Application of Computer Vision Techniques to Soccer Images & Videos

Bernardi Mattia
291712@studenti.unimore.it

Davoli Davide
240442@studenti.unimore.it

Ragazzi Francesco
232701@studenti.unimore.it

Abstract

The aim of this report is to describe computer vision approaches to a soccer match scenario by doing things like player and ball detection, position mapping, action classification and action retrieval.

1 Introduction

The project aims to provide tools capable of processing soccer game scenarios, both static images and dynamic videos, taken from the internet and courtesy to the SoccerNet Project; the basic implemented features are:

1. *Image Processing:* Consists in methods to do player and ball detection with and without the use of Deep Learning methods.

2. *Video Processing:* Poor try to use background subraction in order to achieve team classification.

3. *Geometry Mapping:* We've created a 2D mapping of a game scene using homographic transformations alongside with RANSAC[3], SIFT[7] and Hough Transform[2].

4. *Action Classification:* We've used a combination of Neural Networks, pretrained and made by us, in order to achieve action classification.

5. *Action Retrieval:* Consists in an extension of the Neural Network used for action classification in order to provide descriptors to do action retrieval.

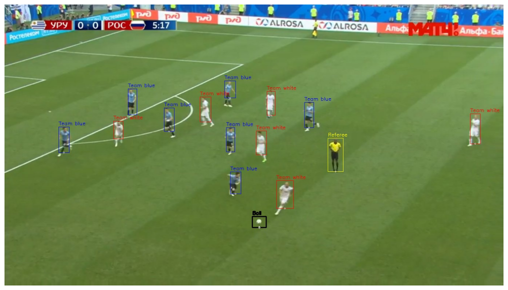


Figure 1: Image Processing Detection Results

2 Related Works

In order to develop a high quality project it is always a good idea to study what has already been done, in order to have a hint of the best techniques applied up until that moment. Starting from that point the decision that can be made is to reinvent the wheel or work on what has already been done. On the internet there are several works on player and ball detection, geometry mapping, video classification and video retrieval most of them use OpenCV or some pre-built neural networks, but it's curious to see how approaches to achieve the same task could be totally different from one another. Starting from player detection, the straightforward method here is to use a convolutional neural network such as YOLOv3[10], the result is guaranteed, but for educational purposes we decided to continue with traditional techniques, which has as an advantage not requiring a dataset and much less computational power. In any case to avoid completely skipping this approach we also used YOLOv3 pretrained by DarkNet[9] on COCO[6] in order to do player and ball detection; going back to the classical computer vision, let's see for example the case of [11] where the technique consists in applying a Gaussian blur, perform a color space transformation, extracting the green color, performing a bitwise or operation in order to get the contours.Moving on we looked into more Deep Learning based approaches to do action classification, for that we looked first into SoccerNet[4] and then into SoccerNetV2[1] in order to retrieve some info on how to work with their videos to achieve action spotting and classification; We of course tried different approaches but none of them were inspired by some older work in particular.

3 Classic Approach

Working on the project, we have taken into account all the related works already done in the past and we have introduced a series of improvements to the several steps of the pipeline.

3.1 Player Detection

The first thing we did was to perform a ball and player detection task based on a color subtraction method.

3.1.1 Pre-processing

First of all we performed a Gaussian Blur in order to improve the image quality and the most of the useless noise, we tried different sigmas and then went with the best one for our task.

(a) Input Image

(b) Result after the application of the Gaussian Filter

3.1.2 Color Space Conversion

The second step was to convert from BGR to HSV in order to get a cleaner and stronger green presence over the entire image; this also helps by making the players pop-out more.

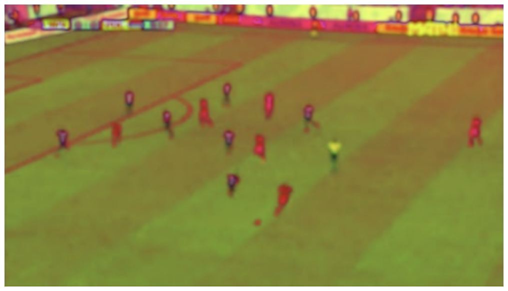


Figure 3: Input image after color space conversion

3.1.3 Green Extraction & Bitwise OR

This steps consists in creating a binary image setting all the green pixels to 1 and the others to 0; after that we performed a *Bitwise OR* operation with this new binary image and the input one, converting the results in a grey level scale.

