CSCI 566 Project Final Report - Beyond the Scoresheet: New Value from Soccer Footage through Computer Vision

Anikeya Aditya
aaditya@usc.edu
Tian Sang
tians@usc.edu
Zeyuan Wu
uwu@usc.edu

Abstract

This project aims to apply computer Vision and transfer learning to create a soccer game analysis system. YOLO (You Only Look Once Jocher et al. [2023a]), a popular algorithm for object detection in computer vision, was used to detect players, the ball, and referees and track their movement across the video frames. As a result, quantitative feedback can be provided on players' movement throughout the game, including speed and distance covered in the game. Additionally, we applied GCN (Graph Convolutional Networks) to predict the position of missing players from the camera, and LSTM (Long Short-Term Memory) to predict the future position with given data.

1 Introduction

In the top-tier European soccer leagues, teams apply data science, statistics, and advanced AI technologies in soccer training to optimize the performance of each player and develop team strategies (Carey, DeBlois-Beaucage). They use advanced GPS wearable and a multi-camera setup to track every movement on the field to provide such feedback (Borodastov [2023]). However, with a limited budget, teams in lower leagues do not have access to such expensive technology or a team of engineers to implement and analyze the data. Financial constraints make it unaffordable for them to purchase costly devices, like precise GPS tracking systems or motion sensors, as well as high-resolution multiple-angle camera sets. The analysts and tactical coaches in these leagues rely heavily on manual observations, note-taking, and memorized player performance and team strategies. This traditional approach is time-consuming, less accurate, and may overlook subtle but critical aspects of the game. This creates a big competitive divide between rich and poor clubs.

Figure 1: Emblem of few soccer clubs in Premier League Arise News [2021].

Preprint. Under review.

The key challenges in this field are technical limitations and data quality issues. First, it is challenging to collect visual data from limited sources of cameras. For example, observing the real-time ball movement from some angles is difficult. On the other hand, it is costly to analyze the whole game for tracking each player. Needless to say, obtaining sufficient annotated video data from lower-tier leagues for training and testing learning models might be difficult as well.

A deep learning model combined with computer vision can produce more quantitative results, converting 3D movements into 2D and tracking the trajectories of the target ball and players. With this tool, analysts and coaches can have personalized training plans for each player to overcome his or her weaknesses, and help the team to manage game time efficiently. Moreover, this cost-effective solution can help the lower leagues to level up and bring a fair game with other teams.

Tracking of players from video footage has been tried using different methods to obtain accurate player positions on the field (Maglo et al. [2023], Yang et al. [2024]).In this project, we have used multiple machine-learning models to develop a game analysis system from video footage. We use YOLOv8 (Jocher et al. [2023b]) to extract player and ball movement data from video game footage. We developed our own code to keep track of player ID across frames and calculate the total distance traveled and instantaneous speed per frame. We tried to implement a GNN-based model like GCN (Scarselli et al. [2009]) to interpolate missing data in frames and predict team formations based on the position of the soccer ball and available players. Also, We used an LSTM (Hochreiter and Schmidhuber [1997]) model to try to predict future player positions.

2 Related Work

There are two major types of the existing methods, including physical devices that assist movement tracking and the data analysis techniques. For the first category, GPS tracking with wearable devices or on clothes can collect data on the players' performance (SoccerTAKE [2023]). These devices can measure moving distance, running speed, acceleration, and heart rate. Also, the multiple camera sets can provide detailed data on player positions, passing, and tactical patterns (Borodastov [2023]). In general, the GPS tracking method can provide physical performance and real-time tracking, but it is not comfortable to wear the GPS vest, and experts need to be involved in interpreting the data. While video can be a good resource of raw data, however, handling and processing large volumes of video data like this needs robust computational resources, which can be costly.

Another recent work (Maglo et al. [2023]) combines a field registration and player tracking algorithm through video pipline. This method enables projection from frame space to pitch space with homographic transformation based on an encoder-decoder model, where heatmaps are generated to locate key points on the pitch. Unlike earlier works that either leverage convolution-based models or use a uniform distribution to represent key points in the pitch space (Nie et al. [2021]) for field registration, this transformer-based approach captures global pitch features to better handle partial views and camera motion. The 2D player tracking algorithm employs a two-phase strategy: initially associating tracklets of consecutive frames with bipartite matching and distance-based criteria, then iteratively merging these tracklets via a self-supervised re-identification (re-ID) scheme that fine-tunes a re-ID model on the specific video. By combining transformer-based field registration with specialized re-ID fine-tuning, this approach can significantly increase the accuracy when extracting the player trajectories from video.

Due to the limitation of single-camera view, some players may be outside the frame and not detected by the model. To solve this, recent contributions from Yang et al. [2024] focus on creating a full-field video via the homography-based merging of two half-field clips, thereby circumventing the limited availability of full-field training data. In contrast to the approach introduced by the article we discussed before, this approach directly integrates detection and tracking within a single attention-driven framework. Unlike the previous method, this solution focuses on reducing ID switches by combining CNN feature extraction with transformer encoders and decoders structure for query-based track maintenance. Although ID increase and switches still happen, this framework includes a correction phase after initial tracking to mitigate identity inflation and inconsistent matching, which is important to our work.

3 Method

We trained two models using YOLO(Jocher et al. [2023a]) architecture. The first model is designed to detect players, referees, and balls from single frames of video footage. The second model identifies vital points on the pitch so that we can use the output from the two models to map players' positions on a 2D pitch, providing a birds-eye view of the game.

