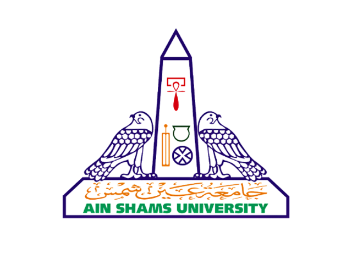
**Ain Shams University**

**Faculty of Computer & Information Sciences**

**Scientific Computing Department**

Game Glimpse

**Football Match Summarization**

**This documentation submitted as required for the degree of bachelor’s in**

**computer and Information Sciences.**

By:

|  |  |
| --- | --- |
| Ahmed Mohamed Abdel-Hamid | [Scientific Computing] |
| Ahmed Tamer El Sayed | [Scientific Computing] |
| Tarek Mohamed Saad | [Scientific Computing] |
| Ahmed Mohamed Abdel-Hady | [Scientific Computing] |
| Mazen Hamada Badr | [Scientific Computing] |
| Khaled Mahmoud Negm | [Scientific Computing] |

**Under Supervision of**

**Prof. Dr. Maryam Nabil Al-Berry**

[Lecturer], Scientific Computing Department,

Faculty of Computer and Information Sciences,

Ain Shams University.

**T.A. Heba Gamal Saleh**

[Teaching Assistant], Software Engineering Department,

Faculty of Computer and Information Sciences,

Ain Shams University.

**July 2024**

Acknowledgements

I would like to express my heartfelt gratitude to all those who have supported me throughout the journey of completing this graduation project. Without their encouragement, guidance, and support, this achievement would not have been possible.

First and foremost, I am deeply indebted to my parents for their unwavering love, patience, and encouragement. Your constant support and belief in my abilities have been a source of immense strength and motivation. I am truly grateful for the countless sacrifices you have made to ensure that I have every opportunity to succeed.

I also extend my sincere thanks to my family and friends who have been a pillar of support throughout this journey. Your understanding, encouragement, and confidence in me have been invaluable, and I am grateful for the numerous ways you have cheered me on.

A special note of thanks goes to my esteemed supervisors, **Prof. Dr. Maryam Nabil Al-Berry** and **TA. Heba Gamal Saleh**. Prof. Dr. Maryam, your insightful feedback, guidance, and expertise have been crucial in shaping this project. Your dedication to my academic growth and your willingness to always make time for my questions have been deeply appreciated. TA. Heba, your practical advice, constant support, and encouragement have played a significant role in the successful completion of this work. I am grateful for your patience and the valuable knowledge you have imparted to me.

Lastly, I would like to thank everyone who has contributed to this project in any capacity. Your support and encouragement have been instrumental in helping me achieve this milestone.

Thank you all for being a part of this journey and for helping me reach this point.

Table of Contents

[Acknowledgements I](#_Toc170202913)

[Table of Contents II](#_Toc170202914)

[List of Tables IV](#_Toc170202915)

[List of Figures V](#_Toc170202916)

[List of Abbreviations VII](#_Toc170202917)

[Abstract VIII](#_Toc170202918)

[chapter 1: Introduction 1](#_Toc170202919)

[1.1 Problem Definition 2](#_Toc170202920)

[1.2 Motivation 3](#_Toc170202921)

[1.3 Objectives 4](#_Toc170202922)

[1.4 Methodology 6](#_Toc170202923)

[1.5 Conclusion 8](#_Toc170202924)

[1.6 Time plan 9](#_Toc170202925)

[1.7 Thesis Outline 10](#_Toc170202926)

[chapter 2: Literature Review 11](#_Toc170202927)

[2.1 Analysis and comparison our study 12](#_Toc170202928)

[2.2 Background 20](#_Toc170202929)

[chapter 3: System Architecture 32](#_Toc170202930)

[3.1 System architecture Layers 33](#_Toc170202931)

[3.2 System Architecture Algorithms 38](#_Toc170202932)

[chapter 4: System Implementation 44](#_Toc170202933)

[4.1 Dataset 45](#_Toc170202934)

[4.2 Experiments & Results in model training 50](#_Toc170202935)

[chapter 5: Run the Application 53](#_Toc170202936)

[5.1 Desktop Application 54](#_Toc170202937)

[5.2 Mobile Application 55](#_Toc170202938)

[chapter 6: Conclusion and Future Work 69](#_Toc170202939)

[6.1 Conclusion 70](#_Toc170202940)

[6.2 Future Work 71](#_Toc170202941)

[chapter 7: Tools 73](#_Toc170202942)

[7.1 Languages 74](#_Toc170202943)

[7.2 Environments 74](#_Toc170202944)

[7.3 frameworks 75](#_Toc170202945)

[chapter 8: References 77](#_Toc170202946)

[8.1 References 78](#_Toc170202947)

List of Tables

[Table ‎2‑1: Survey Evaluation (F1 Score) 17](#_Toc170202948)

[Table ‎2‑2: Surveys Evaluation (AVG. MAP) 18](#_Toc170202949)

[Table ‎2‑3: Machine Learning VS Deep Learning 29](#_Toc170202950)

[Table ‎4‑1: ResNet & LSTM Evaluation Metric 51](#_Toc170202951)

[Table ‎4‑2: E2E Spot Evaluation Metric 52](#_Toc170202952)

List of Figures

[Figure ‎1‑1: Time Plan 9](#_Toc170202953)

[Figure ‎2‑1: History Of Video Summarization 12](#_Toc170202954)

[Figure ‎2‑2: Model in Abstraction 13](#_Toc170202955)

[Figure ‎2‑3: How AI work 21](#_Toc170202956)

[Figure ‎2‑4: Key Components of AI 22](#_Toc170202957)

[Figure ‎2‑5: How Does Machine Learning Work 24](#_Toc170202958)

[Figure ‎2‑6: Classification of Machine Learning 25](#_Toc170202959)

[Figure ‎2‑7: Biological Neural Networks 26](#_Toc170202960)

[Figure ‎2‑8: Artificial Neural Networks 27](#_Toc170202961)

[Figure ‎2‑9: Deep Learning Neural Network Architecture 28](#_Toc170202962)

[Figure ‎2‑10: Difference Between Machine Learning and Deep Learning 30](#_Toc170202963)

[Figure ‎3‑1: System Architecture Layers 33](#_Toc170202964)

[Figure ‎3‑2: E2E Spot Model 40](#_Toc170202965)

[Figure ‎4‑1: Dataset Sample 1 48](#_Toc170202966)

[Figure ‎4‑2: Dataset Sample 2 48](#_Toc170202967)

[Figure ‎4‑3: JSON Representation 49](#_Toc170202968)

[Figure ‎5‑1: MVVM (Model-View-ViewModel) Architecture 56](#_Toc170202969)

[Figure ‎5‑2: Design Patterns 57](#_Toc170202970)

[Figure ‎5‑3: SOLID Principles 59](#_Toc170202971)

[Figure ‎5‑4: Splash Screen 60](#_Toc170202972)

[Figure ‎5‑5: Onboarding Screens 61](#_Toc170202973)

[Figure ‎5‑6: Authentication Screen 62](#_Toc170202974)

[Figure ‎5‑7: Home Layout 63](#_Toc170202975)

[Figure ‎5‑8: Fixtures Screen 64](#_Toc170202976)

[Figure ‎5‑9: Match Details Screens 65](#_Toc170202977)

[Figure ‎5‑10: Leagues Standing Screens 66](#_Toc170202978)

[Figure ‎5‑11: Video Summary Screens 67](#_Toc170202979)

[Figure ‎5‑12: Profile Screen 68](#_Toc170202980)

List of Abbreviations

**NN:** Neural Network.

**CNN:** Convolutional Neural Network.

**RNN:** Recurrent Neural Network.

**DCNN:** Deep Convolutional Neural Network.

**NLP:** Natural language processing.

**AI:** Artificial Intelligence.

**ML:** Machine learning.

**LSTMs:** Long Short-Term Memory Networks.

**MLPs:** Multilayer Perceptrons.

**RELU:** Rectified Linear Units.

**SVM:** Support Vector Machine.

**GANs:** Generative Adversarial Networks.

**VS Code:** Visual Studio Code.

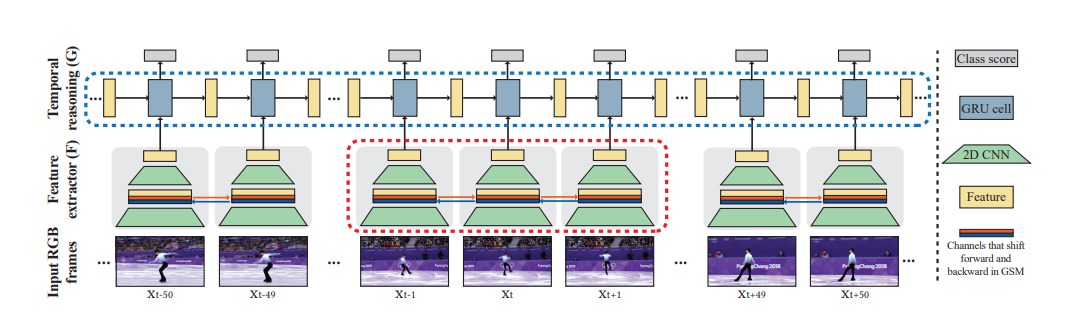
Abstract

In the fast-paced world of sports, fans often struggle to keep up with live soccer matches due to time constraints and busy schedules. To address this issue, our project presents an innovative solution: an intelligent model that automatically summarizes soccer matches, extracting key highlights to provide a comprehensive overview for those who cannot watch the full game.

This project leverages advanced machine learning techniques and deep learning techniques to analyze match footage, identify crucial events such as goals, penalties, and notable plays, and compile them into a concise summary.

Our model delivers a rich, engaging summary that mimics the experience of watching the entire match. This system aims to enhance fan engagement, offering a practical tool for sports enthusiasts to stay updated with minimal time investment. The effectiveness of the model is evaluated through a series of tests comparing the generated summaries with human-compiled highlights, demonstrating its accuracy and relevance.

Our project contributes to the growing field of sports analytics and media consumption, paving the way for more sophisticated automated summarization technologies in the future.



*Model Overview*

# Introduction

1. Problem Definition
   1. History
   2. Applications
2. Motivation
3. Objectives
4. Time plan

## Problem Definition

Soccer, being one of the most popular sports worldwide, attracts millions of viewers and fans. However, due to time constraints and busy schedules, many fans are unable to watch live matches in their entirety.

This leads to a significant demand for concise and informative summaries that capture the essential moments of a match. The challenge lies in creating a model that can automatically and accurately identify and compile these key highlights, providing a comprehensive overview of the game. The aim is to ensure that fans who cannot watch the full match still receive a rich, engaging summary that reflects the most critical aspects of the match.

### History

The concept of video summarization has evolved significantly over the past few decades, with its roots tracing back to the early development of video processing and analysis technologies. Initially, video summarization techniques were rudimentary, relying heavily on manual editing and basic algorithms to extract highlights.

The advent of machine learning and artificial intelligence has revolutionized this field, enabling the automation of highlight detection and summarization. In the context of soccer, early attempts focused on simple event detection, such as goals or fouls, based on predefined rules and basic video analysis. Over time, advancements in computer vision, natural language processing, and deep learning have paved the way for more sophisticated models capable of understanding and interpreting the complexities of a soccer match, including player movements, crowd reactions, and contextual nuances.

### Applications

The applications of soccer match video summarization are extensive and varied, benefiting both fans and industry stakeholders.  
Key applications include:

1. Fan Engagement: Automated summaries provide an accessible way for fans to stay updated on matches they cannot watch live, enhancing their overall engagement and satisfaction.
2. Media Broadcasting: Broadcasters can use summarization models to quickly generate highlight reels for news segments, social media, and post-match analyses, thereby increasing viewer engagement and content reach.
3. Coaching and Analysis: Coaches and analysts can utilize summarized footage to review key moments, strategize for future games, and conduct performance evaluations without sifting through entire match recordings.
4. Content Creation: Sports journalists and bloggers can leverage summarized content to create articles, blogs, and social media posts, providing quick and insightful match analyses to their audience.
5. Historical Archiving: Summarization aids in creating condensed archives of matches for historical reference, making it easier to analyze trends and performances over time.

## Motivation

The motivation for this project stems from the growing need for efficient and effective ways to consume soccer content in an increasingly busy world. Soccer, as the most popular sport globally, commands a vast and passionate audience. However, not all fans can dedicate the time to watch full matches due to demanding schedules and other commitments. This creates a gap between the desire to stay updated with favorite teams and players and the practical limitations of time.

Enhancing Fan Experience:

By developing an intelligent model for summarizing soccer matches, we aim to bridge this gap and enhance the overall fan experience. This project empowers fans by providing them with quick access to the most crucial moments of a match, ensuring they stay informed and connected to the sport they love. This is particularly important in the age of social media, where timely and engaging content is essential for maintaining audience interest and satisfaction.

Leveraging Technological Advancements:

Recent advancements in machine learning, computer vision, and natural language processing present a unique opportunity to create sophisticated summarization models that can deliver high-quality summaries akin to human-generated content. Harnessing these technologies allows us to push the boundaries of automated sports analytics, offering innovative solutions that were previously not feasible.

Addressing Industry Needs:

The media and sports broadcasting industries are continuously seeking ways to optimize content delivery and enhance viewer engagement. Automated summarization can significantly reduce the time and effort required to produce highlight reels and post-match analyses, leading to cost savings and increased efficiency. Moreover, this technology can be a valuable tool for coaches, analysts, and sports journalists, providing them with quick access to relevant match data and insights.

Promoting Accessibility and Inclusivity:

Our project also aims to promote accessibility and inclusivity in sports content consumption. By providing concise and informative summaries, we cater to a diverse audience, including those with limited time, people with disabilities, and casual fans who may not have the inclination to watch full matches. This ensures that soccer remains a universally enjoyable and accessible sport.

Pioneering Future Applications:

Finally, this project serves as a stepping stone towards more advanced applications in sports analytics and media consumption. The development of a robust summarization model lays the groundwork for future innovations, such as real-time summarization, personalized highlight reels, and enhanced interactive viewing experiences.

## Objectives

The primary objective of this project is to develop an intelligent model that can automatically summarize soccer matches by extracting key highlights and deploying this model in a mobile application. This application will also provide additional features such as fixtures, league standings, and match summaries. The detailed objectives of the project are as follows:

1. Develop an Intelligent Summarization Model
   1. Automated Highlight Extraction:
      * + Create a machine learning model capable of analyzing soccer match footage to identify and extract key highlights, such as goals, fouls, penalties, and significant plays.
   2. Multi-modal Analysis:
      * + Integrate audio-visual data and textual commentary to enhance the accuracy and richness of the generated summaries.
   3. Real-time Processing:
      * + Optimize the model for real-time processing to provide quick and timely summaries immediately after or even during the match.
2. Design and Implement a Mobile Application
   1. User-friendly Interface:
      * + Develop a mobile application with an intuitive and easy-to-navigate interface to ensure a seamless user experience.
   2. Fixtures and Standings Display:
      * + Include features to display upcoming fixtures, current league standings, and detailed team and player statistics.
   3. Integrated Match Summaries:
      * + Implement the summarization model within the app to provide users with concise, informative match summaries directly on their mobile devices.
3. Ensure High-Quality Summaries
   1. Accuracy and Relevance:
      * + Continuously evaluate and refine the summarization model to ensure that the highlights are accurate, relevant, and reflective of the most critical aspects of the match.
   2. User Feedback Integration:
      * + Implement mechanisms for users to provide feedback on the summaries, using this feedback to further enhance the model’s performance.
4. Deployment and Maintenance
   1. Scalability:
      * + Design the system architecture to handle a growing user base and increasing data volume efficiently.
   2. Continuous Improvement:
      * + Establish a robust maintenance and update process to keep the application and summarization model up-to-date with the latest advancements and user needs.

By achieving these objectives, the project aims to deliver a comprehensive and engaging mobile application that provides soccer fans with quick and insightful access to match highlights, fixtures, and league standings, enhancing their overall experience and connection to the sport.

