# STATE-OF-THE-ART CNN TECHNIQUES FOR FOOTBALL PLAYERS DETECTION AND CLASSIFICATION

*Daniel Arriazu (s212792), Alberto Caregari (s221794), Iván Serrano (s212477), Davide Venuto (s220331)*
https://github.com/ivans14/Deep_Learning_Proj

Technical University of Denmark

## ABSTRACT

The project focus on a real-world problem of object (player) detection on amateurish soccer video recordings, taken from the Veo Cam 2 system. The task results to be challenging due to the frequent presence of small near-grouped or occluded objects(players and ball), in addition to different illumination and camera viewpoints. [1]
To achieve this task, two of the state of art convolutional neural network models for object detection were used and compared: YOLOv5 and faster R-CNN.

***Index Terms—*** Player identification, CNN, Unsupervised Learning, YoLo, Faster R-CNN

## 1. INTRODUCTION

The fast evolution of Convolutional Neural Networks algorithms starts a new era in the object detection field. After a decade of increasing the quality of object detection based on hand-crafted features, in the year 2010 a saturation point was reached. However, in 2012, CNN's deep learning algorithms were found capable to learn high-level feature representations of an image, and in 2014 with the proposal of the "Region" of detection, a new R-CNN family come into birth.

The demand for performance and statistics data increased in both the professional and amateur sports sectors due to the improvement in object tracking and detection algorithms. In fact, new technologies may have an impact on more than just the video recording features themselves because they enable a deeper analysis of the entire game for purposes ranging from training to sports broadcasting.

This project is based on Veo Technologies challenge for object tracking. In this context, according to VEO's mission was decided to set the project goal on the implementation of a model, for an amateurish sports purpose (analysis on Veo cam 2 recordings), that is capable to identify the ball and the players first and be able to distinguish the two opposing teams in the second iteration. The following step was to decode the previous data onto a 2D representation of the player positions. The YoLov5 and the faster R-CNN algorithms, two state-of-the-art Convolutional Neural Network models for object detection with a one-stage and two-stage detection methodology, were used to complete this task.
For the project development the following milestone were set:

- Data Acquisition and pre-processing (labeling of the ball)
- Player and ball identification using both faster R-CNN and YoLov5 models
- Data handling for teams labeling
- Teams detection with YoLov5 model
- Soccer field demarcation line detection
- 2D mapping of the ball and players

## 2. CONSIDERATIONS

For the first project milestone, regarding the player detection feature, the main goal was to reach a model able to correctly detect all active persons on the field and the ball. For this reason, as a partially labeled data set without any differentiation for the different teams and for the referee was provided, the referees were identified as players.
For the team's detection milestone, the YOLOv5 model was decided to run with the data that were handled in two different ways: the first approach using a dataset of labeled teams obtained from an unsupervised machine learning clustering algorithm; the second one uses a hand-labeled teams data set.
During the process, was decided to redefine part of the first project proposal (2D representation of the player positions). This has to be done as was found out a high discrepancy between the labeled and the real final Veo Cam 2 recording dataset, when it comes to the soccer field demarcation lines. The illumination and the soccer fields' bad conditions, joined with the low camera point of view for the Veo recordings make it very difficult for the algorithm to detect correctly the field border that is the starting point for a Linear perspective analysis. The main intention of this project is to go beyond the techniques explained in the lectures, as well as to understand the power and adaptability of pre-trained state of art models.
Both object detection models have been developed in parallel.

This means that each is contained in one or several code files. For instance, the Faster R-CNN player identification task is held in 2 unique Python scripts for the training step (Dataset.py and Train.py), while testing reuses the same code as the implementations in YoLo.

## 3. DATA PRE-PROCESSING AND HANDLING

The first data set provided by Veo (SN) is composed of a series of folders, each one with 750 frames from a major football match action, and 2 tables related to each of those folders. The two tables had the same data but in different order. Each row of this data set corresponds to an *entity* in a frame, the relevant columns refer to:

- frame number
- track id, for recognizing the *entity* in different frames,
- top left corner of the box where the *entity* is recorded,
- height and width of the box.

