Grassroot football analysis Report

Project Objectif :

The project aims to build and implement a Big Data structure to collect, sort, and manage 500 valuable videos of basic football matches. This data will be used for analyzing player skills, tracking performance, gaining tactical insights, and increasing fan involvement. This effort will support training strategies, improve training programs, create effective game plans, and enhance the fan experience by making engaging content and personalizing the fan experience.

Project organization (Plan)

1. **Data Collection & Storage**
    - Identify 500 grassroots football match videos from sources
    - Annotate metadata like teams, players, tournament info into structured SQL tables
2. **Data Processing**
    - Extract data from storage, clean & transform
3. **Analyze & Model**
    - Access processed videos & metadata from warehouse

- Engineers perform custom analytics using Python/OpenCV/Soupervisor/YOLOV8
- Train ML models to generate player/team insights

4. **Application Development**
    - Create web portal for users to search & retrieve videos
    - Build interactive dashboards & visualizations on extracted insights
    - Support APIs for third-party integration

All Informations that I need to collect

1. Videos Table

- **VideoID** (Primary Key): Unique identifier for each video.
- **Title**: Name or title of the video.
- **Source**: Source of the video (e.g., club, school).
- **DateRecorded**: Date when the video was recorded.
- **Duration**: Length of the video

2. Teams Table

- **TeamID** (Primary Key): Unique identifier for each team.
- **Name**: Name of the team.
- **Category**: Age group or skill level category.
- **Region**: Geographic region or locality of the team.

3. Players Table

- **PlayerID** (Primary Key): Unique identifier for each player.
- **TeamID** (Foreign Key): Identifier linking to the Teams table.
- **Name**: Player’s name.
- **DateOfBirth**: Player’s date of birth.

- **Position**: Playing position (e.g., forward, defender).
- **Stats**: Link to player's performance statistics.

4. Matches Table

- **MatchID** (Primary Key): Unique identifier for each match.
- **Date**: Date of the match.
- **Team1ID, Team2ID** (Foreign Keys): Identifiers for the teams involved.
- **Location**: Venue of the match.
- **Result**: Outcome of the match.
- **VideoID** (Foreign Key): Link to the video of the match.

5. Events Table (Optional, for detailed analysis)

- **EventID** (Primary Key): Unique identifier for each event in a match.
- **MatchID** (Foreign Key): Identifier linking to the Matches table.
- **Time**: Time of the event during the match.
- **Type**: Type of event (goal, foul, etc.).
- **PlayerID** (Foreign Key): Player involved in the event.
- **Description**: Brief description of the event.

6. Metadata Table

- **MetaID** (Primary Key): Unique identifier for each metadata entry.
- **VideoID** (Foreign Key): Identifier linking to the Videos table.
- **Key**: Metadata key or name (e.g., player highlighted).
- **Value**: Metadata value (e.g., player's name).

CMD

```
erDiagram

Videos ||--o{ Matches : "contains"
Videos {
    int VideoID PK
    string Title
    string Source
    date DateRecorded
    time Duration
}

Teams ||--o{ Players : "has"
Teams {
    int TeamID PK
    string Name
    string Category
    string Region
}

Players ||--o{ Events : "involved_in"
Players {
    int PlayerID PK
    int TeamID FK
    string Name
    date DateOfBirth
    string Position
    string Stats
}

Matches ||--o{ Teams : "involves"
Matches ||--o{ Events : "includes"
Matches {
    int MatchID PK
    date Date
    int Team1ID FK
```

```
    int Team2ID FK
    string Location
    string Result
    int VideoID FK
}

Events {
    int EventID PK
    int MatchID FK
    time Time
    string Type
    int PlayerID FK
    string Description
}

Metadata ||--|| Videos : "describes"
Metadata {
    int MetaID PK
    int VideoID FK
    string Key
    string Value
}
```

