Real-Time Football Action Prediction

In this paper we present a study in the real-time football action prediction framework. On one hand, we evaluate a setup that unifies together several steps of classical processing for real time prediction: from object recognition, through tracking to action prediction. On the other hand we evaluate an action prediction pipeline based on CNN and LSTM. Finally, we propose an annotated dataset for football game individual actions and action sequence prediction. In order to provide real-time processing and highest available accuracy, the two studied pipelines are evaluated on every step of processing. Both pipelines have two main components: feature extraction and prediction. The first pipeline start from input image and applies object recognition and tracking using off-the-shelf tools from OpenCV. The second pipeline starts from input image and uses a CNN to extract image features. The second stage in both pipelines are specifically trained LSTM for the sequential task prediction. The best combination of algorithms and parameters is optimized by training and data augmentation. The proposed dataset was created as a set of labeled short videos: each frame is labeled with current action and current scene type. The proposed system obtains an accuracy of 35.5% in predicting the next action performed by the football team. The proposed system is currently the most accurate action prediction system for real-time streamed football games.

1 INTRODUCTION

While in recent years classification, recognition, detection and segmentation received a large amount of attention, much less work and attention has been given to the prediction task. The main reason is that the prediction of action or activity has an additional complexity component, the time.

In fact, the current approaches mix actions and activities, but in general the action prediction approaches are mostly not available. For instance in the UFC-101 [31] dataset there are 101 categories of activities each composed of a movie including several actions. In Sports-1M [13] there is 487 different categories of activities also containing a sequence of actions. The common feature between these datasets is that recognition of the activities is based on the recognition of a sequence of actions. However no specific focus in given to the recognition of the individual actions. The individual actions are not necessarily recognizable, but many various actions from many various activities are expected to be used for activity recognition or prediction.

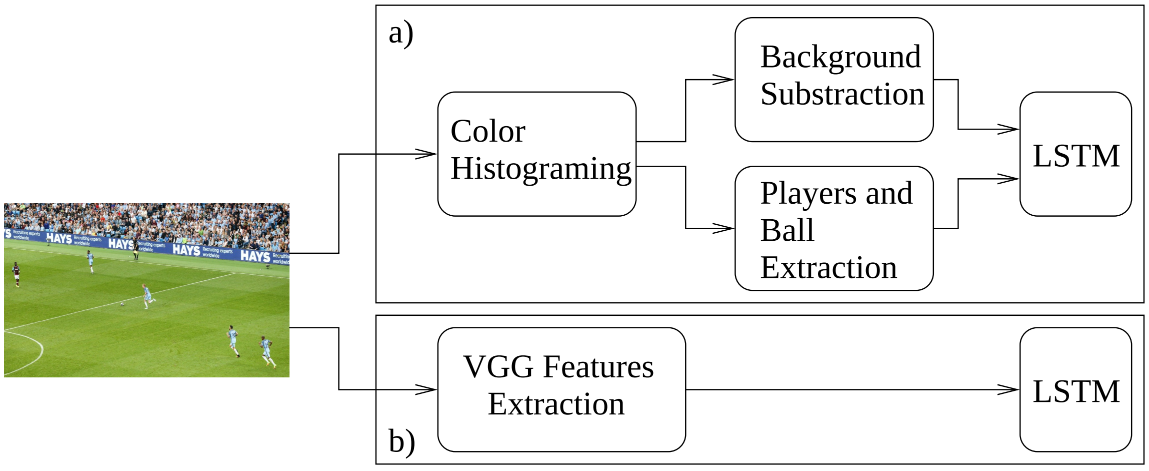


Fig. 1. Schematic description of the two evaluated approaches in this paper.

The prediction of individual activities can be seen as a form of behavior estimation and recognition. Individual behavior has been studied from various point of views such as estimating key points of posture in time [1, 8, 18, 30, 35], using context for behavior estimation from videos [17, 33], tracking changes of interest points in motion [11], part-based recognition [5] as well as the use of convolutional neural networks applied to various information sources [12, 34], using 3D convolution [10] or temporal modeling [25, 29].

The activity prediction can be either done in a short term such as in [15, 16, 19, 20, 28] where the input video is several seconds long and the objective is to predict or determine the category of the activity. The long-term prediction is the prediction of next activities based on the current observed state. Videos in this category are longer and last up to several minutes [14]. The longer term prediction can be seen as intention estimation [20, 23, 26].

Unlike activity prediction, individual action prediction research is much less abundant. One can predict action of a sports team based on visible game elements. In this case the prediction of action is individual agnostic: the predicted action is the result of the game elements configuration and motion hysteresis. Examples of such approaches have been investigated not only for sports but also for games [7] or ice hockey [4].

However, there is not as of yet a action specific database that would allow to focus on predicting various actions within one activity. For instance, predicting next moves within football, basketball, etc,. Recently, action prediction dataset was introduced in the context of cooking in [32] and with the target of predicting actions as a part of food making process.

The target of this work is not to provide a completely new framework for action prediction but to study the feasibility of real-time action prediction using a set of available tools, evaluate their speed and accuracy, and provide new dataset for long-term action prediction as well as continuous action prediction. We provide a comparative analysis of the two studied processing flows. The first processing flow starts by extracting all visible recognizable game elements and tracks the recognized objects form frame to frame. The second processing flow starts by simply extracting visual features using a CNN. The result of each of these two processing flows is then used as input for the prediction of the next action from a set of possible actions (Figure 1). In addition to evaluate the speed and accuracy of the two models we also introduce two modes of accuracy measure: frame-wise and action-wise. The frame-wise action prediction (FWAP) is an estimation approach to determine action for the next frame while the action-wise action prediction (AWAP) estimates the next action that will occur.