(a) Binary Image with green pixels set to 1

(b) Result of the bitwise OR operation

3.1.4 Get Contours

The last steps of this process are to apply a threshold on this last grey level image in order to perform a segmentation task on what ain't green on the field, here we used both OTSU Thresholding [8] and Binary Inverse Thresholding. Moreover we applied the morphological operation called closing and a median blur filter in order to improve the border and image quality obtaining this last image where we then proceed to extract the contours.

3.1.5 YOLO

One last try we did was to use the YOLOv3 network pretrained on DarkNet in order to do player detection and we also combined this method with K-Means to achieve team recognition without specific color information.

Figure 5: Final imgage for border tracing

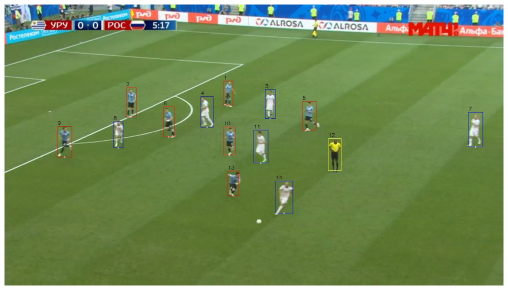


Figure 6: YOLO plus K-Means player detection and team recognition

3.2 Ball Detection

In order to do ball detection without any network we implemented a simple template matching method using the NCC as distance measure.

3.3 Geometry Mapping

Here we wanted to create a 2D mapping starting from a 3D scene combining the detection system we've just built with some geometry techniques in order to get the best results as possible: the main idea was to use the homographic transformation of OpenCV which requires at least 4 anchor points to work; the first approach was to use either a photo editing tool, like *Gimp*, in order to retrieve the exact pixels of the points we were selecting; we then tried to use a function in order to get the coordinates by the mouse click but at the end we stayed with the first one. During the first tests we noticed that 4 points weren't enough due to the small visible portion of the 3D image with the respect to the 2D field. We decided to increase the number of points and to spread them around more in order to cover the entire length of the field, using this approach the results were nearly perfect. After making this work we went ahead and tried to automate the process of finding the keypoints, the first try we did was by using the Hough Transform in order to find the intersections between the field's lines but with poor results; maybe with less satisfying outcome we also tried to use SIFT and RANSAC also switching with different field images but at the end nothing was working in a good enough way also due to the symmetry of the field.

3.4 Video Processing

To end the image processing part we wanted to approach also at least 1 video processing method to do player detection, this was done using background subtraction but the outcome was very poor; we also tried a motion based approach but with no more luck than before.

4 Deep Learning Tasks

One of our goals was to perform the action classification task in order to detect whether an action was in a clip or not. Moreover we wanted to try a retrieval approach based on the results of this classification network.

4.1 Dataset

In order to kick off our methods we need a dataset, luckily for us we found rescue in the SoccerNetV2 dataset, owned by King Abdullah University of Science and Technology, composed of 500 full matches, including also:

- Full untrimmed broadcast videos in both low and high resolution.
- Pre-computed features such as ResNET-152.
- Annotations of actions among 17 classes.
- Annotations of camera replays linked to actions.
- Annotations of camera changes and camera types for 200 games.

In our project we used only a subset of the entire dataset, in fact we used 4 classes: Nothing, Goal, Foul, Corner. The first thing we did was to extract clips from the full games, each of those is composed of a 10 seconds window, 5 before and 5 after, of the action starting at the time provided by the annotation at 3 fps, so each clip is composed of 30 frame images with a 224x398 resolution:

Action	Instances
Corner	3775
Foul	4483
Goal	1531
Action	4490
Total	14279

Table 1: This is how the dataset is composed

The entire set is then split into 3 subsets: training, taking up the 70% of the whole dataset, validation and test which are 15% each.

4.2 Architecture

The architecture of our network is basically split into 2 different networks, a feature extractor and a classifier:

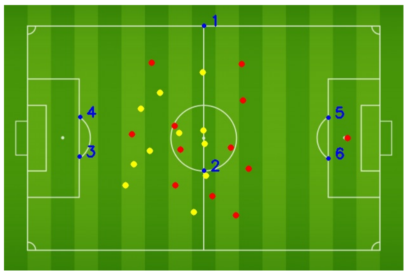
- **Feature Extractor:** It is simply a ResNet18 [5] without the classification layer and global AVG Pooling, here we decided to try 2 different versions, the first with a 2 by 2 AVG Pool, which is different from the original ResNet layer, and the second one without it.
- **Classifier:** For this part we decided to build our own network, it is composed by 3 1D Convolutional layers and 3 Fully Connected one.