YOLO is a state-of-the-art model (Redmon et al. [2016]) that can perform object detection, key point detection, and segmentation tasks on visual data. It uses a single neural network to detect all objects in one pass, giving it a significant speed advantage over other methods that require multiple passes (YOLOv8 Documentation [2023]). Over the years, several versions have been released; for our project, we have used the YOLOv8 model, which was introduced in 2023 and offers higher accuracy and real-time speed than previous versions. It is lightweight and requires fewer computational resources, making it suitable for real-time speed applications (YOLOv8 Documentation [2023]). YOLOv8 consists of a convolutional neural network (CNN) that can be divided into two main parts, the backbone and the head. It has a modified version of CSPDarknet53 architecture Bochkovskiy et al. [2020] as the backbone, which consists of 53 convolutional layers with partial cross-stage connections. The head contains multiple convolution layers followed by fully connected layers for the prediction of labels (Labellerr Blog [2023]). Yolov8 introduces several changes compared to its predecessors, like mosaic data augmentation, anchor-free detection, a C2F module, a decoupled head, and a modified loss function (VISO AI [2023]). These changes improve the model's speed, accuracy, and robustness.

We used a publicly available dataset from Roboflow (Dwyer et al. [2024]) to train and validate our YOLO-v8 models effectively. This dataset contains annotated images featuring outfield players, goalkeepers, referees, and the match ball captured from various game scenarios. Our goal is to improve the performance and make them more suitable for our specific player and pitch keypoint detection tasks. We splitted the dataset into three sections: 92% of them are for training, 3% for testing, and 5% for validation, ensuring a balanced approach to model development and generalization.

Soccer game videos often have some inevitable changes or complex situation like players being blocked, different camera angles, and varying lighting. All of these would make the learning model less reliable. To address this problem, we implemented data augmentation techniques in two steps. For the first step, we fine-tuned the YOLOv8 augmentation parameters systematically through the model's configuration (YAML) files, to change hue shifts, saturation and brightness adjustments, rotations, scaling, and horizontal flips. This approach allowed our model to adapt to the visual distortions that could be encountered when processing soccer footage. In the second stage, we used data augmentation tools to increase the size of the dataset three times larger by adding Gaussian blur up to 4px and introducing noise that affected up to 0.18% of pixels. We chose these two parameters because they are not available in the YOLO model's configuration default settings. As the result, with these two types of data augmentation successfully applied, we created a more powerful detection model.

As mentioned in the related work for data analysis, there is a significant challenge in maintaining the consistency of player tracking, especially in fast-paced soccer environments. Initially, we applied the ByteTracker (Zhang et al. [2022]), which is an advanced tracking algorithm. By assigning unique identifiers to each detected object across the video frames, ByteTracker aims to create consistent tracklets that capture players' movements. This algorithm is capable of working well for the multi-object case with intricate scene interactions.

However, considering the inconsistency of camera views through time, we found that identity inflation is inevitable, making further analysis difficult. To mitigate this problem, we implemented a more nuanced custom tracking method with limited maximum number of players in each time in the camera view. Instead of using ready-made tracking algorithms, we matched player positions between consecutive frames based on spatial similarity and movement patterns. At the same time, we established a distance-based threshold for players who frequently enter and exit the camera's field of view. We set the maximum distance traveled between two frames to 0.5 meters; this was based on the maximum speed in meter per second data recorded for soccer players (Govind [2023]). Since two frames are 0.04 seconds apart, this limit helps keep tracking reliable. These techniques and adjustments allowed us to overcome the limitations of normal tracking algorithms and achieve more accuracy and reliability in retrieving player movements.

Player tracking alone is insufficient for comprehensive analysis of team sports like soccer – it's almost as important to capture the team dynamics and interrelationships between teammates and opponents. We applied machine learning approaches to categorize players into different teams. We first obtained players' crops based on the bounding boxes in the detection model's output, and then we employed Sigmoid loss for Language-Image Pre-training (SigLIP) (Wang et al. [2023]) embedding to generate feature vectors from players' crops. Compared to arguably more influential work like Contrastive Language-Image Pre-Training (CLIP), SigLIP replaced the loss function with pair-wise sigmoid loss, reportedly resulting in better performance in zero-shot classification accuracy (Documentation [2023]). To make the high-dimensional embeddings more manageable, we employed Uniform Manifold Approximation and Projection (UMAP). This approach reduces the 768-dimensional vectors to three-dimensional space while preserving the essential relationships between data points, allowing for more efficient clustering while maintaining the fundamental structural information. Then we leveraged K-means clustering to divide players into two disjoint groups representing each team on the pitch. Since the goalkeepers typically don't wear jerseys of similar color to other players, we separated them from the outfield players and assigned them to the team with the closer team centroid, given that goalkeepers mostly stay inside their half. This holistic approach provides an automated method for team identification that goes beyond basic color-based detection models.

With a clearer understanding of player positions and team momentum, our analysis still lacks the temporal perspective needed to provide predictions about future player movements and team formation changes. This limitation is especially pronounced in our project for two reasons: (1) our input videos are only 30 seconds in length, and (2) players could be missing in many frames due to reasons mentioned earlier. To resolve this problem, we implemented a predictive model using Long Short-Term Memory (LSTM) networks. LSTM is well-suited for our task because it took sequential data to generate predictions. Again, we used the player position data from our detection model as input, splitting it into sliding windows of sequential frames. Each sequence is comprised of twelve frames; eight past frames are used to predict four future frames.

