People Classification and Match Analysis in Soccer

Chiara Cappellino
255264@studenti.unimore.it

Alfredo Onori
249742@studenti.unimore.it

Elisa Zanni
255362@studenti.unimore.it

University of Modena and Reggio Emilia

June 28, 2023

Abstract

The goal of this paper is to offer resources and aids that will make it simpler to understand a game as fast-paced and occasionally chaotic as soccer.
Without the use of any intrusive sensors on the playing surfaces or on the players, it is now possible to automatically extract some statistics to make a useful analysis of the game.
We used the SoccerNet dataset, which is made up of a huge number of soccer frames taken from videos of matches. We implemented our own CNN in addition to using the already-existing Faster RCNN for the detection tasks, and we got excellent results.
This serves as further evidence that new AI technologies, and in particular the Computer Vision domain, can be more than just helpful tools, even in the realm of sports.

1 Introduction

The use of advanced technologies in football has transformed several aspects of the game, improving efficiency, data analysis, and tactical judgment.
This project's first section focuses on the fundamental task of identifying the ball and every person on the football field according to both their physical presence and their assigned roles. The model has been designed to classify individuals as *players, referees, or goalkeepers* using a supervised approach, by utilizing the already labeled data of Soccernet [1] with bounding boxes.
This approach opens the door to a range of interesting possibilities. One of these has been implemented in the second part of our project, which has been built upon the detection achieved in the first task.
Based on what is seen on the playing field, a variety of data and insight can be derived by utilizing the information gathered from the classification process.
This capability provides a unique perspective on the game and enables the extraction of useful data points that can contribute to a deeper analysis and comprehension of football matches.

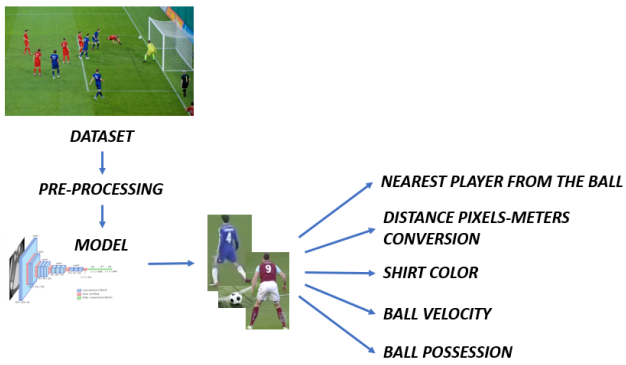

Figure 1: System's inference pipeline

2 Related Works

Our research proposal is in line with the continuing work of numerous companies engaged in artificial intelligence, especially as it relates to soccer. These organizations have actively investigated several aspects of this field after realizing the importance of soccer video comprehension.
One notable example is SoccerNet, which not only offers a large-scale-dataset for soccer video analysis that we utilized to train our models but also hosts contests on a variety of related tasks where teams may compete to always find better results. SoccerNet's extensive dataset is a useful tool for academics and professionals trying to improve the comprehension of soccer games. It includes bounding boxes on annotated videos, which provide essential ground truth information for activities like camera calibration, player re-identification, action-

spotting, and many more.
We have drawn from the existing literature in this field and our project aims to contribute to the growing body of research in soccer video understanding. We add to the body of currently used information and methods while also providing fresh perspectives and ideas. This task expands the scope of existing challenges and research, as it requires addressing multiple classification tasks simultaneously.

3 Datasets

Soccernet-v3 [2] and a custom dataset created to train our CNN were both utilised in our project pipeline. The utilization of these two datasets, combined with their unique characteristics, enabled us to gain a comprehensive understanding of the data required for the training of our models, conducting evaluations, and performing experiments.

