running, jumping, etc.

Implementation of DL-based approaches, specifically CNN for motion detection, has accelerated motion analysis to another level. Modern networks are able to perceive and analyse complex motions on this account, which results in advanced video analysis in sports, resulting in analytics improvement. What makes Basketball complicated is the fact that it involves a lot of sudden changes in all directions; thus, acquiring knowledge about these aspects would give players a chance to improve their skills. Coaches will be able to adapt a more specific approach thanks to the fact that they can analyse the team's motion, which will lead to better team performance. Moreover, the audience may even get an opportunity to experience a deeper level of engagement that can come as part of understanding the context of what's going on on the court [34].

2.4.4 Techniques for Object Detection and Tracking

2.4.4.1 Overview of Detection vs. Tracking

Object Detection and **Object Tracking** are considered the two most popular methods when it comes to data processing and analysis in computer vision. These methods basically involve the same steps of detection and tracking objects, but with different ap-

plications depending on the goals, and they help to solve different problems.

Object detection, in a sense, is focusing on the position and recognition of objects in a frame of a video image. It creates regions of possibilities where it succeeds in classification based on features of players, balls, and objects. Therefore, the players and the objects are distinguished automatically in the game. It helps to track players within the basketball arenas, examine the trajectories of the basketball when they are tossed over the rim, and distinguish different in-game situations. While object tracking involves tracking a specific object, i.e., player movement or basketball tracking, in each frame, in action recognition, we combine player movements.

Modern computer vision and AI systems use action recognition extraction techniques, which are at the forefront of sports analytics, to analyse and understand the players' movements from images and videos. By combining these two, we are capable of capturing player skills, such as their movements, and focus on the most important moments. Image processing techniques are responsible for capturing spatial features, while video analysis techniques deal with temporal dynamics. Likewise, both aspects are necessary to effectively perform action identification. By confronting external factors such as obstructions and other complexities in the game scenario, algorithms can adapt to the scenario. For example, interest point detection is one of the methods that includes feature extraction for Histogram of Oriented Gradients (HOG) and Local Binary Pattern (LBP), as presented in [35], and also frame flow, which is the optical flow used for video.

2.4.4.2 Used Object Detection Method

YOLO [36], which is one of the most employed and well-known approaches, is the one that stood out for us. YOLO is an app that processes data in real time; thus, it provides super-speed and high-accuracy multiple-object detection within the frame of an image or a video. As opposed to traditional object detection methods such as HOG or R-CNN, YOLO serves with a rather different mechanism, partitioning the image into a grid of cells "S × S," where S is the abbreviation of itself in the spatial dimension. Beforehand, it separately confirms the class it supposes the bounding boxes are representing with confidence for the accuracy of its forecasts. The confidence score does so in two ways: the first is providing information about object existence, and the second is the label, which is about how precisely the bounding box circles the object. The final step post Non-Maximum Suppression (NMS) , which follows the ROI generation phase, is the YOLO post-cell processing. K-Nearest Neighbour (KNN) removes the excessive and overlapping bounding boxes by merely retaining the bounding box with the largest confidence value for each object. By using this technique, YOLO will also be able to detect addresses homogenously from left to right in the image. This method stands out not only owing to its near-instant execution rate but also for its generally friendly usage nature.

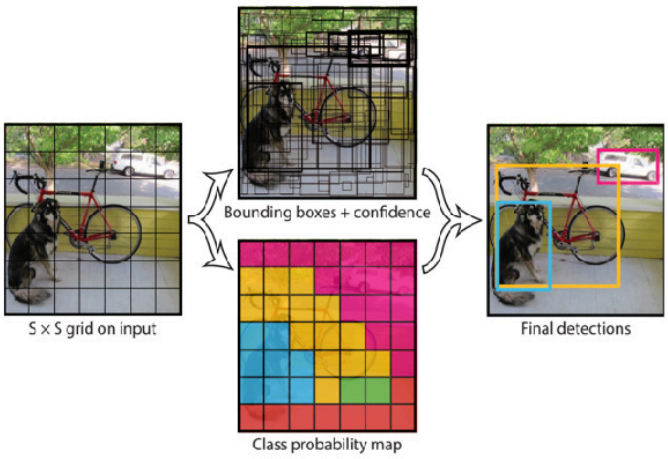


Figure 2.2: The image is divided into S x S grid cells where the bounding boxes are simultaneously predicted with confidence and class probability for a final decision [1].

As depicted in Figure 2.2, one of the key advantages of YOLO is its ability to handle object detection as a regression problem, enabling it to learn to predict bounding box coordinates and class probabilities in a single pass. This approach not only accelerates the detection process but also improves accuracy, making YOLO particularly suitable for this system where both speed and precision are essential.

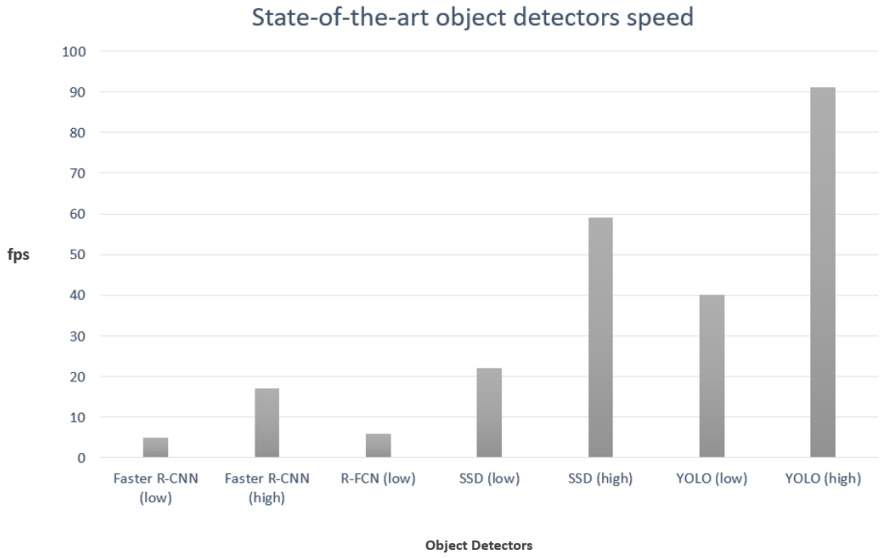


Figure 2.3: YOLO Speed compared to other state-of-the-art object detectors (adapted from [2]).

Our chosen object detector, YOLO, was emphasised in article [2] with the highlighted criterion of the detection speed in frames per second (fps), as we can observe in Figure 2.3. YOLOV3 and YOLOv4 are both available, and each one is another version of the architecture. The last generation of this architecture provided high accuracy and speed. Besides, it proves that it can easily fit in many tasks, such as object tracking, pose detection, and lots more, which are just a few of its functions during the object detection phase. In sport, the YOLO system provides the latest solution to match the different types of sports with the help of better, quicker, and more accurate object detection than what is currently avail-

able. Its important role has accelerated the progress of the development of highly effective tools, elevating the analysis of sports events and changing the way players' movements and interactions are interpreted.

2.4.5 Deep Learning-based Trackers

Sports object recognition often entails the exact and stable tracing of the players as well as the specific elements of the gameplay for the correct determination and examination of particular actions and events that occurred on the playing court. In many cases, ultra-traditional tracking algorithms (like the "Kalman Filter" [37]) tend to fail when facing occlusions, fast movements, and complex interactions that are seen in fast-paced sports events such as Basketball.

Nonetheless, DL tracking has greatly improved the precision of action recognition and motion tracking, thus enabling it in quite intricate environments. These trackers exploit the efficiency of CNN to obtain high-grade feature representations for the video frames. These tracking systems combine object detection and/or tracking together into a coherent process, which makes them capable of sharply processing complex motions and appearance variations. They have the ability to retrieve special features that are suitable for all kinds of scenes and work well under all conditions, which makes them the ideal match for a dynamic sports environment such as Basketball. Regarding action recognition for Basketball, the use of DL-based trackers has proven extremely useful for several key reasons:

- **Robust player tracking:** trackers can properly monitor Basketball players throughout the game, even when they are obscured by opponents, spectators, or the basketball. The excellent tracking guarantees that player motions are correctly captured, providing important spatial and temporal information for action detection;
- **Handling Complex Actions:** Basketball has a variety of complex actions, including dribbling, passing, shooting, and blocking. DL-based trackers can learn to track players and the basketball successfully even during high-speed activities, allowing them to handle these challenging actions;
- **Integration with Action Recognition Models:** DL-based trackers' end-to-end nature allows the integration with action detection algorithms. The trackers' deep features may be immediately fed into the action recognition model, allowing for better and efficient detection of Basketball actions;
- **Adaptation to Changing Scenes:** Basketball games are played in a variety of indoor environments with varying lighting, camera angles, and court layouts. DL trackers can adjust to changing environments by learning flexible feature representations, allowing consistent tracking performance in varying game conditions.

DL-based tracking has yielded a significant reduction in tracking errors and improved the tracking system's adaptability. Approaches such as Siamese Networks [38] and Mask R-CNN [39] appeared to work well in cases involving both object detection and tracking with

a high degree of precision in real-time [40, 41]. These innovations in player tracking or tracking of sports activities keep enhancing the power of analysing the performance level of players in Basketball or any other sports.

Nevertheless, substantial success with the trackers of DL for action recognition in Basketball has quite a few challenges to overcome and concerns to be addressed. For deep trackers, the computation can be too demanding, requiring more resources than are available for real-time tracking in a resource-limited environment. This contributes to the diversity of training data requirements for the models and implies that it is more challenging to collect data for specific categories of action.

2.4.6 Spatio-Temporal Representations in Sports Action Recognition

This field is aimed at exploring sports motion, while combining spatial and temporal analysis studies. Developing algorithms that can extract the patterns of motion and interaction types internalised within sports videos. These, in return, will enable accurate recognition and classification of many sports movements.

In sports action recognition, spatio-temporal representations aim to depict an outline of players and objects on the field as well as the order of sequences of these objects over time as they are undergoing different activities, Figure 2.4 provides an example of this spatio-temporal representations. The use of both spatial and temporal allows for a comprehensive picture of actions or interactions, raising a level of recognition where actions get more elaborate and figurative. In significant cases, the use of spatial-temporal representations is integrated with both 2D and 3D data [42]. We use 2D spatial presentations, like optical flow, where we track the movements of the pixels between subsequent frames. This is how we reveal the motion trajectories of players and objects. Optical flow-based representation is definitely the best format to showcase quick Basketball moves, e.g., dribbling or passing.

Figure 2.4: Example of spatio-temporal action detection.

3D spatial representations ensure that the depth perception is present and facilitate the accurate understanding of the 3D structure of the scene since they contain the depth of the

scene. This particularly concerns sports like Basketball where the elevation of the players, the basketball, and the hoop results in observable actions may become hazy. Temporal representations, however, work to portray the action as it unfolds over a time period. Temporal models like RNN or LSTM models involve the analysis of the sequential movements of the body parts and learning temporal dependencies between frames. By using the time-tracking models, they could completely simulate the movement of the ball, which is crucial in Basketball. The player's movement around the court is also among the factors that will be comprehended while tracking time.

An important improvement is the incorporation of 3D-CNN to the tasks that deal with spatio-temporal representations. Hyunhoon Lee, et al., 2021 [43] tried 3D-CNN for action recognition by showing a lot of advantages of them. It was proven that using various optimisation methods is advantageous in terms of speed without compromising accuracy. These hyper-networks scale traditional 2D-CNN up to process the space and time information entirely; that's why they can see motion patterns and temporal existing dynamics directly from video data. This has greatly improved the accuracy and efficiency of Basketball recognition tasks on sports video grounds.

The knowledge of spatial-temporal representations in sports action recognition provides a significant research area because it is based on the spatial arrangement and the temporal progression of complex actions and interactions in sports videos. By integrating spatial and temporal factors, we are able to create advanced models that are more specific and complex for recognising patterns and actions in sports, thus contributing to improving performance assessment.

2.5 CNN for Action Recognition

2.5.1 An Introduction to CNN: How They Work

Before proceeding with this section on CNN, we need to understand the meaning and the reason for "neural networks." This is an essential discussion on the principles and operations of CNN in DL.
The Neural Networks (NN) framework is the basis of the AI and computer vision theories of CNN; therefore, having a grasp of these concepts is the only way to comprehend them. The name Artificial Neural Network (ANN) comes from the fact that this branch of science has greatly benefited from the studies of different models that try to replicate the brain's computation, which can't be compared with digital computers computing skills [44].

The features of our brain, which are complex computational structures, non-linear, and made of the basic components called neurons, make it feasible to accomplish diverse tasks like perception and recognition. This ability very often goes beyond the processing speed of any existing digital computer, even if it is the fastest one on the market. One example happens when our brain can quickly recognise familiar faces in a crowd, something that

would take a lot more time even with a powerful computer. This raises the question: ”How does the human brain work to achieve such outstanding functionality?” The structure and functionality of our brain is very complex, and based on our memories, we build the rules of behavior. When neurons in a human brain process information, ANN, made up of artificial neurons, replicate this information-processing ability.

Let’s now direct our focus to the central subject of this section: **”What exactly are CNN?”** CNN are a particular class of NN used to great effect for image processing tasks, and they also work well when used for natural language processing. The meaning of DL and AI lies in the ability of NN to adapt to issues raised in quick-moving fields. In contrast with conventional networks consisting of input, hidden, and output layers, CNN does not form any connection between the neurons situated in the layers with different weights. The term ”convolutional” literally refers to the filtering process that is at the heart of a NN architecture, enabling the acceptance of less complex image processing tasks and thus speeding up comprehension.

Like standard NN, CNN are made of layers, but the more advanced layers that give them their unique characteristics are different from other networks. Convolution and pooling layers, that transform incoming data spatially, will be followed by Rectified Linear Unit (ReLU) layer, ensuring non-linearity in the data flowing through the network. The ReLU layer has the unit length dimension, for which the transfer of information is important to maintainin the complexity of the data since it is multidimensional. This mechanism of layers merged into one another provides plenty of capacity to CNN to recognise and understand both textual and visual features, making it a valuable part of modern AI applications. Section 2.5.2 explores these layers of local features that are extracted from CNN.

2.5.2 Architectural Components of CNN

CNN has brought image and video data analysis to a whole new level, and the ways we used to understand Basketball gameplay have changed. The architecture of a CNN has a big role in its capacity to distinguish the most important feature of the complex visual data found in any Basketball game. The convolutional layer is the part of the CNN that provides these features by detecting unique characteristics in the input data. Such layers provide filters that detect patterns, edges, and textures and descend the hierarchical stage. Then, activation functions like ReLU come next to bring in the non-linearity that helps the network capture complex relationships in data.

Pooling layers come next, followed by downsampling feature maps to decrease computational load while preserving essential data. These layers enhance the network’s ability to handle variations in object position and size. On the other hand, fully connected layers establish connections between all neurons from the preceding layer to the subsequent one, contributing to the network’s capability for high-level predictions.

Layers like Batch Normalisation [45] work towards training stability by bringing variety, while elements like Dropout Regularisation [46] mitigate the overfitting problem. Besides that, it becomes able to adopt skip connections for a smoother gradient flow during back-propagation through innovations from ResNet architectures [47]. The latest developments in CNN, like Visual Geometry Group (VGG) and ResNet, have brought a new backend to CNN design. While VGGNet means a deeper network's performance, it doesn't explain the vanishing gradient problem with skip connections like ResNet [48]. They tend to appear in gradients, together with the gradual changes in the networks being trained to become very small. This occurs when the network takes too many steps to establish such information in the initial layers, affecting its ability to conceive correct concepts that might be comprehensive in the data.

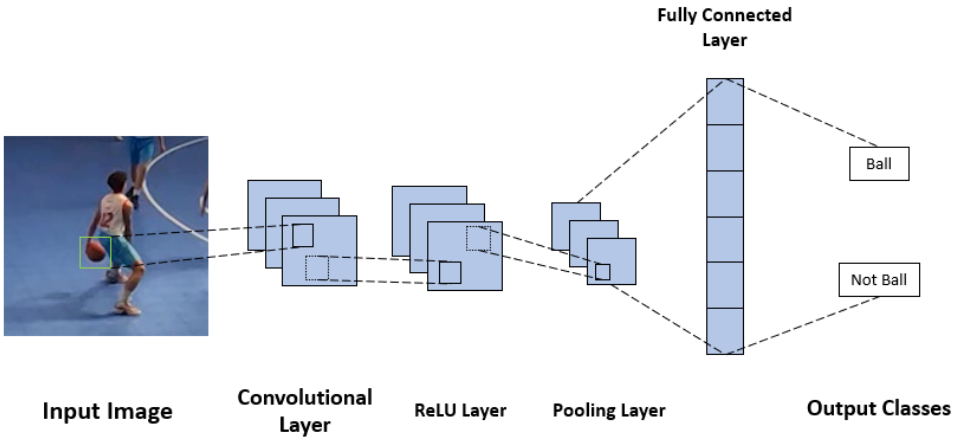


Figure 2.5: An example of 2D CNN architecture for image classifcation.

The fully connected layer is essential due to the fact that it makes the data input recognisable. In other words, this layer gives information about the hidden features in the dataset and defines them through the identified patterns. In contrast, the convolutional layer, being one of the most important layers, computes with a filter across a pixel array that ends up in convoluted feature maps. In the process of pooling, a certain feature map undergoes a reduction so that the number of parameters required to be handled by the network can be limited, speeding up its processing. The result of this process will be a pooled feature map, as shown in Figure 2.5.
When the various architecture-related components are aligned and thoughtfully considered, we can unravel the ability of CNN to use the complex dynamics of Basketball. A properly structured CNN is able to deal with the game's complexity, providing a viewpoint that facilitates the advancement of Basketball analytics and operations.