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix},$$

$$c_t = f \odot c_{t-1} \; + \; i \odot g,$$

$$h_t = o \odot \tanh\big(c_t\big).$$

To deal with the missing player problem, we employed the Graph Convolutional Network (GCN). GCN treats each player's position as a node in the graph and connections between them as edges, accounting for the special relationships between players. With this method, we are allowed to encode contextual information about player interactions into our model and thus make more robust predictions.

$$\mathbf{h}_i^{(k+1)} = \Phi\left(\sum_r \sum_{j \in \mathcal{N}_r^i} c_r^{ij} \mathbf{W}_r^{(k)} \mathbf{h}_j^{(k)} + c_r^i \mathbf{h}_i^{(k)}\right), \tag{1}$$

The propagation rule for GCNs shown in this equation is adapted from the work of Zitnik et al. Zitnik et al. [2018].

4 Results and Discussion

We trained two YOLOv8 models to detect players and key points on the football pitch, respectively. We used the data augmentations that were outlined in the midterm report. The mAP50 metric for the player detection and pitch detection models are given in 2 and 3. This shows that both the models have achieved over 90 % accuracy.

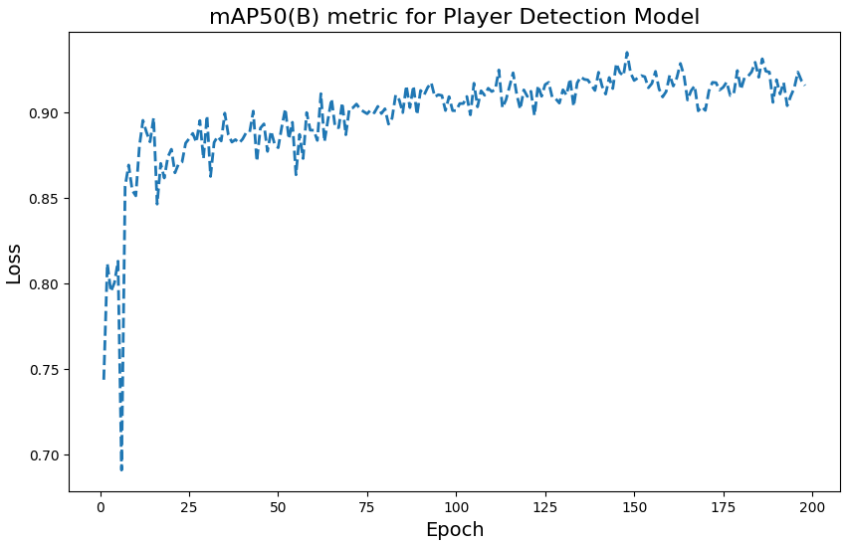

Figure 2: mAP50(B) metric for player detection model. We used the augmentations outlined in the midterm report and trained for a long time of 50 epochs of patience. The training terminated on its own.

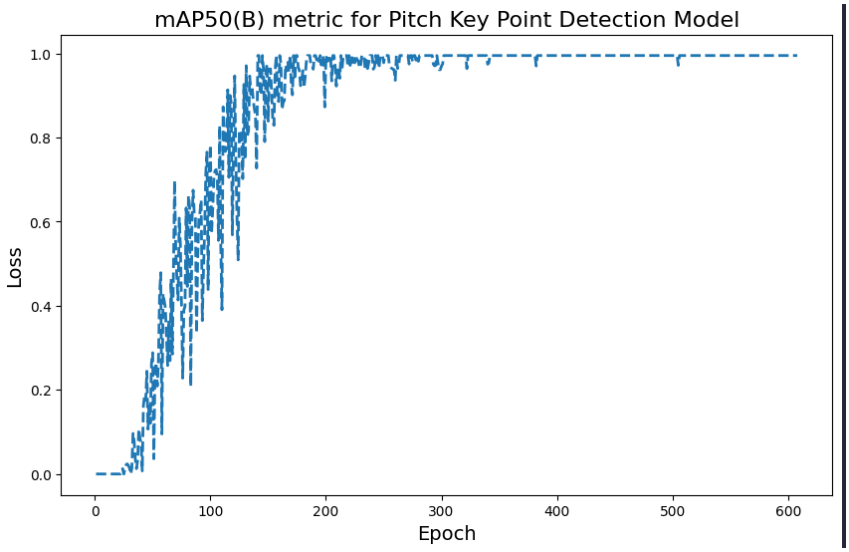

Figure 3: mAP50 metric for the pitch detection model.

Figures 4 to 6 show an example of a frame taken from a video. Figure 4 shows the players, goalkeepers, and referees detected, and Figure 5 shows the players and goalkeepers being categorized into two teams. Then, the homography is used to extract players' x and y positions from the video frames and plotted on a soccer pitch image to develop the bird's eye view of the game, as shown in Figure 6.