3.1 SoccerNet-v3

The first version of SoccerNet dataset consists of videos for 500 games, of which 300 serve as training games, 100 as validation games, and 100 as test games. For our needs, SoccerNet-v3, which offers annotations for 400 SoccerNet games with a split of 290/55/55, was the most appropriate version. It is an exhaustive collection of annotated images that have been extracted as video frames from videos while only maintaining replay and live action frames. We particularly chose it because we didn't need all the temporal data gathered from videos for our tasks.
By utilizing frames rather than full videos, we were able to focus solely on the visual aspects captured in each snapshot and this allowed us to streamline our analysis process and concentrate on the essential visual features present in each frame. The computational efficiency that the SoccerNet-v3 dataset provided for achieving our particular goals was another factor that influenced our choice to use it. Working with individual frames proved to be a computationally lighter technique because our main focus was on solving particular tasks rather than performing intricate temporal analyses.
The latest version of the dataset includes 1,324,732 annotations on 33,986 soccer images, making SoccerNet-v3 the largest dataset for multi-view soccer analysis.
The annotations have many different types, but we focused on the bounding boxes of each player on the field, the ball, and the label that categorizes them.

3.1.1 Adjustment

As we mention before, in order to modify the dataset's contents to meet our needs, processing was necessary. Our primary attention was given to the annotations concerning the ball's and persons' bounding boxes. To achieve this, we took the following steps:

- We kept only the actual action frames, discarding all replay frames. The majority of the frames in the replays were taken from close range, frequently focused exclusively on close-ups rather than capturing the entire soccer field. This observation led us to make this choice. The inclusion of these close-up images in the dataset would have simply interfered with the performance of our models because they were neither pertinent nor helpful for the problems we were trying to solve.

- Each human present on the soccer field was annotated with an individual bounding box and further classified as a left or right player or goalkeeper, a main or side referee, or a staff member.
  For the training of the Faster R-CNN in the multi-class version, we made the decision to divide the human classes into three primary categories: *player, goalkeeper, and referee*. However, in the version with our CNN, we further collapsed these three classes into a single one called *person*.
  The main reason for removing the distinction between individuals was to simplify the task and improve the performance of our models. In this way, we were able to simplify the training procedure and reduce the complexity of the classification task by condensing the human classes and ignoring the precise team affiliation and the distinction between referees.

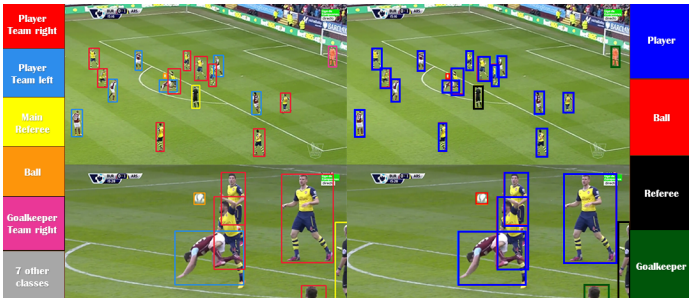

Figure 2: SoccerNet's classes vs adjusted classes

- We removed all other bounding boxes and labels of salient objects, except for those related to the ball.

Description	#Instances
Total number of action frames	10.943
Bounding box for players	130.148
Bounding box for goalkeepers	7.697
Bounding box for referees	13.369
Bounding box for balls	8.382

Table 1: Our SoccerNet version panorama

3.1.2 Football-People

The main goal of our second strategy for the first task was to create a CNN model that could correctly identify a person's class. This required the development of a customized dataset customized to our particular purpose and needs.
We retrieved the three classes' bounding boxes and related labels during the dataset construction procedure. We assured that the dataset contains the information required for training our CNN to recognize and categorize individuals in each of the designated classes by precisely defining and annotating these bounding boxes. By doing this, we could be sure that our CNN would be trained on samples that were extrapolated from the initial dataset and were both relevant and representative.

Figure 3: Images from our dataset

As shown in Table 1, we noticed that the classes were unbalanced. We initially attempted to train the network using an imbalanced dataset, and as expected, the prediction of our CNN consistently favored the player class, which was the predominant one in the dataset.
This result emphasized how the model's behavior was impacted by the class imbalance.
We made the decision to construct a balanced dataset after realizing the need for more precise and balanced predictions across all classes. The CNN's performance has increased significantly as a result of our efforts to give it a more equal representation of each class so that it can learn and generalize more effectively.