2.5.3 Training and Transfer Learning for CNN

In the context of Basketball analytics, ML methods like training and transfer learning, when combined with CNN are very helpful tools that integrate the field with the intricacies of the sport.

Training CNN: Training with CNN is not any different from Basketball players who rigorously practice to improve their skills. The network must use an extensive image and video dataset, Basketball related. The network processes and memorises the visual aspects of the game—players' movements, the court configuration, a Basketball trajectory, and many other things that are taking place during play. The network does this through backpropagation, which allows it to make minor changes to the parameters within it, increasing the overall performance and accuracy of the network in conjunction with Basketball images.

Transfer Learning: For this case, transfer learning becomes the key element of the structure that lets the CNN which is used to solve a range of computer problems such as object classification and detection (e.g., ImageNet [49], Coco Dataset [50] or Open Images Dataset [51]) apply its knowledge to Basketball, after which it is fine-tuned on the basketball There are two main advantages: rapid convergence and better results. The network applies the already existing knowledge of image features, which increases the speed of the training process. Moreover, it equips and further strengthens the base, which provides for Basketball-related insights [52].

Apart from the reliability issue, the use of transfer learning also makes sure that the training process is accelerated and, at the same time, the efficiency of the trained system is increased [53]. It enables the classification with the use of universal attributes, taking into account the details about the basketball. The involvement in transfer learning and Basketball-specific training within the CNN framework allowed us to make the proper choice, recognise participants, define actions, and understand the unique aspects of the game.
Moreover, it is valuable to know that transfer learning, despite being efficient in most cases, multiple iterations or formulations are necessary in order to achieve the desired accuracy for a certain task in a single attempt.

2.5.4 Application of CNN in Basketball Action Recognition

The recognition of Basketball actions is very complex due to the fact that the game is extremely dynamic and players perform movements of varying types. CNN was demonstrated to be an extremely useful tool for this process by offering an automated analysis of Basketball games that provides them with valuable details in return. To do this, unsupervised CNN for Basketball action recognition will be investigated and reviewed from different dimensions and uses in the next chapter.

2.5.4.1 CNN Architecture for Action Recognition

CNN have been designed to efficiently manage the process of visual data; therefore, they are used for action recognition tasks that employ images and videos. The architecture of a CNN includes convolutional layers, pooling layers, fully connected layers, and activation

functions, as we explored previously in Section 2.5.2.
As mentioned previously, a typical CNN architecture can consist of several convolution blocks that capture spatial features in successive frames of the video. The algorithm contains multiple levels of pooling that are used to obviate the loss of spatial dimension and get the desired output of feature extraction. It is these edges between layers that join one another and indulge in the task category, subdividing the different Basketball actions.

2.5.4.2 Training with Labelled Data

The CNN technique is based upon DL and can require large sets of labelled training data, which enable the CNN to recognise different Basketball actions with high precision. CNN are trained on a dataset of labelled Basketball videos showing the Basketball players performing a huge number of actions, including dribbling, passing, shooting, some defensive moves, etc. The basic idea is to demonstrate the association of the movements with each of the visual cues during the training phase. Later, this will be used as a cue during the inference phase [54].

2.5.4.3 Transfer Learning and Pre-trained Models

A shortage of labelled Basketball action data has led to the development of transfer learning, which has become a useful tool. The CNN models with transfer learning from large-scale datasets are further trained using Basketball action dataset for a smaller number of labels. The CNN model is pre-trained and is re-trained to become capable of recognising Basketball actions along with the few last layers, giving it the specific dataset for Basketball. Rather than using the data set entirely for training purposes, this method tries to gain similar accuracy and generalisation with less data.

2.5.4.4 Spatial-Temporal Information

A problem in Basketball activity recognition is that it isn't only about the information but also involves dealing with actions changing over time. In this recent approach, CNN architecture, for instance, in the form of 3D-CNN or Two-Stream Networks, has been utilised because of the possible incorporation of spatial and temporal features to build up action recognition networks weakly.
3D-CNN are able to analyse videos as a sequence of frames and use spatial appearances, which can show how each clip is different from another and its characteristic motion patterns, helping to differentiate actions like layups, two-pointers, three-pointers, or passes.

2.5.4.5 Real-Time Action Recognition

In real-time applications, simplified and low-load NN architectures are applied for fast video stream processing. For instance, some specific networks like Single Shot Multibox Detectors (SSD) and YOLO can be used to quickly carry out real-time situation evaluations. Consequently, coaches and analysts can formulate their quick decisions in real-time.

2.5.4.6 Action Localisation

Apart from being able to tell whether a basketball is being dribbled or not, CNN might also point out when such actions take place in a video. One of CNN's outstanding performance areas is the action localisation ability, in which it successfully learns the spatial and temporal features from the video frames, being fundamental for understanding both the action that is happening and where it happens in the video.

2.6 Challenges in Action Recognition and Sports Analysis

2.6.1 Variability in Player Poses and Actions

Players are characterised by having a unique way of playing the game, with actions like running, jumping, dribbling, passing, shooting, and defending. Such actions can happen in different spatial locations, speeds, and directions at any time. The problem exists when players' movements may not always be the same because of factors such as personal style, gameplay, and external conditions.
In the case of action recognition systems, generalising the views and movements to the collection of different poses and actions requires very accurate algorithms that are able to take care of the complexity and variability that the game presents. This may include the development of ML models that are able to learn from a variety of diversified examples in order to comprehend the complexity of different postures and actions. Zdravko Ivankovic et al. 2014 [55] proposed a technique that merges body part detection based on SVM, latent SVM, k-means clustering [56], and spatial transformation to read player poses in Basketball broadcasts accurately. While their algorithm score was quite high, about 82%, it should be mentioned that they did not execute their approach adequately, and it was not possible to view the actions precisely in the case of occlusions.

The effectiveness of evaluating player performance, tactical analysis, and strategic optimisation can be affected by varying player poses and actions. The analysis of players' motions, interactions, and choices becomes too complicated as a result of having several differences.
Achieving this effect involves the creation of complex algorithms that are capable of adjusting to various player styles and reference points posed by them. This is not only because it highlights the importance of having personalised training datasets but also because of the wide range of motions and scenarios.
Ultimately, having knowledge and control over the different player poses and actions is a critical step in identifying valid and authentic action in usually dynamic and intricate sporting situations, such as Basketball.

2.6.2 Handling Occlusions and Crowded Scenes

In Basketball, occlusions and crowded scenes often disturb the smooth transmission of visual data. Successfully addressing these challenges requires advanced approaches that

combine the capabilities of computer vision with the distinctive complexities of the game.

(a) Obstruction case A.

(b) Obstruction case B.

Figure 2.6: Examples in a game of Basketball where players obstruct each other during a play.

Occlusions occur when players or objects come in between the views of others; something or someone may get hidden, and the visual scene becomes uncertain [57]. This kind of behaviour is typical of players during a fast-paced game, like Basketball, which is performed predominantly around the hoop and in crowded areas of the court, as we can observe in Figure 2.6. Here we see the necessity for the development of more advanced object detection methods. Besides just identifying players, these methods also evaluate their respective positions and motion behaviors, which may be interrupted by physical obstacles. Through this, the system can infer the purpose of the players and their actions, even if these are hidden.
The dynamics of Basketball action also introduce **temporal complexity**. Players tend to move fast, changing their location and interaction within a fraction of a second. In the case of crowded scenes, this temporal missmatch can cause confusion. In this case, the notion of temporal consistency gains significance. The system, in this case, does it by analysing the motion and various obstacles such as occlusion and crowded scenes, thus making it more able to adequately track players through these sequences. It guarantees that players, even those who are not immediately in the camera view, can be predicted next through the learning process from patterns.

Crowded scenes are the most notable scenes characterising Basketball games that increasingly make action recognition a real challenge. The ability to distinguish the players and what they do while they are within the playing court needs both spatial and temporal understanding [58]. This is where CNN comes into play. The visual data is processed through the convolutional layers, which evolve spatial features while their architecture captures temporal dynamics. This may allow decoding even in crowded scenes and help the system to indicate who was the player who passed the basketball, who made the shot, who initiated an assist, etc.
The study conducted by Daniel Weinland, et al., 2010 [59] addresses the limitations that appear when confronted with scenarios involving both these challenges, occluded actions, and crowded scenes. Their approach revolves around a two-pronged strategy: local par-

titioning and hierarchical classification of a 3D HOG descriptor. They have successfully employed the dense HOG-based representation enabling local methodologies, and the results have shown the improvement of resilience to occlusions during action recognition models.

2.6.3 Real-time Processing and Inference

These quick reactions often determine the game's winner. That is the case, which makes it possible to have access to the data immediately and to apply it in real time, which gives the coaches, players, and analysts the tools they need right on the spot. Instant and appropriate analysis of the in-game actions is very important, especially focusing on the characteristics of the game, which are very fast, where quick decisions could change the outcome of the game. Real-time processing and inference are a new technical breakthrough. It brings player performance assessment, strategic decision-making, and game-realistic realisations into play. Here we discuss the role played by real-time processing and inference in sports. We focus on two effects of the same on the sport's dynamics and results.

2.6.3.1 Real-Time Action Recognition Challenges

Quickly identifying and comprehending player movement, basketball play, and important actions are relevant skills not only for coaches and players but also for analysts of Basketball as well. This particular stage of taking in, delivering, and completing these actions while intense gameplay is taking place poses various set of challenges. The multilayered Basketball dynamics, the quick overall pace of the game, and the relevance of real-time conclusions mean that we would need powerful and highly efficient data processing methods.

2.6.3.2 Real-Time Decision-Making and Strategy Adaptation

Real-time processing and inference open new possibilities to players for on-court decisive processing. The reaction time of coaches can be significantly improved when they are immediately aware of player actions such as dribbling, passing, or shooting. This enables them to instantly adjust their tactics by directing their players and finding solutions to the possible inconveniences. This instant feedback improves the team's flexibility, and they can have preventive coaching talks that could likely affect the match outcome.

2.6.3.3 Enhanced Player Performance Assessment

Besides coaching decisions, working with real-time technology helps identify on-court actions and perform performance analysis. Consequently, tennis players get invaluable insights that enable them to make real adjustments throughout the course of a match, build on their strengths, and take care of their weaknesses.

2.6.3.4 Revolutionising Basketball Analysis

The integration of real-time processing and inference into Basketball analysis represents a fundamental change in the way the game is understood. This timely recognition of player movements and actions establishes the framework for performing real-time performance metrics, giving players a deep understanding of their gaming experience.
These insights make it possible for the players to adjust to the moment, benefit from their strong points, and make up for their weak points during gameplay. The quick accessibility of data enables analysts to extract practical insights that can be used for post-game analysis, thus enhancing the team's long-term progress.

2.6.4 Scalability and Generalisation to Different Sports

The detection of players' moves and actions has turned into a next-generation technology, creating data that is highly helpful in making decisions and improving strategies. Action recognition, turns out to be one of the most important factors in Basketball - a sport in which fast movements and complex interactions occur all the time while the players are fully engaged during the game.
But it is not only a benefit for a single sport. More than that, the use of action recognition can be applied to various sport. That means a number of things, such as understanding a number of activities that are happening on the Basketball court. It is also worth taking into consideration the idea of generalisation and the way in which the recognition of actions not only has the potential to affect the world of Basketball but potentially the majority of other sports.

2.6.4.1 Scalability in Action Recognition

Scalability is the fact that a recognition system can handle larger and better datasets without losing performance quality. A recognisable system capable of quickly analysing a diverse range of actions, like running, jumping, shooting, or dribbling, in players, teams, and game situations. Since Basketball is full of events and interactions, scalability is fundamental if we wish to have an accurate and reliable recognition to exist for different levels of a game.

2.6.4.2 Generalisation to Different Sports

While action recognition systems are refined for particular sports, the possibility of enhancing this performance for different sports is an interesting matter, which might lead to improved player skills and also gain new insights.
In this case, the generalisation capability is about the ability of the model, which is trained on Basketball data, to work efficiently in another context, such as Soccer. Soccer's expansive field, with unique player movements and distinct gameplay complexity, requires a delicate recalibration of the model's parameters [60]. The problem is correctly interpreting these ideas and understanding how they change and adapt in different places.

The generalisation we need is that we work on the retention of relevant sport-specific knowledge and the adaptation of unique features of the new sport. This comprises adjusting the algorithms, designing NN, and re-tuning the parameters with the goal of bringing the translated model close to the original portrayal of the sport.

Ultimately, it is generalisation that binds sports together, although on a broader level. Insights drawn from the analysis of player actions in a single sport disseminate within the community of sports analysis, strategic thought, and the artistic touch that distinguishes athletic champions.

Through models that can take advantage of the complex scenarios of gameplay as well as transfer the skills gained from one sport to another, we are able to develop intelligent tools that are adaptable and flexible.

2.7 Conclusion

In this exploration of fundamental concepts and related work, we briefly explored the definition of AI and its various types, establishing a foundational understanding of this evolving field. We then focused on "Action Recognition in Indoor Sports," with a specific emphasis on Basketball, recognising its important role in enhancing performance analysis and strategic decision-making.

Through the exploration of some feature extraction techniques, we begin an in-depth analysis to discover the complexities of capturing both spatial and motion information from Basketball footage. This was truly a rewarding experience, providing invaluable insights that have significantly expanded our understanding of how to use these techniques properly for Basketball analysis. Additionally, object detection and tracking methodologies showcased their importance in identifying players and key actions during gameplay.

Our research lies an exploration of the importance of CNN in action recognition. Through an examination of the strengths and weaknesses inherent in NN, we understood their significance in this field. This knowledge allowed us to use their capabilities correctly, amplifying our understanding of how to use their potential optimally. The application of CNN in Basketball action recognition was a game-changer, showcasing the desired performance in recognising certain Basketball actions.

In summary, the exploration of these topics presented the vast opportunities and transformative capabilities of AI. The integration of advanced image and video processing techniques with the formidable power of CNN elevates the analysis and comprehension of sporting events to new possibilities. As we explore the boundaries of AI applications, a promising future surely awaits, with potential to discover new approaches, encourage innovations, and redefine our interactions with sports and various real-world scenarios.

Chapter 3

Technologies and Tools Used

3.1 Chapter Overview

In this chapter, we shall provide an overview of the technologies and tools we shall use in the system development. The chosen technological tools are Google Colab, CVAT, Alpha Pose (for its future development) and the Open Images Dataset. Hardware-wise, we chose the Hero 10 GoPro for the most significant shots. This chapter will discuss these technologies, their function in the system, and how they add to the system's effectiveness. By conducting such evaluation, our aim here will be to demonstrate our willingness to lead in innovation and banking on modern advancements to reach our targets.

In **Section 3.2**, we explored various considerations that emerged during the development of our system.

In **Section 3.3**, we go into the essential tools and technologies that significantly influenced the development of our system, emphasising their crucial role in achieving our Dissertation goals.

In **Section 3.4**, we present the final conclusions regarding the selection of these technologies and tools.

3.2 Object Detection and Identification Considerations

During the development of this system, multiple scenarios had to be taken into consideration:

1. Which model architectures would be employed for object detection methods to identify the players, the basketball, and made baskets on the court?
2. How will we distinguish the players of both teams during the course of the game?
3. Could it be possible for us to track the made baskets efficiently?
4. How should we address challenges related to occlusion?
5. Would the timing of video capture, whether during the day or night, significantly impact the recorded footage given the different lighting conditions?

So with all these questions/considerations at hand, after a long research, we opted to choose the assets present in Section 3.3. These were found to be the most suitable for this Dissertation.

3.3 Technologies and Tools

Taking the scenarios mentioned in Section 3.2 into account, these technologies and tools were applied in the following manner:

- **GoPro Hero 10 -** A high-speed performing action camera, which gives you a compact design. And also the ability to live stream, video and photo capabilities. It is particularly geared for all water and action sports. Is based on custom-oriented features, which gives the opportunity for shooting amazing pictures and video.
- **Darknet -** An open-source neural network framework that is primarily used for training and implementing DNN. Darknet's compatibility with YOLO has rendered it a valuable choice for adopting the YOLO approach as we mentioned in 2.4.4.2.
- **OpenCV -** An open-source computer vision and machine learning software library. It provides a wide range of tools and functions that enable developers and researchers to work with images and videos, performing tasks like image processing, computer vision, and machine learning.
- **CUDA -** Technology that enables developers to leverage the processing power of NVIDIA GPUs for a wide range of high-performance computing tasks, making it an important tool for scientific and computational applications.
- **COCO Dataset -** Dataset contains images from a wide range of everyday scenes and scenarios, capturing diverse objects in complex contexts. It includes images with multiple objects, often in cluttered scenes, and provides annotations for object categories, object bounding boxes, and instance segmentation masks.
- **CNN -** A category of DL model designed for processing and analysing visual data, such as images and videos. CNN are particularly effective at tasks like image recognition, object detection, and image classification.
- **YOLOv3 -** A state-of-the-art object detection model in computer vision known for its speed and accuracy in real-time object detection tasks.
- **Python -** A versatile and high-level programming language used for a wide range of applications, this programming language was employed consistently throughout the entire system development. Python 3.10.0 was used.
- **VSCode -** A free source-code editor developed by Microsoft. It provided a lightweight yet powerful platform for our Python coding.