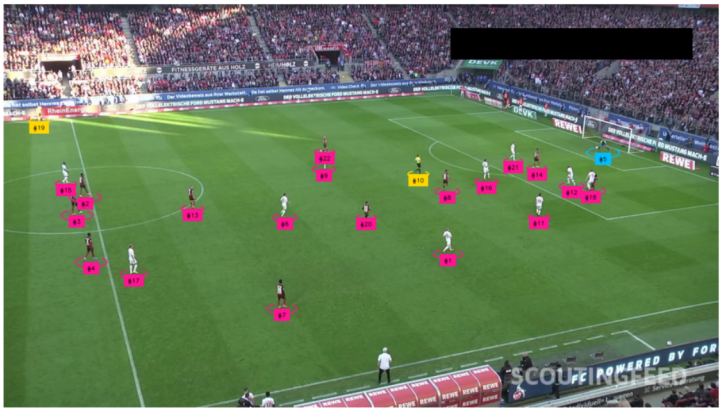

Figure 4: The figure shows a frame from a video that has been analyzed using the Yolo model. Here, the detected players are shown by magenta color labels, the goalkeeper is shown by cyan color labels, and the referees are shown by yellow color labels.

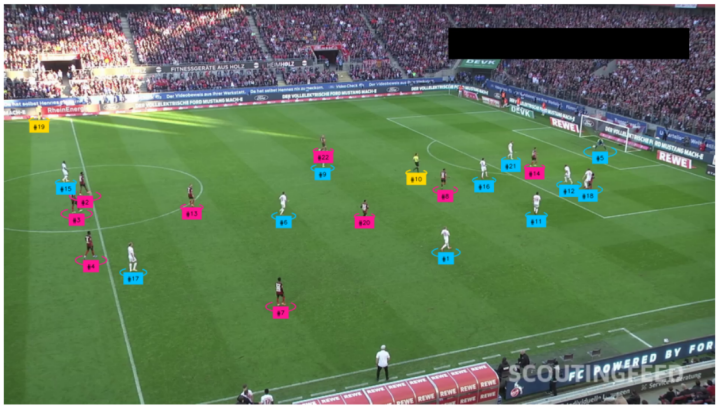

Figure 5: The frame shows the same frame as above 4; here, the players have been categorized into two teams. Here, the players in white jerseys are grouped in one team (cyan label), and players in red jerseys are in another team (magenta label). The goalkeeper has also been categorized into the correct team (blue).

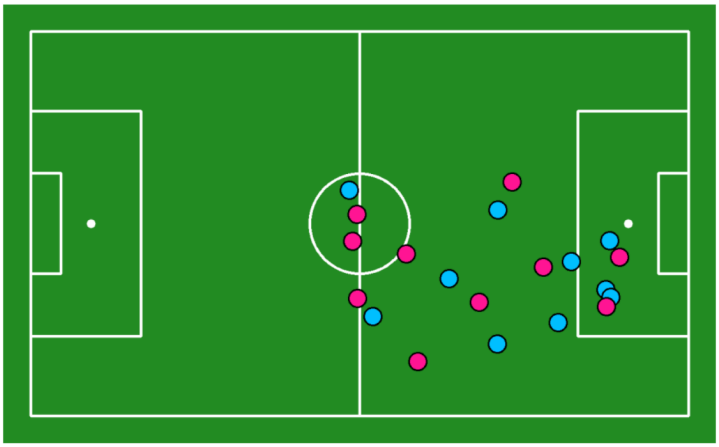

Figure 6: This figure shows the radar map (bird's eye view) of the frame mentioned in 4 and 5. Here only outfield players for each team are shown as they will used for further analysis.

After extracting the player's x and y positions from the video frame, the next challenge was to keep track of individual players across the frames so that their performance metrics like distance traveled and speed can be calculated. To keep a consistent player ID, we developed a code that tracked the players across the frames as described in the methods section. An example of the player ID from a video frame is shown in Figure **??**. Once the players are consistently tagged with ID, we calculated the distance traveled by each player, shown in figure 8, and the instantaneous speed of each player is shown in figure 7.

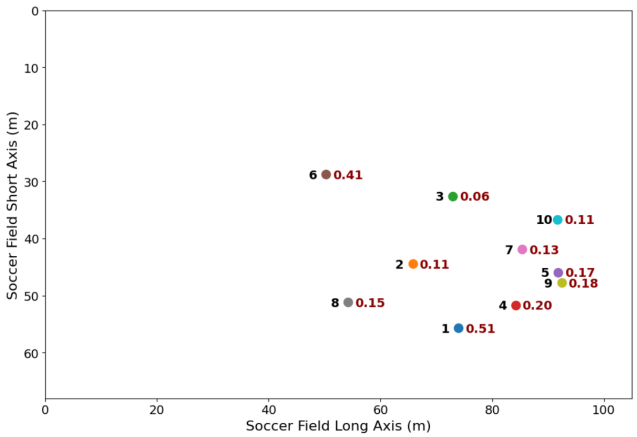

Figure 7: Players were assigned an ID value at the first frame, and then a consistent ID value was maintained by tracking the closest player in subsequent frames from the same team. The figure above shows the id values for the frame shown above. We decided to display only one team (blue dots) for better visual display. We can also calculate each player's speed (red text label) in m/s for a given frame by dividing the distance covered between adjacent frames and dividing by the time between the frames.