To evaluate the methodologies, we introduce a Football Action Prediction Dataset (FAPD). The dataset contains a set of annotated videos of football game. Each video is annotated for action on a frame by frame basis as well as is labeled by a scene type label. The action annotation was performed using an image difference algorithm and actions are based on ball-player information. The actions are classified and limited to only such actions that could have been detected using the proposed autonomous method. The scene type label is performed by a state of art football labeling algorithm []. The data is annotated and can either be used as a single stream or as individual independent videos. The annotated dataset contains all together less than 15000 annotated images. Each short video is of length of up to several minutes and contains a sequence of actions label. The videos have been randomly selected and collected from youtube.

This paper is divided as follows. Section 2 introduces the required background, Section 3 describes the created dataset and Section 4 introduces the method for evaluating the action prediction framework. Section 5 describes the experiments and discusses the results and the paper is concluded by Section 6.

2 BACKGROUND

Commonly used approaches to action prediction are performed using a partially observed video in [28] using a bag-of-words method. The authors combined feature variation over time for representing sequential action over time. In [2] frames from videos were removed and using posterior maximization the missing frames were predicted. The likelihood was estimated from feature reconstruction error. In [16] a structured SVM was used that allowed to consider temporal dynamics of human actions. In [17] the detection of activities was evaluated with respect to the length of the video representing given unfinished action. Finally recently [15] an LSTM based approach was used to recognize activities from sports sequences.

The sequential action prediction studied in this approach is similar to approaches such as in [15] with the difference that we are focusing on one particular context (activity) that is football game. In addition we do not predict the activity to recognize the context but rather we predict action in a larger time horizon. Our target is not to provide a general image to action prediction framework but rather, an specific next action prediction method. The main difference from the existing frameworks is that the limited framework allows us a) to exploit the contextual information to maximum, b) extract specific features to accelerate the whole computation and c) generate data set in a autonomous manner.

3 THE DATA SET

In order train the action prediction system, a data set was created using several existing tools.

3.1 Action Labeling

The action labeling is based on the difference between individual frames of a video stream. An example of labelling a particular scene is shown in Figure 2. Figure 2a (Figure 2c) shows the initial and Figure 2b (Figure 2d) shows the end image used to determine the pass long (short) action. The labeling procedure starts by recognizing all players in the

Table 1. The content of single sample

	#frame	action	team	Ball		Players Number		Coord Player 1		…	Coord Player 22	
		label	controlling ball	x	y	Team 1	Team 2	x	y		x	y
Values	Int	1-12	{1,2}	float	float	1-11	1-11	float	float			

image. This is obtained after several steps of processing that include the background subtraction and histogram based player recognition. Then for every two consecutive frames a) the difference between the ball and closest player and b) the difference between ball position are measured.

The action detection must however filter lost ball, ball rolling along the player or ball passing by a player while being kicked. For this purpose the estimated velocity of the ball from the difference of position between frames is used. The speed of the ball was used to determine what type of pass or goal kick was performed: this result was obtained by density-based clustering and analysis.

Figure 3 shows the result of such clustering. The x axis shows the total number of frames and y axes shows the change of ball's position. In addition, to further limit the misdetection of ball trajectory, we experimentally determined that the time difference between the initial and final coordinates must not exceed 60 frames which represents an time interval of up to 3 seconds. This allowed the clustering to be performed over sequences of frames reliably recognizing either player-to-player ball passes or goal kicks.

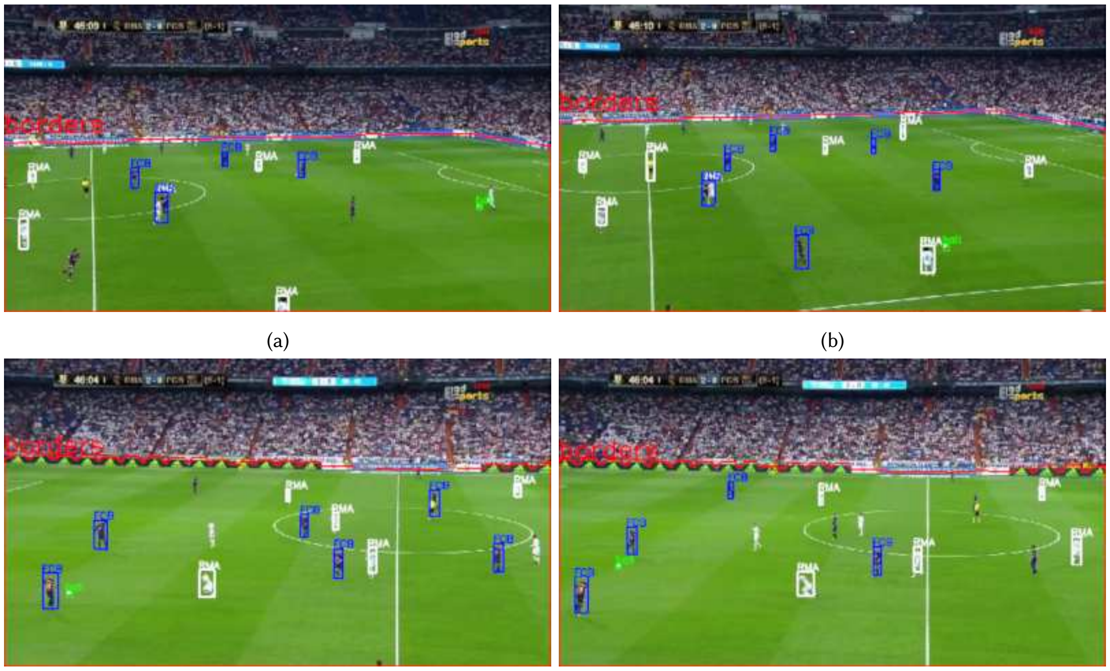


Fig. 2. Examples of start and end of 2a- 2b long pass and 2c- 2d short pass respectively.