- **DeepSort -** An object tracking algorithm that integrates DL for object detection with data association techniques. It excelled in tracking and associating the multiple objects across the frames in our videos, making it valuable for our layer tracking system.
- **TensorFlow -** An open-source machine learning framework. It provided a comprehensive set of tools for building and deploying our machine learning system.
- **Github -** A web-based platform primarily designed as a repository for software development projects. We used it for guidance and cloning key repositories that played a foundational role in shaping our system. Notably, repositories like DeepSort and the Colab notebook were instrumental in shaping and training our custom models.
- **Google Drive -** A cloud-based file storage and synchronisation service developed by Google. It allowed us to store files securely in the cloud, making them accessible in our Google Colab.
- **LateX -** This typesetting system, based on the TeX typesetting language, provided a efficient and flexible way to format the Dissertation report.
- **Google Colab -** A Cloud-based platform for writing and running Python code. It's widely used for tasks like data analysis and machine learning due to its free GPU access and integration with Google services such as Google Drive.
- **CVAT -** Software used to label and annotate the images, for our custom model.

The integration of these essential tools played a fundamental role in the development of the system. These tools not only made coding easier, allowed for effective machine learning model training, and systematic system documentation, but also ensured a streamlined and efficient development process.

3.4 Conclusion

The development process of our system has been possible due to these innovative technologies and powerful tools. The careful choice of such technologies was due to the abundant documentation available that allowed the successful solving of the issues found and good progress. Technologies such as Google Colab, CVAT, COCO Dataset, and the GoPro Hero 10 contributed uniquely to the Dissertation's success.
All these technologies are integrated and have led to the creation of an operable system in the process of this study. The system, with the help of computer vision and DL technology, aims to revolutionise action recognition in Basketball and finally extend to other sports and real-world applications.
Obtaining this knowledge demonstrates the considerable effect of AI, computer vision, and future technology on sport development and the way humans and machines interact.

Chapter 4

Methodology

4.1 Chapter Overview

The ability to interpret the actions being taken in a video is crucial for autonomous awareness in sports activities. Spatial-temporal and action recognition are the two main fields currently being studied in this context. general recognition framework can be broken down into three major steps: feature extraction, dictionary learning from previous video events, and finally classification.
Currently, the level of science and technology applied in official sports games and training is deemed to be relatively low. Players, nowadays, are only capable of mastering the practical essentials through repetition (muscle memory). With current intelligent monitoring, advanced AI regarding human and computer interactions, autonomous tagging, medical diagnosis, and many other applications, the studies that cover most Human Action Recognition (HAR) possess a vast potential for teams looking to improve their performance exponentially. There are numerous factors to consider in sports, such as the variety of movement actions, dynamic and static camera motion angles, and other factors. All of these aspects make it harder to track down and recognise movement. Current research is mostly based on skeleton sequence methods; the studies conducted by Kaiyuan Liu et al., 2022 [61], Danilo Avola et al., 2022 [62], and Jian Liu et al., 2020 [63] provide valuable insights regarding the extraction of motion aspects. Although it sounds like a valid solution to the subject's intent, there are a few restrictions and problems regarding traditional motion recognition, such as different court layouts (depending on the sport), lighting, costs, and many other aspects [64].

As we already mentioned previously in this Dissertation, the subject of action recognition is widely used in many fields that can benefit from its use. An application that's developed for video data analysis will be able to identify the motion in the video through the computer once we have the video motion data, and consequently, we will be able to extract behavior from the objects present in the video. With the current development of advanced depth cameras and sensors, the extraction of data regarding motion recognition has become significantly more efficient. The detection and tracking down players in sport videos, analysing their behaviour and their specific movements, is a crucial stage of analysing this sort of video. With this, professionals can acquire the important data they need to improve.

Current technologies are still unable to capture real-time requirements due to slow recognition speed processing and the enormous amount of features that need to be taken into

consideration. Hence, this Dissertation proposes a video action recognition method for indoor sports based on ML, which will use computer vision to analyse game footage. The current aim is to focus on Basketball games, although in the future it may also be applied to other sports with a similar setup involving players and a ball [65]. Our initial purpose is to design a system that is as low-budget and practical as possible with a view to making it usable in real-world situations and also more widespread by being low-media-cost. Also, the system's efficiency ensured that the primary tasks of the system were precisely performed, available resources were well utilised for higher outputs, and performance improved while remaining affordable. This extensively characterised the need for affordable yet reliable options, making this system usable in various sectors.

Section 4.2, details the systematic approach and practical execution of our proposed solution.

Section 4.3, explores the strategies and processes employed to gather relevant data for our study.

Section 4.4, explores the process of constructing and refining our tracking system, indicating each step and decision made along the way.

Section 4.5, introduces and elaborates on the main Python file for our system.

Section 4.6, presents the Graphical User Interface (GUI) that serves as the primary interaction point for users.

Section 4.7, addresses the moral and responsible aspects of our work, acknowledging the importance of ethical guidelines for the development and deployment of the tracking system.

4.2 Methodology and Implementation

4.2.1 Data Collection

As previously stated, all the analysed data was collected during the Basketball season from the **CDC Under-16 Male** team. All footage used to create the dataset was captured using a **GoPro Hero 10** action camera. The camera specifications were as follows:

- **Camera Model**: GoPro Hero 10 Black;
- **Video Resolution**: Ultra High Definition (UHD), 4K;
- **Frame Rate**: 120 fps;
- **Capturing Mode**: Superview.

Although the GoPro possessed other resolutions and various capturing modes, after running some tests, the **UHD** resolution at **120 fps** with **Superview mode activated** (94.4º Vertical, 122.6º Horizontal FOV) stood out as the optimal choice for the unique requirement of capturing the entire court while simultaneously conserving space within the computer's storage, the videos were then stored in .mp4 format with the H. 265 High Efficiency Video Coding (HEVC) video codecs. This resolution, frame, and capturing mode rate provided an amazing degree of clarity in the gathered footage.

Throughout the recording process, a series of filming spots were tested in an effort to scout for the perfect and most stable places to mount the cameras on in order to get the most optimal field of view and framing, to effectively capture the entire Basketball court without any obstructions. The center location in the final row of spectators was ultimately chosen as the best choice after thorough examination of numerous prospective camera locations to obtain the desired videos, although we encountered certain challenges at a later stage.

During the whole film capturing phase, the same position was continually used, maintaining the best and most useful vantage point for recording purposes. Before each recording session, the camera's settings (resolution, frame rate, and exposure) were correctly adjusted to maintain consistency and precision. To guarantee accurate and high-quality footage, lighting settings and environmental elements were also taken into account throughout the data gathering phase. The footage was deliberately recorded during daylight hours, as a series of tests concluded that the court's nighttime lighting system had a significant negative impact on the overall image quality [24].
Once we had the footage, we needed no obtain each individual frame so that we could annotate the relevant classes for our custom object detector for the "basketball" and "made baskets".
With the following Python script we were able to obtain each individual frame from the collected videos to build our image dataset:

```
import cv2
import os

# Replace with our video file path
video_path = ''

output_dir = 'MBRun'
os.makedirs(output_dir, exist_ok=True)

cap = cv2.VideoCapture(video_path)

if not cap.isOpened():
    print("Error")
    exit(1)

frame_count = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
```

```
17
18 # Read and save all frames
19 frame_index = 0
20 while True:
21     ret, frame = cap.read()
22     if not ret:
23         break
24
25     frame_filename = os.path.join(output_dir, f"frame_{frame_index:04d}.jpg")
26     cv2.imwrite(frame_filename, frame)
27
28     frame_index += 1
29
30 cap.release()
31
32 print(f"Saved to '{output_dir}'")
```

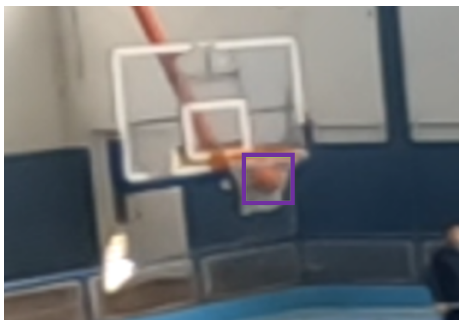
Listing 4.1: Video frame slicing.

4.2.2 Feature Extraction

Initially, LabelImg [66] was used for image annotation, however, this option eventually turned out to be relatively slow. Consequently, Computer Vision Annotation Tool (CVAT) [67] emerged as a valuable alternative, proving to be significantly faster and more efficient in the task of image annotation. All video frames were put through a thorough analysis using CVAT in order to extract features. The ability to annotate and extract important visual objects from footage such as photos or films makes CVAT a well-known and adaptable platform. The first goal was to properly identify basic essential components, namely the basketball and the made baskets, with bounding boxes.

In order to retain the greatest degree of accuracy in the identified features, these annotations where performed frame-by-frame with the utmost care and attention to detail, these annotations can be observed in Figure 4.1.

(a) Basketball.



(b) Made-basket.

Figure 4.1: Features extracted with the help of CVAT tool.

Subsequently, after the manual annotation process, the pertinent features, including the

precise object coordinates and corresponding class labels (i.e., basketball and made-basket), were successfully extracted and saved in the YOLO format, enabling effective and consistent data representation for future use. A total of **4550 frames** were annotated.

4.2.3 Developing a Model for Ball and Player Tracking

The problem of creating our own YOLOv3 object detector was quite considerable, but in Google Colab there were enough tools that allowed to do it even rather quickly. In this section, we will go into detail regarding our custom YOLOv3 model [68] for tracking Basketball movement. Furthermore, we will be exploring the free resources of Graphics Processing Unit (GPU) provided by Google Colab and the integration of Google Drive with it for data storage.

4.2.3.1 Google Colab and GPU Acceleration

Google Colab is a free cloud-based Jupyter notebook environment that provides instant access to powerful GPU resources. This applied especially to the learning of very DNN, which often consume a lot of computational resources. To set up Google Colab for YOLOv3 model development, we:

- Created a Colab notebook.
- Enabled GPU acceleration through Google Colab's settings to begin training.

4.2.3.2 Data Preparation and Storage

To train our custom YOLOv3 object detector, we first needed labelled data containing annotated bounding boxes of the basketball and the made baskets, this step was performed using CVAT as mentioned in Section 4.2.2. This data was then stored on Google Drive, providing easy access and organisation. The data preparation involved:

- Uploading the labelled dataset to Google Drive.
- Structuring the data according to the Darknet format, which includes image files and corresponding text files containing object annotations.

4.2.3.3 Darknet Framework

Darknet is an open-source neural network framework designed specifically for DL tasks like object detection. We used Darknet for creating and training our custom YOLOv3 model. The key steps involved:

- Cloning the Darknet Repository into our Google Colab environment.
- Compiling Darknet with CUDA support to take full advantage of GPU acceleration.
- Customising the YOLOv3 configuration file (yolov3_custom.cfg) to define the network architecture, anchors, and other hyperparameters.

4.2.3.4 Pre-trained Convolutional Layer Weights

Training a YOLOv3 model from scratch can be computationally demanding and time-consuming. To accelerate the training process and enhance accuracy, we used pre-trained convolutional layer weights, specifically those from the darknet53 model. These weights were downloaded from the official Darknet website and where used as a starting point for our custom object detector.

4.2.3.5 Training Process

After preparing the data, configuring Darknet, and loading pre-trained weights, we initiated the training process:

- Specified the path to the custom YOLOv3 configuration file and the pre-trained weights.
- Iteratively adjusted hyperparameters as needed to optimise the model's accuracy and speed.

4.2.4 Custom Model Workflow Overview

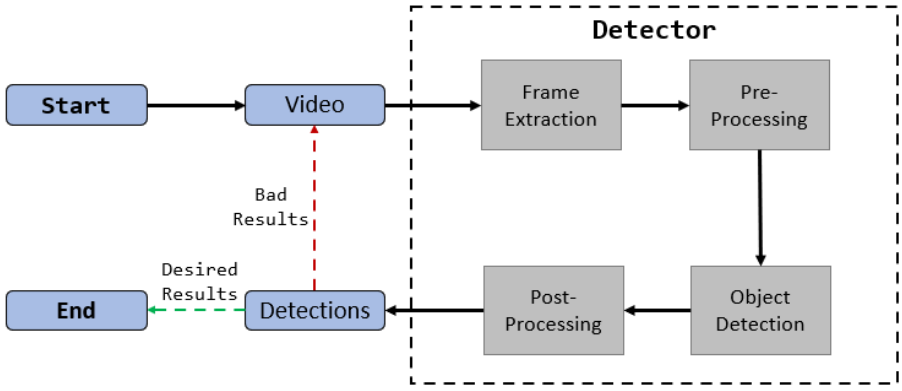


Figure 4.2: Diagram describing the method to detect the basketball and made baskets with our method.

This flow diagram, Figure 4.2, illustrates the key steps and components of the object detection process. Firstly, we did this process by gathering all the videos and retrieving their separate images, which were then inserted in a Google Drive folder. The name for the image retrieval process was called **frame extraction**.

Before the frames were handed to the detector, **the pre-processing** was performed. These included resising, normalisation, and other transformations done in the custom Yolov3.cfg file to make the frames ready for detection.
The object detection process was performed using YOLOV3 for**image detecting**. YOLOV3 algorithm based implementation in every frame of the video allowed for the identification of the bounding boxes and class labels specific to the objects in our dataset. YOLO's object

detection was implemented using Darknet in CUDA acceleration on the GPU.

YOLO detections were then followed up by **post-processing** to further improve accuracy. This, therefore, consisted of iterative threshold value trials to filter out low-confidence pulls; in the end, a threshold of 0.9 was selected as the best fit.

The final output was a video with annotations pointing at the objects ”basketball” and ”made baskets” that were highlighted. If the detection results were as expected, we would end the the analysis and then focus on refining some parameters on the code. If the results were not deemed satisfactory, we needed to take additional videos and expand our dataset to continue training our model until satisfactory results were achieved, and this was exactly what happened.

4.2.5 Training and Transfer Learning

Although traditional ML provides rather good results when identifying relevant use cases, this approach is lacking in some real cases. In a perfect ML scenario, there should be a large number of labelled training data copies that are identical with their testing data distribution. Nevertheless, combining large quantities of labelled training data with actual data can be too expensive, time-consuming, or even impossible in most situations.

Transfer learning is a promising approach to solving this problem, with the idea of transferring information from one domain to another.

Here, the ML technique relies on the fact of using the model with pre-training for the base of the new one. For sports activity recognition, the model can be applied and then modified for analysing the movements of an athlete; on the other hand, the large number of pictures similar to the specific action needed in training a model can hardly be collected; however, fine-tuning a ready model is much easier.

Transfer learning is a concept that is very easy to understand; a conditioned **ML !** (**ML !**)model can be accepted to classify images of a particular sports category, for example, those that are running, jumping, or shooting a basketball. Hence, the machine can perform well, and the results can be more accurate.

An example of this is in our own scenario, where we moved the knowledge from the source domain (Basketball tracking pre-trained model) to the target domain (Basketball action recognition). With this technique, the model learned how to respond more optimally to the situations that occurred on the Basketball court.

4.3 Data Aquisition

4.3.1 Weight Files Generation using Darknet53 Training for the Basketball and Made Baskets

The use of Google Colab's GPU support made it an excellent choice for this task, ensuring efficient model training. We will present the code snippets as we guide you through each step in custom object detection training.

The following cells clone darknet from AlexeyAB's Repository [69], we need to adjust the "Makefile" to enable OPENCV and GPU for darknet and then build darknet.

```
!git clone https://github.com/AlexeyAB/darknet
```

Listing 4.2: Clone darknet repository.

```
!sed -i 's/OPENCV=0/OPENCV=1/' Makefile
!sed -i 's/GPU=0/GPU=1/' Makefile
!sed -i 's/CUDNN=0/CUDNN=1/' Makefile
```

Listing 4.3: Change makefile to have GPU and OPENCV enabled.

```
!/usr/local/cuda/bin/nvcc --version
```

Listing 4.4: Verify CUDA.

```
!make
```

Listing 4.5: Make darknet (build).

YOLOv3 has previously undergone training using the COCO dataset. We will acquire these pre-trained weights to enable the execution of YOLOv3 on these classes and obtain detections. These weights were only used to study the detections and provide a better understading for future analysis, they were not directly used in our system.

```
!wget https://pjreddie.com/media/files/yolov3.weights
```

Listing 4.6: Get YOLOv3 pretrained coco dataset weights.

After completing the previous step, the next step was to facilitate access and use Google Drive files. This was achieved by running the following cell, which mounts our Google Drive into the cloud Virtual Machine (VM). We create a symbolic link between "/content/gdrive/My Drive/" and "/mydrive." This additional step proves valuable, as it addresses potential issues that might occur from spaces in the "My Drive" folder path when executing specific commands.

```
from google.colab import drive
drive.mount('/content/gdrive')
```

Listing 4.7: Mounting Google Drive.

```
!ln -s /content/gdrive/My\ Drive/ /mydrive
!ls /mydrive
```

Listing 4.8: Creating a symbolic link so that now the path /content/gdrive/My Drive/ is equal to /mydrive.

After the successful mounting of Google Drive in our Google Colab notebook, we created our custom object detector all within the cloud environment. If we wish to create a custom YOLOv3 detector, we will need the following:

- **Labelled Custom Dataset**;
- **Custom .cfg file**;
- **obj.data and obj.names files**;
- **train.txt file**.