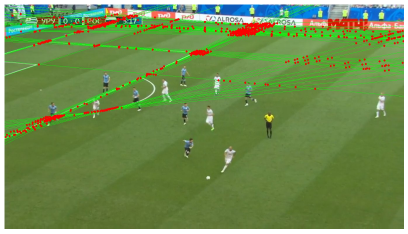
4.3 Action Classification

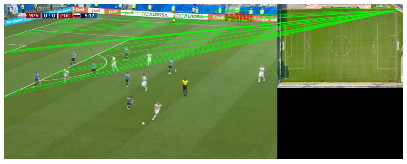
The action classification task is the main one for this part, we considered using a *SOA* network but for edu-

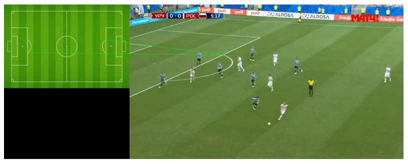
(a) Initial Image with 6 points selected

(b) Player detection system


(c) Final Mapping result


(a) Hough Lines intersections


(b) SIFT + RANSAC 1


(c) SIFT + RANSAC 2

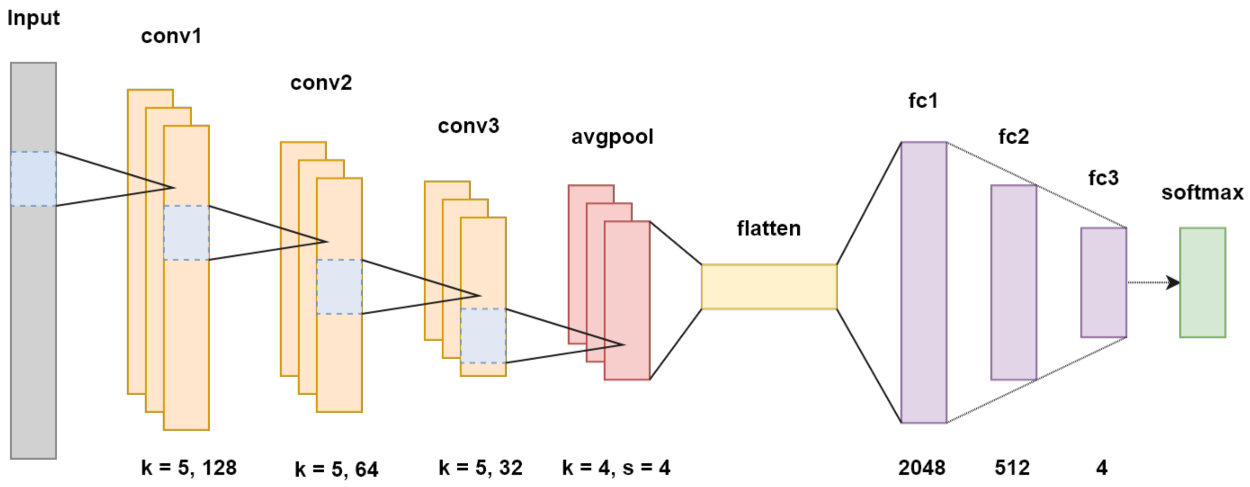


Figure 9: Custom Classifier Architecture

cational purposes we decided to take a pretrained feature extractor but then build our own classification and train it over SoccerNet; this was a very interesting part as we were able to see what are the effects of the different layers on our task, this wouldn't have been possible by using a plug and play network.

4.3.1 Training

The training of our network was doubtless the most difficult part: initilially we built a balanced dataset composed by 1531 clips for each class, we tried some different learning rates and we've obtained good performance on the training set but on the validation set the results were very poor, around 25% accuracy. To try and fix this obvious overfitting problem we first tried to implement some regularization techniques but with little to no effect on the validation and test set; lastly we decided to augmenting the dataset, switching from roughly 6000 clips to well above double of that resulting in the tipping point for our network.

4.4 Action Retrieval

For the retrieval step we decided to reinvent what we already did for the classification task: since our goal was to, given an input clip, retrieve the most similar actions; to achieve this, and to avoid requiring more computational power to train another ad-hoc network, we went ahead and used the outputs of the last but one fully connected layer of the classifier and use those as descriptors on which we then computed some distance measures:

- Euclidian Distance
- Cosine Distance
- Earth Mover Distance
- Minkowski Distance of order 1, Manhattan Distance

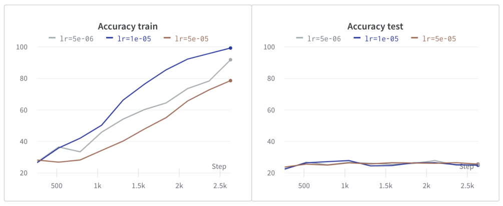
To compute those different measurement types we used the *MAP: Mean Average Precision*, the *MRP: Mean R-Precision* and the *Average Precision at Each Rank*.