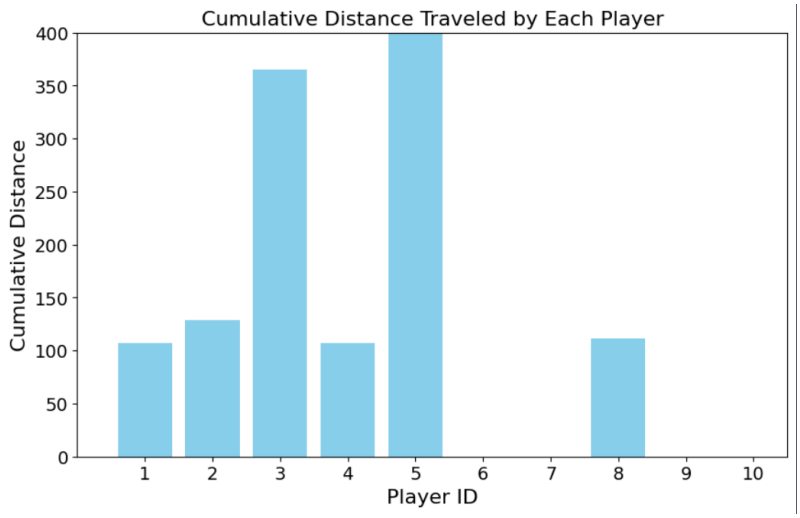

Figure 8: The figure shows the distance traveled by some players of a team from the video clip. Some players do not have a distance value as some players were not detected in many frames of the video.

One of the main challenges we have faced in using YOLO was that data extraction from video footage is inconsistent. If the player's jersey colors are not contrasting with the background or they are overlapping even partially, some players do not get detected, leaving a lot of incomplete data. These lead to incomplete cumulative distance calculation, as shown in Fig 8.

For our case, we had 5 videos of 30 seconds in length with 750 frames each, i.e., 1 frame every 0.04 sec, so our dataset was limited. If we analyze longer videos and can get 1 good frame every 0.1 seconds (here, a good frame means 18 or more players detected out of a possible 20), then we can create a good dataset for overall game analysis and further model training like GNNs to study the gameplay strategy.

With the current 5 videos of 30 seconds data, we decided to use GCN to analyze the player position and predict the missing players' positions. The first step was to identify missing players across various frames in different videos. With the statistical analysis shown in Fig 9, the histogram shows how frequently players' positions were not included in the video and the average number of these missing player is nearly 3 for each frame. This analysis convinced us that with the completeness of the current input data (19 of 22), it could be reasonable to use graph-based modeling to study.

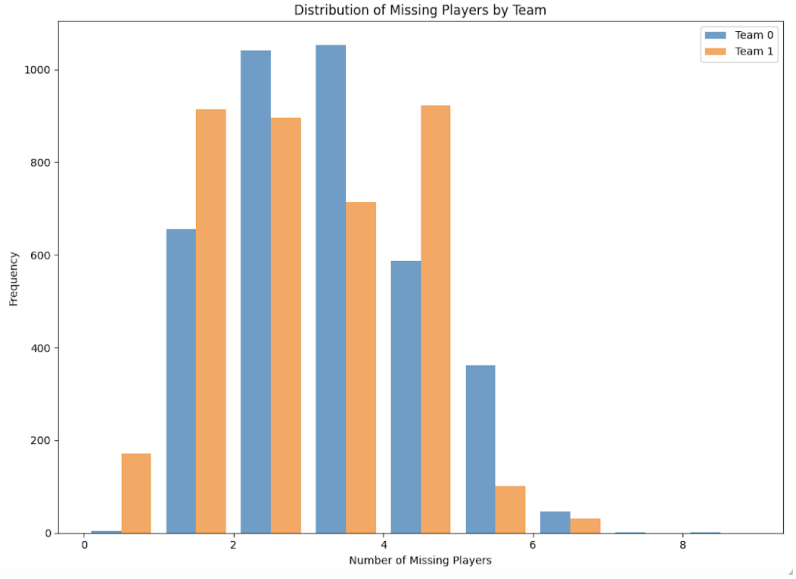

Figure 9: The figure shows the histogram of the frequency of missing player numbers in each team.

During the data preprocessing stage, the positions from input dataset generated by computer vision results were normalized to a standard scale. Each node represents a player with a 5-dimensional feature vector, even those who were absent in certain frames. This is done by inserting placeholder nodes for missing players, each initialed with some neutral coordinates with special flag indicating they were missing. Then the graph was constructed by connecting players from the same team within each frame. The other most important feature for graph structure is the edge features between the nodes, here we used the pairwise Euclidean distance calculated from their positions. Through this data process, our model was provided with uniform data with both present players and placeholders for managing the missing players to maintain a consistent graph and effectively learn the patterns across videos and frames.

With training and validation splitted, our model was trained used a GCN architecture with 3 graph convolutional layers each followed by bach normalization layer and ReLu activation function and dropout layer with probability of 0.2, then followed by a fully connected layer to predict the position x, y for each player. The combined loss funciton shown in Fig 10 plotted with the prediction of presented players instead of the missing ones to confirm that our model is valid without overfitting cases. Theoretically, the missing players position should be inside of the play field but out of the camera range, for example, when players are gathering near a goal for one team, another goalkeeper far away from the camera is missing in this case. For the prediction, we tried to push the missing players outside of the current players maximum and minimum position range, which can be thought as the camera range, and then to avoid overlapping positions, certain threshold was applied to keep players from each other for a small distance. A frame of predicted result is shown in Fig 11, which corresponds to the situation we just discussed with team 0 goalkeeper P6 missing. However, this goalkeeper was missing from the whole video clip that is the input dataset, so the model could not

correctly predict the reasonable position of the missing goalkeeper. As another example frame of prediction shown in Fig 12, we can find the prediction of missing position of team 1 goalkeeper P1 is located in his proper position near the goal, with some other players position reasonably located, except for the P6 case. This confirms that our GCN model combined with some post-processing steps to make realistic prediction demonstrates a well-trained approach to handle incomplete player tracking data.