We have named our labelled custom dataset "obj_train_data.zip." This file contains all the labelled images alongside their corresponding .txt files, each sharing the same nomenclature as seen in Figure 4.3.

frame_0001.jpg	501 591	499 963	JPG File
frame_0001.txt	38	33	Text Document
frame_0002.jpg	483 475	481 673	JPG File
frame_0002.txt	38	34	Text Document
frame_0003.jpg	478 043	476 445	JPG File
frame_0003.txt	38	34	Text Document

Figure 4.3: The "obj_train_data.zip" should be structured as follows.

After completing this step, We will copy the .zip to Google Colab and unzip it to our Cloud VM.

```
!ls /mydrive/yolo
```

Listing 4.9: This is where our zip was stored (we created a YOLOv3 folder where we will get the required files from.

```
!cp /mydrive/yolo/obj_train_data.zip ../
```

Listing 4.10: We copied the .zip file into the root directory of cloud VM.

```
!unzip ../obj_train_data.zip -d data/
```

Listing 4.11: We unzip the zip file and its contents should now be in /darknet/data/ball in the Colab notebook folders.

The next step is to look into our configuration file in the .cfg file. We require editing this file accordingly to our requirements; this process is based on our object detector. AlexeyAB's Repository provided some insight on how to edit the .cfg file:AlexeyAB's Repository provided some insight on how to edit the .cfg file:

- Recommend having **batch = 64** and **subdivisions = 16** for a better performance. In case of running into any sort of issues, the subdivisions to should be increased to 32 or 64;
- Classes will be equal to the number of classes being used, in our case we used two classes, ”basketball” and ”made baskets”;
- The **”max_batches”** will be equal to our number of **classes * 2000**, in our case it will be 4000;
- The **”steps”** will be equal to **max_batches * 0.8** , **max_batches * 0.9**;
- In **[yolo]** sections change ”classes” to our number of classes, and in the **[convolutional]** sections above change to the number of classes + 5 * 3, in our case **21**.

After these edits were made we had our ”yolov3_custom.cfg” ready to be uploaded into Google Colab.

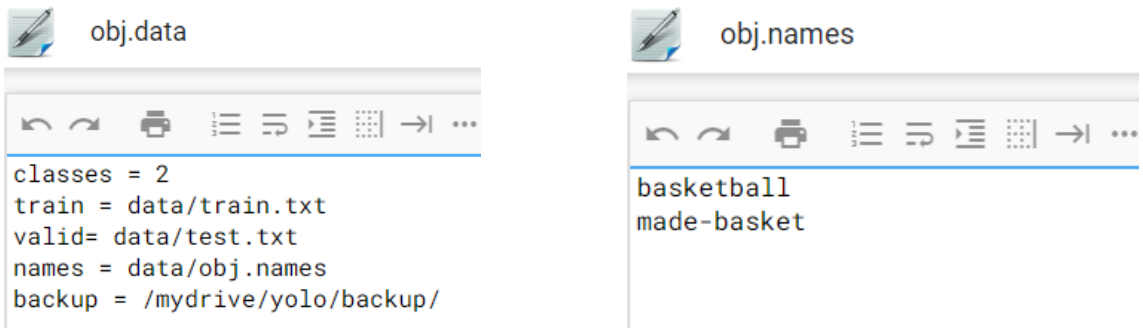
```
!cp /mydrive/yolo/yolov3_custom.cfg ./cfg
```

Listing 4.12: Uploading the custom .cfg back to cloud VM from Google Drive.

We would now generate two files with the help of a text editor (Sublime Text3 was used): ”obj.data” and ”obj.names,” as we can observe in Figure 4.4 The ”obj.names” folder will include the classes which are present within our custom dataset.

```
!cp /mydrive/yolo/obj.data    ./data
!cp /mydrive/yolo/obj.names   ./data
!cp /mydrive/yolo/train.txt   ./data
```

Listing 4.13: Uploading the obj.names, obj.data files and train.txt to cloud VM from Google Drive.



(a) Obj.data file.

(b) Obj.names file.

Figure 4.4: The obj.names and obj.data files.

It’s important to consider the destination for our data. Since we were using Google Drive, the files were being stored in a backup folder within the drive.

To streamline this process, we required a ”train.txt” file. Within this file, we would list all the paths to the images located in the ”obj_train_data” folder. Each line in the file should

follow the format ”data/obj_train_data/(image_name).jpg.” Considering the large number of images, manually adding each line individually would have been a time-consuming task, so we created a small script that would generate this file for us.

```
import os

image_files = []
os.chdir(os.path.join("data", "ball"))
for filename in os.listdir(os.getcwd()):
    if filename.endswith(".jpg"):
        image_files.append("data/ball/" + filename)
os.chdir("..")
with open("train.txt", "w") as outfile:
    for image in image_files:
        outfile.write(image)
        outfile.write("\n")
    outfile.close()
os.chdir("..")
```

Listing 4.14: Creating the generate_train.py script.

We uploaded the Python script from our drive and executed it.

```
!cp /mydrive/yolo/generate_train.py ./
```

Listing 4.15: Uploading the generate$_train.pyscripttocloudVMfromGoogleDrive.$

Having all the files ready to begin our training we downloaded the weights for YOLOv3's convolutional layers, **darknet53**. These pre-trained weights enabled the beginning of the process and enhanced accuracy, as discussed in Section 4.2.3.4.

```
!wget http://pjreddie.com/media/files/darknet53.conv.74
```

Listing 4.16: Uploading pretrained convolutional layer weights.

Once everything was ready, we could finally start training our custom object detector.

```
!./darknet detector train data/obj.data cfg/yolov3_custom.cfg darknet53
    .conv.74 -dont_show
```

Listing 4.17: Training our custom detector.

At a certain point, the training process halted because Google Colab stopped its processing due to an inactivity alert. To prevent the need for restarting the entire training process, the following code was implemented so that we could resume from the last saved checkpoint.

```
!./darknet detector train data/obk.data cfg/yolov3_custom.cfg /mydrive/
    yolo/backup/yolov3_custom_last.weights -dont_show
```

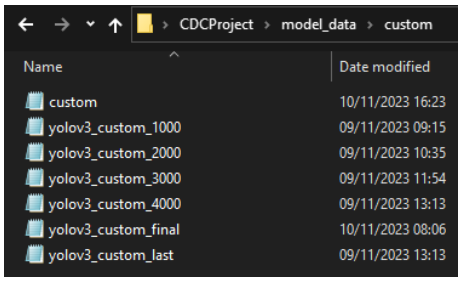
Listing 4.18: Continuing from where we left off.

The trained model weights were stored through Google Drive on ”yolo/backup” path. These weights represent our own custom YOLOv3 object detector trained for tracking the

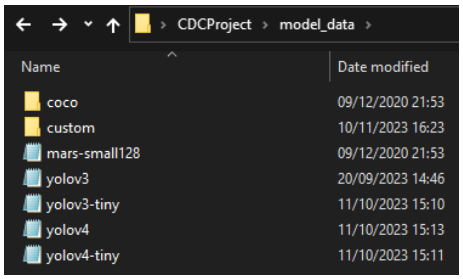
basketball and made baskets in our videos.
We were able to save time by using the pre-trained weights and running the personal tuning process of the YOLOv3 model on Google Colab's free GPU resources. We were able to obtain good accuracy in the tracking of basketballs and made baskets. The notebook used to train our model can be accessed **here**.

4.3.2 Weight Files for the Players

To track players, we considered using either the extensive "Open Images Dataset" [51] or the "COCO Dataset," both large-scale datasets, proceeding to train the acquired dataset from scratch. However, we recognised that this approach would require a bit more time to accomplish, so we opted to employ the default YOLOv3 and YOLOv4 weight files comprehensively [36] and later specified the "Person" class during code implementation in our main Python file. All weight files were located in the "model_data" folder. The weights for "basketball" and "made baskets" were stored in the custom folder, while those for "Person" were stored directly in the 'model_data' folder as we can see in Figure 4.5.



(a) Custom folder with the custom weights.



(b) Model_data folder with the YOLO weights.

Figure 4.5: The "model_data" folder holds YOLO weight files, and the "custom" folder contains our custom YOLO trained weights.

In the "coco" folder, we have the "coco.NAMES" file containing all the 80 classes present in the COCO dataset. Later, we will specifically focus on the 'Person' class.

4.4 Developing our Tracking System

4.4.1 DeepSORT

In order to track the players we found DeepSORT [70], an advanced algorithm for object tracking, to be useful. During some research we discovered that this tool addresses a significant challenge for our tracking system, specifically handling occlusions [71].
We started by cloning the repository using the Command Prompt:

```
1  git clone https://github.com/nwojke/deep_sort.git
```

After its installation, within the deep_sort folder we had the following tracking code Python files:

- **detection.py**: Detection base class.

- **kalman_filter.py**: A Kalman filter implementation and concrete parametrisation for image space filtering.
- **linear_assignment.py**: This module contains code for minimum cost matching and the matching cascade.
- **iou_matching.py**: This module contains the Intersection over Union (IOU) matching metric.
- **nn_matching.py**: A module for a nearest neighbor matching metric.
- **track.py**: The track class contains single-target track data such as Kalman state, number of hits, misses, hit streak, associated feature vectors, etc.
- **tracker.py**: This is the multi-target tracker class.

4.4.1.1 Team Identification Capability

To distiguish both teams we used **openCV** to extract the ratio of light pixels vs total pixels in each players bounding box. By doing a running average of this ratio we were able to identify the team the player is on this. The following Python file was added to the deep_sort folder as "**color_detect.py**".

```
import cv2
import numpy as np

def find_color(roi, threshold=0.0):

  roi_hsv = cv2.cvtColor(roi, cv2.COLOR_RGB2HSV)
  # set a min and max for team colors
  COLOR_MIN = np.array([224, 98, 37])   #Opposing Team Color
  COLOR_MAX = np.array([226, 28, 93]) #CDC Team Color

  # dark teams will remain with this mask
  mask = cv2.inRange(roi_hsv, COLOR_MIN, COLOR_MAX)
  res = cv2.bitwise_and(roi,roi, mask= mask)

  # dark teams should have a higher ratio
  tot_pix = roi.any(axis=-1).sum()
  color_pix = res.any(axis=-1).sum()
  ratio = color_pix/tot_pix

  return(ratio)
```

Listing 4.19: Colour detection.

This script can be found within the "color_detection.py" subdirectory of the DeepSORT folder. The HSV colour codes within the variables "**COLOR_MIN**" and "**COLOR_MAX**" were acquired through the web tool "GetImageColor". These values should be adjusted

accordingly based on the team colours. With this step completed, DeepSORT was successfully configured for further development.

4.4.2 YOLOv3 Folder

The second step was to create a Yolov3 folder containing all the necessary .py files for our YOLO object detection. The folder can be seen in Figure 4.6.

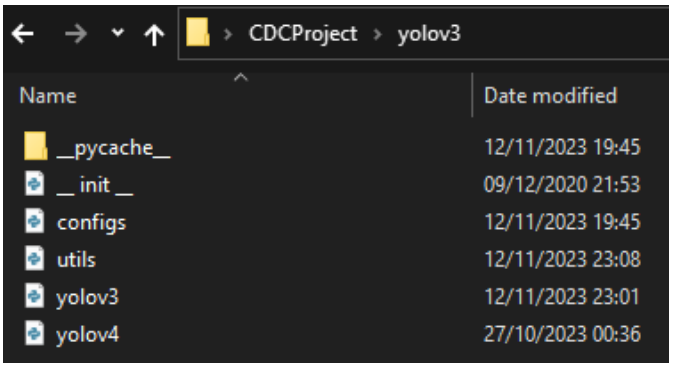


Figure 4.6: Our YOLOv3 folder.

We created a **config.py** file containing all the configuration parameters for training and testing our YOLOV3 object detection model. This file was fundamental for setting up and configuring the YOLO model for both training and testing phases, here we specified our model type, paths to our weights and datasets, training parameters, and other important aspects.
The remaining folders, namely yolov3.py, yolov4.py, and utils.py, were installed by running the following command in the Command Prompt:

```
python3 -m pip install yolov3-tf
```

```
python3 -m pip install yolov4
```

These files where then adapted to handle several aspects related to loading the YOLO model, processing the images, and performing object detection on our videos. Once we had these two tools installed, we were ready to initiate the development of our main Python file, which we named "**object_tracker.py**".

4.5 Object_tracker.py

Within our "CDCProject" folder, "object_tracker.py" assumes an important role as the primary file for analysis the videos. This Python script stands at the forefront, overseeing the complex process of tracking and detecting the objects present in the video sequences. From YOLO-based object detection to DeepSORT-based tracking, this script is employing complex algorithms, ensuring a precise and in-depth understanding of the visual information being analysed.

4.5.1 Installing Requirements.txt

To achieve compatibility, we had to integrate these dependencies. This step took quite a while because we had to make sure that there were no inconsistency issues while installing the different dependencies, and furthermore, although some of these prerequisites were installed without any issues, another critical module called TensorFlow caused some troubles during this installation process. A conflict in version compatibility in the requirements.txt file prevented the setup from completing without any issues.

Still, for an end user, these obstacles could be some sort of stumbling block. The most specific and complex points, primarily those interconnected in ML libraries, should definitely be handled with care concerning version control. Being able to make these packages work with the Python environment is as essential as it is complex. Users who are new should repeatedly pay attention to the versions of the libraries and check for regular updates. The libraries maintain the most up-to-date functionality information and instructions in their official documentation. On the contrary, aiming to institute a high-tech system, we meant to install some difficulties in doing the installation. Proper documentation, community support, and awareness of subsequent software releases are the basis for a better experience.
The dependencies listed in our **requirements.txt** were as follows:

- **numpy**>=1.18.5,<1.22.0
- **scipy**>=1.7.0
- **wget**>=3.2
- **seaborn**>=0.11.2
- **tensorflow**>=2.6.0,<2.7.0
- **opencv-python-headless**==4.5.3.56
- **tqdm**==4.62.3
- **pandas**>=1.3.3
- **awscli**>=1.20.78
- **urllib3**>=1.26.7
- **mss**>=6.2.0

To set up the dependencies listed in requirements.txt, we executed the following command in the Command Prompt.

```
pip install -r requirements.txt
```

4.5.2 Object_tracker.py Code

To begin our coding process for the main file, we had to first import essential packages, classes, modules, and other significant elements.

```
import os
os.environ['CUDA_VISIBLE_DEVICES'] = '0'
import cv2
import subprocess
import numpy as np
import tensorflow as tf
from yolov3.utils import Load_Yolo_model, image_preprocess, postprocess_boxes,
    nms, draw_bbox, read_class_names
from yolov3.configs import *
import time
from keras.datasets import mnist
from deep_sort import nn_matching
from deep_sort.detection import Detection
from deep_sort.tracker import Tracker
from deep_sort import generate_detections as gdet
from deep_sort.color_detect import find_color
from scipy import stats
```

Listing 4.20: Imports for object_tracker.py.

We wrote the main function of our code with the necessary parameters to perform object tracking on our videos, saving the results to the specified output path.

```
# The main function for analysing our videos
def Object_tracking(Yolo, video_path, output_path, input_size=416, show=False,
    CLASSES=YOLO_COCO_CLASSES, score_threshold=0.3, iou_threshold=0.45,
    rectangle_colors='', Track_only = [], custom_yolo=None, custom_classes=
    YOLO_CUSTOM_CLASSES, Custom_track_only=[]):
    # Definition of the parameters
    max_cosine_distance = 0.7
    nn_budget = None
```

Listing 4.21: Object_tracking function in object_tracker.py file.

The following code, represent our custom block of the main code, where we initialise the files within deep_sort folder using the pre-trained model (mars-small128.pb) for object tracking. The code keeps track of possession and made baskets, with certain conditions and cooldown periods applied after a made basket before searching for the next one, guaranteeing accurate made basket tracking.

```
    #initialise deep_sort object
    model_filename = 'model_data/mars-small128.pb'
    encoder = gdet.create_box_encoder(model_filename, batch_size=1)
    metric = nn_matching.NearestNeighborDistanceMetric("cosine",
    max_cosine_distance, nn_budget)
    tracker = Tracker(metric)

```

```
    times, times_2 = [], []
```

Listing 4.22: Custom block in object_tracker.py file.

This code will check if a video path was provided. If yes, it will capture frames from the video, if no path was provided, it will capture frames from the webcam. This allows the flexibility to either test the object detection on pre-recorded videos or in real-time using the webcam.

```
    if video_path:
        vid = cv2.VideoCapture(video_path) # detect on video
    else:
        vid = cv2.VideoCapture(0) # detect from webcam
```

Listing 4.23: Analysis of a video or analysis straight from the webcam.

This block retrieves information about the video frames, such as width, height, and frames per second (fps). It also sets up a VideoWriter object (out) to write the processed frames to an output video file.

```
    # by default VideoCapture returns float instead of int
    width = int(vid.get(cv2.CAP_PROP_FRAME_WIDTH))
    height = int(vid.get(cv2.CAP_PROP_FRAME_HEIGHT))
    fps = int(vid.get(cv2.CAP_PROP_FPS))
    codec = cv2.VideoWriter_fourcc(*'XVID')
    out = cv2.VideoWriter(output_path, codec, fps, (width, height)) #
    output_path must be .mp4
```

Listing 4.24: Video capture setup.