5 Results

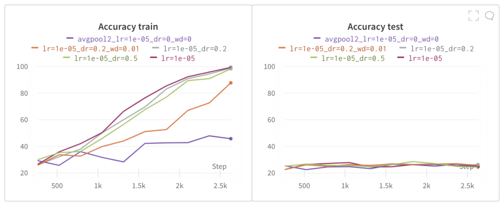
x All the results obtained through the experiments are summarized in tables [2],[3], here there's a brief discussion on those outcomes. First of all let's start with the Image Processing and Geometry section: - colori: velocesemplice ma limitato - yolo + k: generale ed efficiente ma lento - fallimenti: putni di riferfimento influenzano The first method based on green extraction is doubtless faster and simpler but very limited by the scene we are facing, players need to be very visible with the respect to the field and they need to be standing otherwise the detection is very hard; on the other hand the YOLO plus K-Means approach in more general and efficient but is much slower. Lastly, talking about geometry mapping and homographic transformation, we noticed how much the points selection influences the outcome of the algorithm, they need to be spread out across the field and possibly more than 4 to make it work perfectly. Regarding the classification task we can see that results are very similar throughout all the runs and surely enough there is some overfitting, especially on those without the AVG Pool; we tried to use weight decay without any luck. Moreover we can see that the confusion matrix are very good all around, even on the Goal class, which was the one we were most concerned about since we have few examples; the only thing here is a bit of misinterpretation between No Action and Foul, probably because, when a foul occurs, some time passes without players doing anything. For the retrieval part we can see that even if the network was not trained for this scope, we were able to achieve a sufficient level of performance despite the large similarities in every clip due to the fact that from a descriptor point of view it is not easy to extract features related to a specific action. We thought about Bag of Word in order to perform this task but we ended up with a common thought: the soccer environment isn't really BoW friendly due to the enormous similarity from one action to another in term of visual words.

6 Final Discussion

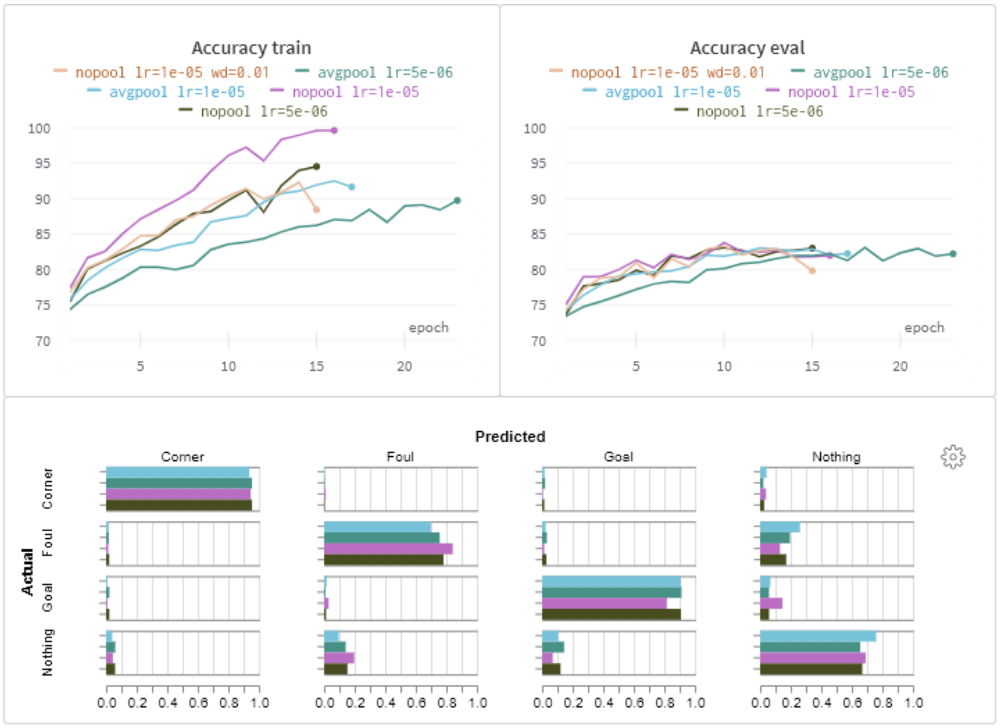
This work's been a good way for us to perceive the difference between classic computer vision techniques and the newer ones involving neural networks. Despite the limited data available for certain classes and the disparity between the number of instances, we are very satisfied by what we achieved. By dedicating