Finally, it was necessary to filter the ball movements for deceptive situations such as multiple passes that average to a similar ball velocity. In addition the ball detection can be partially faulty and therefore we integrated a fail-safe to detect when one action finishes and another starts. Experimentally it was determined that a valid pass is considered to be pass if there are at least 5 consecutive frames with ball coordinates in them (it could be both true and noisy coordinates). If another action lasted for 5 frames or there were no detection of ball, the end of previous action is calculated. In this fashion true frame ranges of passes were deduced. Since true ranges of passes are known, true coordinates of ball can be known by taking only positions belonging to similar clusters (like-colored clusters).

The actions addressed by the labelling can be separated in two general actions: a goal kick or a pass.

- A goal kick is any shot that ends up in the hands of the goal keeper, or near or inside the goal cage.
- A pass is a shot that is not directed to the goal cage and ends up with another player. As such, the pass is one of the following: short, long and traversal.

Using the results of this clustering analysis, the recognized actions are further split as follows:

- A short kick or Short Pass is obtained if the ball is moving between two locations for 8-15 frames,
- A long kick or long Pass lasts for 16 to 40 frames,
- and a traversal pass is longer than 48 frames.

Beside the action classification, the actions were also attributed to a given team. Team which possesses/passes the ball was determined by finding the closest player to the ball in the last 50 frames.

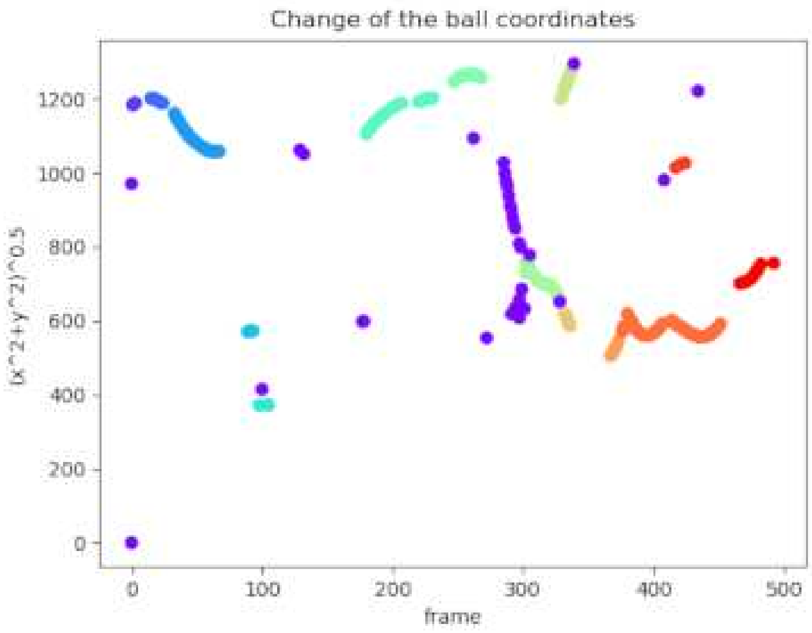


Fig. 3. Clustered ball motion from randomly selected football videos.

The possible labels of actions generated by the data set are: short pass, long pass, self pass, traversal pass and kick. For each action four directions are given depending in which quadrant the ball was kicked into. The actions are represented graphically shown in Figure 4. the recognized and labeled actions are only passes of different duration.

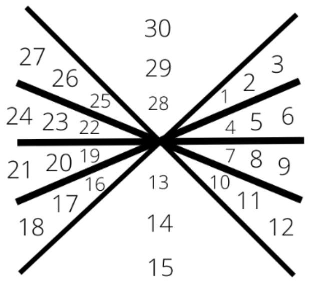


Fig. 4. Label assignment to each recognized action by the autonomous data labelling.

3.2 Scene Labelling

The data set is limited to only certain types of scenes from the whole game. The reason is that because the intention of the system is to be used in a real-time intelligent streaming environment, mane scenes are not relevant and not appropriate for football action predictions. For instance, breaks, faults, penalty, direct kick or corner are very specific and much rarer and actions are mostly determined. For instance during the penalty type only one type of kick occurs; during the corner kick the kick is also the same in the general direction. Therefore the most variety of actions occur during the game play in the mid-filed or when the attack is mounted. Consequently we only focused on most common type of scenes taking place in Midfield or Goal attack (When players initiate an attack) were used.

To detect the ball during the match new ball detection above was used (Transfer Learning on YOLOv3); In addition right and left directions were split into 4 subsectors each; Now actions are labelled as following: 1 - short, 2 - long, 3 - traversal and so on; With such actions the dataset with following frequency for each action was acquired as in the graph. For 45 minute match there are 24016 samples and 51 features (frame, actionLabel, ballcoordinates and etc);

The *Goal Attack* scene type was determined from scene recognition. If number of players in Team 1 is less than in Team 2, it is considered as Team1 Goal Attack and vice versa. *Goal Attack* is identified only if it lasts 15 or more frames.

3.3 Data Gathering

After labeling, each scene was was saved into individual file. Each frame information about current game state is gathered in the format shown in Table 1. In general we will refer to 4th to last column of Table 1 as the game state s^t with t representing a time counter or the value of the frame from column two. Column three from Table 1 is referred to as action a^t.