This code snippet initialises classes for object detection. It reads class names and creates dictionaries (NUM_CLASS, key_list, val_list) for mapping class indices to names.

```
    NUM_CLASS = read_class_names(CLASSES)
    key_list = list(NUM_CLASS.keys())
    val_list = list(NUM_CLASS.values())
```

Listing 4.25: Class and variable initialisation.

We initialise these variables to track the events, such as possessions, made baskets, and related statistics.

```
    # set a bunch of flags and variables for made baskets and possessions
    possession = None
    possession_list = []
    combined_possession_avg = 0.5

    total_basket_count=0
    basket_frame_list = []

    #Changed after training with new weights
    baskets_dict = {"Dark": 0, "Light": 0}

    made_basket_first_frame = 0
```

```
    made_basket_frames = 0
    basket_marked = False
    basketball_center = None

```

Listing 4.26: Basketball-related variables.

If a custom YOLO model is being used, this section initialises classes for it.

```
    if custom_yolo:
      NUM_CUSTOM_CLASS = read_class_names(custom_classes)
      custom_key_list = list(NUM_CUSTOM_CLASS.keys())
      custom_val_list = list(NUM_CUSTOM_CLASS.values())

```

Listing 4.27: Custom YOLO model handling.

In this code snippet we created a loop where we read each frame from the video (vid.read()) and perform operations on it until there are no more frames to be processed. In this loop we convert the frame to the required format, pre-processes it for the model, and, if a custom YOLO model is used, makes predictions.

```
    frame_counter = 0
    # loop through each frame in video
    while True:
        _, frame = vid.read()

        try:
            first_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            original_frame = cv2.cvtColor(first_frame, cv2.COLOR_BGR2RGB)
            frame_counter += 1
        except:
            break

        image_data = image_preprocess(np.copy(first_frame), [input_size,
    input_size])

        image_data = image_data[np.newaxis, ...].astype(np.float32)

        t1 = time.time()
        # CUSTOM BLOCK FOR BASKETBALL
        if custom_yolo:

          if YOLO_FRAMEWORK == "tf":
            # use yolo model to make prediction on the image data
            custom_pred_bbox = custom_yolo.predict(image_data)

          # reshape our data to be in correct form for processing
          custom_pred_bbox = [tf.reshape(x, (-1, tf.shape(x)[-1])) for x in
    custom_pred_bbox]
          custom_pred_bbox = tf.concat(custom_pred_bbox, axis=0)
```

```
28
29          # get boxes based on threshhold
30          custom_bboxes = postprocess_boxes(custom_pred_bbox, original_frame,
    input_size, 0.3)
31
32          # extract bboxes to boxes (x, y, width, height), scores and names
33          custom_boxes, custom_scores, custom_names = [], [], []
34          for bbox in custom_bboxes:
35              if len(Custom_track_only) !=0 and NUM_CUSTOM_CLASS[int(bbox[5])]
    in Custom_track_only or len(Custom_track_only) == 0:
36                  custom_boxes.append([bbox[0].astype(int), bbox[1].astype(int)
    , bbox[2].astype(int)-bbox[0].astype(int), bbox[3].astype(int)-bbox[1].
    astype(int)])
37                  custom_scores.append(bbox[4])
38                  custom_names.append(NUM_CUSTOM_CLASS[int(bbox[5])])
39
40
41          # Obtain all the detections for the given frame.
42          custom_boxes = np.array(custom_boxes)
43          custom_names = np.array(custom_names)
44          custom_scores = np.array(custom_scores)
```

Listing 4.28: Frame processing loop.

The following code snippet is designed to identify and track the custom objects: ”basketball” and ”made baskets.” The system processes detected bounding boxes and associated information for these objects in each frame. For each detected object, it examines its label and associated confidence score.
It identifies objects labelled as ”basketball” and tracks the highest-scoring basketball by comparing confidence scores. It then stores the highest-scoring bounding box coordinates of the basketball.
Similarly, for objects labelled as ”made-basket,” it considers only those with a confidence score higher than 0.85. It keeps track of the highest-scoring made baskets, similar to the basketball detection.

```
1           # take note of the highest "scoring" made basket and basketball obj
    in each frame
2           highest_scoring_basketball = 0
3           basketball_box = None
4           basketball_center = None
5           highest_scoring_made_basket = 0
6           made_basket_box = None
7           for i, bbox in enumerate(custom_bboxes):
8             name = custom_names[i]
9             score = round(custom_scores[i], 3)
10            if name == 'basketball':
11              if score > highest_scoring_basketball:
12                highest_scoring_basketball = score
13                basketball_box = bbox
```

```
            if name == 'made-basket':
              if score > .85 and score > highest_scoring_made_basket:
                highest_scoring_made_basket = score
                made_basket_box = bbox
```

Listing 4.29: Identify and track ”basketball” and ”made basket.”

The following code handles the visualisation of detected objects in the original frame. It specifically deals with marking and annotating the bounding boxes of detected ”basketball” and ”made-basket” objects.
To identify the basketball, a purple-coloured rectangle is highlighted around it. Furthermore, a purple bar with the class ”basketball” and the displayed confidence points is placed above the box. The center coordinates of the mentioned bounding box are also given, which can be used for possession detection afterwards.

Likewise, a green bounding box appears and encompasses recognised-made baskets. Within this box, a green bar labelled ”made basket” and its corresponding confidence score follows. The system keeps track of the number of frames where a made-basket is detected. If three consecutive frames are marked and a made-basket hasn't been marked yet, it marks the basket from that point onwards. A marker is placed to set the position of the first frame in the sequence, which is then replaced once the ”marked basket” flag gets reset (an indication that the system is about to detect and mark the next made basket after 60 frames).

```
        # if it sees a basketball, put a box on it and note the center (for possession)
        if basketball_box is not None:
          cv2.rectangle(original_frame, (int(basketball_box[0]), int(basketball_box[1])), (int(basketball_box[2]), int(basketball_box[3])), (0,0,255), 1)
          cv2.rectangle(original_frame, (int(basketball_box[0]), int(basketball_box[1]-30)), (int(basketball_box[0])+(10)*17, int(basketball_box[1])), (0,0,255), -1)
          cv2.putText(original_frame, "basketball" + "-" + str(highest_scoring_basketball),(int(basketball_box[0]), int(basketball_box[1]-10)),0, 0.5, (255,255,255),1)
          basketball_center = ( (basketball_box[2]+basketball_box[0])/2, (basketball_box[3]+basketball_box[1])/2 )


        if made_basket_box is not None:
          # if theres a made basket put the box on it
          cv2.rectangle(original_frame, (int(made_basket_box[0]), int(made_basket_box[1])), (int(made_basket_box[2]), int(made_basket_box[3])), (0,255,0), 1)
          cv2.rectangle(original_frame, (int(made_basket_box[0]), int(made_basket_box[1]-30)), (int(made_basket_box[0])+(15)*17, int(made_basket_box[1])), (0,255,0), -1)
```

```
            cv2.putText(original_frame, "made-basket" + " " + str(
    highest_scoring_made_basket),(int(made_basket_box[0]), int(made_basket_box
    [1]-10)),0, 0.6, (0,0,0),1)

            if made_basket_frames == 0:
              # if this is the first frame in the sequence
              made_basket_first_frame = frame_counter

            # increment a counter for made basket frames
            made_basket_frames += 1

            # if there were 3 consecuative frames AND we haven't marked the
    basket yet, lets count it!
            if made_basket_frames >= 3 and not basket_marked:
              basket_marked = True
              basket_frame_list.append(made_basket_first_frame)
              if possession:
                # record which "team" scored the basket
                baskets_dict[possession] += 1

          # if no made basket make sure the made basket counter is at zero
          else:
            # no made basket
            made_basket_frames = 0

          # 60 frames after a made basket we can reset the "marked basket" flag
     to False
          # in essence this means we start looking for made baskets again
          if basket_marked and frame_counter > basket_frame_list[-1] + 60:
            basket_marked = False
```

Listing 4.30: Basketball object detection and score tracker in object_tracker.py file.

This code is YOLO's implementation; it detects objects in each frame, and bounding boxes (bboxes) are generated to illustrate their locations. In order to refine the detections, NMS is applied by filtering out replicate boxes and keeping the most significant remaining ones. Afterwards, the filtered boundary boxes are transferred to DeepSORT tracker, where a track is built for each object rather than for a single frame. At the same time, the code extracts patches from the frames containing the tracked "Person" bounding boxes and finds the colour percentages through the colour finding function call (find_color).

```
        if YOLO_FRAMEWORK == "tf":
            pred_bbox = Yolo.predict(image_data)
        elif YOLO_FRAMEWORK == "trt":
            batched_input = tf.constant(image_data)
            result = Yolo(batched_input)
            pred_bbox = []
            for key, value in result.items():
                value = value.numpy()
```

```
                pred_bbox.append(value)

        t2 = time.time()

        pred_bbox = [tf.reshape(x, (-1, tf.shape(x)[-1])) for x in pred_bbox]
        pred_bbox = tf.concat(pred_bbox, axis=0)

        bboxes = postprocess_boxes(pred_bbox, original_frame, input_size,
    score_threshold)
        bboxes = nms(bboxes, iou_threshold, method='nms')

        # extract bboxes to boxes (x, y, width, height), scores and names
        boxes, scores, names = [], [], []
        for bbox in bboxes:
            if len(Track_only) !=0 and NUM_CLASS[int(bbox[5])] in Track_only or
     len(Track_only) == 0:
                w = bbox[2].astype(int)-bbox[0].astype(int)
                h = bbox[3].astype(int)-bbox[1].astype(int)
                if h < height/3 and w < width/4:
                  if h > 120:
                    boxes.append([bbox[0].astype(int), bbox[1].astype(int), w,
    h])
                    scores.append(bbox[4])
                    names.append(NUM_CLASS[int(bbox[5])])

        # Obtain all the detections for the given frame.
        boxes = np.array(boxes)
        names = np.array(names)
        scores = np.array(scores)
```

Listing 4.31: Person detection and tracking block in object_tracker.py file.

The following code extracts patches from the frame based on the bounding boxes of detected objects (presumably players) and calculates the colour ratio for each patch using the find_color function.

```
        # detect jersey color using the tracked persons bounding box
        patches = [gdet.extract_image_patch(frame, box, [box[3], box[2]]) for
    box in boxes]
        color_ratios = [find_color(patch) for patch in patches]
```

Listing 4.32: Jersey colour detection in object_tracker.py file.

```
        features = np.array(encoder(original_frame, boxes))

        # mark the detection
        detections = [Detection(bbox, score, class_name, feature, color_ratio)
    for bbox, score, class_name, feature, color_ratio in zip(boxes, scores,
    names, features, color_ratios)]
```

Listing 4.33: Feature extraction and detection initialisation in object_tracker.py file.

It uses a tracker (DeepSORT tracker) to predict and update the tracks based on the detected objects.

```
        # Pass detections to the deepsort object and obtain the track
    information.
        tracker.predict()
        tracker.update(detections)
```

Listing 4.34: Object tracking in object_tracker.py file.

This loop processes each tracked object, extracts information such as bounding boxes and class names, and checks if there's a basketball in a person's bounding box to determine possession.

```
        # Obtain info from the tracks
        tracked_bboxes = []
        color_ratio_list = []
        check_possession = False
        for track in tracker.tracks:
            if not track.is_confirmed() or track.time_since_update > 5:
                continue

            color_ratio = track.get_color_ratio()
            color_ratio_list.append(color_ratio)

            bbox = track.to_tlbr() # Get the corrected/predicted bounding box
            class_name = track.get_class() #Get the class name of particular
    object
            tracking_id = track.track_id # Get the ID for the particular track
            index = key_list[val_list.index(class_name)] # Get predicted object
     index by object name

            tracked_bboxes.append(bbox.tolist() + [tracking_id, index]) #
    Structure data, that we could use it with our draw_bbox function

            # if there is a basketball in the frame, and its "in" a persons
    bounding box, check what box it is in for possession
            if basketball_center:
              if basketball_center[0] >= bbox[0] and basketball_center[0] <=
    bbox[2]:
                if basketball_center[1] >= bbox[1] and basketball_center[1] <=
    bbox[3]:
                  check_possession = True
                  if color_ratio <= .2:
                    # lighter team
                    possession_list.append(0)
                  else:
                    # darker team
                    possession_list.append(1)

```

```
            else:
              # no basketball in frame
              pass
```

Listing 4.35: Processing tracked objects in object_tracker.py file.

This code snippet updates possession tracking based on colour ratios and calculates a combined possession average.

```
      # if the ball is in a bounding box, update out possession tracker
      if check_possession:
        if len(possession_list) > 60:
          # this function takes an average of the last 60 posessions marked
to determine current position
          # it weights the most recent detections more
          # WIP algorithm
          possession_list = possession_list[-60:]
          last_60_avg = sum(possession_list[-60:])/60
          last_30_avg = sum(possession_list[-30:])/30
          last_15_avg = sum(possession_list[-15:])/15
          last_5_avg = sum(possession_list[-5:])/5

          combined_possession_avg = round((last_60_avg + last_30_avg +
last_15_avg + last_5_avg)/4,3)

        else:
          combined_possession_avg = round(sum(possession_list)/len(
possession_list),3)

        # use possession average to determine who has the ball atm
        if combined_possession_avg < 0.5:
          possession = "Light"
        elif combined_possession_avg > 0.5:
          possession = "Dark"
```

Listing 4.36: Possession tracking and averaging in object_tracker.py file.

In this code snippet we employ YOLOV3 for object detection. The video frames are processed, and the YOLO model predicts bounding boxes, scores, and class names for detected objects. For custom YOLO models, predictions are obtained using TensorFlow. The DeepSORT tracker is then used to associate detections with object tracks, and information about these tracks is collected for further analysis.

```
      # draw detection on frame
      image = draw_bbox(original_frame, tracked_bboxes, color_ratios=
color_ratio_list, CLASSES=CLASSES, tracking=True)

      t3 = time.time()
      times.append(t2-t1)
      times_2.append(t3-t1)

```

```
        times = times[-20:]
        times_2 = times_2[-20:]

        ms = sum(times)/len(times)*1000
        fps = 1000 / ms
        fps2 = 1000 / (sum(times_2)/len(times_2)*1000)

        if possession == "Light":
          image = cv2.putText(image, "Posession: {}".format(possession), (width
    -400, 30), cv2.FONT_HERSHEY_COMPLEX_SMALL, 1, (50, 255, 255), 2)
        else:
          image = cv2.putText(image, "Posession: {}".format(possession), (width
    -400, 30), cv2.FONT_HERSHEY_COMPLEX_SMALL, 1, (0, 0, 255), 2)


        image = cv2.putText(image, "Posession Avg: {}".format(
    combined_possession_avg), (400, 30), cv2.FONT_HERSHEY_COMPLEX_SMALL, 1, (0,
     0, 255), 2)

        image = cv2.putText(image, "Time: {:.1f} FPS".format(fps), (0, 30), cv2
    .FONT_HERSHEY_COMPLEX_SMALL, 1, (0, 0, 255), 2)


        print("Time: {:.2f}ms, Detection FPS: {:.1f}, total FPS: {:.1f}".format
    (ms, fps, fps2))
```

Listing 4.37: Drawing bounding boxes and text on the frame in object_tracker.py file.

This snippet involves drawing bounding boxes around detected objects using the draw_bbox function. Additional text is added to the frame to indicate possession, possession average, and time-related metrics. The frame is displayed or saved as specified.

```
        if output_path != '': out.write(image)
        if show:
            cv2.imshow('output', image)

            if cv2.waitKey(25) & 0xFF == ord("q"):
                cv2.destroyAllWindows()
                break
```

Listing 4.38: Displaying and saving in object_tracker.py file.

Here we calculate and display performance metrics, including detection time and fps. The function returns a dictionary with information about detected made baskets and frames with marked baskets. This data can be used for further analysis or visualisation outside the function.

```
    cv2.destroyAllWindows()
    return_data = {"baskets_dict": baskets_dict, "basket_frame_list":
    basket_frame_list}
    print("video saved to {}".format(output_path))
```

```
    return(return_data)
```

Listing 4.39: Performance metrics and return data in object_tracker.py file.

Finishing we initialise the video path, load a YOLO model using the Load_Yolo_model() function, and performs object tracking on the specified video. The tracking is focused on persons (Track_only=["person"]). The processed video is saved at "`C:/Users/guipo/OneDrive/AmbientedeTrabalho/CDCProject/results/vid2\_result2.avi`."

After tracking, it checks if there are any objects labelled as "Dark" in the tracked results. If the count is greater than 0, it runs a darknet command. The darknet command involves running a demo of a custom YOLO model on a video file. This darknet command will only be executed if the conditions specified in the if statement are met.

```
video_path = "C:\\Users\\guipo\\OneDrive\\Ambiente de Trabalho\\CDCProject\\
    input_videos\\video2.mp4"


yolo = Load_Yolo_model()
Object_tracking(yolo, video_path, "C:\\Users\\guipo\\OneDrive\\Ambiente de
    Trabalho\\CDCProject\\results\\vid2_result2.avi", input_size=
    YOLO_INPUT_SIZE, show=False, iou_threshold=0.1, rectangle_colors=(255,0,0),
     Track_only = ["person"])

result_data = Object_tracking(yolo, video_path, "C:\\Users\\guipo\\OneDrive\\
    Ambiente de Trabalho\\CDCProject\\results\\vid2_result2.avi", input_size=
    YOLO_INPUT_SIZE, show=False, iou_threshold=0.1, rectangle_colors=(255,0,0),
     Track_only=["person"])

if result_data["baskets_dict"]["Dark"] > 0:
    # If the condition was met - run
    darknet_command = 'darknet.exe detector demo data/obj.data cfg/yolov3-
    custom.cfg backup/weights2/yolov3_custom_final.weights recorte.mp4 -
    out_filename result.avi'
    subprocess.run(darknet_command, shell=True)
```

Listing 4.40: Video paths initialisation in object_tracker.py file.