Initially, the aim was to detect not only the *entities* but also classify them as person or ball. The option to manually label each image was not taken into consideration, instead, the data set has been exploited: there was the possibility to extract other insight from what was available. As a first approach, the *track_id* has been used as a key to label the ball: since during a football match the ball should be always on the screen, the entity that compares the most in each table should be labeled as the ball. After an accurate inspection was clear that this was not the correct way complete the task: sometimes more the one ball was present in the image (e.g. a player shoots the ball into the stands and the ball-buster throws another ball to the goalkeeper) the *track_id* is coherent and the id of the ball entity is changed. Another approach could be to consider the box of the ball's corresponding area compared to the player's one. This approach has another problem: sometimes the camera can't record the whole player (e.g. only a leg, or the head can be seen). Since they have a row dedicated to the data set thus also a box with an area that is very similar or even smaller than the ball's box area. The solution to this problem was to use a combination of these two ideas. A double filter on the dimensions of the box and a filter on the frequency's entity were used to automatically detect the ball with a high degree of confidence.

At this point, the data set was ready to be used by the deep learning method. Since the results were encouraging (more information further) has been decided to make the data set more complicated. Now the new labels for the entities are three: player team 1, player team 2, and others (that can compare ball, goalkeeper, and referee). Once again the option to manually label thousands of images were not covered.

Here the approach was different: the key to label each player team was the color of their uniform. In order to compute the mean color of the box, each pixel's color has been transformed in a GBR code (three numeric coordinates from 0 to 255 to describe the amount of green red, and blue of the image) and then computed the mean in the usual way, finally, the result was converted in a color to visualize it. Since the box is rectangular and the uniform covers only a small percentage of the box area, ignoring a frame means ignoring some pixels in the computation, this can help to accentuate differences between uniforms. Using the GBR coordinates was easy to link the mean color of each box to a point in a three dimensions space and use a clustering algorithm to find a set of similar colors to label those points. Two different algorithms were taken into consideration: k nearest neighbor (KNN) and Gaussian mixture model (GMM) (see fig. 1). The last one has been chosen because can describe better the shape of the clouds since they should come from one color (that become one point using the GBR code) spoiled by random noise (for more information see [2]). With this procedure, the data set obtained one new column referring to the team to who each player belongs or to a bonus category for all hard-to-classify entities.

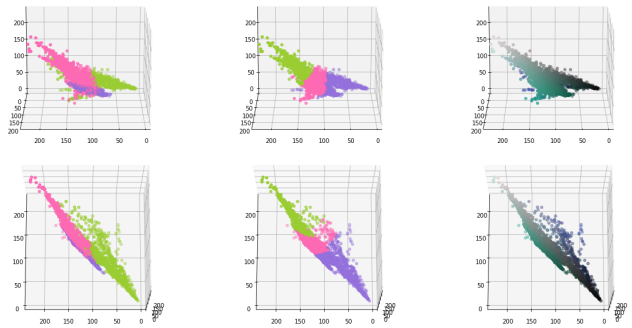

**Fig. 1**. In the left panel is possible to see the division in clusters using GMM, in the middle using KNN and on the right the original mean color of each image. It is clear player's GMM algorithm can better catch the clouds' shape.

A python class called `dataset` has been implemented to fast manage the data set. Thanks to this class can be easily computed the dimension id the boxes for every player and visualize the result in the complete picture with the complete set of players (for an example see Fig. 9).