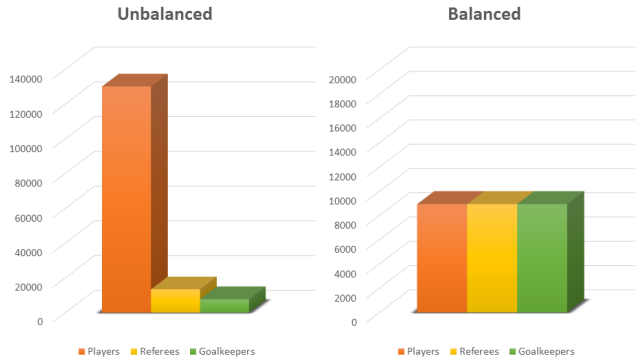

Figure 4: Unbalanced vs Balanced dataset

4 The Approach

Our project is divided into two main tasks, as was already mentioned. For the first task, which focuses on object detection, we have chosen to analyze two different scenarios. In the first, we use a Faster R-CNN [3] model to train a multi-class detection system that can distinguish between people and their various roles.
On the other hand, the second approach involves using a Faster R-CNN model solely for detecting individuals, while our CNN classifies them based on their roles.
By exploring both of these scenarios, we aim to compare and evaluate the performances of the two approaches in terms of accuracy, efficiency, and suitability for our specific task of role-based people detection. This allows us to analyze the strengths and limitations of each approach and make informed decisions regarding the most effective methodology.
Calculating the ball's velocity, the player who is closest to it, and the color of his jersey are all part of the second task. For the purpose of identifying ball possession in a particular video, these statistics are crucial. In this task, we use non-deep techniques and occasionally compare them with deep approaches. By making these comparisons, we aim to better understand the benefits and constraints of traditional Computer Vision methods in regard to our particular goals.

4.1 Detection with Faster R-CNN

Faster R-CNN (Region-based Convolutional Neural Network) with ResNet-50 and FPN (Feature Pyramid Network) is a popular object detection framework that combines state-of-the-art convolutional neural network architectures to achieve accurate and efficient detection of objects in images.
This framework takes advantage of ResNet-50's strong feature representation capabilities, while Faster R-CNN and FPN architecture improve the network's ability to

recognize objects of various shapes and sizes. We chose to train our object detection model using a Faster R-CNN FPN for several reasons:

- **Precision and accuracy**: Recognize and represent fine details in images is a strength of ResNet-50. Given the nature of our project, keeping track of and analyzing minute facets in people's aspect and behavior is crucial to correctly recognizing and categorizing the roles of people like players, goalkeepers, and referees.
- **Robustness to scale variations**: Our model can successfully handle objects of various sizes and scales by utilizing FPN. This is especially important in situations where items could appear at different distances or display different sizes.
  We had to manage scale variations because Soccernet uses cameras at various angles. By utilizing FPN in combination with Faster R-CNN and ResNet-50, we were able to address this challenge and achieve robust object detection.
- **State-of-the-art performance**: One of the most sophisticated and popular methods for object detection is the combination of Faster R-CNN, ResNet-50, and FPN. This powerful combination utilizes the advantages of every component to produce outstanding outcomes.

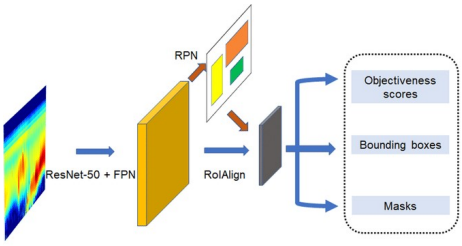

Figure 5: Faster R-CNN FPN model

4.1.1 Model adaptation

We made several modifications to the Faster R-CNN using ResNet-50 and FPN model. First, we adjusted the predictor by modifying the output neurons. The model had 91 output neurons by default, but we adjusted it to 5 to better represent the classes we identified (4) and an additional class for the background, as required by the model.
Another modification we experimented with was the introduction of dropout in the fully connected (FC) layers. Dropout is a regularization method used to prevent neural networks from overfitting. By applying dropout to the FC layers, we aimed to improve the generalization and performance of the model.
Additionally, during the training process, we initially froze the weights of the first layers of the network and used pre-trained weights from the IMAGENET [4] dataset. Through this method, we were able to make use of the information gained from a sizable image dataset. In order to further optimize the network for our job, we also tried training the models without freezing the initial layers.
By customizing the predictor, introducing dropout regularization, and exploring different training strategies, we aimed to optimize the model's accuracy and adapt it to our specific detection requirements.