4 REAL-TIME FOOTBALL ACTION PREDICTION

In this work, a similar setup to the one in [15] is used as a starting point. Let the sequence of image frames be given by $x = \{x^1, \dots, x^t\}$, $l = \{l^1, \dots, l^t\}$ being a set of labels representing the types of football scenes and $a = \{a^1, \dots, a^t\}$ representing actions occurring at each frame. The task is then to estimate $a^{t+1} = \mathcal{L}(a, x, l)$.

As one of the two foci of this paper is real-time football action estimation. Therefore the first studied pipeline is based on well known, highly optimized computer vision and image processing elements. we combined several highly optimized methods into a processing flow which is schematically depicted in Figure 1(a). The first step starts by the extraction from the input image $i \in x$ of a set of descriptors that are either symbolic or numerical values $d = \{d_0, \dots, d_m\}$. The second step is to estimate the scene label - scene type. Then using the set of descriptors d ad he scene label l we estimate the next action a^{t+1}.

We will recognize several configuration of the problem at hand:

(1) Task $\mathcal{P}_0$, the set a is a sequence of consecutive actions and predicting next action is to predict next element of a
(2) Task $\mathcal{P}_1$, the set a is a sequence of action labeled image frames and predicting next action is to predict action label of the next frame
(3) Task $\mathcal{P}^o$, the mapping from image to d results in a set of objects recognized and localized on the image. The images frames x^t are processed for object recognition and tracking. Let $o = \{o_1, \dots, o_m\}$ be the set of available objects (object labels and coordinates) , then the task becomes $a^{t+1} = \mathcal{L}(a, o, l)$.
(4) Task $\mathcal{P}^f$, the mapping from image to d results in image features. The image frame x^t are processed for labelling of scenes only and the task is then only $a^{t+1} = \mathcal{L}(a, f, l)$.

Note that as it is the sets x and l are limited to few past actions $0 < k \ll t$.

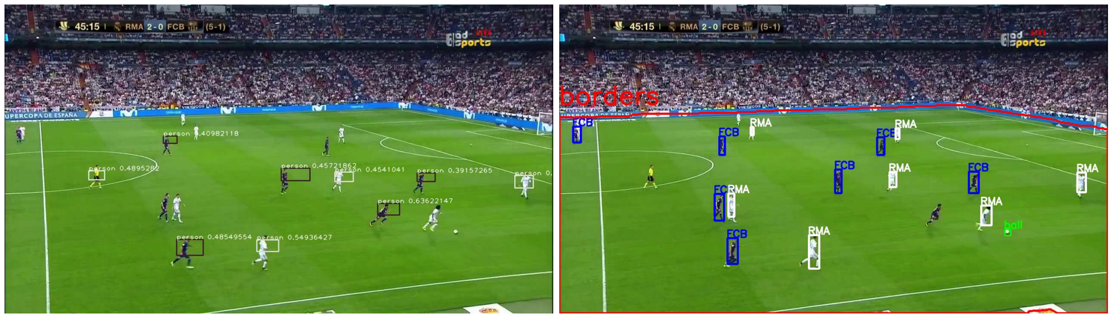
As a starting point for processing we evaluate several components for each stage of the processing. We use a set of classical tools for extracting, recognizing and tracking elements of the football game and we compare them with more advanced neural networks based approaches. However, the proposed method flow is not an end to end learning approach: the detection, recognition and tracking part have been already well investigated within the football context [3, 6, 21, 22, 24, 36]. Therefore, the proposed system starts by first extracting a set of objects o^I from image I

Table 2. Comparison of football players and ball detection using YOLO and OpenCV

		YOLO	OpenCV Color Histogram
Accuracy (avg.)		52%	80%
Detection Speed	Visual Representation	0.51/fps	1.04fps
	No Representation	0.95fps	1.61fps

(Figure **??**). This process is represented by a pixel wise labelling $\lambda : x_{ij} \longrightarrow o_{ij}$. The objects recognized are the players of each team, the referee and the ball. The individual players are not recognized individually and are only tracked while visible. The recognized and tracked players are labeled by colors representing their team.

The recognized objects o^I are tracked, and are used to construct a sequence of positions of labeled objects. Such sequence is used to train an LSTM that predict the next action.



(a) (b)

Fig. 5. Comparison of players and ball detection using YOLO 5a and opencv 5b.

4.1 Object Tracking/Detection

Object Detection is performed using only image processing tools (OpenCV library) in a similar fashion to [3, 6, 24, 36]. The reason for such choice is the fact that the environment has almost a constant background: the football field allows to subtract the color of the background and therefore the recognition and tracking of elements of interest is faster using traditional image processing approaches than for instance using a Convolutional Neural Network (CNN) to extract features.

For instance, Figure 5 shows the results of detection of football players and the ball using (a) YOLO [27] and (b) openCV accelerated detection. While both approaches are similar, it is simple to see that in this particular example the engineered features approach is more accurate. A comparative experiment is described in Table 2.

As can be seen the average speed and accuracy of openCV and color histogram is faster than using the YOLO approach. Therefore for this first stage of extracting game elements, the recognition using color histograms is preferred. The whole process of detection and tracking is done using a sequence of operations as follows:

(1) Every frame $x_i \in x$ is read and converted from RGB to HSV and HSL formats. This conversion facilitates the color specification for background subtraction and players recognition and tracking. The colors of each team jerseys can either be specified manually or are extracted by clustering from a sequence of k initial video frames.
(2) The green color is autonomously obtained by clustering pixels from k initial frames.
(3) The green color is removed and everything above the gren area is filtered
(4) The remaining colors (wihtout the black) are clustered and the dominant frequencies are the team colors.
(5) To remove the noise present due to overlapping and shadowing between the individual players (and in the crowd) and to make detection easier - 3 different morphological transformation operations are performed respectively for player detection, border detection and ball detection.