## 4. PLAYER IDENTIFICATION

This section introduces the team's first milestone: *player and ball identification* on the pitch for any given football game. A vast majority of the latest state-of-the-art object detection architectures, such as *Single Shot Detector* or *R-CNN*, include a set of pre-trained weights downloaded and recorded in the cache when used. This means that the original pre-trained network is, by default, confident enough to predict categories (e.g. persons) previously "seen". Some advantages

of working with pre-trained weights/models are:

- In terms of player (person) recognition, object detection algorithms reach high accuracy levels with a lower number of training instances.
- As a result of the previous point, training is computationally less expensive. Hence, it becomes easier to experiment with a higher number of epochs or increase the number of output classes since less resources are required.
- Training and validation datasets can be simplified since the model already has pre-trained weights to identify important features required to recognize a person or ball.

The following subsections present both frameworks used for players and ball identification tasks.

### 4.1. Faster R-CNN ResNet50

#### *4.1.1. Faster R-CNN architecture*

The so-called term *object detection* stands for the computer vision technique to locate a target of interest from a given image, accurately determine the category it belongs and give the bounding box of each target [3]. Faster R-CNN is by definition made up of two separate models: RPN or Region Proposition Network and a region-based CNN, such as the latest Fast R-CNN.

The former corresponds to a fully convolutional network and are defined to predict, as efficiently as possible, regions of interest of various shapes. The latter corresponds to the region-based object detector that, by means of the proposed regions ("where to look"), classify such proposal into object categories or, otherwise, background. Bounding boxes are not obtained in this step.
In practice, this has been one of the major drawbacks when training and testing the football dataset with the team's hardware. Due to this, results ambitions have been lowered so that experiments could have been carried out (i.e. using less data).

Merging RPNs with the Fast R-CNN network is made by switching between fine-tuning for both the object identification task and the region proposal one. The former would keep the region proposals acquired. As a result, a new model is obtained that combines both convolutional frameworks: the *Faster R-CNN*.
See the architecture of the model in Figure 2.

Several evaluations of the model on some well-known benchmarks can be seen in Shaoqing Ren et.al. work [4].

Lastly, the later model works on a ResNet-50-FPN backbone, a CNN architecture that was originally built for image classification and is used for feature extraction.

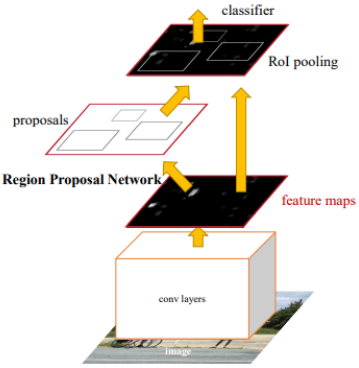


**Fig. 2**. Overview of the Faster R-CNN architecture. ResNet50 is used as the R-CNN backbone. Output is based on a *softmax* function for each class detected.

#### *4.1.2. Model training and testing*

This section brings up the bases for the Faster R-CNN implementation for the football player and ball identification task.
The training model lays makes use of *COCO* pre-trained weights in order to minimize training time, computational costs, and the data set size [5]. Note that COCO benchmarks were produced by following a similar training recipe as on [4], likewise, it was trained with 80 of the most common object categories (including persons and balls).

**Train**

Since training ran in the team member's own hardware, the batch size had to be adjusted so that the data partition could fit the memory of the device. Thus, after a few tests, the train batch size was set to 2. On the other side, the number of epochs was set to a reasonable value of 8. Notice that working on a pre-trained model helped reach convergence faster, hence no more epochs were necessary.
Training details are listed hereunder:

- **Training dataset** contains 750 images from one single football match from the SoccerNet dataset.
- **Images scale** is unique (size 1080x1920) and has not been re-scaled.
- For **image augmentation**, a single *HorizontalFlip* transform has been used.
- The total number of **output classes** is set to 3, where class 0 corresponds to the background.
- The **Gradient Descent Optimizer** (SGD) learning-rate is initially set to 0.005, but it varies following a scheduler with step size 3.

As a result, training provided a weighted model for later use in our test dataset.

**Test**

Unlike training, testing has been performed in both the SoccerNet dataset and Veo's. When it comes to the former

case, once again 750 images were used. This time from different football matches. Testing has been done using the test framework designed for the YoLo model. Check the respective Jupyter notebook.