## Methodology

1. Data Collection and Preprocessing
   1. Match Footage Acquisition:
      * + Collect a diverse dataset of soccer match videos from various leagues and competitions.
        + Ensure the dataset includes a range of match scenarios, including goals, fouls, penalties, and other significant events.
   2. Annotation:
      * + Manually annotate the collected videos with timestamps and descriptions of key events.
   3. Data Augmentation:
      * + Apply data augmentation techniques to enhance the dataset's diversity and improve the model's robustness.
2. Model Development
   1. Feature Extraction:
      * + Use computer vision techniques to extract visual features from video frames.
        + Utilize natural language processing to analyze and extract key information from textual commentary.
   2. Event Detection:
      * + Develop and train machine learning models (e.g., convolutional neural networks, recurrent neural networks) to detect and classify key events in the match footage.
   3. Highlight Generation:
      * + Implement algorithms to compile detected events into a coherent and concise match summary.
        + Ensure the summarization captures the flow and critical moments of the match.
3. Mobile Application Development
   1. Design and User Interface:
      * + Design an intuitive and user-friendly interface for the mobile application.
        + Ensure the app provides easy access to fixtures, league standings, and match summaries.
   2. Integration:
      * + Integrate the summarization model into the mobile application.
        + Develop APIs to facilitate data exchange between the model and the app.
4. Evaluation and Testing
   1. Model Evaluation:
      * + Test the summarization model on a separate validation dataset to assess its accuracy and relevance.
        + Use metrics such as precision, recall, and F1-score to evaluate model performance.
   2. User Testing:
      * + Conduct user testing to gather feedback on the app's usability, interface, and summarization quality.
        + Iterate on the app design and functionality based on user feedback.
5. Deployment and Maintenance
   1. Deployment:
      * + Deploy the mobile application on iOS and Android platforms.
        + Ensure the app is scalable to handle a growing user base.
   2. Monitoring and Updates:
      * + Continuously monitor the app's performance and user engagement.
        + Implement regular updates to improve features, fix bugs, and incorporate new advancements in summarization technology.
6. Continuous Improvement
   1. Research and Development:
      * + Stay abreast of the latest developments in machine learning, computer vision, and natural language processing.
        + Incorporate new techniques and innovations to continuously improve the summarization model's performance and accuracy.

By following this methodology, we aim to develop a robust and user-friendly mobile application that provides high-quality soccer match summaries, fixtures, and league standings, enhancing the overall experience for soccer fans worldwide.

## Conclusion

This project aims to address the challenge of delivering concise and informative soccer match summaries for fans with limited time. By developing an intelligent summarization model and integrating it into a mobile application, we provide an innovative solution that enhances fan engagement and accessibility.

The application not only offers match highlights but also includes features like fixtures and league standings, ensuring a comprehensive user experience.

The successful implementation of this project will bridge the gap between fans and the sport, allowing them to stay updated and connected despite their busy schedules. Additionally, this project lays the groundwork for future advancements in sports analytics and media consumption, demonstrating the potential of artificial intelligence in transforming how sports content is delivered and consumed. Ultimately, our solution aims to make soccer more enjoyable and accessible for a global audience, fostering greater engagement and appreciation for the sport.

## Time plan

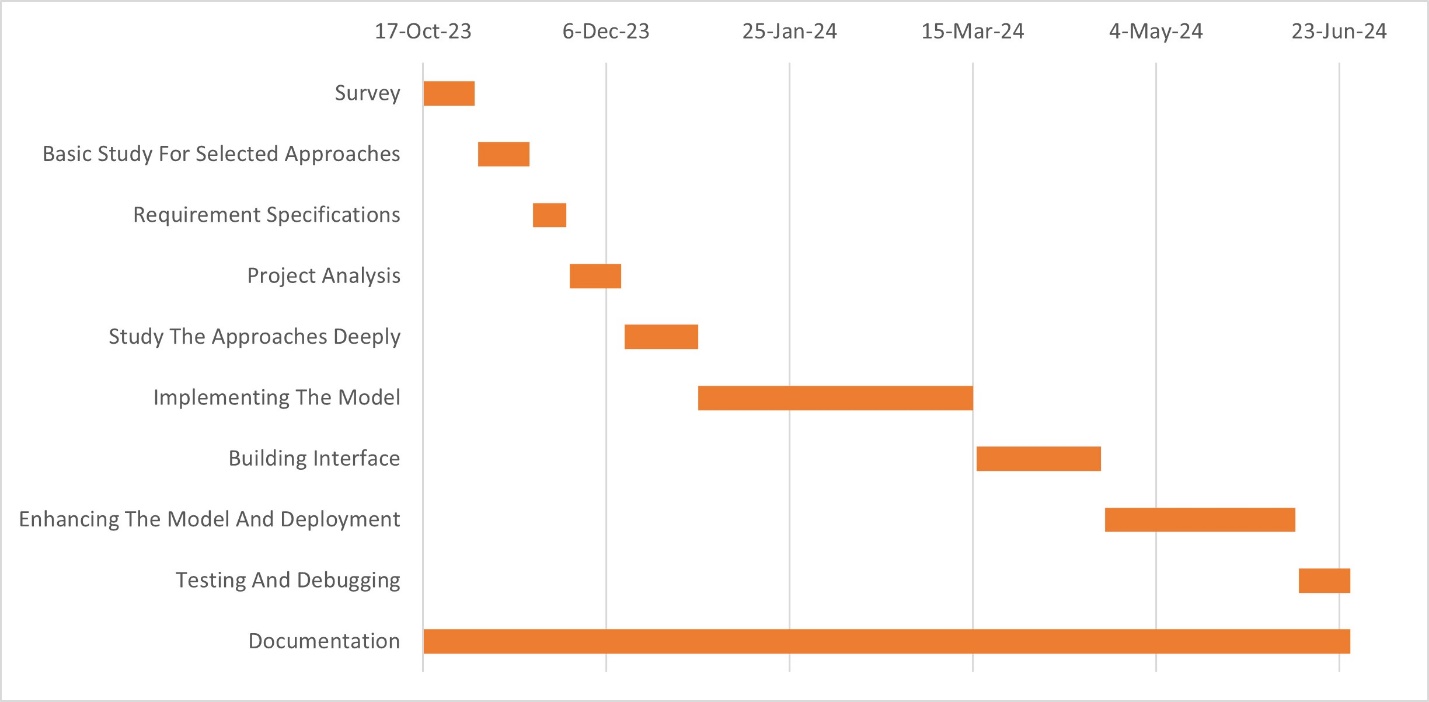


Figure ‎1‑1: Time Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Tasks** | **Start Date** | **End Date** | **Duration** |
| Survey | 10/17/2023 | 10/31/2023 | 2 Weeks |
| Basic Study For Selected Approaches | 11/1/2023 | 11/15/2023 | 2 Weeks |
| Requirement Specifications | 11/16/2023 | 11/25/2023 | 9 Days |
| Project Analysis | 11/26/2023 | 12/10/2023 | 2 Weeks |
| Study The Approaches Deeply | 12/11/2023 | 12/31/2023 | 20 Days |
| Implementing The Model | 12/31/2023 | 3/15/2024 | 2 Month & 2 Weeks |
| Building Interface | 3/16/2024 | 4/19/2024 | 1 Month & 4 Days |
| Enhancing The Model And Deployment | 4/20/2024 | 6/11/2024 | 1 Month & 3 Weeks |
| Testing And Debugging | 6/12/2024 | 6/26/2024 | 2 Weeks |
| Documentation | 10/17/2023 | 6/26/2024 | 253 Days |

## Thesis Outline

Chapter 2: Literature Review

This section covers the project area of deep learning and deepfake, exploring its scientific foundations, analysis, and research. The goal is to offer a detailed and complete overview of the field, encompassing its history, evolution, and current status. Furthermore, the section includes results from surveys and studies on the topic, emphasizing significant discoveries and insights.

Chapter 3: System Architecture and Methods

This section concentrates on outlining the system architecture and its primary modules. It seeks to give readers a comprehensive understanding of the project's technical components. Furthermore, it will also provide a detailed description of the methods and procedures employed in the development and implementation of the system.

Chapter 4: System Implementation and Results

This section will provide a detailed description of the dataset used in the project and the software tools employed. Additionally, it will cover the configuration steps involved in the application, the design of experiments, and the results obtained. The aim is to give readers a comprehensive understanding of how the dataset was utilized in the project and how the results were achieved. This section will offer a complete overview of the technical processes involved, helping readers understand the methodology used in the project development.

Chapter 5: Run the Application

This section provides a step-by-step guide for users to learn how to use the desktop application, along with the user manual for the mobile application.

Chapter 6: Conclusion and Future work

The section provides a comprehensive summary of the project, including the obtained results. It will also offer insights into potential improvements for the project's performance and suggest additional features or functions to enhance its capabilities.

Chapter 7: Tools

This section offers a detailed overview of the tools, programming languages, development environments, and frameworks used in the project. It will explain how each component was utilized, providing insights into their roles and contributions to the project's development.

# Literature Review

1. Analysis and comparison our study
   1. History Of Video Summarization
   2. Surveys
2. Background

## Analysis and comparison our study

We conduct surveys to save time, as they are cost-effective and beneficial for our project. Surveys typically incur a minimal cost per participant while yielding a large number of potential responses. Collecting results can be done swiftly, and one method to guarantee the success of our project is by applying insights gained from surveys.

By conducting surveys to analyze the most effective techniques, we can ensure the accuracy of the information we gather. With this valuable data at our disposal, we can make well-informed decisions on how to advance our project, ultimately achieving the best possible outcomes.

### History Of Video Summarization



Figure ‎2‑1: History Of Video Summarization

1. Early Development (1990s):
   * + - The concept of video summarization emerged in the 1990s alongside advancements in multimedia processing and digital video technologies.
       - Initial efforts focused on extracting key frames from videos to create static summaries, mainly for archival and retrieval purposes.
2. 2000s:
   * + - With the increase in video content, more sophisticated methods, including shot boundary detection and keyframe extraction, were developed.
       - Researchers began incorporating audio and text data to enhance summaries, resulting in more informative and contextually relevant summaries.
       - Machine learning algorithms started being used to improve the accuracy and relevance of video summaries.
3. 2010s:
   * + - The rise of deep learning significantly impacted video summarization techniques, allowing for more complex and accurate models.
       - Techniques such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) were utilized to create dynamic and adaptive summaries.
       - User-generated content platforms like YouTube and social media increased the demand for automated video summarization to handle large volumes of video data.
4. Recent Advances (2020s):
   * + - Advanced deep learning models, including transformers and attention mechanisms, have further improved the quality and efficiency of video summarization.
       - Real-time summarization and personalized summaries have become areas of active research, focusing on creating summaries tailored to individual user preferences.
       - Applications have expanded to various fields, including sports, entertainment, surveillance, and education, reflecting the broad utility of video summarization technology.

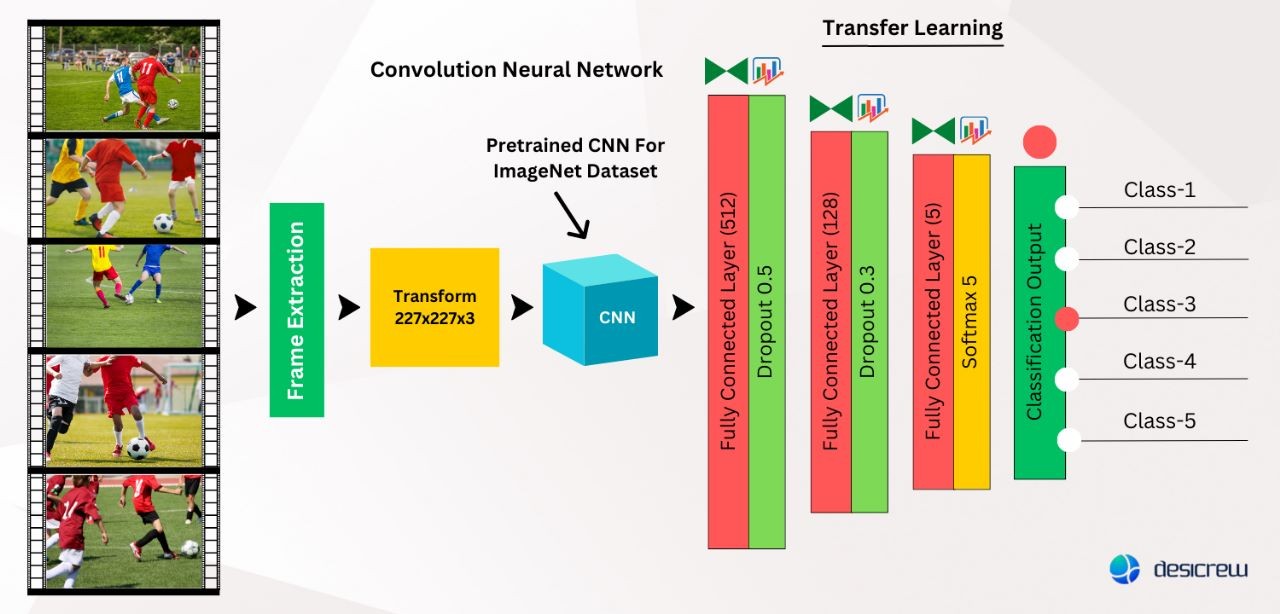


Figure ‎2‑2: Model in Abstraction

### Surveys

The paper "E2E Spot" utilizes the RegNet Y architecture combined with GRU (Gated Recurrent Unit) on the SoccerNet v2 dataset, achieving an average MAP (Mean Average Precision) of 82.7%. This methodology enhances end-to-end event spotting in soccer videos by effectively integrating deep learning techniques to accurately detect and classify events throughout matches as shown in Table ‎2‑2 [1].

The paper "Flexible Automatic Football Filming and Summarization" utilizes Convolutional Neural Networks (CNNs) and a self-made dataset to automatically film and summarize football matches, achieving an F1 score of 79.18%. This system tracks key events and creates concise highlight reels to enhance the viewer experience as shown in Table ‎2‑1 [1].

The paper "Slicing and Dicing Soccer: Automatic Detection of Complex Events" employs Interval Temporal Logics (ITL) and the SoccER dataset to automatically detect complex soccer events, achieving an F1 score of 94%. This methodology enhances the precision of event detection, contributing to more accurate and informative soccer analysis as shown in Table ‎2‑1 [2].

The paper "Machine Learning-Based Soccer Video Summarization System" utilizes SVM, Gabor filters, and Hough transform methodologies on YouTube videos, achieving an F1 score of 99%. This approach provides highly accurate soccer video summarization by effectively identifying and highlighting key events as shown in Table ‎2‑1 [3].

The paper "Deep Learning Based Automated Sports Video Summarization using YOLO" employs YOLO (You Only Look Once) with Optical Character Recognition (OCR) on a self-made dataset, achieving an impressive F1 score of 99.8%. This methodology excels in accurately summarizing sports videos by detecting and recognizing key visual and textual elements, enhancing the efficiency and effectiveness of video content analysis as shown in Table ‎2‑1 [4].

The paper "Dense Anchors Revisited" utilizes Dense Anchors (DU), Spatial Attention Module (SAM), mixup technique, and Soft-NMS on the SoccerNet-v2 dataset, achieving an average MAP (mean Average Precision) of 79%. This methodology enhances object detection accuracy in soccer videos, providing robust evaluation metrics for effective video analysis and summarization as shown in Table ‎2‑2 [2].

The paper "COMEDIAN" employs a Spatio-Temporal Transformer combined with knowledge distillation and self-supervised learning techniques on the SoccerNet-v2 dataset, achieving an average MAP (Mean Average Precision) of 77.6%. This methodology enhances video analysis by effectively capturing spatio-temporal features in soccer videos, leading to improved performance in event detection and summarization tasks as shown in Table ‎2‑2 [3].

The paper "A Context-Aware Loss Function for Action Spotting in Soccer Videos" utilizes CALF (Context-Aware Loss Function) combined with YOLO (You Only Look Once) on the SoccerNet-v2 dataset, achieving an average MAP (Mean Average Precision) of 40.7%. This methodology focuses on enhancing action spotting accuracy in soccer videos by leveraging contextual information and efficient object detection techniques as shown in Table ‎2‑2 [4].

The paper "SoccerNet-v2: A Dataset and Benchmarks for Holistic Understanding of Broadcast Soccer Videos" introduces the AudioVid methodology applied to the SoccerNet-v2 dataset, achieving an average MAP (Mean Average Precision) of 39.9%. This comprehensive approach aims to enhance the understanding of broadcast soccer videos by integrating audio-visual analysis techniques for improved video content understanding and event detection as shown in Table ‎2‑2 [5].