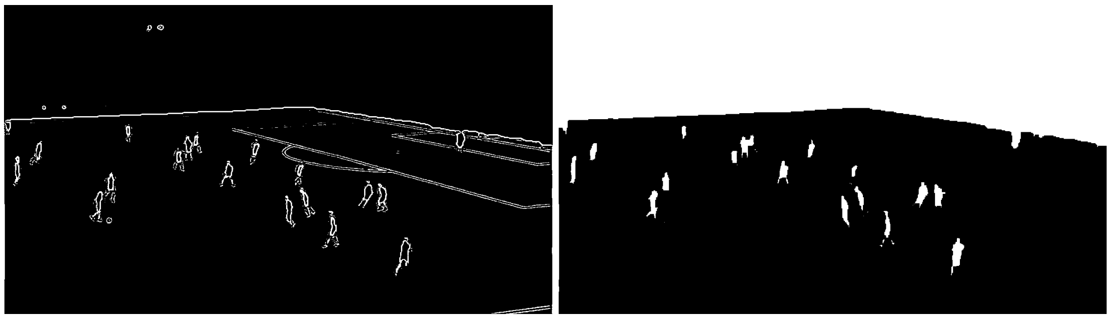

(a) (b)

Fig. 6. Image processed by adaptive threshold 6a and constant threshold for player detection 6b.

4.1.1 Player Detection. To separate each player from each other as much as possible, we used morphological closing which is Dilation followed by Erosion.The example of morphological closing on player can be seen in the Figure 6 and 7.

4.1.2 Border Detection. To recognize the football field borders, we invert the image colors and dilate the football pitch as much as possible to detect its borders.

4.1.3 Ball Detection. A contour of the ball was very small compared to the players, therefore first an adaptive threshold was used instead of a global preset value. However, in order to get full contours of the ball and players dilation with larger kernel (as compared to players and to border regions) must be used, which would decrease ball detection frequency near players. Instead, image processed by Adaptive Threshold is combined with player detection threshold image followed by morphological closing with lower kernel value. This method prevents the crossing between contours of players and ball compared to performing morphological closure with high kernel value. The comparison of detection accuracy of two methods can be seen in the Figure 7. Second, in order to remove noise data presented in the crowd, ball detection is applied only within the borders of the football pitch.

Finally, the colors only, do not allow to always distinguish the players, the borders and the ball. Therefore, in order to differentiate contours of ball and players we check the height and width of contours. The spatial conditions for detection of each of the detected objects are shown in Table 3.

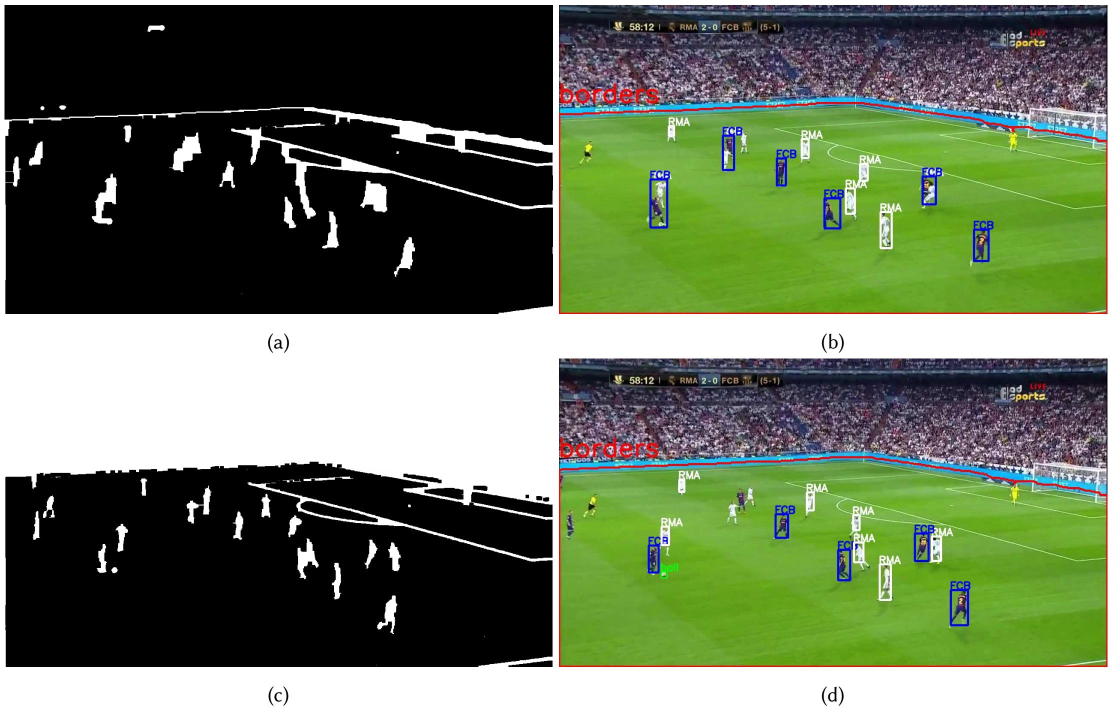


Fig. 7. Examples of ball detection using adaptive threshold and morphological closing with big kernel value 7c- 7d vs the combination of adaptive threshold, constant threshold and morphological closing 7a- 7b

Table 3. Spatial limits experimentally determined for distinguishing the relative size of recognized and tracked football game elements.