SQL Code

```
-- Create database
CREATE DATABASE grassroots_football;

-- Use the created database
USE grassroots_football;

-- Create Videos Table
CREATE TABLE Videos (
    VideoID INT PRIMARY KEY,
```

```
    Title VARCHAR(255),
    Source VARCHAR(255),
    DateRecorded DATE,
    Duration TIME,
    Quality VARCHAR(50),
    Format VARCHAR(50),
    FilePath VARCHAR(255),
    Status VARCHAR(50)
);

-- Create Teams Table
CREATE TABLE Teams (
    TeamID INT PRIMARY KEY,
    Name VARCHAR(255),
    Category VARCHAR(100),
    Region VARCHAR(100)
);

-- Create Players Table
CREATE TABLE Players (
    PlayerID INT PRIMARY KEY,
    TeamID INT,
    Name VARCHAR(255),
    DateOfBirth DATE,
    Position VARCHAR(50),
    Stats VARCHAR(255),
    FOREIGN KEY (TeamID) REFERENCES Teams(TeamID)
);

-- Create Matches Table
CREATE TABLE Matches (
    MatchID INT PRIMARY KEY,
    Date DATE,
    Team1ID INT,
    Team2ID INT,
    Location VARCHAR(255),
```

```
    Result VARCHAR(100),
    VideoID INT,
    FOREIGN KEY (Team1ID) REFERENCES Teams(TeamID),
    FOREIGN KEY (Team2ID) REFERENCES Teams(TeamID),
    FOREIGN KEY (VideoID) REFERENCES Videos(VideoID)
);

-- Create Events Table
CREATE TABLE Events (
    EventID INT PRIMARY KEY,
    MatchID INT,
    Time TIME,
    Type VARCHAR(100),
    PlayerID INT,
    Description TEXT,
    FOREIGN KEY (MatchID) REFERENCES Matches(MatchID),
    FOREIGN KEY (PlayerID) REFERENCES Players(PlayerID)
);

-- Create Metadata Table
CREATE TABLE Metadata (
    MetaID INT PRIMARY KEY,
    VideoID INT,
    Key VARCHAR(255),
    Value TEXT,
    FOREIGN KEY (VideoID) REFERENCES Videos(VideoID)
);

-- Create Users Table
CREATE TABLE Users (
    UserID INT PRIMARY KEY,
    Username VARCHAR(255),
    Password VARCHAR(255),
    Role VARCHAR(100)
);
```

Data Collection :

The types of grassroots football matches in England that I can consider for the video analysis project:

1. Youth Football Leagues

- Under 8s to Under 18s Leagues (https://www.londonfa.com/cups-and-competitions/cups/2022-2023/london-u-13-youth-cup-saturday/results)
- Regional youth leagues like Midlands Junior Premier League
- Youth teams associated with professional clubs

1. Adult Amateur Leagues

- Sunday League Football
- Veterans League Football (over 35 years old)
- County Leagues like Middlesex County Football League

1. Schools & Colleges Football

- Inter-school tournaments between academies/high schools
- College football events and competitions
- Intramural football played within universities

1. Charity & Community Matches

- Football festivals organized by communities
- Special Olympics local area games
- Corporate football events

1. Recreational Football

- 5, 7 a side social matches
- Local park kickabouts
- Friendly games between informal teams

For the data collection, I first thought of gathering everything from official club and school websites using Python web scraping. However, I realized I didn't have any information on the players, nor videos of the unofficial matches in England.

After two days of searching with no results, I changed my goal. I decided to collect the 500 videos (just the links, title, number of views, and any other available information). So, I used JavaScript to directly collect the videos from YouTube channels.

```
//Videos from channel
var scroll = setInterval(function(){ window.scrollBy(0, 1000)},

window.clearInterval(scroll);
console.clear(); urls = $$('a');
urls.forEach(function(v,i,a){if (v.id=="video-title-link" && v.l

//Videos from playlist
console.clear();
let goToBottom = setInterval(() => window.scrollBy(0, 400), 100

clearInterval(goToBottom);
    let arrayVideos = [];
    console.log('\n'.repeat(50));
    const links = document.querySelectorAll('a');
    for (const link of links) {
    if (link.id === "video-title") {
        link.href = link.href.split('&list=')[0];
        arrayVideos.push(link.title + ';' + link.href);
        console.log(link.title + '\t' + link.href);
    }
    }
```

With this script, I collected the videos in an Excel file and then converted it to a CSV file.

Data Processing

In my project focused on the analysis and processing of grassroots football match videos, I developed a series of Python functions that are key to preparing and analyzing the dataset. Here's a rundown of the work I've done:

1. `nan_count`: I created this function to identify and count missing values in a DataFrame. It not only gives me a detailed count of NaNs per column but also the total NaNs present in the dataset. Additionally, it checks for duplicate rows, providing a clear picture of data cleanliness.
2. `preprocess_csv_data`: Following the initial data gathering, I implemented this function to clean the DataFrame by removing rows with missing values and duplicates. It also specifically targets the 'Views' column, cleaning and converting string values to integers, making the data analysis-ready.
3. `is_youtube_video`: To ensure the URLs I collected were pointing to valid YouTube videos, I used this function to analyze each URL. It adds a new column, 'Is_Video', to indicate whether the URL is a YouTube video link. This step is crucial to ensure the data's relevance for subsequent analyses.
4. `get_youtube_video_duration`: Leveraging YouTube's Data API, this function enriches the DataFrame with video duration information. By creating a YouTube API client, extracting video IDs from URLs, and querying the API for each video's duration, I was able to add this valuable data directly to the DataFrame.
5. `convert_duration_to_minutes`: After fetching the video durations in ISO 8601 format, I wrote this function to convert those duration strings into minutes. This simplification makes it easier to analyze video lengths, enabling straightforward comparisons and filtering based on duration.
6. `filter_videos_by_duration`: This function helps me focus on videos that are long enough to be meaningful for analysis by filtering the DataFrame to include only videos longer than a specified duration. It's a critical step for narrowing down the dataset to the most relevant videos. With this function I save only the videos link that have duration over 15min.
7. `save_dataframe_to_csv`: Lastly, I used this function to save the cleaned and enriched DataFrame as a CSV file. This step is vital for data sharing and further analysis, making the processed information easily accessible for future steps in the project or more in-depth analysis.