The paper "SoccerNet-v2: A Dataset and Benchmarks for Holistic Understanding of Broadcast Soccer Videos" utilizes the NetVLAD methodology on the SoccerNet-v2 dataset, achieving an average MAP (Mean Average Precision) of 77.6%. This approach enhances the understanding of broadcast soccer videos by effectively capturing and encoding visual features for improved video analysis and event detection as shown in Table ‎2‑2 [6].

The paper "RMS-Net: Regression and Masking for Soccer Event Spotting" employs the RMS-Net methodology on the SoccerNet dataset, achieving an average MAP (Mean Average Precision) of 75.1%. This approach enhances soccer event spotting by integrating regression and masking techniques, improving the accuracy and efficiency of event detection in soccer videos as shown in Table ‎2‑2 [7].

The paper "Spotting Football Events Using Two-Stream Convolutional Neural Network and Dilated Recurrent Neural Network" utilizes a combination of Two-stream CNN (Convolutional Neural Network) and Dilated RNN (Recurrent Neural Network) methodologies on the SoccerNet dataset, achieving an average MAP (Mean Average Precision) of 63.3%. This approach enhances the detection and spotting of football events in videos by effectively capturing both spatial and temporal dependencies, thereby improving the overall accuracy of event recognition in soccer matches as shown in Table ‎2‑2 [8].

The paper "Event Detection in Coarsely Annotated Sports Videos via Parallel Multi Receptive Field 1D Convolutions" employs the Multi-tower CNN methodology on the SoccerNet dataset, achieving an average MAP (Mean Average Precision) of 60.1%. This approach enhances event detection in sports videos by utilizing parallel multi receptive field 1D convolutions, improving the ability to accurately identify and classify events in dynamically changing contexts within soccer matches as shown in Table ‎2‑2 [9].

The paper "Real-Time Detection of Events in Soccer Videos using 3D Convolutional Neural Networks" utilizes 3D CNN methodology on the SoccerNet dataset, achieving an average MAP (Mean Average Precision) of 32. This approach focuses on real-time event detection in soccer videos by leveraging spatio-temporal features, aiming to accurately identify and classify events as they occur during matches as shown in Table ‎2‑2 [10].

The paper "SSET: A Dataset for Shot Segmentation, Event Detection, and Player Tracking in Soccer Videos" utilizes a combination of VGG, GoogLeNet, ResNet, and C3D methodologies on the SSET dataset, achieving an average MAP (Mean Average Precision) of 28.57%. This comprehensive approach enhances shot segmentation, event detection, and player tracking in soccer videos by leveraging deep learning architectures to analyze and classify complex visual and spatio-temporal data as shown in Table ‎2‑2 [11].

The paper "Soccer Video Summarization using Deep Learning" employs the 3D ResNet-LSTM methodology on the Soccer-5 dataset, achieving an evaluation metric of 96.81% accuracy. This approach enhances soccer video summarization by effectively capturing spatio-temporal features and sequences, ensuring precise and informative summaries of soccer matches as shown in Table ‎2‑1[5].

Table ‎2‑1: Survey Evaluation (F1 Score)

|  |  |  |  |
| --- | --- | --- | --- |
| **PAPER** | **METHODOLGY** | **DATASET** | **EVALUATION (F1 Score)** |
| Flexible Automatic Football Filming and Summarization [1] | CNN | Self-Made Dataset | 79.18 |
| Slicing and dicing soccer: automatic detection of complex events [2] | Interval Temporal Logics (ITL) | SoccER | 94 |
| Machine Learning-Based Soccer Video Summarization System [3] | SVM + Gabor ﬁlter and Hough transform | Youtube Videos | 99 |
| Deep Learning Based Automated Sports Video Summarization using YOLO [4] | YOLO + OCR | Self-Made Dataset | 99.8 |
| Soccer  Video Summarization using Deep Learning [5] | 3D ResNet - LSTM | Soccer-5 | 96.81 |

Table ‎2‑2: Surveys Evaluation (AVG. MAP)

|  |  |  |  |
| --- | --- | --- | --- |
| **PAPER** | **METHODOLGY** | **DATASET** | **EVALUATION (AVG. MAP)** |
| E2E Spot Action [1] | RegNet Y + GRU | SoccerNet v2 | 82.7 |
| Dense Anchors Revisited [2] | DU + SAM + mixup + Soft-NMS | SoccerNet-v2 | 79 |
| COMEDIAN [3] | Spatio-Temporal Transformer + knowledge Distillation + Self-Supervised Learning | SoccerNet-v2 | 77.6 |
| A Context-Aware Loss Function for Action Spotting in Soccer Videos [4] | CALF (Context Aware Loss Function) + YOLO | SoccerNet-v2 | 40.7 |
| SoccerNet-v2: A Dataset and Benchmarks for Holistic Understanding of Broadcast Soccer Videos [5] | AudioVid | SoccerNet-v2 | 39.9 |
| SoccerNet-v2: A Dataset and Benchmarks for Holistic Understanding of Broadcast Soccer Videos [6] | NetVLAD | SoccerNet-v2 | 77.6 |
| RMS-Net: Regression and Masking for Soccer Event Spotting [7] | RMS + Net | SoccerNet | 75.1 |
| Spotting Football Events Using Two-Stream Convolutional Neural Network and Dilated Recurrent Neural Network [8] | Two-stream CNN + Dilated RNN | SoccerNet | 63.3 |
| Event detection in coarsely annotated sports videos via parallel multi receptive field 1D convolutions [9] | Multi-tower CNN | SoccerNet | 60.1 |
| Real-Time Detection of Events in Soccer Videosusing 3D Convolutional Neural Networks [10] | 3D CNN | SoccerNet | 32 |
| SSET: a dataset for shot segmentation, event detection, player tracking in soccer videos [11] | VGG + GoogLeNet + ResNet + C3D | SSET | 28.57 |

## Background

### Artificial Intelligence:

#### What Is Artificial Intelligence (AI)?

Artificial intelligence (AI) involves creating machines that can perform tasks and make decisions in ways that mimic human thought processes. This includes programming machines to learn from experience and solve problems, essentially replicating human cognitive abilities.

#### Applications for artificial

Artificial intelligence has limitless applications across various sectors and industries. In healthcare, AI is currently being trialed and utilized for administering precise drug dosages and delivering customized treatments to individual patients. It is also assisting in surgical procedures within operating rooms.

#### Types of Artificial Intelligence

Artificial intelligence is categorized into two main types: narrow and general. Narrow AI is designed for specific tasks, exemplified by video games like chess and personal assistants such as Amazon's Alexa and Apple's Siri, which respond to user queries effectively.

On the other hand, general AI, also known as strong AI, performs tasks that are typically human-like and complex. These systems are capable of problem-solving independently, such as in self-driving cars or in surgical operations within hospitals.

#### How does AI work?

Initially, an AI system receives input data in forms such as speech, text, or images. It proceeds by applying rules and algorithms to process the data, interpreting, predicting, and taking action based on the input. Following processing, the system generates an outcome whether it indicates success or failure in handling the input data. The results are then evaluated through analysis, uncovering insights, and receiving feedback.

Finally, the system utilizes these assessments to refine its input data, rules, algorithms, and desired outcomes, adapting and improving over time.

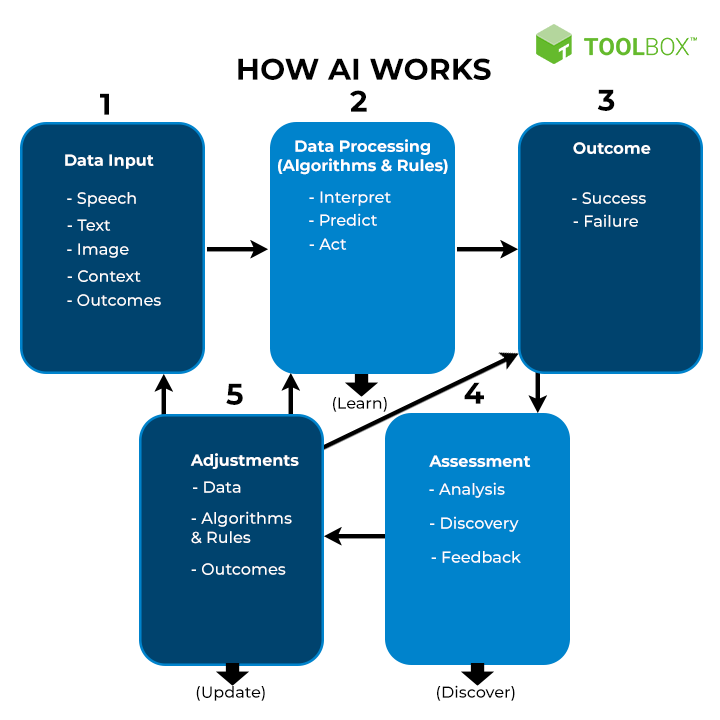


Figure ‎2‑3: How AI work

#### Key Components of AI

Artificial Intelligence (AI) comprises several key components, including:

* + - Machine Learning:
      * Algorithms that enable systems to learn and improve from experience without explicit programming.
    - Deep Learning:
      * A subset of machine learning that uses neural networks with many layers (deep neural networks) to learn hierarchical representations of data.
    - Neural Networks:
      * Computational models inspired by the structure and function of the human brain, used in various AI applications for pattern recognition and prediction.
    - Cognitive Computing:
      * Systems designed to simulate human thought processes, often incorporating elements of AI, machine learning, natural language processing, and more.
    - Natural Language Processing (NLP):
      * AI techniques that enable computers to understand, interpret, and generate human language, facilitating interactions between computers and humans via natural language.
    - Computer Vision:
      * AI technology enabling computers to interpret and understand visual information from the world, including image and video analysis, object recognition, and scene understanding.

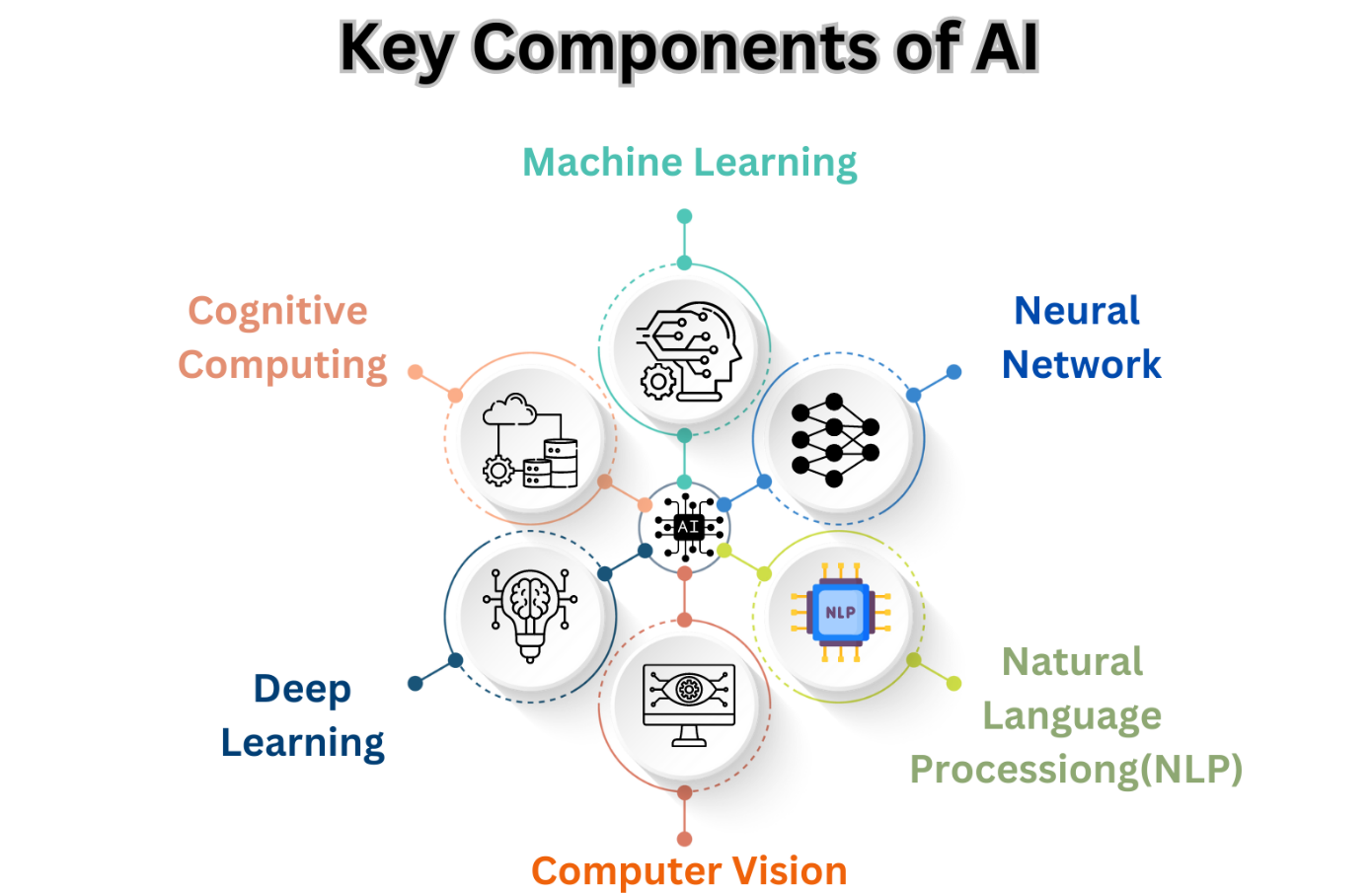


Figure ‎2‑4: Key Components of AI

### Machine Learning:

#### What Is Machine Learning?

Machine learning, a branch of artificial intelligence (AI), empowers systems to enhance their performance through experience without direct programming. It revolves around creating algorithms that can access and utilize data for autonomous learning. As new data is introduced, machine learning applications can autonomously learn, evolve, develop, and adjust independently of explicit programming. This capability enables computers to operate more intelligently over time as they encounter new information and scenarios.

#### How does machine learning work?

Machine learning works by enabling computers to learn from data and make decisions or predictions based on that learning, without being explicitly programmed for every possible scenario. Here’s a simplified explanation of how machine learning typically works:

* Data Collection:
  + - The first step involves gathering relevant data that is representative of the problem or task the machine learning model aims to solve. This data can include text, images, numerical values, etc.
* Data Preprocessing:
  + - Once collected, the data needs to be cleaned and prepared for analysis. This step involves tasks such as handling missing values, scaling numerical features, encoding categorical variables, and splitting the data into training and testing sets.
* Feature Extraction and Selection:
  + - In this step, relevant features (or attributes) from the data are extracted. This can involve techniques to reduce the dimensionality of the data or transform it into a more suitable format for the model.
* Choosing a Model:
  + - Machine learning models come in various types (e.g., decision trees, neural networks, support vector machines). The choice of model depends on the specific problem and data characteristics.
* Training the Model:
  + - Training involves feeding the prepared data into the chosen model. During training, the model learns the patterns and relationships in the data by adjusting its parameters iteratively to minimize the difference between predicted and actual outcomes.
* Evaluation:
  + - Once trained, the model is evaluated using a separate set of data (test data) that it hasn’t seen before. This step assesses how well the model generalizes to new data and performs the intended task (e.g., making predictions).
* Model Optimization and Tuning:
  + - If the model's performance is not satisfactory, parameters can be fine-tuned or different models can be tested to improve performance.