Player	$width > 4px$, $height > 6px$ $height > 1.5 * width$
Ball	$20px > width > 4px$, $20px > height > 6px$ $height < 1.2 * width$, $width < 1.2 * height$
Border:	$width > 500px$

This approach however is not very satisfactory due to the required parameterization. Therefore we compared it with a machine learning approach. We used transfer learning from a pre-trained YOLOv3 model in the training. We gathered a collection of 336 (266 train + 70 validation) match image samples and marked the positions of the ball in them. This approach was then evaluated for speed and for accuracy with open CV. The resul is shown in Table 4. The accuracy was averaged over sampled of 100 frames and testing batch also had frames with no ball. In such case if ball was not detected and indeed there was no ball, it was considered as a correct detection;

4.2 Scene Recognition

Beside recognizing and tracking the game elements, each image x^j in the video sequence was labeled with a scene label $l^j \in l$. For the scene labeling, the ImageAI library was used to classify each frame $x^t i \in x$ from each sequence

Table 4. Evaluation of ball detection approaches.

Ball Detecting Approach	Accuracy) (correct/total)	Time Performance (sec)
OpenCV	50% (50/100)	0.0235
Transfer Learning (YOLOv3)	70% (70/100)	0.0605

into l_k. The total number of labels for scene classification was 11 with the individual labelse being: $l = \{Corner$, $CounterAttack$, $Foul$, $GoalAttack$, $Goalkeeper$, $LogoView$, $Midfield$, $Outfield$, $Penalty$, $Player(close-up)$, $Referee\}$. These 11 labels were selected due to both their visual prominence in streamed football games as well as to their relevance for further game analysis. The The scene classification was done using the Densely Connected Convolutional Networks (DenseNet) [9] and the training parameters are: number of training epochs was 200, batch size of 16.

4.3 Data preparation for LSTM Training

The gathered data has the general form of (s^t, a^t) , however because one action in general spans several samples the sequence of samples can be also written from the point of view of actions. For instance, Table 5 shows the sequence of 6 actions each associated with corresponding number of samples. Table 5 shows that action 1 lasted for 50 frames,

Table 5. Example of a sequence of actions with the associated number of samples per action

# frames	50	20	60	40	25	15
action	1	2	11	3	1	7

action 2 followed for 20 frames and so on. To train on LSTM training data must have fixed shape similar to (#of samples, time_frames, features). Therefore, to accommodate this variable number of frames per action, we consider the time_frames variable to be the largest possible number of frames per action. For example: if prediction is based on 3 previous action: max is taken among (50+20+60), (20+60+35), (60+35+45),(35+45+10),(45+10+25). Then data is pre-padded (filled with zeros if duration of sequence is less than max_sequence).

Thus, shape of input to LSTM is
(#of samples, number_of_frames_in_max_sequences, features). Since the number of previous actions varied, shape of inputs to LSTM was as following: (14000, 240, 51), (14000, 210, 51), ... Experiments with no player coordinates were also made. Shape of input was (#of_sample,max_sequence, 7)

The training data set consists of 14650 images, which were gathered by cropping real football games videos. The action distribution is shown in Figure 8.

Note that the resulting labeled data is not completely balanced. In particular the actions 1,2,7 and 8 that are the short and middle pass to the left and to the right are dominating the actions distribution.

4.4 Action Prediction

To final step in the processing flow is the module that predict football actions. For this task the labelled dataset acquired in the previous step (Section 3) was used.

Since, our data contains continuous information flow, LSTM implementation was used. In addition, to get confidence for each predicted action, the problem was transformed to multi-class classification problem. In other words, 1

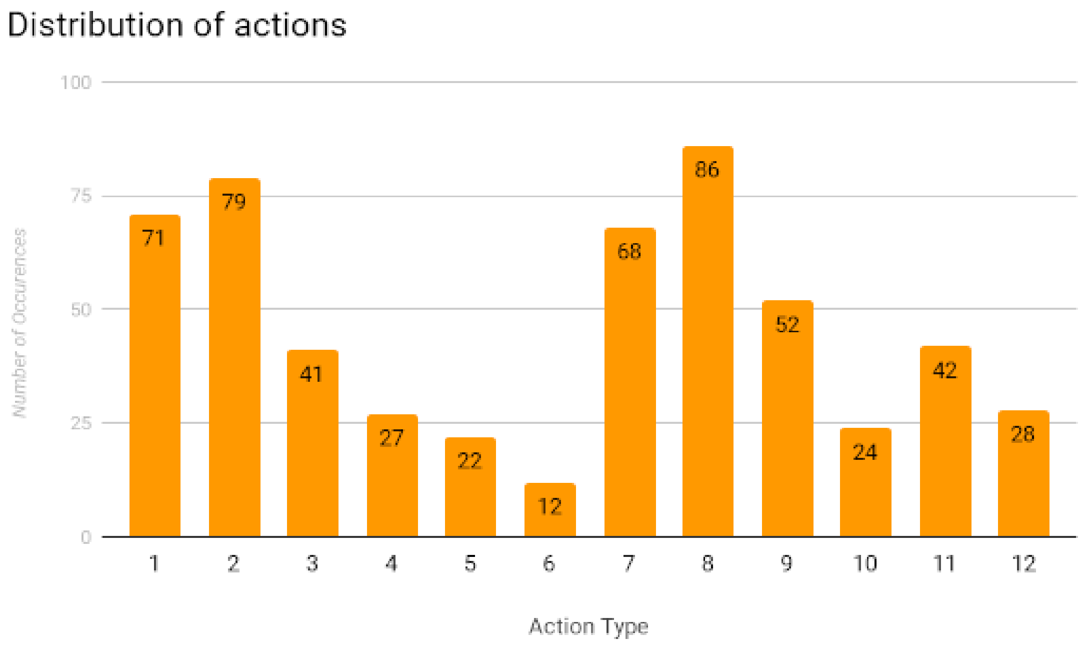


Fig. 8

numbered labels were transformed to one-hot array encoding (action 2: 0, 0, 1, 0, 0, 0, 0, 0, 0, 0). Shape of X_train must be in the form: (number of samples, number of time series, features), in our case: (number of samples, 2, X). The number of samples X depends on whether the experiment was using the spatial coordinates of players and of the ball for learning and inference.