Together, these functions automate the essential steps of cleaning, preparing, and enriching the video data, laying a solid foundation for advanced analysis and the development of applications based on insights from grassroots football match data.

Analyze & Model

For this section I decided to follow these steps :

1. **Techniques and Tactics**:
    - **Player Positioning**: Analyze the positioning of players on the field to understand formation strategies and movements.
    - **Ball Possession**: Calculate the time of possession for each team.
    - **Passes**: Count the number of successful and unsuccessful passes.
    - **Shots**: Track the number of shots, shots on target, and shots off target.
    - **Dribbles**: Evaluate the effectiveness of dribbles and attempted ball retrievals.
2. **Physical Performance**:
    - **Distance Covered**: Measure the distance covered by each player during the match.
    - **Sprints**: Count the number and duration of sprints.
    - **Intensity**: Assess periods of high activity and rest.
3. **Match Events**:
    - **Goals**: Record the moments and circumstances of each goal.
    - **Cards**: Note fouls and yellow or red cards.
    - **Substitutions**: Track player changes and their impact on the game.
4. **Team Movement Analysis**:
    - **Offensive and Defensive Transitions**: Observe how the team repositions between defense and attack.
    - **Pressing**: Analyze how the team presses the opponent when not in possession of the ball.
5. **Advanced Statistics**:

- **xG (Expected Goals)**: Use statistical models to estimate the probability that a goal-scoring opportunity will result in a goal.
- **Heatmaps**: Create heat maps to show where players are most active on the field.
- **Passes Networks**: Visualize passing networks between players to determine strong and weak links.

1. **Techniques and Tactics**:

To begin with, I focused on video collection and frame extraction. Here's a detailed report on what I accomplished:

Video Collection:

- Initially, I loaded a dataset from a CSV file named 'Grassroot_data_15.csv' that I preprocessed, which contains URLs to various grassroots football matches.

  To facilitate the work for my computer, the dataset was filtered to select a random sample of 5 videos for a focused analysis.

- I implemented two Python functions to download videos from YouTube. The first function, `download_videos`, was designed to download videos directly using their URLs from the DataFrame to a specified directory. However, to ensure the quality of the videos, I opted for a more refined approach with the `download_best_available` function. This function prioritizes downloading videos in 720p resolution but falls back to the best available resolution if 720p isn't available. This approach helped me secure high-quality video content for analysis.
- The videos were successfully downloaded to a local directory, setting the stage for the next step in my project.

Frame Extraction:

- With the videos downloaded, I then focused on extracting frames from these video files. Frame extraction is crucial for analyzing specific moments in the matches and applying computer vision techniques.
- I used OpenCV, a powerful library for computer vision tasks, to implement the frame extraction process. Two functions were created for this purpose: `extract_frames` extracts frames from a single video file, while `extract_frames_from_all_videos`

automates this process for all videos in a specified directory. The latter function ensures that frames from each video are organized into separate directories, making it easier to manage and access specific frames for analysis.

- This organized approach allowed me to collect frames from all downloaded videos without skipping any frames, ensuring a comprehensive dataset for detailed visual analysis.

For the next part of my analysis, I had to work on just one frame at first because my computer would crash every time I tried to work directly with the videos (I tried with just one video). I ended up picking a random frame to start my analysis. I also used Colab because my PC kept crashing as I had to load all the frames into a variable before choosing just one.

In the next part of my project analyzing grassroots football match videos, I focused on detailed analysis using just one frame, due to technical issues I faced when trying to process entire videos. My goal was to identify the positions of players and classify them by team based on the color of their jerseys. Here's a summary of this part of the project:

Frame Selection and Preparation:
After downloading the videos, my computer struggled to process the full videos, which led me to select a single frame to begin my analysis. I turned to Google Colab to bypass my PC's performance limitations, allowing me to efficiently load and analyze the frames.