After the creation of all important files/folders, as we can see in Figure 4.7, and with the requirements.txt successfully installed we could finally run "object_tracker.py" folder.

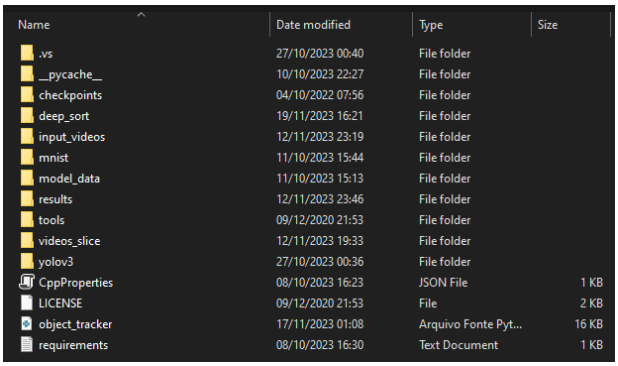


Figure 4.7: CDCProject folder containing all the necessary folders.

4.5.3 Executing our System

To execute our system, we started by placing the desired video in the designated input folder, ”input_videos”. Subsequently, in the ”object_tracker.py” file, we specified the video path along with its corresponding name.

```
video_path = "C:\\Users\\guipo\\OneDrive\\Ambiente de Trabalho\\CDCProject\\
    input_videos\\video.mp4"
```

Listing 4.41: object_tracker.py.

Additionally, we indicate the relevant output path and file name. It is suggested opting for the .avi extension for the video format, although .mp4 is also a suitable choice.

```
Object_tracking(yolo, video_path, "C:\\Users\\guipo\\OneDrive\\Ambiente de
    Trabalho\\CDCProject\\results\\video_result.avi", input_size=
    YOLO_INPUT_SIZE, show=False, iou_threshold=0.1, rectangle_colors=(255,0,0),
     Track_only = ["person"])
```

Listing 4.42: object_tracker.py.

We then executed the following command prompt and waited for the processing to finish:

```
  python object_tracker.py
```

4.6 System Interface

We have developed a user-friendly GUI Python interface that facilitates the interaction with our Basketball player tracking system. The interface, created using Tkinter [72], provides a simple yet effective way for users to upload a video file and initiate the analysis process, as we can observe in Figure 4.8.

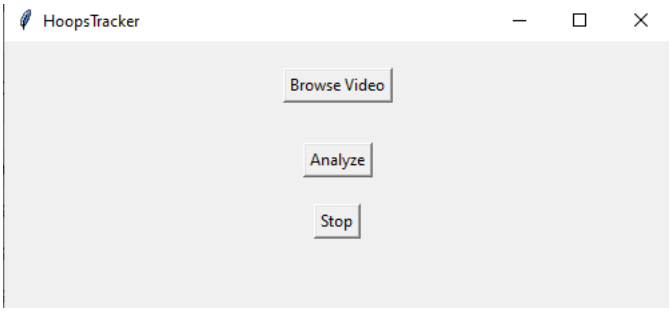


Figure 4.8: Python GUI Interface Design.

The design is equipped with a 'Browse Video' button where the user can just click and choose the video file from the local directories. After the video has been uploaded, the "object_tracker.py" file gets updated within "object_tracker.py" the "video_path" variable is positioned.
For analysis to start, we click on the 'Analyse' button. This button will automatically call the execution of the "object_tracker.py" file, thus tracking the specified video. The system manages to register players and their movements on the screen, providing an integral input into the understanding of the game.
Furthermore, there is a 'Stop button' that lets users break the analysis right away. This feature contributes for improved user control over the tracking process itself, creating a smooth and flexible experience for the user.
The GUI interface utilises the Tkinter framework for the graphical display integrated with the "object_tracker.py" file. This interface, therefore, provides a platform that is fun to work with and, above all, simple enough for everyone, regardless of their level of technical knowledge.

4.7 Ethical Considerations

We need to consider the ethical context that drives the origin of these technologies. AI applications lead to the introduction of many new ethical considerations, for example, player and basketball tracking, that require a cautious and precautionary approach. Below, we outline some key ethical facets that we found relevant to our system:

- **Privacy Concerns** - The system uses and transmits visual data, including players' pictures and videos. Thus, such supervision due to privacy turns into a new problem in which the players must have a right to not be filmed or monitored in the training room or outside the arena. It's important for companies to develop fundamental privacy policies and data processing protocols that will assure the protection of people's rights and privacy.

- **Data Security** - The collected data, which can be very sensitive in nature, must be secured from breaches and unauthorised interferences by ensuring proper storage and protection. The main obstacle is to guarantee the data security and integrity of this information in order to reject any possible misuse or harm.

- **Consent and Informed Participation** - Players and people concerned are supposed to be informed about the use of this process and then give their consent to data collection as well as analysis. To ensure trust and ethical standards, there needs to be good communication.
- **Accountability** - It is necessary to make clear who is in charge of the operation, the maintenance, and the supervision of this system. Accountability should be the fundamental aspect of any procedure aiming to solve problems or mistakes on the spot.
- **Ethical Data Use** - Data acquisition through this system should only be used for legitimate and ethical purposes, like enhancing player performance, coaching strategies, and making the sport fair enough.

While this system offers potential in enhancing the understanding and analysis of Basketball games, we need to take into account these ethical considerations outlined above. It is our responsibility to ensure that the implementation of such technology aligns with ethical principles and respects the rights and dignity of all individuals involved in the sport.

4.8 Conclusion

This chapter provided a comprehensive overview of the approach taken to develop our action recognition system for Basketball. Remarkably, the level of technology currently used in sports games and training is regarded as low, and players rely heavily on muscle memory for skill development. Our purpose was to create an affordable and effective system that can be used in real-world situations without much adjustment. We believe that our system can lead to innovative technology in the field of sports analytics. The ultimate choice, then, of creating an algorithm for Basketball and player tracking, manipulating the Google Colab platform and GPU acceleration, as well as temporary pre-trained convolutional layers, among others, proved to be good. The technical solutions in this area made it possible to develop not only the Basketball tracking model but also the players' one so quickly. The DeepSORT integration was an amazing choice that proved its high efficiency for player tracking, and, as a result, improving the quality of the analysis is less challenging in contrast to individual labelling of each player, which would take a considerable amount of time. This chapter summarises in a systematic and detailed way the actions undertaken, the tools used, and the ethical issues considered while working on developing our Basketball action recognition platform.

Chapter 5

Results and Discussion

5.1 Chapter Overview

This chapter presents the fundamental tasks undertaken to ensure the successful development of this Dissertation. This section illustrates the comprehensive results of the study up to date, as well as indicating the challenges encountered and corresponding solutions that aid in the demonstration of the progress made and the final status of this Dissertation. A detailed examination of the outcomes is provided to ensure a complete vision of what has been accomplished, contributing to a better understanding of where the Dissertation headed and its wider implications in the area of sport action recognition.

Section 5.2, outlines the various tasks and challenges we took into account during the development of our tracking system.

Section 5.3, details the technical specifications, configurations, and components of the computer were we developed our system.

Section 5.4, presents and analyses the findings from the implementation and testing phases.

Section 5.5, explores the limitations and main obstacles encountered during the development and implementation process.

Section 5.6, outlines potential enhancements and refinements that could be implemented to further advance the capabilities and performance of the system.

5.2 Considered Tasks

In this section we outline the specific tasks initially planned for our system. These tasks are carefully tough out to address key objectives, ensuring good performance of our model in action recognition for Basketball games. Each task, as initially projected, played a significant role in establishing a plan so that we could achieve our Dissertation's main goals. The outlined tasks included:

1. Track the basketball and made baskets;
2. Detect the players on court;

3. Recognise the players from each team;
4. Recognise actions from a single player;
5. Map the players from a 3D representation to a 2D, for positioning analysis.

Tasks **1)**, **2)** and **3)** constituted the primary goals of this Dissertation, whereas **4)** and **5)** outline the future objectives of the Dissertation.

5.3 Computer Specifications

This section presents a detailed overview of the computer specifications employed in our Dissertation. We explore into key components, offering insights into the hardware and configurations that define the computational landscape of our study.
The computer specifications were as follows:

- **Operating System:** Windows 10 Pro 64-bit (10.0, Build 19045).
- **Processor:** AMD Ryzen 5 1600 Six-Core Processor (12 CPUs), ~3.2GHz.
- **Memory:** 16384MB RAM.
- **GPU:** NVIDIA GeForce GTX 1060 6GB.

The results presented are based on the current system specifications, equipped with an AMD Ryzen 5 1600 Six-Core Processor running at approximately 3.2GHz, 16GB RAM, and a NVIDIA GeForce GTX 1060 6GB graphics card.
It's also important to note that **a one-minute video took approximately 35 minutes when analysed at 60 fps**. Even though the final results from the analysis were good, it took too long to analyse the data. With an enhanced GPU, we could've noticed some notable improvements in the efficiency of the analysis, shortening the time it tooks for total video processing.

5.4 Results

In this section, we indicate the effective results of our system's deployment. With the efficient completion of our thorough planning and development exploration, we present the already-acquired outcomes from the Basketball action recognition model. At present, our system has provided data about the movement of different elements in the Basketball game environment, which is fairly precise. It reveals the movements and positions of players with speed, captures the basketball into its trajectory, and detects accurate instances of successful shots. This level of accuracy showcases the ability of our tracking algorithms to follow player movements and also that of the basketball. Even though the system is skilled in only identifying successful shots, this does demonstrate the success of our implementation due to its practical and effective nature.

5.4.1 Custom Dataset Results

Figure 5.1 illustrates the Mean Average Precision (mAP) graph values obtained during the training of our custom dataset. The mAP is a key metric in evaluating the performance of object detection models, providing insights into their accuracy and precision across different classes. This graph serves as a visual representation of the training performance, showcasing the model's progression over training iterations.

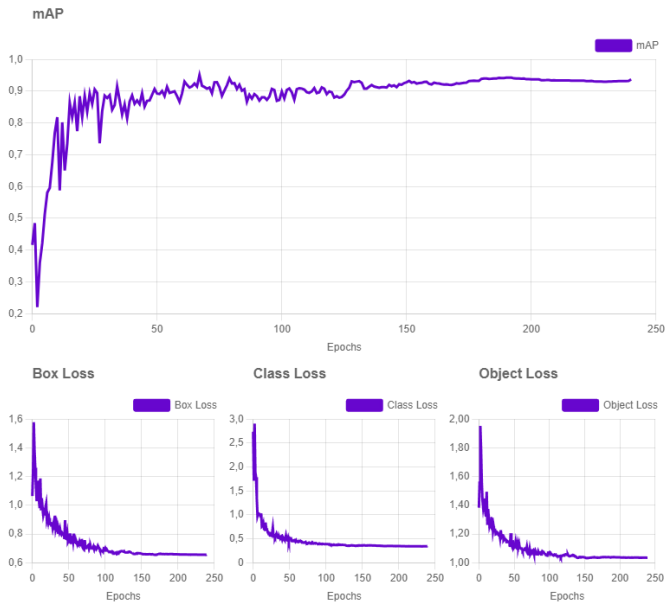


Figure 5.1: Custom dataset training mAP graphs.

The mAP graph evolution in our dataset training results is a spike that hints at a steady improvement over time. The straightline ascension graph up to epoch 240 can be interpreted as the model enhancement throughout the training, which is recorded in terms of incrementing the accuracy valuation from a low value to near 1 at the 240 epoch.

Concerning **Box Loss**, **Class Loss**, and **Object Loss**, our model showed a non-optimal performance in the first stages of the training. The error in locating and classifying the object with our model progressively decreases along with the decrease in loss values, indicating that our model was being correctly trained.

Breaking down each loss graph:

- **Box Loss**: Starting around 1.6 and decreasing over epochs indicates that the model was becoming more accurate in predicting the bounding box coordinates of objects.
- **Class Loss**: Similarly, a decrease from an initial value of 2.7 implies that the model was getting better at correctly classifying objects into their respective categories.
- **Object Loss**: The decline from 1.94 indicates an overall improvement in object detection, combining advancements in both location and classification.

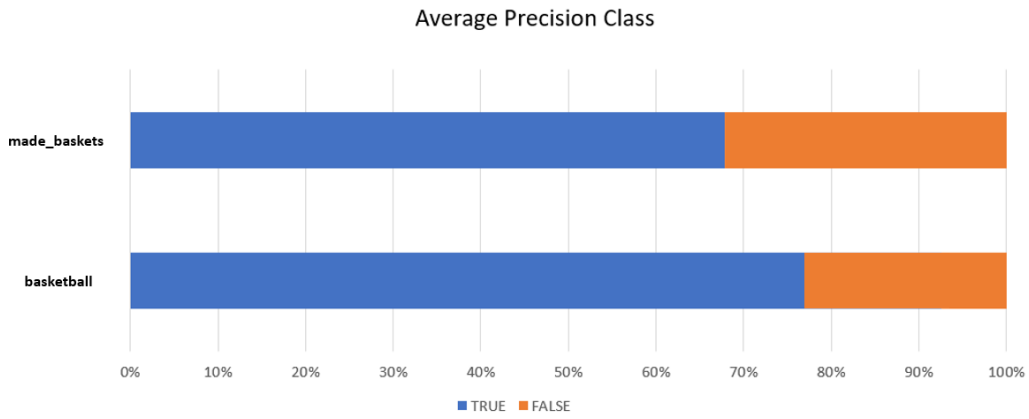


Figure 5.2: Custom dataset average precision by class during testing.

Our tracking system showcased high performance and successfully captured most of the basketball shots in 68% of the instances. This success was primarily due to the strength of the implementation of the DeepSORT tracker, which ensures the correct cause-and-effect connection of a specific object. Nevertheless, challenges emerged, resulting in 23% of failed trackballs, as we can observe in Figure 5.2.
The leading cause of failure in tracking was the interference that was created during the gameplay itself. Obscuring the basketball by players or other parts of the court in some instances will make it tough to get accurate results. The system depends on unobstructed views over the court to precisely evaluate the position and update the basketball trajectory. Such features reveal that sports tracking is an extremely complex area where quick movements, player interactions, and dynamic game patterns can prevent constant tracking of the basketball. A possible solution would be a revision of the algorithm to make it work better in complex environments or looking for other technological approaches, for example, multi-sensor fusion, to improve the resilience of tracking in complex situations.

5.4.2 Tracking Abilities

(a) Players and basketball. (b) Made basket. (c) Team distinction.

Figure 5.3: Main objects being tracked during video analysis.

Our system successfully followed and labelled the basketball, the made baskets, and the players' jerseys of varying colours from the two teams when undergoing video analysis. However, the results were not perfect and need to be enhanced, especially in showing the difference between them, the blue team in particular. It became apparent that the most significant obstacles to precise monitoring were the lighting consequences and the

resemblance of the blue court colour to the blue team's uniform, which made it impossible, at times, to differentiate as it can be observed in Figure 5.3c.

Although the made baskets were accurately registered, additional optimisation possibilities exist by means of a more extensive dataset of successfully made baskets. However, the basketball, which was constantly tracked with relative ease, showed steady tracking throughout the whole analysis, with its tracking percentage holding at a high value.

5.5 Weaknesses and Main Challenges

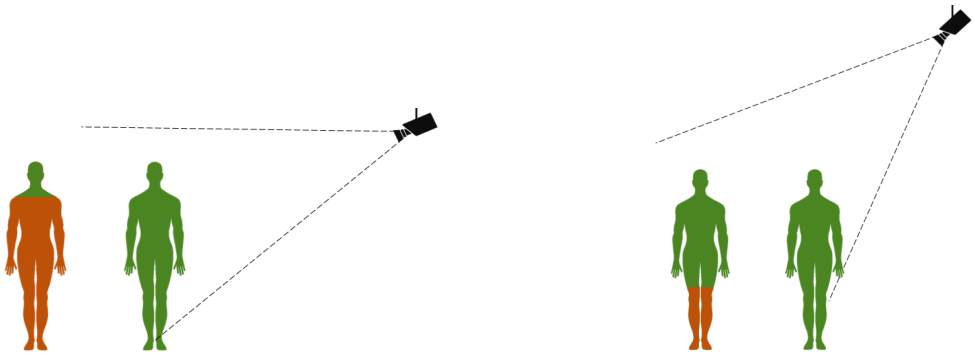
5.5.1 Optimal Capturing Device

In the initial stages of planning this Dissertation, careful consideration was given to the method of recording Basketball games, an important aspect of our research. The first choice was to employ a **Xiaomi Redmi Note 8T** smartphone for capturing the game footage. However, it became apparent that this device, due to limitations in its depth perception, couldn't deliver the video quality required for our analysis.

Consequently, we made the decision to explore alternative recording solutions. After thorough evaluation, we acquired a **GoPro Hero 10** camera. This approach was truly beneficial in order to improve the quality of our recordings, solving the issues we unfortunately found with the smartphone in early stages. The GoPro 10 enabled us to capture high-quality video footage, free from the initial limitations and errors we experienced with the previous equipment. This decision not only improved the overall quality of our data but also ensured the reliability and consistency necessary for good analysis during our research. Taking into account the post-annotation phase, we reached a conclusion that to potentially enhance annotating speed, we could have involved lowering the fps, as this could have provided us with more time to accurately study of other implementation aspects.