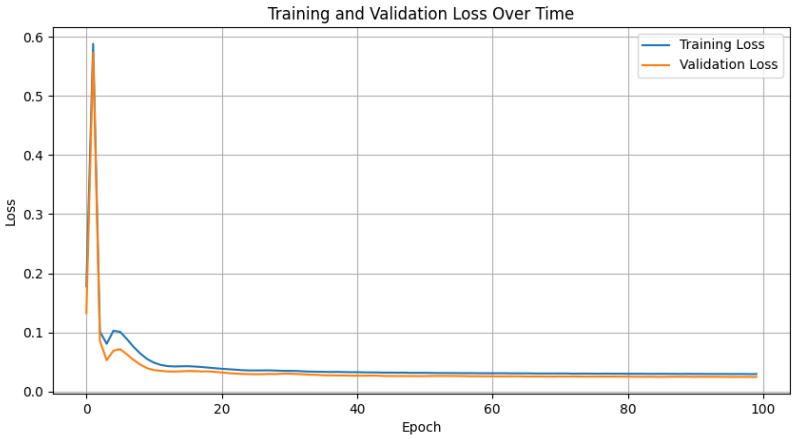

Figure 10: The figure shows the training and validation loss for the GCN model for all presented players. This combined loss function incorporated position loss MSE with temporal loss, which penalized large or sudden changes in player movement during prediction.

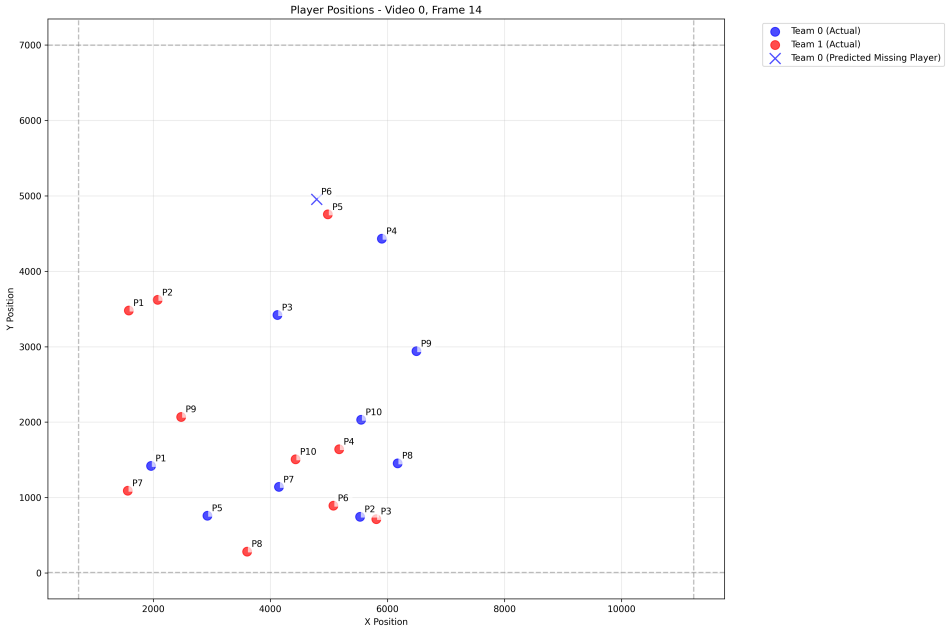

Figure 11: The figure shows the first video and frame 14, actual position of the players in dot, and predicted player position in cross for team 0 as blue and team 1 as red.

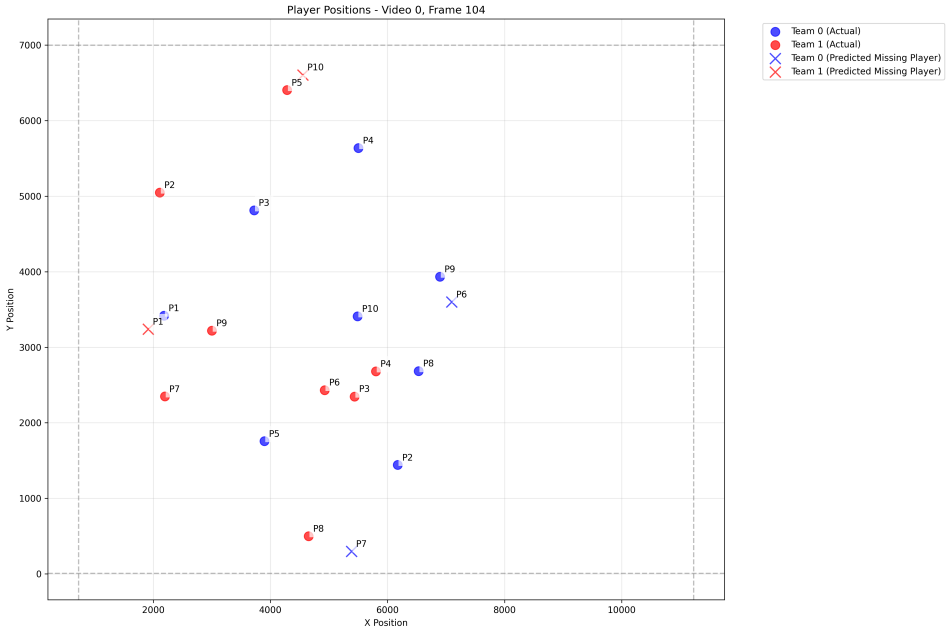

Figure 12: The figure shows another frame, actual position of the players in dot, and predicted player position in cross for team 0 as blue and team 1 as red.