## 4.2. YoLo

### *4.2.1. Yolo architecture*

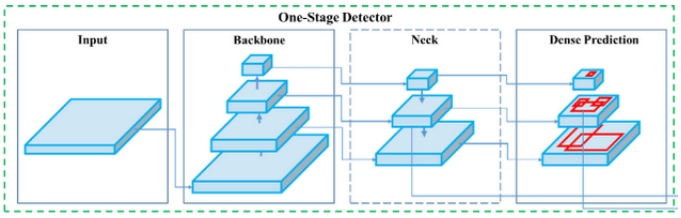


**Fig. 3**. Overview of the YoLov5 architecture.

The second algorithm used to carry out this project is YoLo (you only look once), in particular YoLov5, referring to a group of models with similar structure and architecture but with varying sizes in terms of memory usage intended to make its use flexible and accessible.

The general YOLOv5 architecture consists of multiple convolutional layers followed by a series of upsampling and prediction layers. These former layers extract features from the input image, while the latter increases the resolution of the feature maps. The prediction layers predict the bounding boxes and class probabilities for the objects in the image [6].

One key feature of YOLOv5 is its use of anchor boxes, which are predetermined bounding boxes used to detect objects, many of them coming in different shapes and sizes. The model predicts the coordinates of the bounding boxes, as well as the class probabilities for each box, which allows the model to handle a wide range of object sizes and shapes, producing a more robust and accurate result than some other object detection models.

In addition to the anchor boxes, this architecture also uses a series of prior boxes to provide additional context for object detection. These prior boxes are generated using k-means clustering and are used to regularize the prediction of bounding boxes.

Figure 4 shows the differences between all the different models in terms of performance, which will be relevant when discussing the training process for the team classification prediction challenge. As can be expected, lighter models perform worse than those with large memory usage, which may result too demanding for some workstations such as ours, reason why only the medium, small and nano version of YoLov5 have been used.

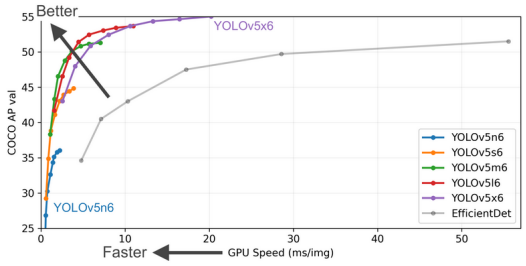


**Fig. 4**. Performance metrics of the different YoLov5 models.

### *4.2.2. Training with YoLo*

The first task for which this powerful model will be used is the same one as the FasterRCNN model, identifying the ball and players as accurately as possible, similarly to [7]. A major advantage to using YoLo is its easy-to-use framework to categorize and feed the dataset into the model for training. Once our dataset is labeled and split accordingly, the training parameters need to be tuned, which in this case are as follows:

- **Training dataset** contains 500 images from one single football match from the SoccerNet dataset.
- **Images scale** is unique (size 1080x1920) and has been re-scaled due to memory allocation.
- For **image augmentation**, since only images from one match are pulled for this session, the parameter corresponding to image augmentations has been set to high, which automatically applies HSV transformations, rotations, translations and flips along the vertical and horizontal axis. These transformations are assigned, a probability of being used for each image in the training dataset.
- The total number of **output classes** is set to 2, where class 0 and 1 correspond to the ball and players respectively.
- The **Gradient Descent Optimizer** (SGD) learning-rate is initially set to 0.01, although it is variable for each cycle.

It is important to remark that both the batch size and number of epochs have been kept the same for both Faster-RCNN and YoLov5 models, as the comparison needs to be as accurate and just as possible.

## 5. TEAM CLASSIFICATION

The second major task for this project consists of classifying each player to a different team, along with identifying the ball as previously explained, and the addition of referee identification. This denotes a significant step in terms of complexity as well as usefulness in future work, since accurately distinguishing all sides can provide many insightful statistics in a football match.