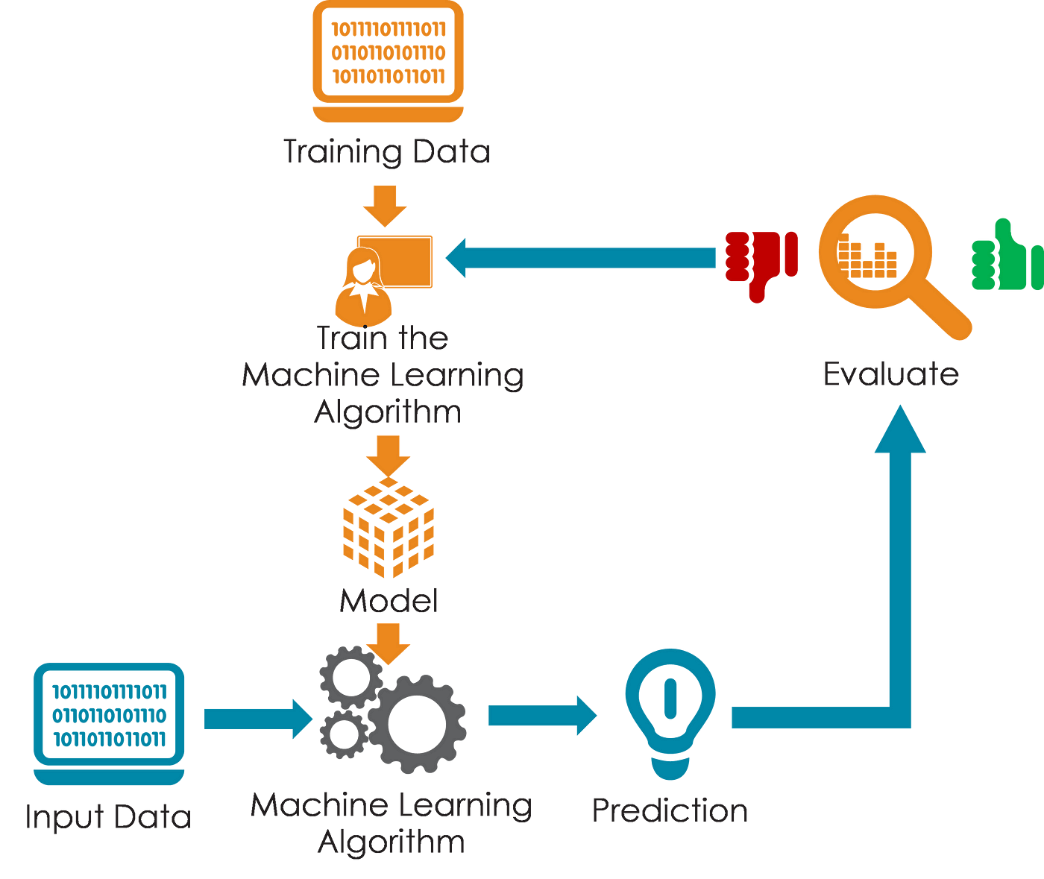


Figure ‎2‑5: How Does Machine Learning Work

#### Classification of Machine Learning

The key components of Machine Learning classification are:

* Supervised Learning:
  + - * A type of machine learning where the model is trained on labeled data, and it learns to map input data to the desired output. The goal is to predict the output for new, unseen data based on the learned patterns.
* Unsupervised Learning:
  + - * In this type of machine learning, the model is trained on unlabeled data and learns patterns and structures from the input data without specific output labels. Common tasks include clustering and dimensionality reduction.
* Reinforcement Learning:
  + - * An area of machine learning where an agent learns to make decisions by interacting with an environment. The agent receives feedback in the form of rewards or penalties as it navigates through a problem, aiming to maximize cumulative rewards over time.

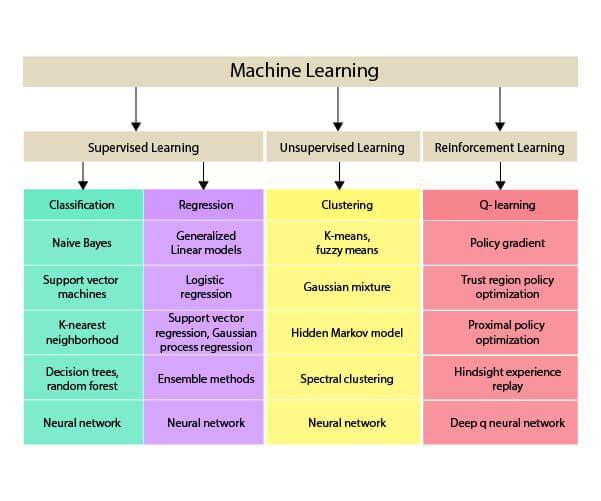


Figure ‎2‑6: Classification of Machine Learning

### Deep Learning

#### What Is Deep Learning?

Deep learning is a subset of machine learning that focuses on algorithms inspired by the structure and function of the human brain, specifically artificial neural networks with multiple layers (hence the term "deep"). It aims to learn hierarchical representations of data by using multiple layers of non-linear processing units or neurons.

Deep learning works on multiple neural networks of three or more layers and attempts to simulate the behavior of the human brain. It allows statisticians to learn from large amounts of data and interpret trends

#### How Do Deep Learning Neural Networks Work?

* BIOLOGICAL NEURAL NETWORKS:

Artificial neural networks draw inspiration from the complex networks of biological neurons found in the human brain. While simplified, they mimic some fundamental aspects of biological neural networks. To better understand artificial neural networks, it's insightful to examine the structure and function of their biological counterparts.

Biological neural networks are composed of vast numbers of neurons. Each neuron consists of essential components: a cell body, dendrites (which receive signals), and an axon (which transmits signals). Signals propagate through these neurons via electrochemical processes, where synaptic connections play a crucial role. When a neuron's membrane voltage reaches a certain threshold, it generates an action potential—an electrochemical pulse that travels swiftly along the axon, activating synaptic connections

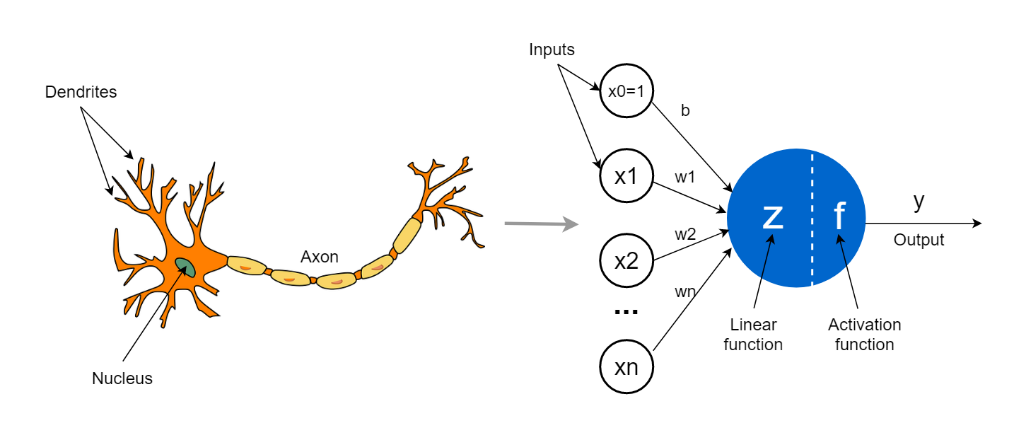


Figure ‎2‑7: Biological Neural Networks

* ARTIFICIAL NEURAL NETWORKS:

Having grasped the fundamental workings of biological neural networks, let's delve into the structure of artificial neural networks. Artificial neural networks are typically comprised of interconnected units or nodes. These nodes, referred to as neurons, mimic, albeit in a simplified manner, the biological neurons found in the human brain.

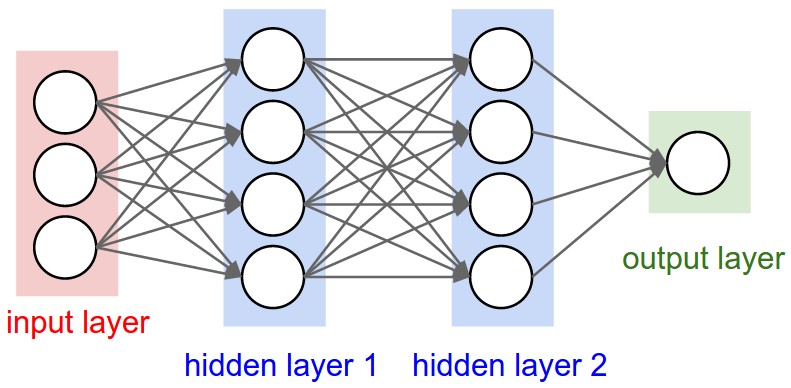


Figure ‎2‑8: Artificial Neural Networks

In the context of artificial neural networks, a neuron represents a numeric value in graphical form. Analogous to biological axons, connections between artificial neurons are established using numerical values known as weights. During the learning process of an artificial neural network, these weights adjust to strengthen or weaken connections between neurons.

When training an artificial neural network, the objective is to find optimal weights that enable the network to perform specific tasks, such as classifying numbers based on provided training data. These optimal weights are unique to each task and dataset, and their values cannot be predicted beforehand. Therefore, the neural network learns these weights through a process known as training.

#### Deep Learning Neural Network Architecture

In a typical neural network architecture, there are several layers, beginning with the input layer and ending with the output layer.

* Input Layer:

The input layer receives the input data 𝑥, which in the case of classifying handwritten numbers, represents images where each entry in 𝑥 corresponds to a pixel in the image.

* Hidden Layers:

Between the input and output layers are one or more hidden layers. These layers perform mathematical operations on the input data to extract and learn features that are useful for making predictions or classifications. The number of neurons in each hidden layer and the number of hidden layers themselves can vary depending on the complexity of the problem and the architecture chosen.

* Output Layer:

The final layer is the output layer, which produces a vector 𝑦 representing the neural network’s output. For example, in a classification task of handwritten numbers, each neuron in the output layer might represent a different class (e.g., digits 0-9), and the values of these neurons indicate the network's confidence or probability for each class.

* Connections Between Layers:

The connections between neurons in adjacent layers are established through weights. Each connection has a weight associated with it, which determines the strength of influence one neuron has on another. During the training process, these weights are adjusted based on the error between predicted and actual outputs, allowing the network to improve its performance over time.

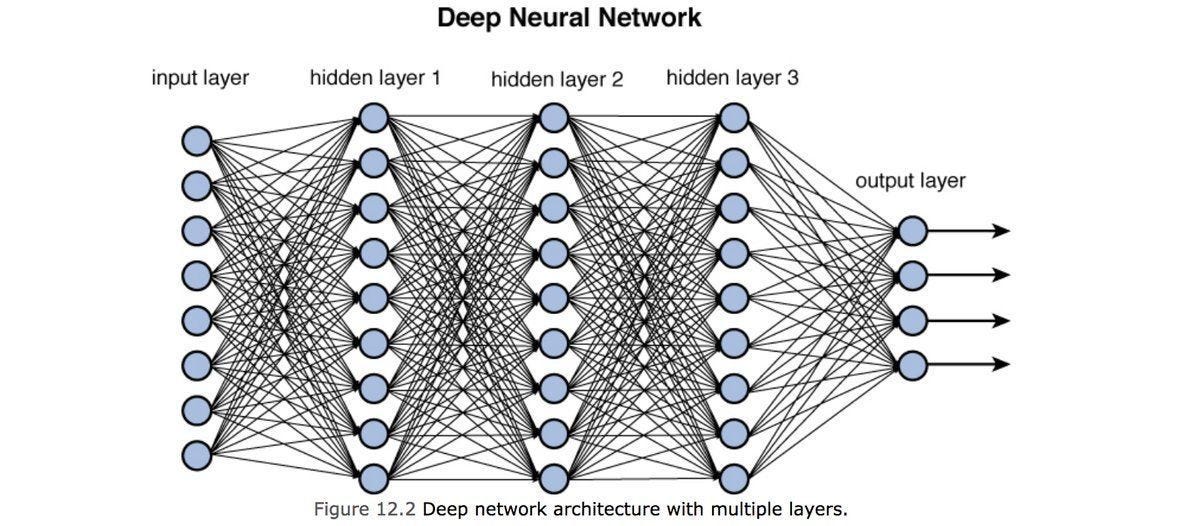


Figure ‎2‑9: Deep Learning Neural Network Architecture

#### Difference Between Machine Learning and Deep Learning?

Table ‎2‑3: Machine Learning VS Deep Learning

|  |  |  |
| --- | --- | --- |
|  | **Machine Learning** | **Deep Learning** |
| **Definition** | Machine learning involves algorithms that enable computers to learn from data and make decisions or predictions without being explicitly programmed for specific tasks. | Deep learning is a specialized subset of machine learning that uses neural networks with many layers (deep neural networks) to automatically learn hierarchical representations of data. |
| **Algorithms** | Common algorithms include decision trees, support vector machines, k-nearest neighbors, and linear regression. | Primarily involves neural networks, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and deep belief networks (DBNs). |
| **Feature Engineering** | Machine learning often requires manual feature extraction and selection, where domain experts identify relevant features from raw data. | Deep learning models automatically learn features from raw data, reducing the need for manual feature engineering. |
| **Model Complexity** | Models can be relatively simple and less computationally intensive. | Models are highly complex and computationally intensive, often requiring specialized hardware such as GPUs or TPUs for efficient training. |
| **Data Requirements** | Can work effectively with smaller datasets, although performance improves with more data. | Requires large amounts of data to achieve high performance due to the many parameters in deep neural networks. |
| **Applications** | Includes a wide range of applications such as spam filtering, recommendation systems, predictive maintenance, and financial forecasting. | Excels in tasks such as image and speech recognition, natural language processing, autonomous driving, and game playing, where high-dimensional data and complex patterns are involved. |

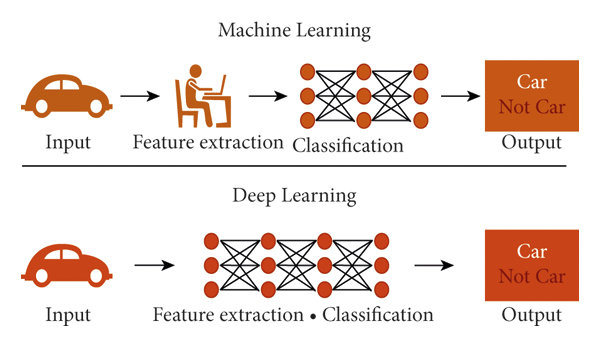


Figure ‎2‑10: Difference Between Machine Learning and Deep Learning

# System Architecture

1. System Architecture Layers
2. System Architecture Algorithms

## System architecture Layers

A diagram of a software application

Description automatically generated

Figure ‎3‑1: System Architecture Layers

System architecture plays an important role in defining the structure and behavior of our project. It helps us to better understand how the project works and ensures that it functions as intended throughout its lifecycle. This includes defining the relationships between components, data flows, service compositions, and subsystems. by designing an effective system architecture that prioritizes scalability, and interoperability, among other objectives, and helps us to have greater confidence in the system's ability to meet its requirements and objectives [36].

System architecture has five layers: Application layer, Logic layer, E2E Video Summarizer layer, database layer and API layer as shown figure 3.1, we explain more details about each layer in the following section.

### Application Layer

The Android application provides users with comprehensive access to football match information. Users can easily register using their email and password or directly through their Gmail account. Once registered, they can log in to unlock a variety of features. These include live match results as they happen, detailed outcomes of past matches, and timely updates on upcoming matches.

Users can also view league standings to track their favorite teams' positions. The standout feature of our application is its ability to deliver match summaries and key actions immediately after each match concludes. This ensures that users stay informed about all the important moments and developments across various football games. Whether it's goals, penalties, or other significant events, our app keeps fans updated in real-time.

In essence, our application serves as an indispensable tool for football enthusiasts, offering not only live updates and historical data but also ensuring they never miss a beat when it comes to their favorite sport.

### Logic Layer

Firebase Authentication is a service provided by Google Firebase that simplifies the process of adding authentication and authorization to web and mobile applications. It supports multiple authentication methods, including email and password, phone authentication, and integration with popular federated identity providers like Google, Facebook, Twitter, and GitHub.

Key Strengths of Firebase Authentication:

* Ease of Integration:
  + - Simple Setup: Firebase Authentication is easy to integrate with existing applications, providing ready-to-use SDKs for iOS, Android, and web platforms.
    - Unified Authentication: It offers a unified authentication model, making it straightforward to manage different types of authentication methods within a single system.
* Multiple Authentication Methods:
  + - Diverse Options: Supports various authentication methods, including email/password, phone number, and third-party providers (Google, Facebook, Twitter, GitHub).
    - Anonymous Authentication: Allows users to start using the app without sign-up and later upgrade to a fully authenticated account.
* Security:
  + - Secure Tokens: Utilizes secure token-based authentication mechanisms, reducing the risk of common security issues like session hijacking.
    - Compliance: Meets industry standards for security and privacy, making it suitable for handling sensitive user data.
* User Management:
  + - Comprehensive User Management: Provides a comprehensive set of tools for managing users, including account creation, password reset, and email verification.
    - Admin SDK: Enables developers to manage users and authentication programmatically via the Admin SDK.
* Scalability:
  + - Handles Growth: Designed to scale effortlessly with the growth of your application, handling a large number of users without significant performance issues.
* Integration with Firebase Ecosystem:
  + - Seamless Integration: Integrates seamlessly with other Firebase services, such as Firestore for real-time database needs, Firebase Analytics for user behavior tracking, and Firebase Cloud Messaging for notifications.