5.5.2 Capturing Device Positioning

In retrospect, it becomes clear that the correct positioning of the camera within the arena was very important for capturing the important footage. With the aim of capturing the game from the most favorable location in mind, a closer examination suggests that the camera's placement could have been optimised further. One key consideration that emerges was the elevation of the camera. However, it's important to note that mounting the camera higher, as desirable as it may have been, posed a certain challenge. Due to the lack of suitable arena conditions for an elevated installation, the camera needed to be mounted in a fixed position on the wall to ensure that some spectators remained outside the frame, something that was not possible at all.



(a) Low capturing angle. (b) High capturing angle.

Figure 5.4: The camera's ability to capture players considering potential obstructions. Orange - obstructed, Green - captured.

Placing the camera at a higher vantage point would have allowed for a more comprehensive and unobstructed view of the players and the basketball in action, as we can observe in Figure 5.4. Given this high vantage point, the view should be obvious and distraction-free from spectators, as well as undesirable elements that could have occurred inside the arena.

The camera placement can definitely be upgraded, which for sure will result in better data quality with a clearer, more dynamic, and more involving video display. Therefore, this would have led to the acquisition of higher-quality videos to be used in the upgrading of the performance of our custom detector focusing on objects. With this, we would not only able to maintain the veracity of the recorded footage, but we would also reduce the likelihood of any possible distractions or unnecessary interruptions while the important data was being gathered.

Moreover, a decision taken at an early stage indicated that the camera should've been static, but after detailed research in that regard, we found that it could be dynamic as well. With this approach, camera rotation could be from one court sideline to the other, improving the capability to present Basketball game coverage in a more dynamic way. The knowledge gained can be used in subsequent improvements to the recording technique, providing chances for optimisation of game analysis in the future.

5.5.3 Occlusions and Crowded Scenes

During the process of capturing footage, we encountered three notable challenges related to occlusions and crowded scenes, the spectators, side court interferences, and the players, Figure 5.5 illustrates all the areas where we encountered obstructions during the acquisition of game footage.

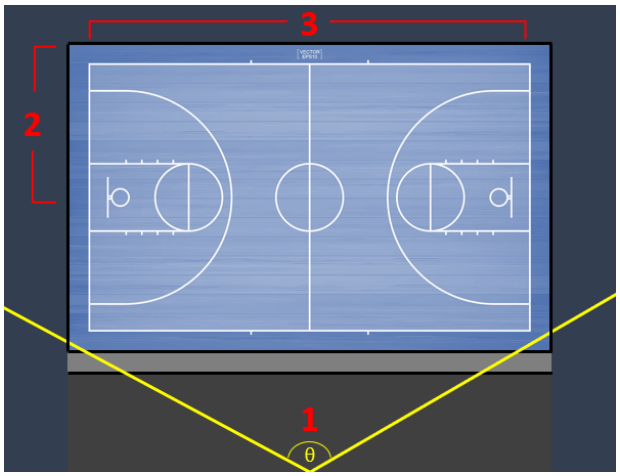


Figure 5.5: Interferences found during game footage capture. 'θ' represents the horizontal FOV of 122.6º in Superview mode.

1. **The Spectators**: Firstly, the presence of dynamic spectators, while an essential part of the game's ambiance, can occasionally introduce obstructions, as we can see in Figure 5.6. People passing or sitting close to the court would not knowingly block the camera view in the process of getting the data. While the arena provided a good platform, its imperfect conditions imposed some drawbacks. Ideally, we would've liked to place the camera in a higher location to get an unobstructed view of the court, but we did not have the ability to do so. As a result of this problem, it became necessary for us to test a variety of methods and approaches so that the data captured and tracked was still reliable, even when the camera view was not perfect.

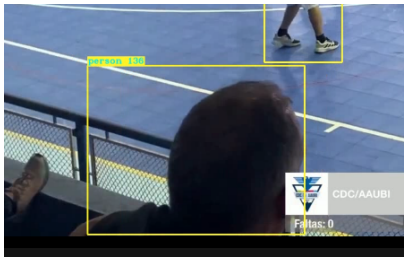


Figure 5.6: Detection of a spectator during video analysis.

2. **Side Interferences**: Among the processes of capturing video footage, there were some occasions when the recording location included numerous activities simultaneously. These actions included children along with others who were walking around and even playing at side hoops from within the arena. These activities happened occasionally without being part of the recording process as planned.

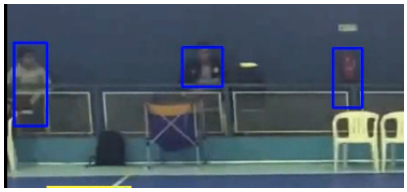


Figure 5.7: Detection of a side interference during video analysis.

As a result, during the detection process, some of these incidental activities and individuals were wrongly included in the detection phase and where wrongly labelled, as we can observe in Figure 5.7. What is especially noteworthy is that these unplanned labels did not stand for particular associations but rather are the results of natural circumstances during the recording process. Steps should be taken to minimise these types of negative perceptions, and as a matter of procedure, quality control must be enforced in order to solicit real and reliable results.

3. **The Players**: Although DeepSORT proved useful in dealing with occlusions, player tracking emerged nonetheless as the most formidable challenge within the scope of this problem. Given the non-stop player movements in proximity to one another, maintaining persistent tracking of an individual player posed as a significant challenge. In the context of future research, our aim would be to further investigate this issue.

   Additionally, it's worth noting that during significant plays, there were instances when benched players would momentarily walk onto the court to celebrate, further adding complexity to player tracking, as we can see in Figure 5.8. Such moments of celebration made it a dynamic process, as players got together for a short while and then went back to the bench or court. These examples show that for the system to be flexible enough to adjust to unexpected tactical moves, specific tracking solutions have to be implemented. In reference to dynamic gameplay, including undependable player movements, it should be stated that unexpected action provides a chance for creativity.

Figure 5.8: Detection of benched players during video analysis.

5.5.4 Reduced Image Dataset

Obtaining the required "data" constitutes an exceptionally time-consuming process. This process involves meticulous frame-by-frame capture from the recorded footage and subsequently annotating each one. To provide context, in an attempt to attain a desired level of accuracy, we annotated a total of 4550 images.
In an ideal case, we would have had enough images from a larger dataset that would include different environment conditions. Expanding the dataset means that we will dramatically increase the quality of predictions, especially in the case of new events in different environments. This broader dataset would have greatly reduced the false positives we identified, as Figure 5.9 illustrates, especially in cases involving made baskets and basket-

ball tracking during plays when players either obstruct each other or are being fouled.

(a) A false positive of a sneaker being labelled as a basketball.

(b) A false positive regarding a made basket.

Figure 5.9: Examples of false positives found during testing.

5.6 Future Improvements

5.6.1 Court Mapping

As part of our future plans, we explored the possibility of employing homography for court mapping. This prospective approach aims to transform the Basketball court's perspective, converting it from a 3D representation to a more accessible 2D visualisation by means of a transformation matrix. This innovative idea was under consideration for potential implementation [73].

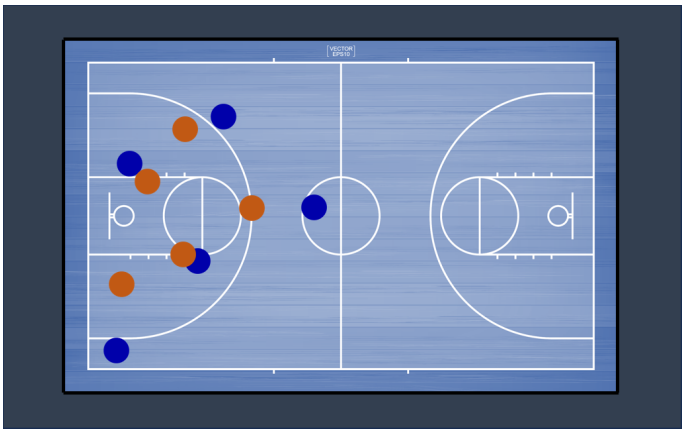


Figure 5.10: 2D Basketball court player mapping.

Having learned the exact measurements of the court's dimensions, our next step was computing a 3x3 homography matrix through an affine mapping. It would become the system's vital tool. With this method, we are able to place the players in a scene that had originally been in a three-dimensional space and accurately project this onto a two-dimensional representation of the court.
This representation would make it easier to visualise the players' positions and movements on the Basketball court, enhancing the accuracy and clarity of the tracking system and improving the ability to analyse player interactions and actions during the game, as

seen in Figure 5.10. This future plan was significantly shaped by the research conducted by Yash Pandya, et al., 2023 in [74]. They brought sensor data from wearables to guide and locate players in-screen with easy and more precise. It derives from a field registration homography based on the information from the Radio Frequency Identification (RFID) sensors and the coordinates of on-screen players as a result. The use of sports tech for tracking players can open up a whole new world of data analysis, which in turn can be beneficial for almost any sport, where comprehending players' data can provide insights for tactics.

5.6.2 System's Analysis Options

In the process of our research work, we devoted time to get informed of the research done by Nurettin ACI and Muhammed Fatih, 2023 [75], they used a connection with computer vision and DL methods intended for precise detection in sport training. This kind of research gave rise to the idea of onscreen courses with action recognition-based systems. With this in mind, we examined two potential approaches: game footage video analysis (after and during game respectively).

5.6.2.1 Post Game Video Analysis

Just like in any sport, the war isn't really over even when the final whistle in blown. Such a process continues even after the game is over, where all the moments, movements, and tactics are considered as case studies. This moment is key for post-game video analysis, where a system able to recognise actions can be put to good use.
It is very easy to carefully study and analyse each video recorded match by watching player actions, team interactions, and tactical plays frame by frame. The system may not only be able to identify actions as categories but is also capable of offering us a separate way to determine player performance as well as the dynamics of the game. Taking into consideration the importance of these tools, we decided to study two instruments that could play a significant role in handling this given situation.

The first tool explored was **OpenPose** [76], a tool for human pose estimation that enables the recognition of key body joints and their movements in real-time. In post-game video analysis, OpenPose can be instrumental in understanding player kinetics.
As part of the research, we studied work by Newman and Maleko, et al., 2019 [74] in conjunction with a posture analysis model that was able to predict Basketball free throws successfully. Interestingly, there are demonstrated high levels of accuracy, which clearly show OpenPose as a viable tool for posture analysis without the need for expensive hardware. It successfully depicts the exact poses of the limbs, joints, and head positioning of the players. This precise thought pattern enables performance analysis from different dimensions.
Post-game, OpenPose can assist in:

- **Player Biomechanics Analysis**: Understanding the kinetics of player movements

helps in assessing the efficiency of techniques, shooting styles, and defensive strategies.

- **Injury Prevention**: By analysing body postures, it's possible to identify potential stress points or irregularities in movements that could lead to injuries.
- **Tactical Review**: Tracking the spatial coordination of players helps in evaluating the effectiveness of team strategies, positioning, and collaborative plays.

Secondly we investigated **AlphaPose** [77], this method is highly accurate and it is used for estimating the multi-person posture. In Basketball when we are considering this tool, it will contribute in a meaningful way to understanding the dynamics of player interactions [78].
In post-game analysis, AlphaPose can be beneficial for:

- **Action Classification**: Identifying specific player actions such as dribbling, passing, shooting, and defensive maneuvers.
- **Player Heatmaps**: Generating heatmaps based on key actions provides a visual representation of player activity, highlighting relevaant areas on the court.
- **Statistical Insights**: Integrating action recognition data with game statistics offers a comprehensive overview of player contributions and areas for improvement.

5.6.2.2 Real-time Video Analysis

As the game progresses, the system accurately determines player movements, identifies team dynamics, and analyses tactical plays, all simultaneously. With the capacity to process the recordings frame by frame, this technology provides precious time for the implementation of proactive measures. Coaches, analysts, and fans can quickly make decisive moves to increase efficiency, gain an advantage, and gain a new, interesting view of the sport with the help of this innovative technique.

PoseNet [79] offers instantaneous pose estimation for players on the court. In live Basketball analysis, PoseNet's capabilities become crucial:

- **Live Player Tracking**: As players move across the court, PoseNet provides a real-time map of their poses, enabling dynamic tracking and spatial analysis.
- **Immediate Biomechanical Feedback**: Coaches can receive instantaneous feedback on player movements, helping in real-time adjustments to techniques and strategies.
- **Interactive Coaching**: With PoseNet, coaches can interactively guide players based on live pose analysis, gaining immediate improvements.

EfficientPose[80] contributes to the real-time dynamism by providing efficient multi-person pose estimation. In the live Basketball, its significance is crucial:

- **Swift Multi-Person Pose Estimation**: EfficientPose's agility ensures that even during a dynamic game with multiple players, pose estimation remains fast and accurate.
- **Strategic Decision Support**: Analysts can use real-time pose data to understand team dynamics, assess player interactions, and make recommendations during the game.
- **Enhanced Fan Experience**: Fans, too, benefit from the real-time insights, gaining a deeper understanding of player movements and strategic plays.

In conclusion, the integration of post-game and real-time video analysis tools can provide useful data for evaluating player performance, enhancing coaching strategies, and offering a more enjoyable experience for both analysts and fans.

5.6.3 Integration of MNIST Dataset for Jersey Number Tracking

The existing implementation is based on the computer vision analytics approach used for tracking the Basketball players, the basketball, and made baskets. To develop participant identification, we can proceed with the improvement of the methodology used in the work by examining the MNIST dataset [81]. The MNIST dataset is a collection of labelled handwriting digits that is mainly used in the ML context for developing DL algorithms. In our case, it can involve training a model to recognise jersey numbers with good accuracy.

5.6.3.1 Advantages of MNIST Integration

Integrating the MNIST dataset offers several advantages:

- **Jersey Number Recognition:** MNIST provides a vast collection of labelled handwritten digits, assisting in training the model to recognise jersey numbers accurately.
- **Numeric Identification:** The system, with MNIST integration, can specifically identify and track players based on their jersey numbers, contributing to more accurate player tracking and statistics collection during the game.
- **Improved Player Profiling:** By associating jersey numbers with individual players, it becomes possible to create more detailed player profiles, including performance statistics, player movements, and strategic analyses.

5.6.3.2 Implementation and Challenges Considerations

In order to generate successful jersey number recognition performance, transfer learning algorithms can be applied to customise ready-made models from the MNIST dataset. This technique covers each number application, from a variety of fonts and styles to vertical or horizontal design elements on jerseys. Moreover, the live jersey number recognition feature accelerates the system's performance instantly updating the players' information

thus facilitate a smooth and dynamic tracking system.

While incorporating the MNIST dataset in our system can bring some advantages, it also carries along certain challenges, which need to be paid a close attention. The uniform identity of a sport team may vary in terms of appearance depending on a variety of styles across different teams and environmental conditions, which needs adaptation. Moreover, the system is expected to automatically make adjustments to the dynamic gameplay scenarios to cover situations where players can be discovered to be partially occluded or moving and also possess different lighting conditions.

5.7 Conclusion

With this chapter's development, we had the opportunity to consider the work done so far and find out the effectiveness of the system developed and its achievements. Looking at this Dissertation experience, the final results are better than we had initially envisioned, taking into consideration that initially there was no background knowledge in the given technical field.
Furthermore, there were different factors that slowed the Dissertation's progress. Even though we had a successful outcome, these inconsistent lighting conditions at the arena, players/basketball obstruction by spectators, the inability to build a custom dataset, different jersey colours, and so on, were the main obstacles that prevented us from achieving a more complete system. These obstacles allowed us to improve the outcome of our efforts and discover multiple ways the system could still be improved to reach its maximum potential.
In conclusion, we could not underestimate our accomplishment, as it was a big step forward from where we stood at the beginning of the project. While our AI-based custom object tracking, which aimed at achieving good accuracy levels, has indeed performed even better than had been initially planned, we are surprised of what was achieved despite training with a dataset that was really small. As demonstrated in Section 5.4.1 about our given test result, we got these values, which were considered good in the end. Nevertheless, we understand the fact that, with a larger dataset and some coding improvements, we would likely obtain better results in the future.

Chapter 6

Conclusions and Further Work

6.1 Chapter Overview

With the conclusion of this Dissertation, I understand its enormous significance. The research performed has brought valuable insights together through a theoretical and practical approach. The Dissertation's representation provides a real-world application example as well.
Having gone through the complicated research procedure, my Dissertation represents not only the information obtained but also the abilities developed within the critical thinking, problem-solving, and effective communication fields. Performing the development process in detail, the Dissertation provides a thorough analysis of the methods used as well as the endpoint of the system that has been structured.

6.2 Proposed Goals vs. Achieved

As outlined in Section 5.2, I learned that the performance of the system, contrary to my initial beliefs, became more productive after I had completed it, and the accuracy values I achieved in the end exceeded the expectations regarding the system's capability. The camera's recording angle, although I opted for a fixed position during recordings, was a challenge as well, being studied throughout the entire Dissertation on what I could've changed to achieve better footage without the obstacles present in Section 5.5.
The following checklist provides an overview of the accomplishments made during the development of this Dissertation:

- ☑ Track the basketball and made-baskets.
- ☑ Detect the players in the playing court.
- ☑ Recognise the players from each team.
- ☒ Map the players from a 3D representation to a 2D, for positioning analysis.
- ☒ Recognise specific player actions such as shooting, passing, dribbling, and blocking.
- ☑ Build a user-friendly Python GUI for enhancing system usage.