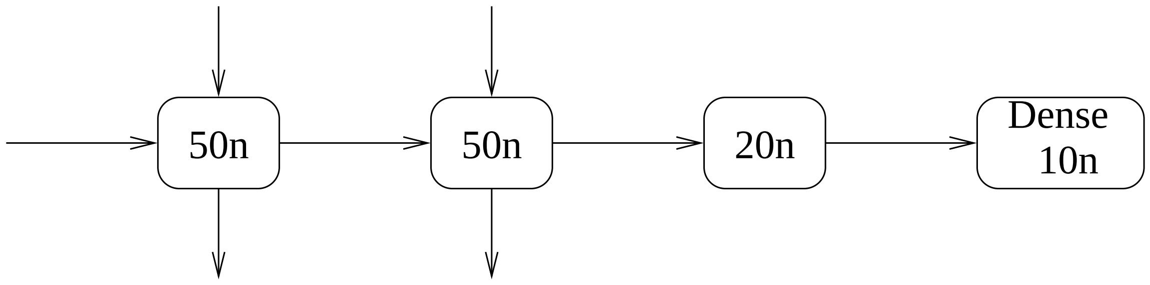


Fig. 9. Schematic representation of used LSTM in action prediction task

To get confidence distribution softmax activation at Dense layer was used. The general scheme of the used LSTM is shown in Figure 9.

The used LSTM is fed a sequence of actions and the target of the inference is to estimate the next action. This means that the LSTM is trained to predict next action after the current one and one that is different than the current action.

Table 6. Summary of action prediction results

Experiment	Settings	Accuracy				
1	Unbalanced	{50,50,20}	100ep	z=-2	s_s	35.5%
1	Unbalanced	{50,50,20}	300ep	z=-1	s_s	33.3%
1	Unbalanced	{50,50,20}	100ep	z=-3	s_s	35.7%
1	Balanced	{200,100,50}	500ep	z=-2	s_s	30%
2	Balanced	{50,50,20}	100ep	z=-2	s_c	25.7%
2	Balanced	{50,50,20}	200ep	z=-2	s_c	26.1%
2	Balanced	{50,50,20}	1000ep	z=-2	s_c	27.6%
3	Unbalanced	{50,50,20}	200ep	z=-2	s_c	33.74%
3	Unbalanced	{50,50,20}	200ep	z=-10	s_c	34.71%

The input to the LSTM is a sequence of samples spanning several actions with the goal of predicting the next action. To evaluate the LSTM predictive power, the prediction based on 1, 2 or 3 previous actions was evaluated.

5 EXPERIMENTS

In order to verify the task difficulty and the ability to predict next action form the given information, several experiments have been run.

First a set of experiments used the game states s^t as input but without taking into account the spatial position of the players and of the ball. This means that the data samples contained only the following information: s_s ={team possessing the ball [0,1], number of visible players from each team [1-11],[1-11], scene label [0,1], current action [1-12]}. The LSTM was trained using one, two and three immediate past actions and was evaluated at the prediction of the next action. This is referred to as experiment 1 in Table 6.

Experiment 2 predicts again the next possible action in the game, however the input state vector contains the information from the s_s sample data and additionally all the spatial coordinates of all recognized game elements. This input information is referred to as s_c.

Finally experiment 3 shows the frame-wise prediction. Unlike the action-wise prediction, frame-wise prediction is intended to predict next action for each next frame. For instance, given is a $\{x^{t-3}, x^{t-2}, x^{t-1}, x^t\}$ annotated frames with actions $\{a^{t-3}, a^{t-2}, a^{t-1}, a^t\}$ the target is to predict for x^{t+1} the action a^{t+1}. Therefore, instead of predicting next action from a set of previous actions, we predict an action for the next frame.

Table 6 shows the accuracy of estimation next action in the third column and the type of experiment in the first column. The second columns shows i the training and testing data was re-balanced in the relative number of action instances. The fourth column shows the configuration of the LSTM (neurons/layer), the fifth column shows the number of epochs during training, the sixth column shows how many previous actions are considered for next action estimation and finally the type of dataset used (s_s or s_c).

The results show several facts about the dataset and the problem. When using the unbalanced dataset, the results are in general higher than when using a balanced dataset for training and testing. The reason for this is that in the unbalanced dataset there is a prevalence of four main actions 1,2,7,8 (Figure 8). The resulting LSTM would always decide that next action is one of the four, 1,2,7,8 and therefore the accuracy would result in above 30%.

When using the balanced dataset, that is a normalized dataset where occurrences of all actions are equal, the accuracy drops to around and below 30%. Additionally, considering that results from experiment 2 are in general lower than the results from experiment 1, indicates that when using s_c for input a more complex LSTM is needed. Also, the prediction setup always trains the LSTM to predict the next action from a set of sequences corresponding to finished actions. The low accuracy of these experiments indicates that the system should also be trained to predict actions on a frame by frame basis.

Note that the frame-wise action prediction is equivalent no to an estimation of next action but rather to classification of current incomplete action in to a label. This task is similar to current approaches in this category such as [15].

Additionally, the used LSTM architecture might require more complex structure to take into account the relations between the positions of the players, their velocities and the position of the ball and its velocity to estimate more accurately the possible actions.