Analysis of the Selected Frame:
I used pre-trained YOLO models to identify players and the ball on the selected frame. This method accurately detected on-field elements, including players, by extracting their positions and annotating them on the image.

Classification by Jersey Color:
The next step was to classify players based on their jersey colors. I utilized a custom function to annotate the frame with ellipses around the players, using predefined color codes to differentiate between the teams.

Interacting with OpenAI API:
To refine the analysis, I integrated interactions with the OpenAI API, sending the annotated frame and receiving classifications based on the jersey colors of the players. This step was crucial for understanding which players belonged to which teams, based on the visual analysis performed by AI.

Results and Visualization:
The outcomes were converted into a usable format, allowing me to update the class identifiers of the detections based on the API's responses. I then visualized the frame with updated annotations, reflecting the positioning and team affiliations of the players, greatly enriching my analysis.

Now I'm working on tracking the ball. I'm still at the beginning and plan to continue using Supervisor, YOLO, and OpenCV for tracking both the players and the ball. I also intend to find a way to save all the data I'll collect on the players directly into a database.

What I Have Left to Do:

First, I need to finish analyzing the videos and generalize the process across all 500 collected videos. Afterward, I will be able to save the data gathered from the videos into a database to conduct statistical analysis and, if possible, create a dashboard.

Completing the Plan:

1. **Generalize Video Analysis**:
    - Develop a scalable and efficient process to apply the analysis performed on sample videos to all 500 videos. This may involve automating the detection and tracking workflows and ensuring that the process can run with minimal manual intervention.
2. **Database Integration**:
    - Design and implement a database schema (review my first schema that I did) that can store the extracted metadata, player performance metrics, and game insights in a structured format. This step involves choosing the right database technology SQL based on the data's nature and the analysis needs.
3. **Data Storage**:
    - Write scripts or use ETL (Extract, Transform, Load) tools to transfer the analyzed data into the database. This includes ensuring data integrity and optimizing storage for efficient querying.

4. **Statistical Analysis**:
    - Once the data is in the database, perform comprehensive statistical analyses to uncover trends, patterns, and insights about player performances and game strategies. This could involve using statistical software or custom Python scripts with libraries such as Pandas and SciPy.
5. **Dashboard Development**:
    - Design and develop an interactive dashboard that visually presents the analysis findings. I think, I'll use PowerBI because it seems easy and free.
6. **Testing and Optimization**:
    - Thoroughly test the data pipeline, database, and dashboard to ensure they work as expected and efficiently handle the data volume. This includes performance tuning and security enhancements.

Challenges:

The main challenge I faced during this project was data collection. I spent two days without finding anything, but with advice from seniors in Discord groups and Stack Overflow, I managed to change my strategy to better adapt to the project's requirements.

I had a significant limitation regarding my computer (my RAM and GPU are very limited), and my PC would crash every time it had to process a somewhat large amount of data.

I found myself forced to use tools I had never used before, such as JavaScript, Supervisor, and the GPT-4 API, but I was very pleased to learn them.

I was also disappointed that I couldn't create my own model and have the freedom to collect everything I wanted from the videos because there were too many images per frame (a single video gave me about 390,000 frames, which is huge to label one by one on Stack Overflow. Moreover, I was severely limited by time, so I had to use a pre-trained model, and there were only two classes that corresponded to my project (detection of "person" and "sport ball").

What I learn and How I fixed the Challenges:

To overcome these hurdles, I turned to cloud computing resources and platforms like Google Colab, which provided the necessary computational power without the need for a high-end PC. This approach allowed me to process data more efficiently and avoid system crashes.

Leveraging community support and online forums became an invaluable part of my learning and problem-solving process. Engaging with experienced developers and data scientists helped me navigate the complexities of machine learning and computer vision technologies.

Accepting the limitations of pre-trained models, I focused on optimizing the analysis and detection processes within the constraints of available resources. This pragmatic approach enabled me to make progress despite the challenges.

Adapting to new technologies and tools not only broadened my skill set but also enhanced my project's overall quality. Learning to use JavaScript, Supervisor, and GPT-4 API opened up new possibilities for data collection and analysis, enriching my experience and the project's outcomes.

While the inability to create a custom model was a setback, it taught me the importance of flexibility and resourcefulness in data science projects. Using pre-trained models as a starting point, I was able to focus on other aspects of the project, such as data preprocessing and interpretation, which were crucial for achieving my objectives.

Despite facing many challenges, I was truly happy to work on this project because I learned a lot, even though not all methods were the best. This experience has made me even more excited about my internship because it has broadened my understanding of what I will need to learn during this time. I'm looking forward to having the opportunity to learn all this and to discover the best practices for an engineer or project manager when tackling a complete project.