We decided to train an LSTM model to be able to predict the future positions of a player and also use it to fill in the missing data points from the YOLO dataset. We first tried to develop an LSTM model for the entire dataset (5 videos) but realized that it would not work, so we decided to train one model for each player, and this showed great performance, as shown in the figures below. The figure 13 shows the train and test loss of the LSTM model for a player. Figure 14 shows the true data and predicted data for LSTM for a given player. The blue line shows the true X position as a function of time(frame number), and the green line is true Y as a function of time. The predicted values are shown in red and orange. The vertical red line shows the cut-off of training and test data. The frames beyond the red line were not used for training. Figure 15 shows the information in figure 14 as an X-Y trajectory on the pitch. The green diamond marks the starting position, and the red diamond marks the final position. Such an LSTM model per player can be useful in the prediction of short distances and directions that players would travel in critical moments like corner kicks and free kicks.

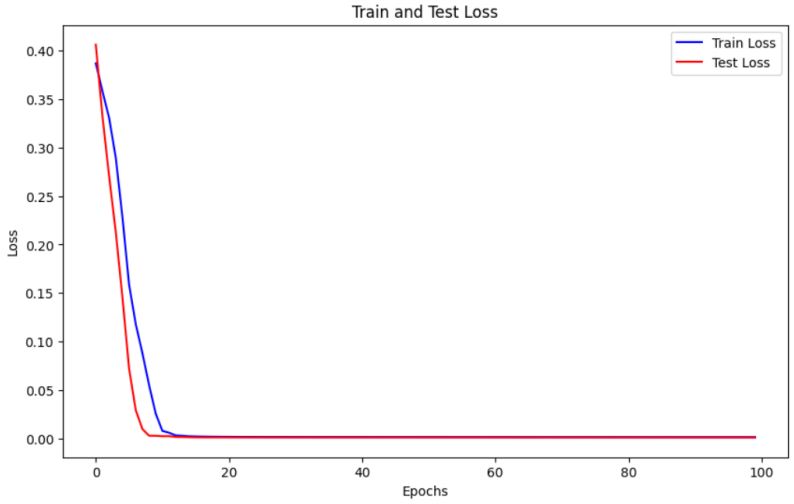

Figure 13: The figure shows the training and test loss for the LSTM model for a single player. The LSTM model takes eight past steps to predict four future steps.

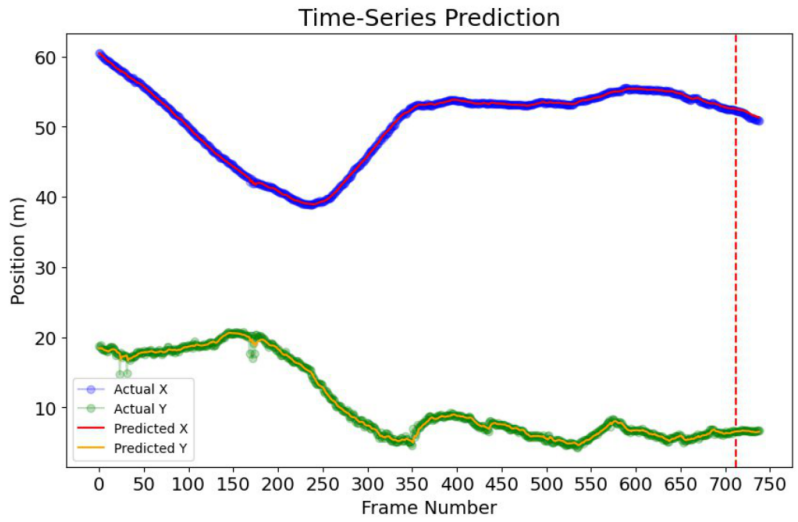

Figure 14: The figure shows the performance of the LSTM model in prediction. The blue and red lines show the X values of the player, and the green and orange lines show the Y values prediction. The red dotted line shows the train test split line. The first 712 frames are used in training, and the last 38 steps are used in testing.

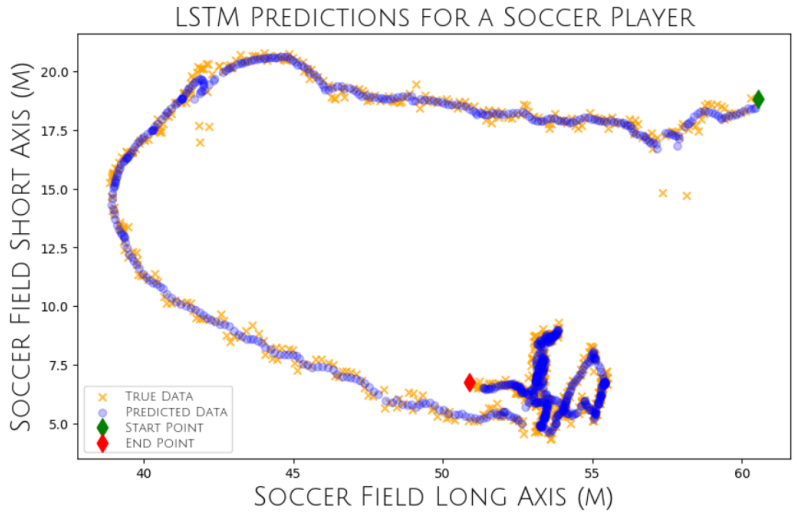

Figure 15: The figure shows the true trajectory and predicted trajectory of a player from Fig. 14. The green diamond marks the player's starting position in the video clip, and the red diamond marks the final position in the video.