4.1.2 Pre-processing

After loading the dataset described in section 3.1, we further processed the data through various preprocessing operations. These pre-processing procedures enable better training and increase the model's capacity to handle different variations and difficulties existing in the dataset by ensuring that the input data is suitably transformed and improved.

The preprocessing steps performed on the dataset are as follows:

- **Resize**: To ensure consistency in input sizes and facilitate efficient processing, we resized the images to a specific resolution (1920, 1080). By resizing the images, we bring them to a uniform size, which is important for the subsequent steps of the model pipeline.
- **Tensor transformation**: Tensor inputs are often used by deep learning models. As a result, we changed the images into tensors, which are multi-dimensional arrays that neural networks can handle. This conversion enables efficient computation and facilitates compatibility with the model architecture.
- **Random Horizontal Flip**: We applied random horizontal flips to the images to expand the dataset and provide variability. With a given probability, this technique horizontally reflects the pictures, generating new training examples with altered orientations. The model can manage differences in object appearances and generalize more effectively thanks to this augmentation strategy.
- **Random Photometric Distortion**: It was another technique we used to enhance the data. This method simulates various lighting situations by

adding random variation in image hue and brightness. By applying random photometric distortion, we increase the robustness of the model to changes in lighting conditions, making it more adaptable to real-world scenarios.

Figure 6: Data augmentation

4.1.3 Training

We trained our Faster R-CNN with ResNet-50 and FPN model using a manual cross-validation approach, and we utilized the following parameters:

- **Epochs**: it represents the number of times the entire dataset is used to train the model. By default, it is set to 300.
- **Patience**: During training, we included the ability for early stopping. When the model has reached its maximum capacity for generalization on the training data, training is stopped earlier in order to prevent overfitting.
- **Batch Size**: it specifies the number of training examples used in each iteration during weight optimization. We can alter training efficiency and speed by changing the batch size starting at 4. Although larger batch sizes can speed up training, they also call for greater GPU memory. Smaller batch sizes can reduce memory requirements but may result in longer overall training time.

The following three variables are used to change the weights of the model's parameters using the SGD optimizer [5]:

- **Momentum**: It reflects the amount of inertia that stochastic gradient descent applies to the weight updates. 0.9 is the default value. The influence of inertia is amplified as momentum increases, which may accelerate the model's convergence. However, a value that is too high may cause undesired oscillations during optimization.
- **Weight Decay**: It regulates the weight decay (L2 regularization) that is applied to the model's weights during training. By restricting weight values, it aids in preventing overfitting. 1e-4 is the default value. Increasing the weight decay strengthens the regularization effect, but a too high value may excessively penalize weights and result in decreased model performance.
- **Learning Rate**: It is an indicator of the learning speed used for weight optimization. In the context of parameter tuning, we specifically opted for a higher learning rate of 0.01 during the initial stages of training. This choice was motivated by the fact that higher learning rates can facilitate faster initial progress in terms of weight updates. However, in the later stages of training, when the model gets closer to convergence, we switched to a lower learning rate of 0.001. This adjustment allowed for more precise fine-tuning and fine-grained adjustments to the model's parameters, leading to improved overall performance and better convergence.