Firebase Authentication offers a robust, secure, and easy-to-integrate solution for managing user authentication in web and mobile applications. Its strengths lie in its simplicity, support for multiple authentication methods, strong security measures, comprehensive user management tools, scalability, and seamless integration with the broader Firebase ecosystem.

### Database Layer

The Database Layer acts as a central repository for storing and managing system data. It provides the infrastructure for administrators and database managers to control and manipulate system resources.

This layer is responsible for storing the generated videos, user data.

The Database Layer supports functionalities such as video creation, editing, deletion, and addition, facilitating seamless integration with the other layers of the system.

Firebase Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform. It offers real-time data synchronization, powerful querying capabilities, and a flexible data model. Firestore is designed to handle a wide range of data storage needs, from simple key-value pairs to complex, hierarchical data structures.

Key Features of Firebase Firestore:

Real-Time Synchronization:

Instant Updates: Changes to the database are instantly synchronized across all connected clients, providing a seamless real-time experience.

Offline Support: Firestore caches data locally, allowing applications to remain responsive even when offline. Changes made offline are synchronized once the connection is reestablished.

Powerful Querying:

Rich Queries: Supports a wide range of queries, including compound queries, array-contains queries, and in queries. You can filter, sort, and paginate data effortlessly.

Indexing: Automatically indexes your data, optimizing query performance without requiring manual intervention.

Scalability:

Automatic Scaling: Designed to scale automatically with your application, handling large amounts of data and high traffic without significant performance degradation.

Distributed Architecture: Uses a distributed architecture to ensure high availability and low-latency access to data.

Flexible Data Model:

Document-Oriented: Stores data in documents, which are organized into collections. Documents can contain complex nested data structures, including arrays and maps.

Schema-Less: Allows for a flexible, schema-less data model, enabling easy iteration and evolution of your data structure as your application grows.

Security:

Granular Security Rules: Provides a robust security model with granular access controls through Firestore Security Rules. You can define who has access to which parts of your data and under what conditions.

Integration with Firebase Authentication: Seamlessly integrates with Firebase Authentication, enabling easy implementation of authenticated access controls.

Serverless Development:

Firestore Functions: Can be used with Cloud Functions for Firebase to create serverless backend logic that responds to database changes, enabling powerful event-driven architectures.

Client SDKs: Offers SDKs for iOS, Android, and web, making it easy to integrate with your application across multiple platforms.

Firebase Firestore is a robust, real-time database that offers powerful querying capabilities, automatic scaling, and offline support. Its flexible, schema-less data model and seamless integration with Firebase services make it an excellent choice for building modern, real-time applications.

With Firestore, developers can focus on building their applications without worrying about managing the underlying infrastructure or dealing with complex database configurations.

### API Layer

The API layer is the backbone of our application, supplying all necessary data regarding live matches, match statistics, league standings, and more. It acts as a central hub where our application retrieves real-time updates and comprehensive information about ongoing matches. This includes live scores, player statistics, team performance metrics, and detailed match summaries.

Furthermore, the API facilitates access to historical data, enabling users to delve into past match results, player performances over time, and statistical trends. This wealth of information empowers users to analyze and compare teams and players effectively.

In addition to match-specific data, the API also serves up-to-date league standings, reflecting the current positions of teams based on their performance throughout the season. This feature allows fans to track their favorite teams' progress and standings in real-time.

Overall, the API layer plays a crucial role in ensuring our application delivers accurate, timely, and comprehensive football data, enhancing the user experience by providing all the information needed to stay informed and engaged with the sport.

## System Architecture Algorithms

### E2E Spot Video Summarizer

#### Video Pre-Processing

1. Frame Extraction:

* Firstly, we get the complete video and extract frames of video so that all videos with different frame rate have the same frame rate. It doesn’t matter what the frame rate of video, we take always two frames per second from each video so it makes training more consistent. We calculate stride by dividing frame rate of complete video by wanted frame rate, in our case wanted frame rate is 2 frames per second, and we get the frames using this stride.

1. Divide frames into groups:

* We will divide all frames extracted in the first step into groups so that each group has same number of frames because We will use each group as training example and we want in our model to calculate the score and class of each frame so that we want all training examples to have same number of frames to have same number of neurons in output layer and make our model end to end. In our case we divide frames into groups so that each group

1. Data Augmentation for each training example:

We train E2E-Spot on 100-frame-long clips sampled randomly and use standard data-augmentations

1. Resizing:

* Resizing is an important step in the E2E Spot Model, as it ensures that the input data has a consistent size and shape, which is necessary for the model to learn the action spotting task more consistently. Resizing involves changing the dimensions of driving video frames to a fixed size, such as 224x224 pixels. This can be done using image processing libraries such as OpenCV or Pillow. Resizing the images to a smaller size can also help reduce the computational load on the model during training and inference. By standardizing the input dimensions, the model can process data more efficiently, leading to improved performance and accuracy in spotting actions.

1. Cropping:

* Cropping is another essential preprocessing step in the E2E Spot Model. It focuses on removing irrelevant parts of the video frames and concentrating on the region of interest. This technique can enhance the model's ability to detect and recognize actions by eliminating background noise and distractions. Cropping can be implemented using image processing libraries like OpenCV, where specific coordinates are defined to extract the desired portion of the frame. By focusing on key areas, cropping not only improves the model's accuracy but also reduces the amount of data it needs to process, thereby optimizing computational resources and speeding up both training and inference phases.

1. Mix UP:

* Mix-Up is a data augmentation technique that plays a vital role in enhancing the robustness and generalization capability of the E2E Spot Model. It involves combining pairs of images and their corresponding labels to create new training samples. Specifically, two images are blended together with a certain weight, and their labels are also mixed in the same proportion. This can be done using libraries such as NumPy or custom functions within deep learning frameworks. Mix-Up helps the model to generalize better by providing it with more diverse training examples, thus making it less likely to overfit on the training data. This technique has been shown to improve the model's performance on unseen data, making it more effective in real-world action spotting scenarios.

#### E2E Spot Model

This is the core component of this layer and it is responsible to get the tag for each frame. It is mainly made up of 4 components: RegNet Y, Gate Shift module (GSM), GRU, MLP.

E2E-Spot treats a video classification network as part of a sequence model, so that processing a clip of N frames results in 100 output features and N per-frame predictions. Figure 2 illustrates our pipeline. Frames from each RGB video group are first fed to a local spatial-temporal feature extractor F, which produces a dense feature vector for each frame This lightweight feature extractor incorporates Gate Shift Modules (GSM) [57] into a generic 2D convolutional neural network (CNN) [46]. The feature sequence is then further processed by a sequence model G, which builds a long-scale temporal context and outputs a class prediction for every frame, including a ‘background’ class to indicate when no event was detected.

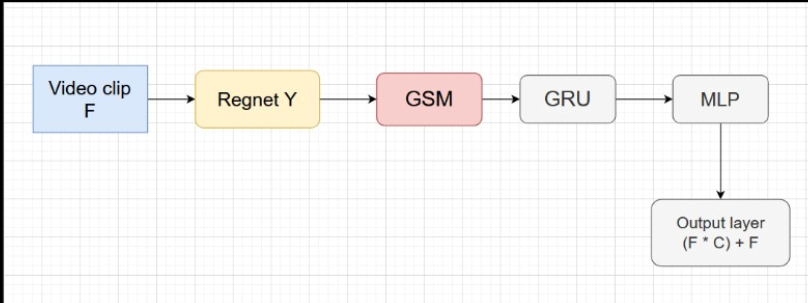


Figure ‎3‑2: E2E Spot Model

* Local Spatial-Temporal Feature Extractor and GSM The first stage of our pipeline extracts spatial-temporal features for each frame. We strive to keep the feature extractor as lightweight as possible, but found that a simple 2D CNN that processes frames independently [9,23,60,63] is often insufficient for precise spotting. This is because a 2D CNN does not capture the spatially-local temporal correlations between frames. In videos that are densely sampled (24–30 FPS), this temporal signal is critical to learn features that can robustly differentiate otherwise very similar frames: for instance, the speed and travel direction of a ball, when each frame likely exhibits motion blur. To obtain more expressive, motion-sensitive features we implement F as a 2D CNN with Gate Shift Modules (GSM) [57].

We choose RegNet-Y [46], a recent and compact CNN, as the 2D backbone. Our feature extractor is similar to models for video classification [37, 57, 63], but with two key differences: (1) it samples frames densely and (2) it uses no final temporal consensus/pooling because our goal is to obtain one output per frame, rather than one for the whole video or multi-frame segment. We choose RegNet-Y [46] over the more commonly used ResNet [26] family of 2D CNNs because the former is more recent and compact (RegNet-Y 200MF has 3.2M parameters vs. 11.7M parameters for ResNet-18), while exhibiting generally better performance on image classification benchmarks [64]. E2E-Spot, however, can be implemented with any 2D CNN architecture.

* Long-term Temporal Reasoning Module To gather long-term temporal information, we use a 1-layer bidirectional Gated Recurrent Unit (GRU [8]) network G, which processes the dense per-frame features produced by F. We set the hidden dimension of G to match that of F. We found that a single-layer GRU suffices and that more complex sequence models such as MS-TCN [20] or a deeper GRU do not necessarily improve accuracy We hypothesize that as a result of end-to-end training, the features produced by F capture subtle temporal cues that are specific to a given activities and task’s requirements. This shifts the burden of representations to F so that G only needs to propagate the temporal context.
* MLP Finally, we apply a fully connected layer and SoftMax on the GRU outputs to make a per-frame K+1 way prediction (including 1 ‘no-event’ background class).

**Create Summary Component:**

After the E2e Spot Model predicts the output for each frame, we apply post-processing techniques to generate the summarized video. This process includes the following steps:

1. **Non-Maximum Suppression (NMS)**

Non-maximum suppression is a crucial step in refining the model's predictions. It helps in eliminating redundant or overlapping detections, ensuring that the most relevant and precise predictions are retained. We evaluated the model predictions with and without non-maximum suppression. For temporally precise datasets, we used a window of ±1 frames to suppress non-maximum scores. However, for SoccerNetv2, which operates at 2 FPS, we use a window of ±2 frames. The effectiveness of NMS in a temporally precise setting depends on several factors, including frame-level tolerance, the specific dataset, and the model's characteristics. Therefore, the decision to apply NMS should be tailored to the specific application and task requirements, ensuring optimal performance and relevance of the summarized content.

1. **Frames Concatenation**

After obtaining the predictions for each frame, we perform frames concatenation to compile the summarized video. We select and concatenate frames that have a prediction score greater than 0.5 and correspond to one of the desired action classes. This threshold ensures that only frames with significant confidence in their action detection are included, thereby enhancing the relevance and clarity of the summarized video. By focusing on high-scoring frames, we effectively create a concise and informative summary that highlights the key moments and actions within the original video.

1. Additional Post-Processing Techniques In addition to NMS and frame concatenation, other post-processing techniques can be applied to further enhance the quality of the summarized video.

These may include:

**Smoothing Transitions:**

Applying techniques to smooth transitions between concatenated frames can help in creating a more visually coherent and pleasant viewing experience.

**Frame Rate Adjustment:**

Adjusting the frame rate of the summarized video to ensure it matches the desired playback speed, enhancing the temporal coherence and making the summary more natural to watch.

**Highlight Enhancements:**

Adding visual or audio highlights to key actions or moments can make the summary more engaging and easier to understand for viewers.

# System Implementation

1. Dataset
2. Experiments & Results in model training

## Dataset

### Introduction

A dataset is a set of structured and interconnected data, is considered an essential component for the development of machine learning and data learning models and is considered a powerful tool for understanding data and using it to achieve various goals in various fields and applications.

Documentation for a dataset should include information about the data sources, the preprocessing techniques used, and the format of the dataset. It should also include any specific instructions for loading and using the dataset in a machine learning framework. Additionally, it is helpful to include information about any potential biases in the data and any limitations or caveats to be aware of when using the dataset.

Overall, creating a high-quality dataset is a critical step in building effective machine learning models. By following best practices for data collection, preprocessing, and organization, you can ensure that your dataset is clean, consistent, and relevant to your problem, making it easier to train and test your models effectively.

### Challenges

In the Beginning we spend too much time to get the dataset that contain label for each frame for many matches videos. It was very difficult task, fortunately we find soccer-net v2 [7] dataset that contain label for each frame for 500 videos of matches in different countries such as England, Germany, Spain. But there was a very big problem that dataset is about 800 Giga Bytes and We don’t have resources to deal with that.

We decide to train the model in batches to overcome this problem and it was a great solution and we trained the model using this approach

### Description

SoccerNet v2 is a comprehensive and richly annotated dataset specifically designed for research in video understanding and action spotting within the context of soccer matches. It builds upon the original SoccerNet dataset by providing an expanded and more detailed collection of soccer videos and their corresponding annotations.

Here are the key features and components of the SoccerNet v2 dataset:

1. Content and Scope:
   * + - SoccerNet v2 contains full-length soccer match videos sourced from various leagues and competitions. The dataset includes both men's and women's matches, providing a diverse range of soccer gameplay.
       - The dataset comprises thousands of hours of video footage, ensuring extensive coverage of different game scenarios and events.
2. Annotations:
   * + - Event Annotations: SoccerNet v2 provides detailed annotations for various types of events occurring during the matches. These include goals, substitutions, yellow and red cards, corner kicks, free kicks, and more.
       - Temporal Annotations: Each annotated event includes precise temporal information, indicating the exact start and end times of the event within the video. This level of detail is crucial for training and evaluating action spotting models.
       - Spatial Annotations: In addition to temporal annotations, some events are also annotated with spatial information, highlighting the regions of the frame where the event takes place.
3. Data Format:
   * + - The dataset is provided in a structured format that includes video files and accompanying annotation files. Annotations are typically available in JSON or CSV format, making it easy to parse and utilize the data for various machine learning tasks.
       - Videos are provided in standardized formats to ensure consistency in preprocessing and analysis.
4. Use Cases:
   * + - Action Spotting: SoccerNet v2 is ideal for training models to detect and classify specific actions within soccer matches. The detailed temporal annotations enable precise action spotting, which is essential for applications such as highlight generation and game analysis.
       - Video Understanding: Researchers can use SoccerNet v2 to develop and evaluate algorithms for broader video understanding tasks, such as activity recognition, player tracking, and game strategy analysis.
       - Computer Vision and Machine Learning: The dataset supports a wide range of computer vision and machine learning research, including supervised learning, unsupervised learning, and reinforcement learning.
5. Evaluation Metrics:
   * + - SoccerNet v2 includes standardized evaluation metrics for assessing the performance of models on action spotting and other tasks. These metrics ensure fair and consistent comparison of different approaches and methodologies.
6. Community and Benchmarking:
   * + - SoccerNet v2 serves as a benchmark dataset for the soccer video understanding research community. It facilitates the comparison of different models and techniques, promoting advancements in the field. The dataset is often used in challenges and competitions, encouraging collaboration and innovation among researchers. In summary, SoccerNet v2 is a vital resource for advancing the state-of-the-art in video understanding and action spotting within the context of soccer. Its rich annotations, extensive coverage, and standardized evaluation protocols make it an essential tool for researchers and practitioners in the field.

### Dataset Samples

### 

Figure ‎4‑1: Dataset Sample 1

### A collage of a football game Description automatically generated

Figure ‎4‑2: Dataset Sample 2

A screenshot of a computer code

Description automatically generated

Figure ‎4‑3: JSON Representation

### Dataset Preprocessing

Functions that used in dataset Preprocessing:

1. Worker (Video)
   1. It gets frame rate of video
   2. Then it calculates stride by dividing frame rate of video by desired frame rate which is 2
   3. It extracts all frames in video and get frames using stride calculated in pervious strep
   4. It saves the frames in folder called frames
2. ImageDataGenerator ()

Make data augmentation

* 1. Resizing
  2. Rotation
  3. Cropping
  4. Mix Up

1. Create traindataset ()
   1. This function takes the frames folder and divide the frames into equally numbered groups each has 100 frames
   2. It takes annotations and get the annotations for frames in each group
   3. It converts annotations in Json format to corresponding output.