5 Conclusion

We had set out to develop a low-cost soccer game analysis system, BTS, using just the game footage. We were able to achieve most of our targets. We were able to train YOLO models to detect players and key points of the soccer pitch to develop a bird's eye view of the football pitch. Then we developed code to tag players with consistent IDs across the frames to calculate each performance metric like distance traveled and speed with which they run. We also successfully developed LSTM models to predict each player's future steps. In the future, we want to continue the development of the BTS system. We plan to first work on the LSTM model to incorporate the ability to predict missing positions where interpolation methods cannot work. We are not sure how successful LSTM would be for such tasks as LSTM lacks spatial awareness and cannot capture spatial relationships between

players on the field. Due to this, we are also exploring graph neural networks (GNN), such as graph convolution networks (GCN) and temporal graph neural networks (TGN). We can use the player's position and ball position as nodes, the distance between players of the same team as one type of edge, and the distance between players of other teams as another type of edge interaction. We want to have a model that can predict team formation (player position), provided the location of the ball on the pitch and what players are present on the field.

References

Arise News. Outrage, threats of ban as 12 european football teams plan to form breakaway 'super league', april 2021. URL https://www.arise.tv/outrage-threats-of-ban-as-12-european-football-teams-plan-to-form-breakaway-super-league/.

Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection, 2020. URL https://arxiv.org/abs/2004.10934.

Mikhail Borodastov. Tracking data: The most detailed and accurate information about players' actions on the pitch. https://footsci.medium.com/tracking-data-the-most-detailed-and-accurate-information-about-players-actions-on-the-pitch-c9 2023.

Mark Carey. How data departments have evolved and spread across English football clubs — nytimes.com. https://www.nytimes.com/athletic/5697684/2024/09/03/football-analytics-uk-evolution/?linked_regilite=google&auth=linked_regilite-google. [Accessed 12-12-2024].

Jérémi DeBlois-Beaucage. Data Science Applied to Soccer: the xPSG Sports Analytics Challenge — medium.com. https://medium.com/the-sports-scientist/data-science-applied-to-soccer-the-xpsg-sports-analytics-challenge-c85402a7193e. [Accessed 12-12-2024].

Hugging Face Documentation. Siglip model doc. https://huggingface.co/docs/transformers/en/model_doc/siglip, 2023.

B. Dwyer, J. Nelson, T. Hansen, et al. Roboflow (version 1.0), 2024. URL https://roboflow.com. Software.

Govind. Top 10 fastest footballers in history based on metres per second covered, December 2023. URL https://khelnow.com/football/2023-12-world-football-fastest-footballers-based-metres-per-second-covered. Accessed: 2024-12-13.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9 (8):1735–1780, 11 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLO. https://github.com/ultralytics/ultralytics, January 2023a. Version 8.0.0, AGPL-3.0 License.

Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolov8, 2023b. URL https://github.com/ultralytics/ultralytics.

Labellerr Blog. Understanding yolov8 architecture, applications, and features, 2023. URL https://www.labellerr.com/blog/understanding-yolov8-architecture-applications-features/.

Adrien Maglo, Astrid Orcesi, Julien Denize, and Quoc Cuong Pham. Individual locating of soccer players from a single moving view. *Sensors*, 23(18):7938, September 2023. ISSN 1424-8220. doi: 10.3390/s23187938. URL http://dx.doi.org/10.3390/s23187938.

Xiaohan Nie, Shixing Chen, and Raffay Hamid. A robust and efficient framework for sports-field registration. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1936–1944, 2021. URL https://openaccess.thecvf.com/content/WACV2021/html/Nie_A_Robust_and_Efficient_Framework_for_Sports-Field_Registration_WACV_2021_paper.html.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. doi: 10.1109/CVPR.2016.91.

F. Scarselli, M. Gori, Ah Chung Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, January 2009. ISSN 1941-0093. doi: 10.1109/tnn.2008.2005605. URL http://dx.doi.org/10.1109/TNN.2008.2005605.

SoccerTAKE. Soccer analytics: How data is changing the game. https://www.soccertake.com/performance/soccer-analytics-how-data-is-changing-the-game, 2023.

VISO AI. YOLOv8 Guide, 2023. URL https://viso.ai/deep-learning/yolov8-guide/#elementor-toc__heading-anchor-14.

Kai Wang et al. Siglip: ... https://arxiv.org/abs/2303.15343, 2023.

Chao Yang, Meng Yang, Hongyu Li, Linlu Jiang, Xiang Suo, Zhen Li, Weiliang Meng, and Lijuan Mao. Soccer player tracking and data correction based on attention with full-field videos. *The Visual Computer*, 40(12):9141–9153, March 2024. ISSN 1432-2315. doi: 10.1007/s00371-024-03300-x. URL http://dx.doi.org/10.1007/s00371-024-03300-x.

YOLOv8 Documentation. YOLOv8 Architecture, 2023. URL https://yolov8.org/yolov8-architecture/.

Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box, 2022. URL https://arxiv.org/abs/2110.06864.

Marinka Zitnik, Monica Agrawal, and Jure Leskovec. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 34(13):i457–i466, 2018.