(a) Initial Accuracy with little data



(b) Little data with regularization technique



(c) Final results with augmented dataset

Learning Rate	AVG Pool	Weight Decay	Train Acc. %	Eval Acc. %	Test Acc. %
1×10^{-5}	No	No	96.1%	83.8%	81.9%
1×10^{-5}	Yes	No	89.5%	83.1%	81.2%
5×10^{-6}	No	No	89.8%	83.1%	81.0%
5×10^{-6}	Yes	No	88.5%	83.1%	81.8%
1×10^{-5}	No	No	90.4%	83.3%	82.0%

Table 2: Classification Performances results with all the different combinations

more time to retrieve lacking instances and applying some more augmentation we are sure that we will obtain results very near to more sophisticated architecture with complex layers. We also tried tweaking

Distance	MAP @ 11	P @ 1/6	P @ 2/6	P @ 3/6	P @ 4/6	P @ 5/6	P @ 6/6	MRP @ 8
Euclidian	0.621	0.713	0.628	0.591	0.565	0.549	0.548	0.516
Cosine	0.618	0.688	0.625	0.589	0.575	0.564	0.560	0.528
Earth Mover	0.558	0.660	0.562	0.515	0.479	0.463	0.453	0.516
Minkowski	0.619	0.709	0.629	0.592	0.567	0.552	0.547	0.516
Average	0.604	0.693	0.611	0.572	0.547	0.532	0.527	0.519

Table 3: Retrieval performances among all distances and measurements

the parameters of the algorithm but it can't be compared with the neural network.
Making this project was very inspiring and stimulating since we were able to try a lot of different methods and algorithms in a practical way, letting us understand the differences between theoretical studies and real implementations; moreover we were able to understand which kind of techniques work better in a certain environment with respect to other ones.

References

[1] Adrien Deliège et al. *SoccerNet-v2: A Dataset and Benchmarks for Holistic Understanding of Broadcast Soccer Videos.* 2021. arXiv: `2011.13367 [cs.CV]`.

[2] Richard O Duda and Peter E Hart. "Use of the Hough transformation to detect lines and curves in pictures". In: *Communications of the ACM* 15.1 (1972), pp. 11–15.

[3] M. Fischler and R. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". In: *Communications of the ACM* 24.6 (1981), pp. 381–395. URL: `/brokenurl#%20http://publication.wilsonwong.me/load.php?id=233282275`.

[4] Silvio Giancola et al. "SoccerNet: A Scalable Dataset for Action Spotting in Soccer Videos". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (June 2018). DOI: `10.1109/cvprw.2018.00223`. URL: `http://dx.doi.org/10.1109/CVPRW.2018.00223`.

[5] Kaiming He et al. *Deep Residual Learning for Image Recognition.* 2015. arXiv: `1512.03385 [cs.CV]`.

[6] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context.* cite arxiv:1405.0312Comment: 1) updated annotation pipeline description and figures; 2) added new section describing datasets splits; 3) updated author list. 2014. URL: `http://arxiv.org/abs/1405.0312`.

[7] David G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". In: *Int. J. Comput. Vision* 60.2 (Nov. 2004), pp. 91–110. ISSN: 0920-5691. DOI: `10.1023/B:VISI.0000029664.99615.94`. URL: `http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94`.

[8] Nobuyuki Otsu. "A Threshold Selection Method from Gray-Level Histograms". In: *IEEE Transactions on Systems, Man and Cybernetics* 9.1 (1979), pp. 62–66. DOI: `10.1109/TSMC.1979.4310076`. URL: `http://dx.doi.org/10.1109/TSMC.1979.4310076`.

[9] Joseph Redmon. *Darknet: Open Source Neural Networks in C.* `http://pjreddie.com/darknet/`. 2013–2016.

[10] Joseph Redmon and Ali Farhadi. *YOLOv3: An Incremental Improvement.* 2018. arXiv: `1804.02767 [cs.CV]`.

[11] Kanan Vyas. "Player and football detection using Opencv-Python in FIFA match." In: (July 2018).