## Experiments & Results in model training

### ResNet & LSTM

* Training Procedure
  + Firstly, we try to use Action Recognition Task to Summary Highlights in football matches.

We use Action Recognition model that consist of ResNet CNN for Feature Extraction [3] then LSTM [5] for learning sequence in videos inspired from [4]. We use soccer DB dataset in this model and we adjust hyperparameters used in [4], We also make loss function for classes that isn’t back ground 5 times larger than background class, we also make strong data augmentation.

The Training procedure use this hyperparameters:

* 3d-ResNet Input Configuration
  + - The input shape is L\*w\*h\*C where L is number of stacked frames

We Experiment Using different values of L: L= {10,16,18,26,30,32,64,128}

We found that optimal value is L = 64

* Learning Rate
  + - Initial Learning Rate is 3\*10-3

Learning rate divided by 10 after every 10,000 iterations.

Decay Rate is 5 \* 10-4

* Optimizer
  + - We use Stochastic Gradient Descent
* Batch Size
  + - We use batch size = 16
* Evaluation Metric

Table ‎4‑1: ResNet & LSTM Evaluation Metric

|  |  |  |  |
| --- | --- | --- | --- |
| **Paper/Metric** | **Accuracy** | **precision** | **recall** |
| LSTM + ResNet [4] | 96.81% | 93% | 91% |
| LSTM + ResNet (ours) | 98.3 % | 97% | 96% |

* Problems in this Model

Because the model is based on Action Recognition So Entire Action should be in video being predicted but actually, we split Complete Football Match Video into parts and make action recognition in each part. what will happen if Action is at end of one part and continuing in start of next part? The model will not predict the Action Correctly so we come to Action Spotting Task Not Action Recognition Task.

### E2E Spot Model

This is our final model we make it and This is the model we use it. we inspire from [8].

**Modifications we made:**

1. Adjusting Hyperparameter
2. Decrease Number of classes into 11 Class We are Interested In.
3. Train Data in Batches Which are more quickly

**Training Procedure:**

1. E2E-Spot model trained using 100-frame long clips and a batch size of 8 clips.
2. Training batches randomly sampled from the training videos.
3. Training cycle comprises 625 steps, equivalent to a pseudo-epoch of 500K frames.
4. Each training cycle runs approximately:
   1. 8.5 minutes for the 200MF variant.
   2. 14 minutes for the 800MF variant on a single A5000 GPU [44].
5. Datasets trained and cycle durations:
   1. SoccerNet-v2:
      1. 200MF model: 150 cycles.
      2. 800MF model: 200 cycles.
6. AdamW optimizer [43] used with a base learning rate of 10-4 including 3 linear warmup cycles and cosine decay [42].

Loss Function is cross entropy loss and to mitigate imbalance arising from the rarity of precise events (< 3% of frames), we boost the loss weight of the foreground classes (5×) relative to the background.

**Evaluation Metric:**

We measure Average Precision within a tolerance of δ frames (AP @ δ). AP is computed for each event class, and MAP is the mean across classes. We focus on tight tolerances such as δ = 1 and δ = 2. Precise temporal events are rare as a percentage of frames (0.2–2.9%), so metrics such as frame-level accuracy are not meaningful for precise spotting.

Table ‎4‑2: E2E Spot Evaluation Metric

|  |  |  |
| --- | --- | --- |
| **Model/Metric** | **MAP (Tight)** | **MAP (Loose)** |
| Baseline | 58% | 61% |
| E2eSpot [8] | 78.6% | 80.5% |
| E2eSpot Fine Tuning Only(ours) | 79.5% | 81.25% |
| E2eSpot with 11 Classes | 86.7% | 87.5% |

Tight means with tolerance 0-5 seconds and loose with tolerance 30-60 seconds only.

# Run the Application

1. Desktop Application
2. Mobile Application

## Desktop Application

## Mobile Application

GameGlimpse, a mobile application recently developed using Flutter for Android devices, offers soccer enthusiasts a comprehensive platform to stay updated with the latest fixtures, league standings, match summaries, and more. Whether you're a passionate fan tracking your favorite teams or a casual observer keeping tabs on the latest soccer action, GameGlimpse provides a seamless user experience with its intuitive interface and up-to-date information. With features designed to enhance the soccer viewing experience, GameGlimpse is your go-to app for everything soccer-related on Android devices.

we delve into the architecture and design patterns utilized in the development of the mobile application. By leveraging robust architectural principles and proven design patterns, we ensured the application is scalable, maintainable, and efficient.

### Architecture

The mobile application is built using the MVVM (Model-View-ViewModel) architecture. This architecture separates the application's logic into three main components, facilitating clear separation of concerns and enhancing testability.

1. Model: Represents the data and business logic of the application. The Model component manages the data, handles network requests, and communicates with the database. It encapsulates the data structures and state of the application.

* Data Models: Classes representing the structure of the data used in the app (e.g., Match, Team, User).
* Repositories: Components responsible for data retrieval from remote or local data sources such as Firebase Firestore, Firebase Storage, or Google Drive Storage.

1. View: The user interface of the application. The View component displays the data and receives user input. It is responsible for rendering the UI elements and interacting with the user.

* Widgets: Flutter's building blocks for UI, such as Scaffold, ListView, Text, and Button.
* State Management: Mechanisms to manage the state of the UI components, typically using providers or stateful widgets.

1. ViewModel: Acts as an intermediary between the Model and the View. The ViewModel handles the presentation logic and prepares data for display in the View. It receives user inputs from the View, processes them, and updates the Model accordingly.

* Cubit: A subset of the BLoC pattern that simplifies state management by using a single state object and a series of events to trigger state changes. Cubit is used for managing the state of various features in the application.State Notifiers: Objects that notify the View of any changes in the data or state, ensuring the UI is always up-to-date.
* State Notifiers: Objects that notify the View of any changes in the data or state, ensuring the UI is always up-to-date.

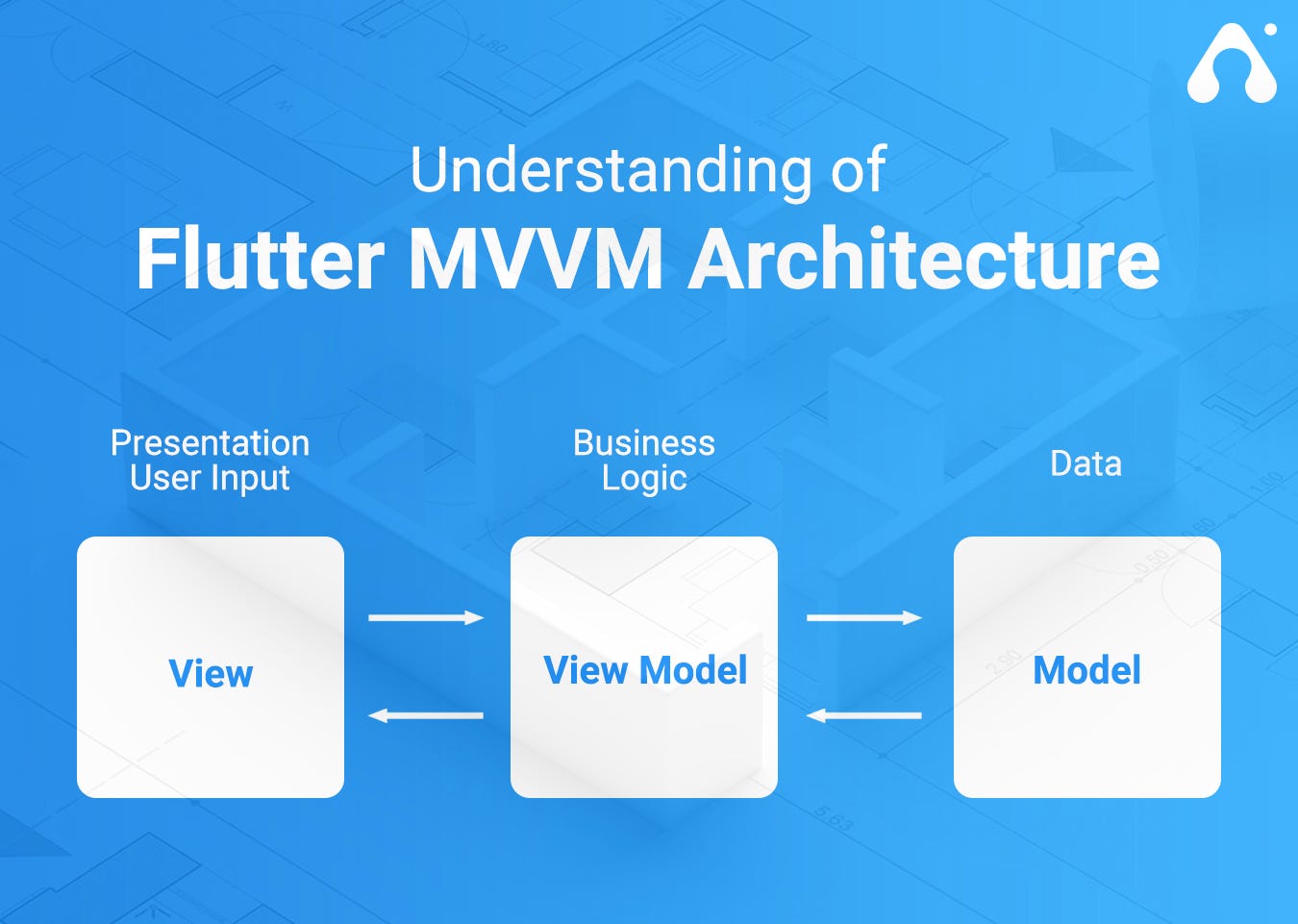


Figure ‎5‑1: MVVM (Model-View-ViewModel) Architecture

### Design Patterns

To complement the MVVM architecture, several design patterns were implemented to enhance the structure and functionality of the mobile application:

1. Singleton Pattern: Ensures that a class has only one instance and provides a global point of access to it. This pattern is used for managing shared resources like network clients, database connections, and configuration settings.

* Example: A DatabaseService class that provides a single instance of the Firebase Firestore client to the entire application.

1. Repository Pattern: Abstracts the data layer, providing a clean API for data access. This pattern helps to decouple the data retrieval logic from the business logic, making it easier to manage and test.

* Example: MatchRepository that fetches match data from Firebase Firestore and exposes methods like getMatchById () and getAllMatches ().

1. Provider Pattern: Used for state management in Flutter. Providers offer a way to manage and propagate state changes throughout the application without the need for manual prop drilling.

* Example: ChangeNotifierProvider that manages the state of the user's profile, notifying the UI whenever the profile data changes.

1. Factory Pattern: Provides a way to create objects without specifying the exact class of the object that will be created. This pattern is useful for managing object creation and ensuring that objects are instantiated in a consistent manner.

* Example: A SummaryFactory class that creates different types of match summaries based on the match data.

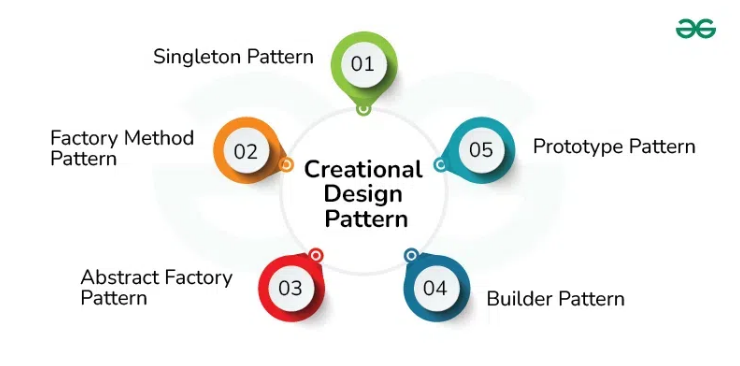


Figure ‎5‑2: Design Patterns

### SOLID Principles

To ensure the application codebase is robust, maintainable, and scalable, we adhered to the SOLID principles of object-oriented design:

1. Single Responsibility Principle (SRP): Each class and module should have only one reason to change, meaning they should only have one responsibility.

* Example: The MatchRepository class is solely responsible for data operations related to matches. It does not handle UI logic or user authentication, which are managed by separate classes.

1. Open/Closed Principle (OCP): Software entities (classes, modules, functions, etc.) should be open for extension but closed for modification.

* Example: Adding a new feature, such as a new type of match summary, can be done by extending existing classes or implementing new ones without altering the existing code.

1. Liskov Substitution Principle (LSP): Objects of a superclass should be replaceable with objects of a subclass without affecting the correctness of the program.

* Example: The User class can be extended to create a PremiumUser class. Any function expecting a User object should work seamlessly with a PremiumUser object.

1. Interface Segregation Principle (ISP): Clients should not be forced to depend on interfaces they do not use. This principle encourages creating small, specific interfaces rather than large, general-purpose ones.

* Example: Interfaces such as UserRepository and MatchRepository are kept specific to their respective data operations, ensuring that classes implementing these interfaces are not burdened with methods they do not need.

1. Dependency Inversion Principle (DIP): High-level modules should not depend on low-level modules. Both should depend on abstractions. Abstractions should not depend on details. Details should depend on abstractions.

* Example: The application uses dependency injection to provide instances of repositories and services to the ViewModels (Cubits). This decouples the high-level business logic from low-level data access details.

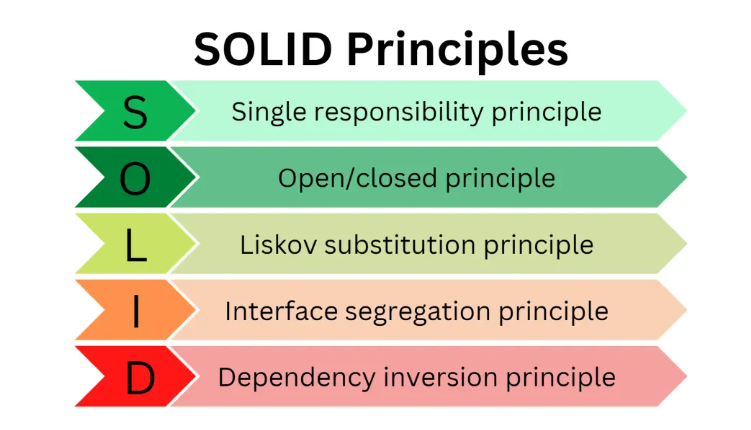


Figure ‎5‑3: SOLID Principles

By implementing these architectural principles, design patterns, and adhering to SOLID principles, the mobile application is structured to handle complexity efficiently, ensuring high performance, scalability, and ease of maintenance. This robust foundation allows for future enhancements and adaptations to be integrated seamlessly, supporting the long-term success of the project.

### Splash Screen

The Splash Screen serves as a fundamental component of our application, greeting users upon launch as the initial interface they encounter. It visually embodies the application's branding, delivering a concise introduction and aesthetic appeal to captivate users. Designed with animation, the splash screen exemplifies a dynamic representation, as illustrated in the figure 5-1 provided.

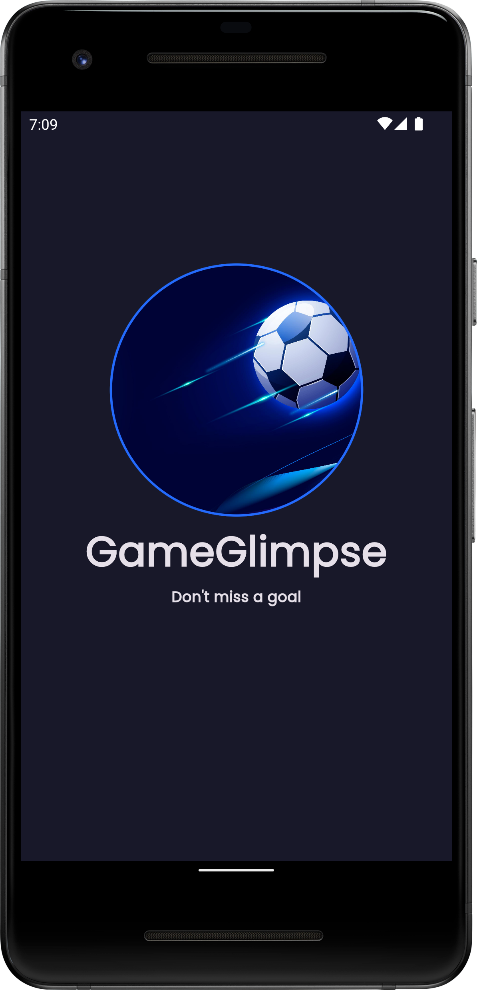


Figure ‎5‑4: Splash Screen

### Onboarding Screens

Onboarding screens play a crucial role in our application, providing new users with a seamless and engaging introduction. Specifically designed for Android applications, these screens are displayed only during the initial launch, ensuring a user-friendly experience from the outset.