It is important to remark that goalkeepers will be labeled

as part of their team even though it is acknowledged the high degree of diversionsy in accurately predicting their presence, given the non-correlation between the colors of outfield players and goalkeepers.

YoLov5 will be the model used to carry out this task, using similar techniques as the ones explained in the previous section.

### 5.1. Data management and training

A dataset consisting of automatically produced labels resulting from the process explained in section 3 was initially used, as it was thought to be hugely instructive to analyze its potential compared to manual labeling, which was also used to realize this comparison.

In terms of training parameters, they are as follows:

- **Training dataset** contains 300 images from six different football matches from the SoccerNet dataset. The increase in match diversity is a result of the presumed difficulty in differentiating between two sides when only one reference is provided in terms of color differentiation.
- **Images scale** is unique (size 1080x1920) and has been re-scaled due to memory allocation. However, the final size of the images is far closer to the original than in the previous task, given the feature-dense nature of this job.
- For **image augmentation**, since the image size is comparatively higher, the parameter corresponding to image augmentations has been lowered in order to decrease the memory needed. Nonetheless, the same class of transformations are applied, even if it is at a lower rate.
- The total number of **output classes** is set to 4, corresponding to ball, referee, team 1 and team 2.
- The **Gradient Descent Optimizer** (SGD) learning-rate is initially set to 0.01, although it is variable for each cycle.

Finally, given the added complexity of the task, the number of epochs used for this exercise has been increased to forty, while the batch size has remained the same, two.

## 6. RESULTS

According to the structure of this project, results will split in two subsections just like in chapters 4 and 5. In addition, some extra remarks on the unsupervised learning task are given.

### 6.1. Team differentiation using unsupervised learning

The result of this task strongly depends on some factors in the tested image. If the players are far from the camera (so they are smaller with respect to some players in the foreground) the area of the box is small and the mean color is more prone to variations due to background noise. The brightness of the picture is another problem that causes uncertainty in the detection of the players, since can change the perceived color.

Anyway, the algorithm in good condition performs in a decent way for the classes *player_1* and *player_2*, when the *referee* label is evaluated, results start to become no more acceptable. See Figure 11 and section 6.3 for more details.

### 6.2. Evaluation of the player and ball identification task

YoLov5 and Faster RCNN ResNet50 object detection models can be easily evaluated by means of an F1 score graph within the player/ball identification task.

An F1 vs confidence graph is a plot that shows the relationship between the F1 score (a measure of the accuracy of a classifier) and the confidence of the predictions made by the classifier. The F1 score is a metric that combines precision and recall, where the former refers to the measure of the fraction of true positive predictions made by the classifier and the latter refers to the degree of the fraction of actual positive examples that the classifier was able to identify.

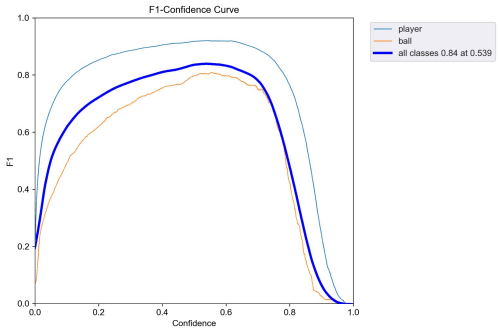


**Fig. 5**. F1 curve for different values of confidences - Faster RCNN

Faster RCNN and YoLov5 have a higher player accuracy w.r.t. the ball for all the confidence range, as Figures 5 and 6 respectively show. Notice that this might be due to the size of the wall and its lack of intrinsic features compared to a person. Additionally, since images are forced to downscale in YoLo, CNN algorithms experience difficulties learning features as the number of pixels per image is significantly reduced. Hence, the Faster RCNN F1 score for ball detection is slightly higher.