4.1.4 Evaluation metric

Because it takes into account both the precision in object localization (how precisely the bounding boxes are detected and delineated) and the precision in object classification (the accuracy in predicting the correct classes for the detected objects), we chose Mean Average Precision (mAP) [6] as the metric to evaluate the performance of our object detection models.

$$mAP = \frac{\sum_{i=1}^{N}(AP_i)}{N} \tag{1}$$

To compute mAP, we first calculate the Average Precision (AP) for each class separately. The precision-recall curve is generated by varying the detection threshold and calculating precision and recall values at each threshold.
In our situation, we tested whether a predicted bounding box matched the actual one using the Intersection over Union (IoU) metric as the evaluation criterion. The IoU threshold establishes the minimal overlap between the predicted and actual bounding boxes that must exist for the detection to be considered successful.

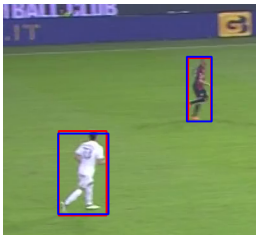

Figure 7: IoU between Prediction and Ground truth

After experimenting with various IoU values, we chose [50, 75] as the list of thresholds to be taken into consideration.
At first, we computed the mAP using the standard measure. We soon discovered, nevertheless, that this strategy was inappropriate for our scenario. The reason was that the class *ball* had either a single instance or was completely absent in many cases, while the other classes had a significantly larger number of instances. Since the *ball* class had a weight that was excessively high in relation to the number of instances in the class in comparison to the total, the classifier's errors only focused on that class significantly reduced the mAP. The model's performance on the other classes, which had a larger number of instances, was therefore misrepresented by the mAP.
To ensure a balanced evaluation, we chose a weighted average precision that takes into account class distribution and gives classes with many instances greater weights.

$$mAP_\omega = \frac{\sum_{i=1}^{N}(\omega_i \cdot AP_i)}{\sum_{i=1}^{N}\omega_i} \tag{2}$$

where ω_i represent the number of istances for class i.

4.2 Different Approach with our CNN

Being able to recognize the assigned roles of individuals is a particularly complex task, because there are minimal visual cues in the clothing that differentiate individuals involved in the game.
As a result, in addition to visual clues, the task of identifying and allocating roles in soccer significantly depends on context, player positions, motions, and other contextual information. Even though Faster R-CNN gave us good performance, we still needed to assess the effectiveness of the model's classifier component in particular.
Therefore, we conducted additional experiments by training an handcrafted CNN to further assess the classification capabilities.
Instead of differentiating between players, goalkeepers, and referees, we chose to train the Faster R-CNN to simply detect the ball and people. Then, to categorize each identified person into one of the three potential roles, we added a custom CNN that was trained using the dataset described in Section 3.1.2
Our CNN's architecture includes a series of layers that sequentially extract useful features from the input data using convolution and ReLU activation.
Its configuration is described in the following image.

Figure 8: Our CNN structure

The specific configuration and number of convolutional and fully connected layers may vary depending on the design choices and complexity of the CNN architecture. In our case, this structure has proven effective in extracting and classifying the essential features for role recognition in soccer videos.

4.2.1 Evaluation metric

With our CNN, we achieved a good classification score, evaluating it with the F1 score metric [7] we got a result of 90.39%, that combines recall and precision to offer a comprehensive evaluation of the model's classification performance.

$$\text{F1-score} = 2 \cdot \frac{precision \times recall}{precision + recall} \tag{3}$$

We used a confusion matrix to offer a thorough analysis of our findings. By showing the counts of true positive, true negative, false positive, and false negative cases for each class, the confusion matrix visually highlights the model's predictions.

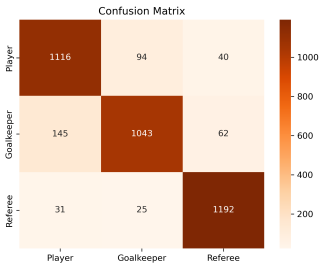

Figure 9: Confusion Matrix

4.3 Detection Results

To find the best model between the multi-class Faster R-CNN and the Faster R-CNN with two classes, to which we added our custom CNN for classification, we trained our models using a variety of techniques.
We tried two distinct methods, as indicated in Section 4.1.1: using a pretrained backbone or training it from scratch, and adding or removing dropout from the Faster R-CNN's last two layers. These modifications were explored to assess their impact on model performance and determine the best configuration for our specific tasks of object detection and classification.
The results obtained using various methods are summarized in the table below:

Approach	Backbone*	Dropout**	Valid mAP
Faster R-CNN - multi-class	✓	✓	70.70%
	✓		76.71%
		✓	**85.03%**
Faster R-CNN - two-classes	✓	✓	65.93%
	✓		76.71%
		✓	82.92%

Figure 10: Result's table

∗: Pre-trained backbone with IMAGENET weights
∗∗: Adding dropout in fully connected layers

We got the best result in validation with the Faster R-CNN in the multi-class version, adding dropout and training the backbone from scratch. We can assume that this result is due to the fact that IMAGENET is composed of images to different from the ones used to test our models.

4.4 Match Analysis

We wanted to go beyond simple object detection in the analysis task and explore more intricate data that would help us better understand a soccer match. The calculation of ball-related measures, such as its velocity and closeness to players, was a crucial area on which we concentrated.
In addition, we aimed to identify ball possession, an important component of soccer analysis. We were able to distinguish between the two teams by determining the color of the players' jerseys and link the ball to the appropriate team during the game.
As previously mentioned, our major strategy for the analytical task involved using non-deep approaches to determine various statistical variables that would be helpful when evaluating a soccer match. We included a deep version of the analysis task to maintain a cohesive workflow and provide a link between the deep task and the analysis that follows.
Our analysis task pipeline consists of the following steps:

1. **Ball detection**
   From the first task, we extract the bounding box of the ball. This choice was driven by the fact that we initially attempted to detect the ball using non-deep methods such as Canny [8] and Hough Transform [9]. However, it was difficult to precisely spot the ball using these non-deep methods due to the low-resolution photos provided by Soccernet. Since ball detection was a key element of the second task, we decided to make use of the bounding box output from the first task, which used a deep learning approach, to ensure more accurate ball detection.

Figure 11: Canny approach

2. **People Detection**
   We used the Histogram of Oriented Gradients (HOG) [10] descriptor and deep learning as two separate methods to identify players. These strategies were used to evaluate their effectiveness and demonstrate the higher level of precision attained by deep learning models compared to conventional approaches.
   The HOG descriptor is a popular feature extraction technique that captures the local gradients and shape information of an image. It examines how gradient orientations are distributed within image patches and visualizes them as feature vectors.
   The HOG descriptor, however, performs noticeably worse when compared to deep learning techniques in terms of accuracy and precision.

3. **Nearest player**
   Once the individuals were detected, we proceeded to calculate the distance between the center of the bounding box of the ball and the center of the bounding boxes of nearby players. This allowed

us to identify the closest player in each frame.
Using a straightforward Euclidean distance formula, the distance was calculated.
The Euclidean distance between two points in a two-dimensional space (x_1, y_1) and (x_2, y_2) can be calculated using the following formula:

$$distance = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1^2)} \quad (4)$$

4. **Distance ball-player conversion**
The size of the ball was used as a reference unit in the estimation of the actual distance between a player and the ball. We used this knowledge to convert the distance in pixels to a distance in meters because it is well known that a soccer ball has a diameter of 23 cm.
We were able to determine a scaling factor by taking into account the ball's diameter in centimeters and converting centimeters to pixels. We managed to translate the distances estimated in pixels from the image into actual distances in meters thanks to this scaling factor. This enabled us to accurately determine the proximity of the players to the ball.

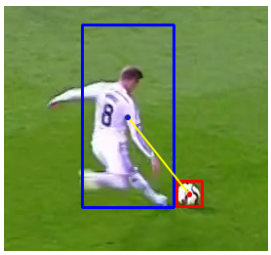

Figure 12: Distance between ball and nearest player

5. **Jersey color**
We then extracted the color of the player's shirt to find out their team affiliation after the player had been located and their distance from the ball had been established. To do this, we extracted the upper part of the player's bounding box and used two masks—a gray mask and a green mask. The gray mask was used to remove white objects such as the goalposts, while the green mask was employed to eliminate the presence of the field itself.
We used the K-means method to extract the main colors from the masked image. The most prevalent color in the resulting clusters was typically black, attributable to the mask. The player's jersey color was represented by the second cluster, allowing us to identify the player's team based on this predominate color.