These screens serve as an opportunity to acquaint users with the application's features effectively. For example, they can guide users on selecting or discovering images of statues while considering spatial constraints. Furthermore, onboarding screens highlight the app's advantages and provide a comprehensive overview of its purpose.

Moreover, these screens demonstrate how to utilize various functionalities, such as interactive chat, ensuring users grasp both how to use them and the benefits they offer. Overall, onboarding screens are instrumental in ensuring users comprehend the app's capabilities, encouraging them to maximize its potential.

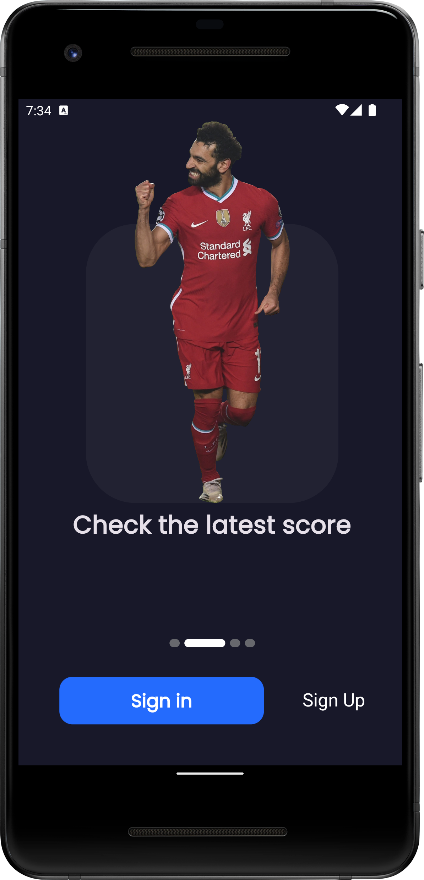
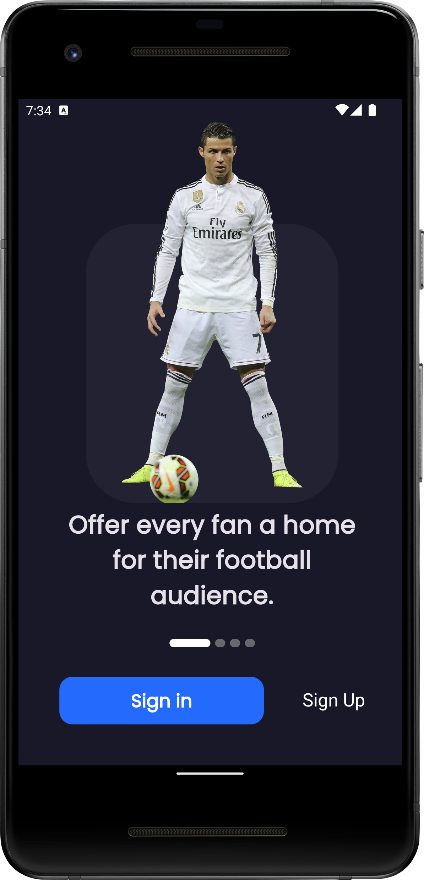


Figure ‎5‑5: Onboarding Screens

### Authentication Screen

The Authentication Screen is pivotal in our application, offering users streamlined access through both Sign In and Sign Up options. Whether opting for traditional email and password authentication or the convenience of Google sign-in, users can seamlessly enter the app.

For those signing up, the process is equally straightforward, with options to create an account using email and password or to swiftly register via Google. Additionally, users have the flexibility to join anonymously as an incognito user, ensuring privacy and convenience.

Furthermore, the Authentication Screen includes a Forgot Password feature, allowing users to securely reset their credentials if needed, enhancing user accessibility and security.

With these functionalities seamlessly integrated, our Authentication Screen ensures a user-friendly experience, prioritizing convenience, security, and user choice. security flutter

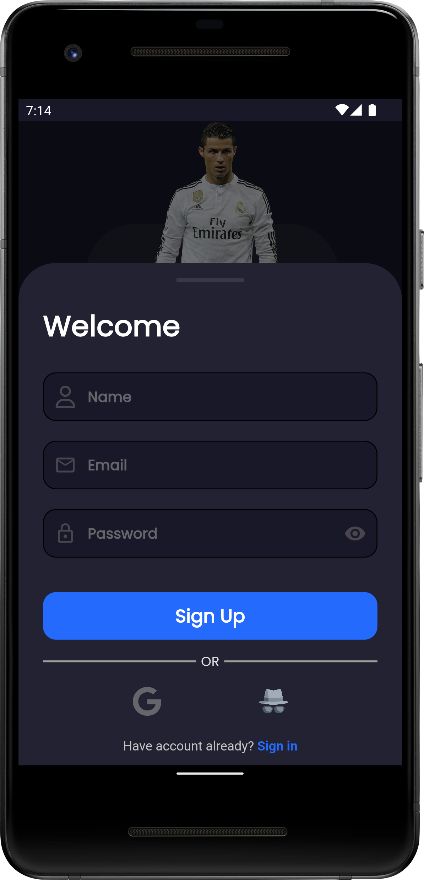
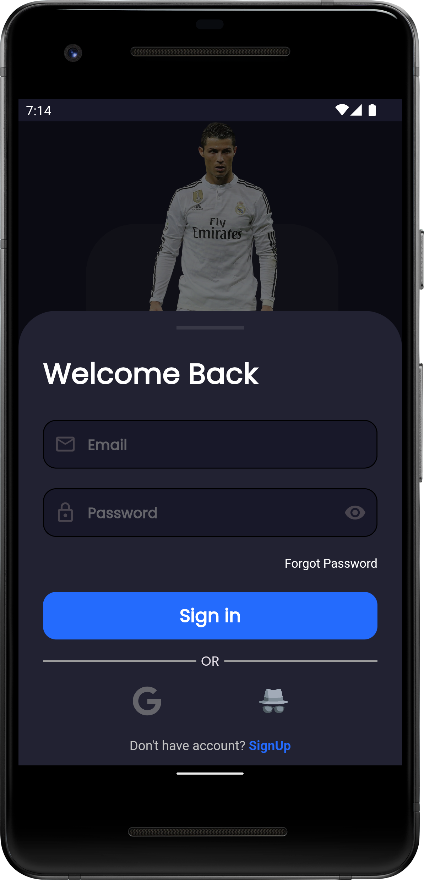


Figure ‎5‑6: Authentication Screen

### Home Layout

The Home Layout of our application features a dynamic Bottom Navigation Bar, meticulously designed to enhance user navigation across four distinct pages: Fixtures, Leagues Standing, Video Summary, and Profile.

This intuitive interface ensures effortless access to vital soccer information. The Fixtures page keeps fans updated on upcoming matches, ensuring they never miss a game. The Leagues Standing page provides comprehensive standings, allowing users to track their favorite teams' progress throughout the season.

For those seeking visual updates, the Video Summary page offers concise recaps and highlights of recent matches, delivering engaging content at a glance. Lastly, the Profile page allows users to personalize their experience, manage preferences, and stay connected with their favorite teams and leagues.

With seamless navigation and informative content, our Bottom Navigation Bar transforms the soccer viewing experience, making our app a must-have companion for every enthusiast.

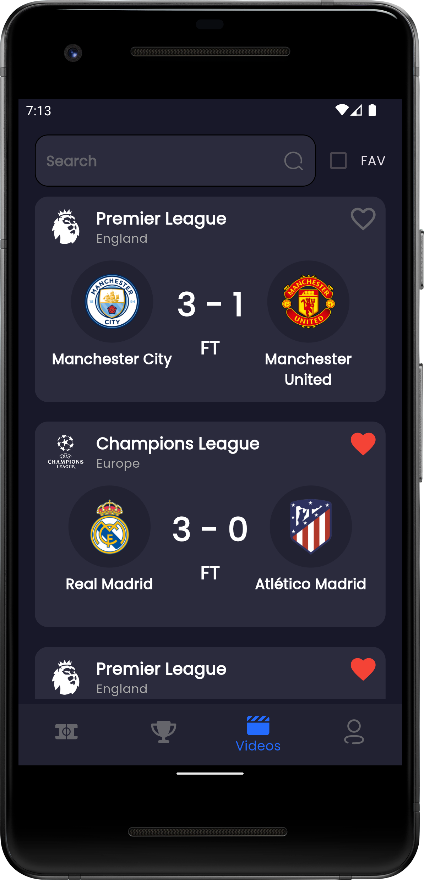
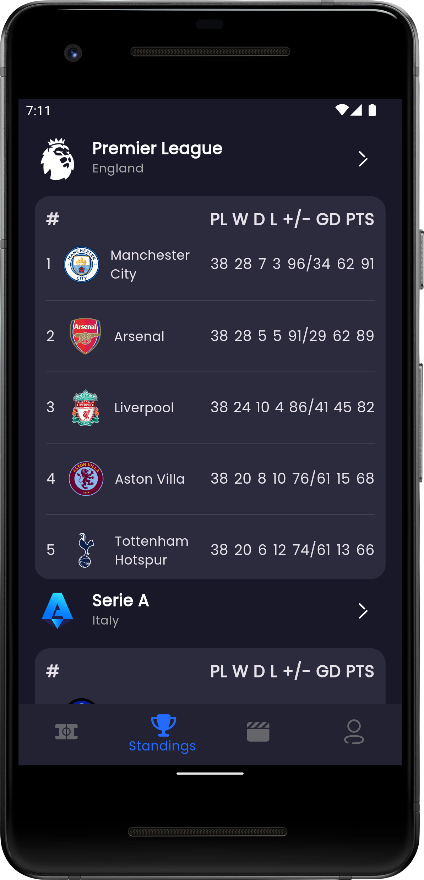
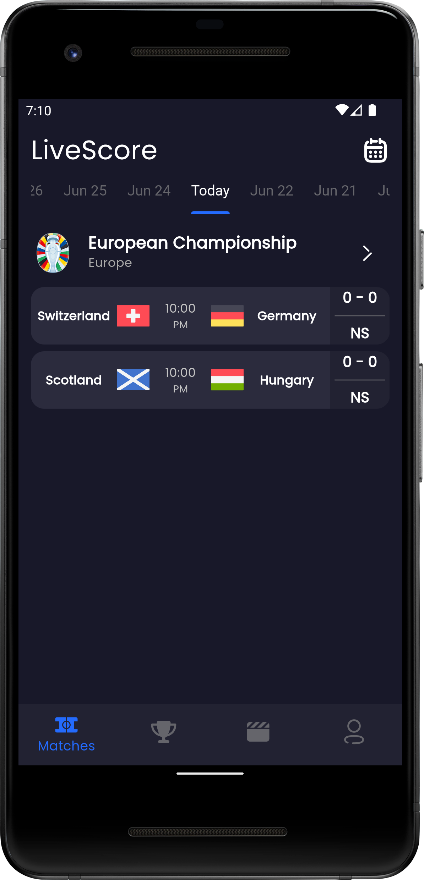


Figure ‎5‑7: Home Layout

### Fixtures Screen

The Day Fixtures Screen in our application is designed for easy access and navigation, featuring a Tab Bar that allows users to effortlessly select the specific day they wish to view fixtures for. Alternatively, users can opt to choose a day directly from a calendar interface to explore the day's scheduled matches.

This intuitive design ensures that soccer enthusiasts can quickly find the fixtures they're interested in, whether they prefer browsing through tabs for immediate access or using the calendar for precise date selection. By providing these options, the Day Fixtures Screen enhances user convenience and flexibility, catering to various preferences and needs when following their favorite teams' schedules.

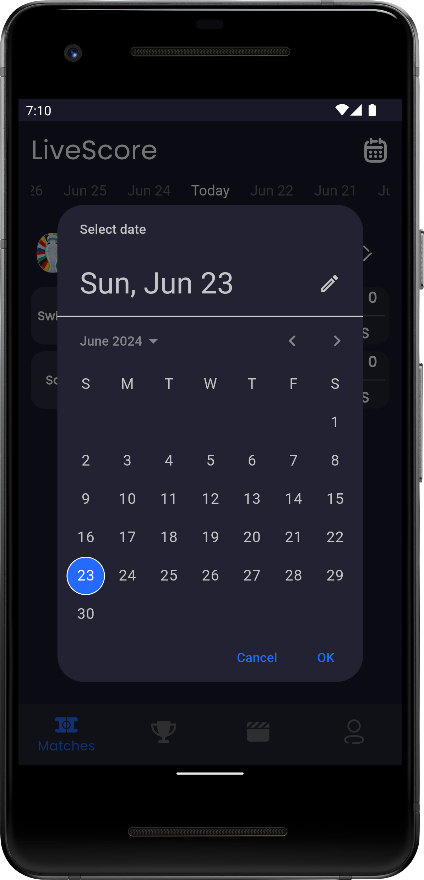
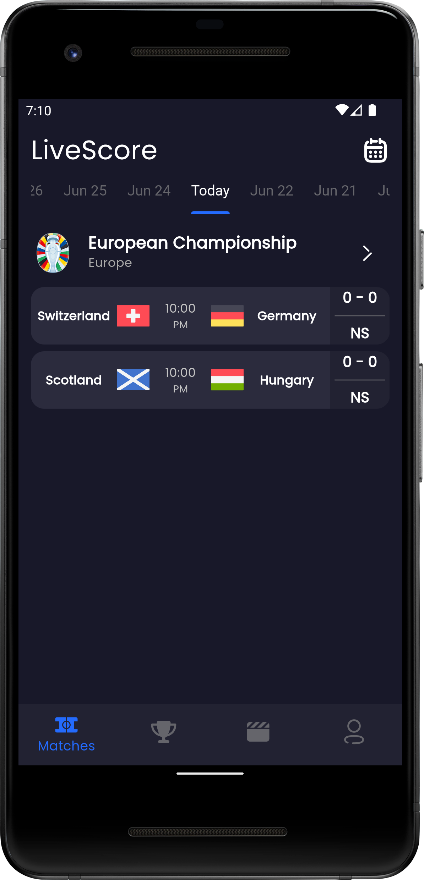


Figure ‎5‑8: Fixtures Screen

### Match Details

The Match Details Screen in our application offers a detailed breakdown of each match, providing users with comprehensive information on scores, statistics, and starting lineups for both teams.

This screen serves as a central hub where users can delve into key aspects of the game, including goals scored, assists, possession percentages, and shots on target. Additionally, it displays the starting lineups for each team, highlighting key players and formations.

With its user-friendly interface and real-time updates, the Match Details Screen enhances the soccer viewing experience, ensuring fans have access to all pertinent information to follow and analyze their favorite matches with depth and insight.

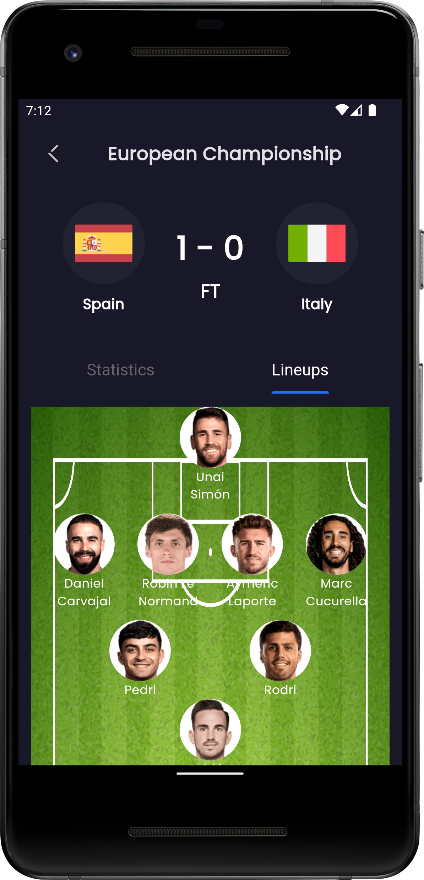


Figure ‎5‑9: Match Details Screens

### Leagues Standings

The Leagues Standings Screen in our application provides a comprehensive view of team rankings across various leagues. Users can easily navigate through different leagues to see the standings of each team displayed in an organized order.