Both methods reveal, at higher confidence values (i.e. $\sim 1$) very low F1 scores. This means that none of them guarantee decent precision results in extreme certainty. This could be due to the model being not complex enough or having a few parameters relative to the size of the training dataset, leading to poor generalization to new, unseen data.

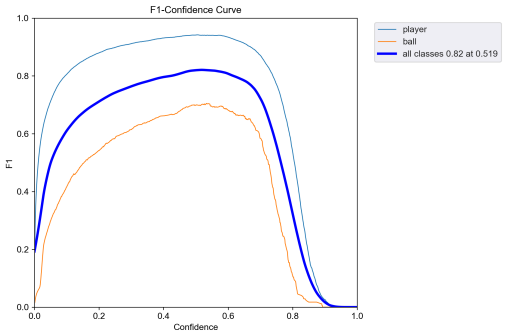


**Fig. 6**. F1 curve for different values of confidences - YoLov5

### 6.3. Evaluation of the team classification task

As has been the case with the previous section, this task will be assessed firstly by analyzing the resulting f1-score vs confidence graph computed during the training of the network and seen below:

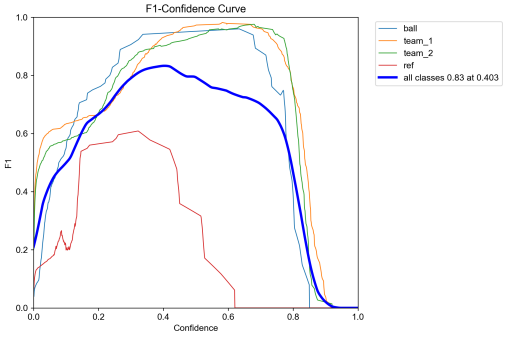


**Fig. 7**. F1 curve for different for all elements' values of confidences - YoLov5

The main observations to be drawn from this figure are that the peak corresponding to the referee detection curve is very low in comparison to the rest, meaning that the model can not detect the referees accurately. Furthermore, the curve is skewed to the left of the plot, which signifies relatively low confidence of the model in the referee´s classification. Given that they are similar in shape to the rest of the players, it is expected that during testing the referees might be classified as part of one of the teams, which is exactly what can be seen in picture x in the annex.

Finally, it is also noteworthy to remark on the results of the confusion matrix, also relevant when analyzing the performance of the model. Figure 8 shows the problem just mentioned, where referees are the group that manifests the worst results. Unsurprisingly, the rest of the elements tested perform at a high level, with the caveat of the ball not being recognized among the background, as is the case with all previous tasks and models described in this report.

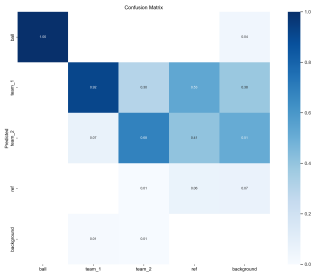

**Fig. 8**. Confusion matrix for all elements in team detection - YoLov5

## 7. CONCLUSION

The main difference between the two approaches could be seen in the different times required from the two. In fact, the YoLo v5 model, thanks to its less computationally complex pipelines, results to be faster and able to process up to 91 Frames per Second. This characteristic makes of YoLo a perfect candidate for real-time processing systems. Faster R-CNN, on the other hand, still maintains a theoretically slightly better accuracy rate, especially when it comes to smaller objects such as the ball, and no computational cost considerations are done.

Both models, YoLo v5 and faster R-CNN, have limitations when it comes to identifying small objects that appear in groups or further away from the camera object. Since bounding boxes are learned from data, models also could struggle with objects found in unusual perspective ratios.

The goalkeeper's automatic labeling has been found a really challenging task. Since he has a different color uniform from the other same team players the model can not label him as a player of the same team, this labeling can also lead to a referee mislabelling, since also he has a different uniform from the teams and, eventually, similar to the goalkeeper.