Figure 13: Jersey color extraction

6. **Ball possession**
Once the nearest players to the ball for each frame were identified, we utilized K-means clustering on the jersey colors of these players to determine ball possession throughout the match. We searched for two unique clusters that represented various teams by doing clustering on the jersey colors. As a result, we were able to analyze each frame individually and precisely determine how many frames each team controlled the ball.

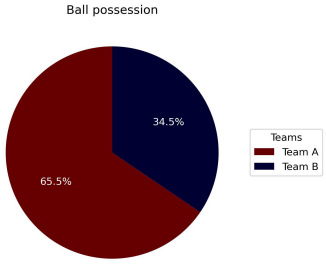

Figure 14: Example of ball possession chart

7. **Ball's speed**
We used a tracking-by-detection strategy to determine the ball's speed as a final metric. In order to do this, the ball's position had to be tracked over frames, and each frame's displacement had to be calculated. We divided the displacement by the frame rate (FPS) to get the ball's speed.

$$Ball\_Speed(m/s) = \frac{Ball\_Displacement(m)}{Time\_Interval(s)} \quad (5)$$

where:

- $Ball\_Displacement$ is the Euclidian distance between the ball's positions in consecutive frames.
- $Time\_Interval$ is calculated as $1/FPS$.

5 Future Purposes

This project opens up a number of fascinating perspectives for further study and application in the area of soccer analysis.

The proposed framework can be modified to handle real-time video streams, enabling in-game analysis and insights. This would offer immediate feedback and tactical information to coaches, analysts, and fans. This structure can also be used to analyze team performance, including game plan optimization, formation analysis, and tactics evaluation.
The incorporation of additional sensor data into the computer vision system, such as GPS tracking or player biometric data, is another potential future direction. Additionally, this project can operate as a building block for the creation of immersive and interactive soccer viewing experiences. Fans can interact with soccer matches in novel ways by utilizing virtual reality or augmented reality technologies, such as by experiencing the game from the perspective of a player or by having real-time statistics and insights overlayed over the live video. Last but not least, this paradigm can be used to other sports like basketball, hockey, or American football that have comparable dynamics and difficulties.

6 Conclusions

In conclusion, we are pleased with the results of our research and the algorithms we used. All of the approaches we successfully investigated for object detection showed excellent results. Through extensive experimentation, we have gained valuable insights into various architectures and gained a deeper understanding of their respective strengths and weaknesses. Finally, we are satisfied with how the two tasks — detection and analysis — were combined since it created a sense of continuity in our method of work. This makes it possible to apply and improve our model even further in order to increase its functionalities.

References

[1] S. Giancola, M. Amine, T. Dghaily, B. Ghanem *"SoccerNet: A Scalable Dataset for Action Spotting in Soccer Videos"*

[2] A. Cioppa, A. Deliège, S.Giancola, B. Ghanem, M. Van Droogenbroeck *"Scaling up SoccerNet with multi-view spatial localization and re-identifcation"*

[3] S. Ren, K. He, R. Girshick, J. Sun *"Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks"*

[4] : J. Deng, W. Dong, R. Socher, L. -J. Li, Kai Li and Li Fei-Fei *"ImageNet: A large-scale hierarchical image database"*

[5] : S. Ruder *"An overview of gradient descent optimization algorithms"*

[6] : A.F. Gad *"Evaluating Object Detection Models Using Mean Average Precision (mAP)"*

[7] : Y. Sasaki *"The truth of the F-measure"*

[8] : J.F. Canny *"Canny Edge detection"*

[9] : A. S. Hassanein, S. Mohammad, M. Sameer, and M. Ehab Ragab *"A Survey on Hough Transform, Theory, Techniques and Applications"*

[10] : N. Dalal, B. Triggs *"Histograms of Oriented Gradients for Human Detection"*