This screen offers soccer enthusiasts a detailed perspective on team performances, allowing them to track their favorite clubs' positions throughout the season. With intuitive design and up-to-date information, the Leagues Standings Screen ensures users stay informed about current standings and league dynamics, making it an essential tool for passionate followers of the sport.

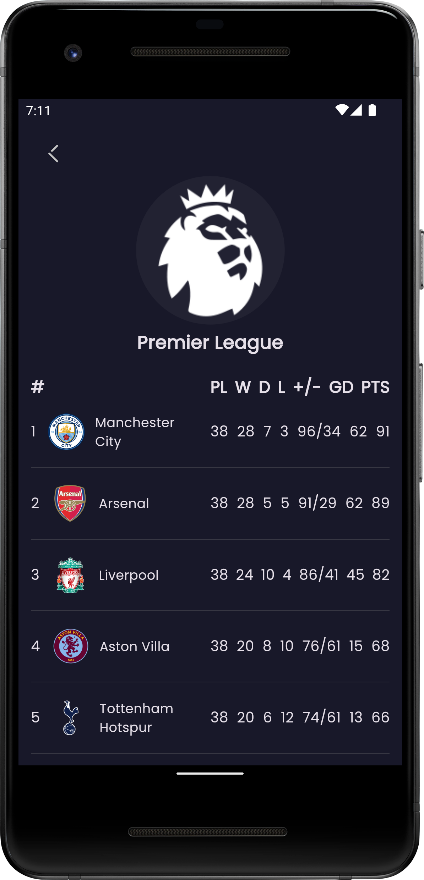
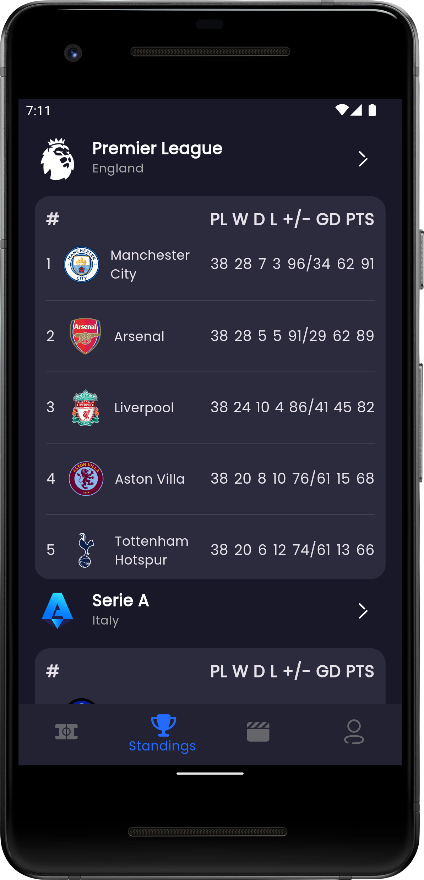


Figure ‎5‑10*:* Leagues Standing Screens

### Video Summary

The Video Summary Screen in our application offers users a curated list of matches, each categorized into four classes: Full Summary, Goals, Shots, and Red Cards. Users can select any match category to view specific highlights and key moments.

Additionally, the screen features a functionality allowing users to mark matches as favorites, enabling easy access for future viewing. This personalized feature ensures users can quickly revisit their preferred matches without searching through the entire list.

Furthermore, the Video Summary Screen includes search and filter options, enabling users to efficiently navigate through matches. Whether searching for a specific match or filtering by favorites, these tools enhance user experience by providing quick access to desired content.

With its intuitive design and robust features, the Video Summary Screen enriches the soccer viewing experience, offering comprehensive match highlights and convenient navigation options for enthusiasts.

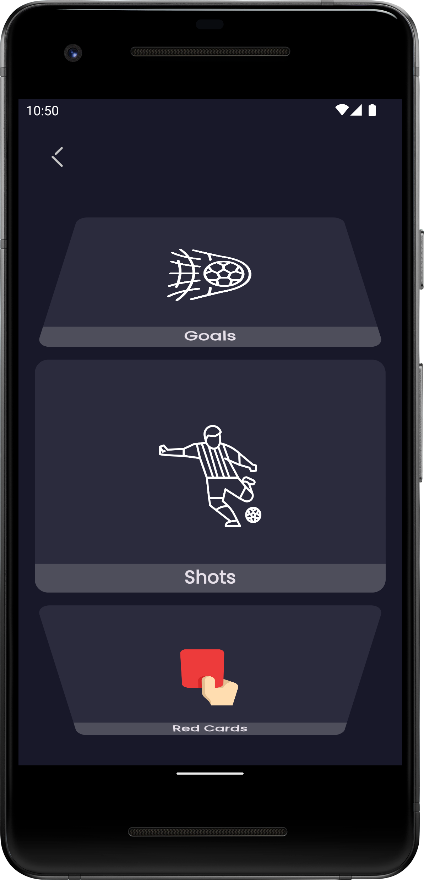
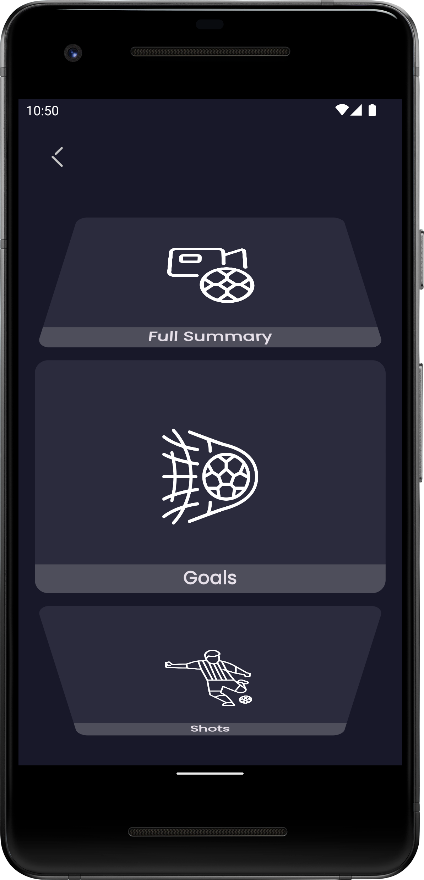
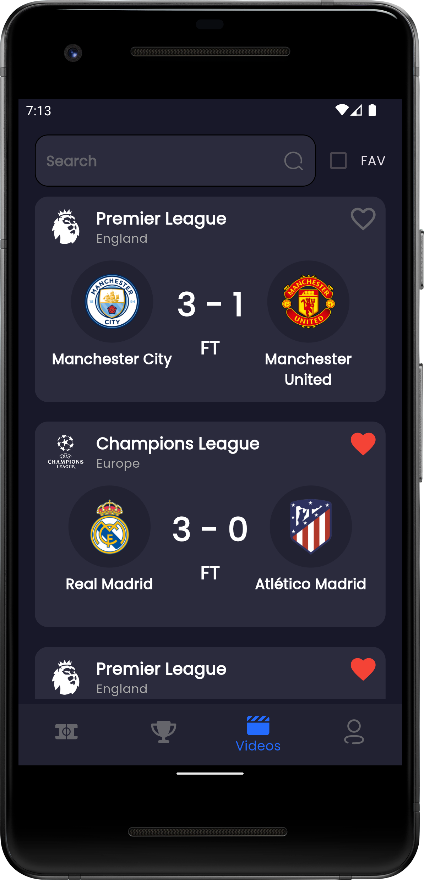


Figure ‎5‑11: Video Summary Screens

### Profile Screen

The Profile Screen in our application provides users with a personalized space to manage their identity and preferences. It prominently displays the user's name, bio, and profile image, offering a glimpse into their profile details.

Users have the option to edit their profile information directly from this screen. They can easily update their name, modify their bio to reflect current interests or statuses, and change their profile image to personalize their account further.

This screen ensures a seamless user experience by allowing quick and straightforward profile management within the app. Whether users wish to update their details or personalize their profile image, the Profile Screen offers intuitive functionality to meet their needs effectively.

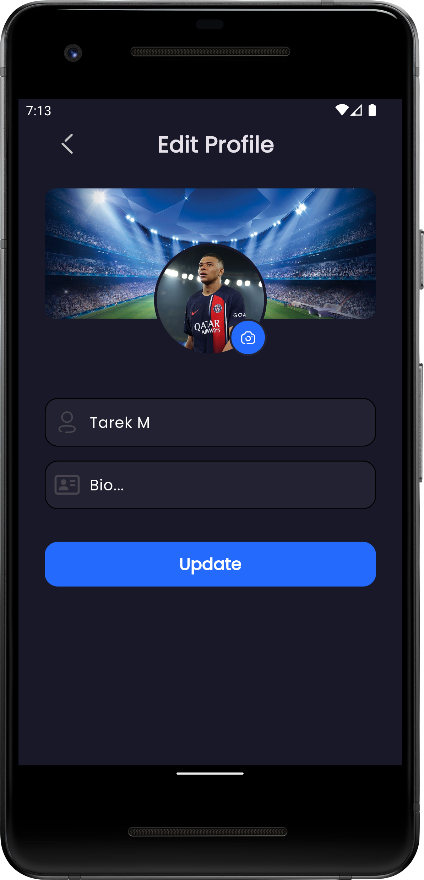
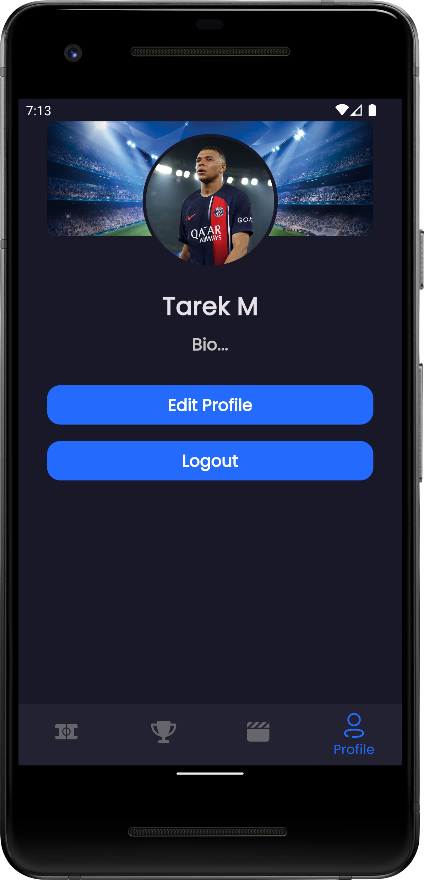


Figure ‎5‑12: Profile Screen

# Conclusion and Future Work

1. Conclusion
2. Future Work

## Conclusion

This project successfully addresses the challenge of delivering concise and informative soccer match summaries for fans with limited time. By developing an intelligent summarization model and integrating it into a mobile application, we have created an innovative solution that enhances fan engagement and accessibility.

The application not only offers match highlights but also includes features such as fixtures and league standings, ensuring a comprehensive user experience.

The intelligent summarization model leverages advanced machine learning and deep learning techniques to analyze match footage, identify crucial events such as goals, penalties, and notable plays, and compile them into a concise summary.

The model's effectiveness was evaluated through a series of tests comparing the generated summaries with human-compiled highlights, demonstrating its accuracy and relevance.

The successful implementation of this project bridges the gap between fans and the sport, allowing them to stay updated and connected despite their busy schedules. Additionally, this project lays the groundwork for future advancements in sports analytics and media consumption, demonstrating the potential of artificial intelligence in transforming how sports content is delivered and consumed.

Ultimately, our solution aims to make soccer more enjoyable and accessible for a global audience, fostering greater engagement and appreciation for the sport.

## Future Work

While the project has achieved its primary objectives, there are several avenues for future work to further enhance the application:

1. Enhanced Personalization:

Implementing more sophisticated machine learning algorithms to provide personalized content recommendations based on user preferences and viewing history. This could involve tailoring summaries to highlight favorite teams, players, or specific types of plays.

1. Real-time Summarization:

Developing capabilities for real-time summarization, allowing users to receive highlights and summaries during live matches. This would enable fans to stay updated with the most crucial moments as they happen, enhancing the live viewing experience.

1. Expanded Coverage:

Extending the application to cover more sports and leagues, providing a broader range of content for users. This expansion would involve adapting the summarization model to different sports, each with unique events and play styles.

1. Social Integration:

Adding social features that enable users to share highlights, comment on matches, and engage with other fans within the app. social interaction can increase user engagement and create a community around the app.

1. Performance Optimization:

Continuously improving the performance and accuracy of the summarization model by incorporating the latest advancements in machine learning, computer vision, and natural language processing. Regular updates and refinements will ensure that the application remains state-of-the-art.

1. Multilingual Support:

Integrating multilingual support to cater to a global audience. This would involve translating the app’s interface and summaries into multiple languages, making it accessible to non-English speaking users.

1. Integration with Wearable Devices:

Exploring the possibility of integrating the application with wearable devices to provide instant notifications and updates, enhancing the convenience for users on the go.

1. Detailed Analytics and Insights:

Adding features to provide detailed analytics and insights based on the summarized matches. This could benefit coaches, analysts, and sports journalists who require in-depth analysis of match performances.

1. User Feedback and Adaptation:

Implementing a robust feedback system to gather user input on the summarization quality and app features. This feedback will be crucial for iterative improvements and adapting the application to better meet user needs.

1. Cross-platform Compatibility:

Ensuring the application is compatible with various operating systems and devices, including iOS, Android, and web platforms, to maximize accessibility and user reach.

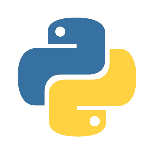
# Tools

1. Languages
2. Environments
3. Frameworks

In this chapter, we provide a detailed overview of the various tools and technologies utilized throughout the development and implementation of this project. These tools are classified into three categories: Languages, Environments, and Frameworks.

## Languages

1. Python: A versatile programming language used extensively for machine learning, data analysis, and backend development. Python's simplicity and robust libraries make it a preferred choice for developing the summarization model and handling data processing tasks.



1. Dart: The primary programming language used with the Flutter framework for developing the mobile application. Dart is optimized for building high-performance, cross-platform applications with a single codebase.



## Environments

1. Kaggle: An online community and platform for data science competitions and collaborative projects. Kaggle provides datasets, notebooks, and an integrated development environment (IDE) for experimenting with machine learning models.



1. Google Colab: A cloud-based Jupyter notebook environment that allows for collaborative development and execution of Python code. Google Colab is particularly useful for running intensive machine learning experiments on Google's hardware accelerators (GPUs and TPUs).



1. Jupyter: An open-source web application that enables the creation and sharing of documents containing live code, equations, visualizations, and narrative text. Jupyter notebooks are extensively used for data analysis and model development.



1. PyCharm: An integrated development environment (IDE) for Python, providing tools for code editing, debugging, and project management. PyCharm enhances productivity and code quality through its intelligent code assistance features.



## frameworks

1. Flutter: An open-source UI software development toolkit created by Google. Flutter allows for the creation of natively compiled applications for mobile, web, and desktop from a single codebase. It is used in conjunction with Dart for developing the project's mobile application.



1. TensorFlow: An open-source machine learning framework developed by Google. TensorFlow provides a comprehensive ecosystem of tools, libraries, and community resources for building and deploying machine learning models.



1. PyTorch: An open-source machine learning library developed by Facebook’s AI Research lab (FAIR). PyTorch is known for its flexibility and ease of use, particularly in research and prototyping of machine learning models.



1. Firebase Firestore: A NoSQL cloud database that allows for real-time data synchronization and storage. Firestore is used for managing and storing application data, providing seamless integration with Firebase services.



1. Firebase Storage: A cloud storage solution by Google that enables secure file uploads and downloads directly from the mobile application. Firebase Storage is used for storing media files such as images and videos related to the summarization model.



1. Google Drive Storage: A cloud storage service by Google that allows users to store files in the cloud, synchronize files across devices, and share files. Google Drive Storage is used for storing project documentation, datasets, and other important resources accessible to the development team.



By leveraging these tools, we were able to efficiently develop, test, and deploy the summarization model and mobile application. Each tool played a crucial role in different phases of the project, contributing to the overall success and functionality of our solution.

# References

1. References

## References