### 7.1. Further work

The unsupervised learning algorithm was implemented in this work to test the limits of the main method by adding the new class for the two different teams, the quality of these results were not comparable with the previous attempt. In further work, an unsupervised machine learning algorithm could be added to the main method taking advantage of the quality of the result achieved with the convolutional neural network and applying unsupervised methods afterwords to detect the players team. In particular, this could be added without a complete redesign of the model in the second section of the faster R-CNN model.

## 8. REFERENCES


[1] Zoran Kalafatić, Tomislav Hrkać, and Karla Brkić, "Multiple object tracking for football game analysis," in *2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)*, 2022, pp. 936–941.

[2] Diego Oliva, Mohamed Abd Elaziz, and Salvador Hinojosa, *Image Segmentation by Gaussian Mixture*, pp. 149–155, Springer International Publishing, Cham, 2019.

[3] Jun Deng, Xiaojing Xuan, Weifeng Wang, Zhao Li, Hanwen Yao, and Zhiqiang Wang, "A review of research on object detection based on deep learning," *Journal of Physics: Conference Series*, vol. 1684, no. 1, pp. 012028, nov 2020.

[4] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2015.

[5] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár, "Microsoft coco: Common objects in context," 2014.

[6] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao, "Yolov4: Optimal speed and accuracy of object detection," 2020.

[7] Anthony Cioppa, Silvio Giancola, Adrien Deliege, Le Kang, Xin Zhou, Zhiyu Cheng, Bernard Ghanem, and Marc Van Droogenbroeck, "Soccernet-tracking: Multiple object tracking dataset and benchmark in soccer videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3491–3502.

## 9. ANNEX


**Fig. 9**. Testing result sample for detection of players and ball using Faster-RCNN model


**Fig. 10**. Testing result sample for detection of players and ball using YoLov5 model

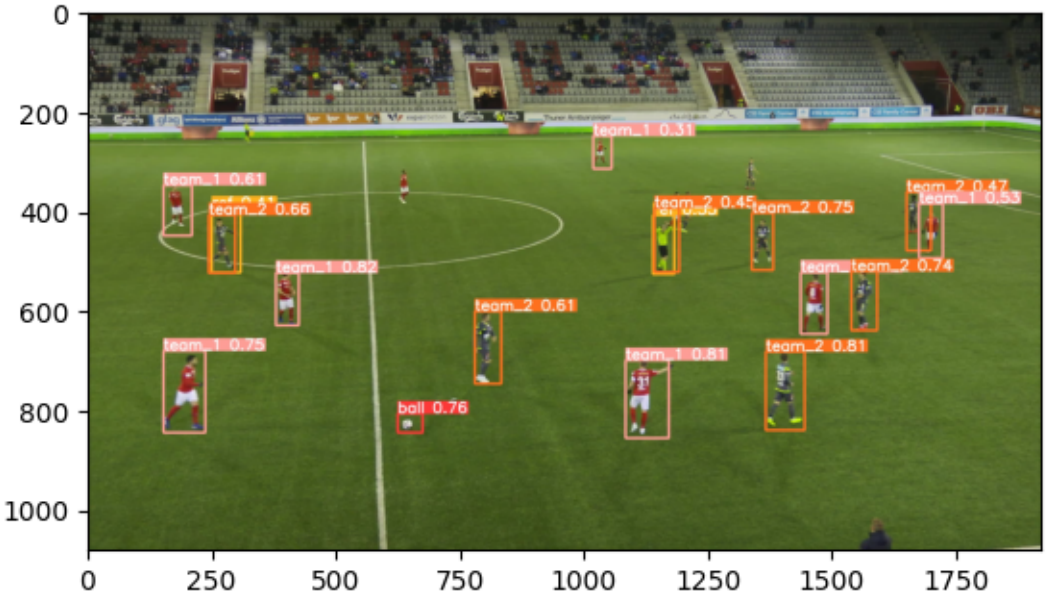


**Fig. 11**. Testing result sample for detection of all actors in a football match