While the following checklist highlights the achievements made during the Dissertation development, it's important to note that the unaccomplished items represent potential opportunities for future implementation and enhancement, as seen in Section 5.6.

6.3 Further Work

However, for the time being, the existing process has proven to be successfully developed; newer chances of development, which are more effective, can occur in the future. Precise attention to achieving higher accuracy should be a priority, especially in connection to the topics of Section 5.5. There is a necessity to discuss details like operating in different lighting situations and minimising the risk of obstacles, as mentioned in the previous section.
As I look to the future, incorporating advanced machine learning algorithms, I can anticipate even more refined systems. Extending the dataset with more diverse sets will facilitate the completion of the system by making it adaptable and robust. These issues will be elaborated on in Section 5.6 where I look at the possible directions for future improvement. In addition to this, the ability of a system under review to deal with changing gameplay scenarios in real-time should also be a notable factor, thus providing it with the needed effectiveness under multiple conditions. Continuous research and development efforts will be the key element in developing a more accurate and better Basketball player tracking system. This continuous process is important as it allows the system to be flexible enough to operate in a wide range of situations. This adaptable system can adapt and deliver services that are within the capabilities of the Basketball environment and conditions.

6.4 Main Conclusions

The main goal of this Dissertation was to develop a system that offered a practical solution catering to the needs of sports-related organisations, particularly those operating with limited financial resources. I aimed to provide a versatile tool capable of enhancing team performance through continuous data analysis in both official matches and training sessions. While the utility of this tool is evident, it remains in its early stages of development, needing further refinements and enhancements to train the algorithm and optimise its performance for the final iteration.

At the outset of this Dissertation, my knowledge and experience in the field of AI and object detection were very limited. Nevertheless, I saw this as an opportunity to expand my horizons and develop new skills. Although not all initially set goals were achieved, the experience provided a profound understanding of AI, object detection techniques, and the complexities of real-world system implementation. Throughout this Dissertation, I viewed setbacks and challenges not as signs of failure but as opportunities for personal and professional development. This Dissertation's conclusion marks a significant milestone in my academic path, and the enthusiasm gained drives my desire to advance and make future contributions to this field, should the opportunity arise.

Bibliography

[1] M. P. Matija Buric and M. Ivasic-Kos. (2019) Adapting yolo network for ball and player detection. [Online]. Available at: https://www.semanticscholar.org/paper/Adapting-YOLO-Network-for-Ball-and-Player-Detection-Buric-Pobar/c853c8e2649d02e26598aa4f825ba89000b0c95a. Last Checked: July xvii, 18

[2] S. Sánchez Hernández, H. Romero, and A. Morales, "A review: Comparison of performance metrics of pretrained models for object detection using the tensorflow framework," *IOP Conference Series: Materials Science and Engineering*, vol. 844, p. 012024, 06 2020. xvii, 18

[3] S. University. (2022) John mccarthy. [Online]. Available at: https://computerhistory.org/profile/john-mccarthy/. Last Checked: February 5

[4] D. dos Santos Gomes, "Inteligência artificial: Conceitos e aplicações," 2010. 6

[5] J. Frankenfield. (2022) Artificial intelligence: What it is and how it is used. [Online]. Available at: https://www.investopedia.com/terms/a/artificial-intelligence-AI.asp. Last Checked: October 6

[6] A. Schroer. (2022) What is artificial intelligence? [Online]. Available at: https://builtin.com/artificial-intelligence. Last Checked: November 6

[7] K. Kelley. (2023) What is artificial intelligence: Types, history, and future. [Online]. Available at: https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/what-is-artificial-intelligence. Last Checked: October 6

[8] B. Marr. What is weak (narrow) ai? here are 8 practical examples. [Online]. Available at: https://bernardmarr.com/what-is-weak-narrow-ai-here-are-8-practical-examples/. Last Checked: November 6

[9] G. Cheng, Y. Wan, A. Saudagar, K. Namuduri, and B. Buckles, "Advances in human action recognition: A survey," *ArXiv*, vol. abs/1501.05964, p. 8, 01 2015. 7

[10] M. Buchheit and B. Simpson, "Player tracking technology: Half-full or half-empty glass?" *International journal of sports physiology and performance*, vol. 12, pp. 1–23, 12 2016. 8

[11] R. Ji, "Research on basketball shooting action based on image feature extraction and machine learning," *IEEE Access*, vol. 8, no. 3, pp. 138 743–138 751, 2020. 8

[12] P. Poulios, A. Serlis, P. Groumpos, and I. Gliatis, "Artificial intelligence and data processing in injury diagnosis and prevention in competitive sports: A literature review," *MOJ Orthopedics Rheumatology*, vol. 13, pp. 34–37, 05 2021. 9

[13] S. S. Sports Analytics. (2006) Catapult. [Online]. Available at: https://www.catapult.com/pt/. Last Checked: April 9

[14] S. G. Limited. (2008) Statsports. [Online]. Available at: https://statsports.com. Last Checked: April 9

[15] S. Software. (2010) Playsight. [Online]. Available at: https://playsight.com. Last Checked: April 9

[16] Kinexon. (2012) Kinexon. [Online]. Available at: https://kinexon.com. Last Checked: April 10

[17] H.-E. Innovations. (2001) Pioneering inspiring change in sport. [Online]. Available at: https://www.hawkeyeinnovations.com. Last Checked: April 10

[18] G. Sports. (2013) The next way of seeing sports. [Online]. Available at: https://www.secondspectrum.com/index.html. Last Checked: April 10

[19] D. R. Bruce Ianni. (2013) Every stat. instantly. [Online]. Available at: https://shottracker.com. Last Checked: April 10

[20] N. Team. (2017) Imagine a new world of play. [Online]. Available at: https://www.nex.inc. Last Checked: April 10

[21] N. Slegers, D. Lee, and G. Wong, "The relationship of intra-individual release variability with distance and shooting performance in basketball," *Journal of Sports Science and Medicine*, vol. 20, pp. 508–515, 06 2021. 11

[22] R. Cañal-Bruland, L. Balch, and L. Niesert, "Judgement bias in predicting the success of one's own basketball free throws but not those of others," *Psychological research*, vol. 79, p. 12, 06 2014. 11

[23] K. Zuo and X. Su, "Three-dimensional action recognition for basketball teaching coupled with deep neural network," *Electronics*, vol. 11, p. 3797, 11 2022. 11

[24] J. Xiao, W. Tian, and L. Ding, "Basketball action recognition method of deep neural network based on dynamic residual attention mechanism," *Information*, vol. 14, p. 13, 12 2022. 11, 39

[25] K. RangaNarayana and G. Rao, "Action recognition in low resolution videos using fo-svm," *Indian Journal of Computer Science and Engineering*, vol. 12, p. 1162, 08 2021. 12, 14

[26] R. Colindres. (2023) Temporal convolutional and recurrent neural networks for sequence modeling. [Online]. Available at: https://shorturl.at/qIO29. Last Checked: June 12

[27] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2016, p. 1941. 13

[28] A. Tejero-de Pablos, Y. Nakashima, T. Sato, N. Yokoya, M. Linna, and E. Rahtu, "Summarization of user-generated sports video by using deep action recognition features," *IEEE Transactions on Multimedia*, vol. 20, no. 8, pp. 2000–2011, 2018. 13

[29] C. Wyawahare. (2020) Denoising noisy documents. [Online]. Available at: https://towardsdatascience.com/denoising-noisy-documents-6807c34730c4. Last Checked: June 14

[30] Y. Benezeth, P.-M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, "Comparative study of background subtraction algorithms," *Journal of Electronic Imaging*, vol. 19, p. 31, 07 2010. 14

[31] L. Sevilla-Lara, Y. Liao, F. Guney, V. Jampani, A. Geiger, and M. Black, "On the integration of optical flow and action recognition," *ArXiv*, p. 297, 02 2019. 14, 16

[32] Y.-H. Byeon, D. Kim, J. Lee, and K.-C. Kwak, "Body and hand–object roi-based behavior recognition using deep learning," *Sensors*, vol. 21, p. 1838, 03 2021. 14

[33] Q. Zeng, M. O. Tezcan, and J. Konrad, "Dynamic equilibrium module for action recognition," *IEEE Access*, vol. 9, no. 1, pp. 168 015–168 025, 2021. 16

[34] J. Zayas. Data in sports: Revolutionizing the way we play and watch. [Online]. Available at: https://medium.com/@curador/data-in-sports-revolutionizing-the-way-we-play-and-watch-4f4e5c9dbdc2. Last Checked: July 16

[35] S. Al-Obaidi, H. Al-Khafaji, and C. Abhayaratne, "Modeling temporal visual salience for human action recognition enabled visual anonymity preservation," *IEEE Access*, vol. 8, no. 3, pp. 213 806–213 824, 2020. 17

[36] Pjreddie. Yolo: Real-time object detection. [Online]. Available at: https://pjreddie.com/darknet/yolo/. Last Checked: January 17, 48

[37] B. Alsadik. (2019) Adjustment models in 3d geomatics and computational geophysics. [Online]. Available at: https://www.sciencedirect.com/topics/earth-and-planetary-sciences/kalman-filter. Last Checked: July 19

[38] R. Nag. (2022) A comprehensive guide to siamese neural networks. [Online]. Available at: https://medium.com/@rinkinag24/a-comprehensive-guide-to-siamese-neural-networks-3358658c0513. Last Checked: July 19

[39] E. Odemakinde. Everything about mask r-cnn: A beginner's guide. [Online]. Available at: https://viso.ai/deep-learning/mask-r-cnn/. Last Checked: July 19

[40] L. Zhou, X. Yao, and J. Zhang, "Accurate positioning siamese network for real-time object tracking," *IEEE Access*, vol. 1, p. 1, 06 2019. 20

[41] R. Abdrakhmanov, M. Elemesova, B. Zhussipbek, I. Bainazarova, T. Turymbetov, and Z. Mendibayev, "Mask r-cnn approach to real-time lane detection for autonomous vehicles," *International Journal of Advanced Computer Science and Applications*, vol. 14, p. 16, 01 2023. 20

[42] H. Wei and N. Kehtarnavaz, "Simultaneous utilization of inertial and video sensing for action detection and recognition in continuous action streams," *IEEE Sensors Journal*, vol. 20, no. 11, pp. 6055–6063, 2020. 20

[43] H. Lee, Y.-S. Kim, M. Kim, and Y. Lee, "Low-cost network scheduling of 3d-cnn processing for embedded action recognition," *IEEE Access*, vol. 9, pp. 83 901–83 912, 2021. 21

[44] M. H. B. Martin T. Hagan, Howard B. Demuth and O. D. Jesús, *Neural Network Design*, 2014. 21

[45] J. Brownlee. (2019) A gentle introduction to batch normalization for deep neural networks. [Online]. Available at: https://machinelearningmastery.com/batch-normalization-for-training-of-deep-neural-networks/. Last Checked: July 23

[46] F. Ndiritu. (2021) Dropout regularization to handle overfitting in deep learning models. [Online]. Available at: https://www.section.io/engineering-education/dropout-regularization-to-handle-overfitting-in-deep-learning-models/. Last Checked: July 23

[47] S. Sarin. (2019) Vggnet vs resnet. [Online]. Available at: https://towardsdatascience.com/vggnet-vs-resnet-924e9573ca5c. Last Checked: July 23

[48] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, p. 116, 04 1998. 23

[49] D. W. S. R. L. L. J. L. K. Deng, Jia and L. Fei Fei. (2009) Imagenet: A large-scale hierarchical image database. [Online]. Available at: https://www.image-net.org/index.php. Last Checked: September 24

[50] info@cocodataset.org. (2014) Coco dataset. [Online]. Available at: https://cocodataset.org/#home. Last Checked: July 24

[51] Google. Open images dataset v7 and extensions. [Online]. Available at: https://storage.googleapis.com/openimages/web/index.html. Last Checked: September 24, 48

[52] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, p. 76, 1 2021. 24

[53] A. Hosna, E. Merry, J. Gyalmo, Z. Alom, Z. Aung, and M. Azim, "Transfer learning: a friendly introduction," *Journal of Big Data*, vol. 9, 10 2022. 24

[54] X. Bu, "Human motion gesture recognition algorithm in video based on convolutional neural features of training images," *IEEE Access*, vol. 8, no. 2, pp. 160 025–160 039, 2020. 25

[55] Z. Ivankovic, M. Racković, and M. Ivkovic, "Automatic player position detection in basketball games," *Multimedia Tools and Applications*, vol. 72, p. 7, 10 2014. 26

[56] Y. Li and H. Wu, "A clustering method based on k-means algorithm," *Physics Procedia*, vol. 25, pp. 1104–1109, 12 2012. 26

[57] L. Kong, D. Huang, J. Qin, and Y. Wang, "A joint framework for athlete tracking and action recognition in sports videos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 2, pp. 532–548, 2020. 27

[58] T. Li, H. Chang, M. Wang, B. ni, R. Hong, and S. Yan, "Crowded scene analysis: A survey," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, 02 2015. 27

[59] D. Weinland, M. Ozuysal, and P. Fua, "Making action recognition robust to occlusions and viewpoint changes," vol. 6313, 12 2010. 27

[60] J. Xiong, L. Lu, H. Wang, J. Yang, and G. Gui, "Object-level trajectories based fine-grained action recognition in visual iot applications," *IEEE Access*, vol. 7, no. 5, pp. 103 629–103 638, 2019. 29

[61] K. Liu, Y. Li, Y. Xu, S. Liu, and S. Liu, "Spatial focus attention for fine-grained skeleton-based action tasks," *IEEE Signal Processing Letters*, vol. 29, no. 2, pp. 1883–1887, 2022. 37

[62] D. Avola, M. Cascio, L. Cinque, G. L. Foresti, C. Massaroni, and E. Rodolà, "2-d skeleton-based action recognition via two-branch stacked lstm-rnns," *IEEE Transactions on Multimedia*, vol. 22, no. 10, pp. 2481–2496, 2020. 37

[63] J. Liu, N. Akhtar, and A. Mian, "Adversarial attack on skeleton-based human action recognition," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 4, pp. 1609–1622, 2022. 37

[64] S. Sharma. (2022) Deep learning architectures for action recognition. [Online]. Available at: https://towardsdatascience.com/deep-learning-architectures-for-action-recognition-83e5061ddf90. Last Checked: January 37

[65] (2022) The future of ai in sports decision making. [Online]. Available at: https://www.thecable.ng/the-future-of-ai-in-sports-decision-making. Last Checked: December 38

[66] N. S. Max Tkachenko and S. Zhuk. (2018) Labelimg. [Online]. Available at: https://github.com/HumanSignal/labelImg. Last Checked: March 40

[67] CVAT.ai. (2018) Cvat. [Online]. Available at: https://www.cvat.ai. Last Checked: March 40

[68] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *ArXiv*, vol. abs/1804.02767, 2018. 41

[69] Pjreddie/darknet. (2018) How to train (to detect your custom objects). [Online]. Available at: https://github.com/AlexeyAB/darknet#how-to-train-to-detect-your-custom-objects. Last Checked: September 44

[70] B. A. P. Wojke, Nicolai and Dietrich. (2017) Deep_sort. [Online]. Available at: https://getimagecolor.com. Last Checked: October 48

[71] A. Kouidri. (2023) Mastering deep sort: The future of object tracking explained. [Online]. Available at: https://www.ikomia.ai/blog/deep-sort-object-tracking-guide. Last Checked: October 48

[72] Python. (2024) tkinter — python interface to tcl/tk. [Online]. Available at: https://docs.python.org/3/library/tkinter.html. Last Checked: January 63

[73] J. Chen, F. Zhu, and J. J. Little, "A two-point method for ptz camera calibration in sports," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, vol. 1, no. 1, 2018, p. 295. 75

[74] M. Nakai, Y. Tsunoda, H. Hayashi, and H. Murakoshi, "Prediction of basketball free throw shooting by openpose," in *New Frontiers in Artificial Intelligence*. Springer International Publishing, 10 2019, p. 446. 76

[75] N. ACI and M. Kuluöztürk, "Accuracy detection in some sports training using computer vision and deep learning techniques," *Bitlis Eren University Journal of Science and Technology*, vol. 13, p. 26, 12 2023. 76

[76] Python. (2019) Openpose documentation. [Online]. Available at: https://docs.python.org/3/library/tkinter. Last Checked: November 76

[77] H.-S. Fang, J. Li, H. Tang, C. Xu, H. Zhu, Y. Xiu, Y.-L. Li, and C. Lu, "Alphapose: Whole-body regional multi-person pose estimation and tracking in real-time," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 6, p. 7173, 6 2023. 77

[78] A. Kouidri. (2023) Human pose estimation with deep learning – ultimate overview in 2024. [Online]. Available at: https://viso.ai/deep-learning/pose-estimation-ultimate-overview/. Last Checked: November 77

[79] Geeksforgeeks. (2023) Posenet pose estimation. [Online]. Available at: https://www.geeksforgeeks.org/posenet-pose-estimation/. Last Checked: November 77

[80] Y. Bukschat and M. Vetter, “Efficientpose - an efficient, accurate and scalable end-to-end 6d multi object pose estimation approach,” *ArXiv*, vol. abs/2011.04307, p. 24, 11 2020. 77

[81] TensorFlow. (2023) Mnist. [Online]. Available at: https://www.tensorflow.org/datasets/catalog/mnist. Last Checked: December 78