The experiments in this paper also show that using directly actions might be too coarse of an approach. This can be seen because in experiment 3 the accuracy is increasing with the increased number of past frames. While using too many previous actions to predict next one in experiments 1 and 2 might be too far to be meaningful, allowing a large number of previous frames to be used directly for prediction is more reasonable.

A final set of experiments was conducted to evaluate a end-to-end machine learning approach as a base line. For this we evaluated a VGG feature extractor directly feeding the features into an LSTM. The configuration of the LSTM is same for both experiments and the main difference is that the VGG based approach feeds image features to the LSTM.

We compared three different models described in Table 7.

Table 7. Models for action detection details trained on the RMA-FCB game

Model name	Description	Training
m1	OpenCV model	200 epochs
m2	VGG model	20 epochs
m3	VGG model	200 epochs

The comparison of the trained models was done over the accuracy of action prediction as well as over the speed and for each separate stage separately. The comparison was done over the test set of the created RMA-FCB game and it is shown in Tabel 8.

Table 8 shows several results. First, the first and third columns show the stages in each approach. The second and fourth columns show the times measured in seconds for each stage. The bottom lines shows the average accuracies when each algorithm is tested on the validation dataset. the first observation is that the proposed approach has more individual stages while the base line end-to end learning is simpler in general. The second main observation that the stages using the end-to-end learning are in general longer, i.e. the processing is slower. Finally, the accuracies of prediction are similar for all three algorithms. As a result the proposed approach is performing quite well compared to the more modern base line.

However, while the results of action estimations are not very satisfying, a closer look at the estimation results shows that the tested LSTM approach is not most suitable. Therefore in the future a more elaborate approach is expected to be applied.

Table 8. Comparison of the base line model with the proposed approach for the processing speed

VGG	Secs per frame	OpenCV	Secs per frame
		K-means fitting 2 dominant colors	5.804
		Scene recognition time(once in 15 frames)	0.047
Measure VGG feature prediction	0.0168	Detection of players and s ball using color histogram and K-mean	0.038
Dataset appending (each frame)	0.00231	Ball data noise removal (DB-clustering)	0.00165
Dataset merging	130.983	Dataset appending (each frame)	0.00015
Measure each epoch while training model	m2: 30.64 m3: 29.61	Measure each epoch while training	m1: 21.870
Prediction action model	m2: 0.0289 m3: 0.027	Predicting action	m1: 0.0149
Average Accuracies:	m1:26.3%	m2: 23.11%	m3:25.8%

6 CONCLUSIONS

In this paper we presented a study on the predictability of a group sports next action. The prediction levels remain relatively low at this stage but the created dataset can be used for further improvements and study.

The future work is to increase the granularity of the proposed system and instead of predicting next different action perform a frame by frame statistical estimation of actions. This allows to exploit the continuity between spatial coordinates in the visual information and the strategic continuity between the actions performed by the team.

REFERENCES

[1] Bregonzio, M.; Gong, S.; and Xiang, T. 2009. Recognising action as clouds of space-time interest points. *2009 IEEE Conference on Computer Vision and Pattern Recognition* 1948–1955.

[2] Cao, Y.; Barrett, D.; Barbu, A.; Narayanaswamy, S.; Yu, H.; Michaux, A.; Lin, Y.; Dickinson, S.; Siskind, J. M.; and Wang, S. 2013. Recognize human activities from partially observed videos. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2658–2665.

[3] Dearden, A.; Demiris, Y.; and Grau, O. 2006. Tracking football player movement from a single moving camera using particle filters. In *3rd European Conference on Visual Media Production (CVMP 2006). Part of the 2nd Multimedia Conference 2006.* IEE.

[4] Fahong Li, and Woodham, R. J. 2005. Analysis of player actions in selected hockey game situations. In *The 2nd Canadian Conference on Computer and Robot Vision (CRV'05)*, 152–159.

[5] Fanti, C.; Zelnik-Manor, L.; and Perona, P. 2005. Hybrid models for human motion recognition. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, 1166–1173 vol. 1.

[6] Gedikli, S.; Bandouch, J.; von Hoyningen-Huene, N.; Kirchlechner, B.; and Beetz, M. 2007. An adaptive vision system for tracking soccer players from variable camera settings. In *ICVS 2007.*

[7] Gómez, A. B.; Albacete, E.; Marrero, I.; and Saez, Y. 2016. Real-time prediction of gamers behavior using variable order markov and big data technology: A case of study. *International Journal of Interactive Multimedia and Artificial Intelligence* 3(6):44–51.

[8] Gorelick, L.; Blank, M.; Shechtman, E.; Irani, M.; and Basri, R. 2007. Actions as space-time shapes. *IEEE Trans. Pattern Anal. Mach. Intell.* 29(12):2247–2253.

[9] Huang, G.; Liu, Z.; and Weinberger, K. Q. 2016. Densely connected convolutional networks. *CoRR* abs/1608.06993.

[10] Ji, S.; Xu, W.; Yang, M.; and Yu, K. 2013. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35(1):221–231.

[11] Ju Sun; Xiao Wu; Shuicheng Yan; Cheong, L.; Chua, T.; and Jintao Li. 2009. Hierarchical spatio-temporal context modeling for action recognition. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2004–2011.

[12] Kar, A.; Rai, N.; Sikka, K.; and Sharma, G. 2016. Adascan: Adaptive scan pooling in deep convolutional neural networks for human action recognition in videos. *CoRR* abs/1611.08240.

[13] Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; and Fei-Fei, L. 2014. Large-scale video classification with convolutional neural networks. In *CVPR*.

[14] Kong, Y., and Fu, Y. 2018. Human action recognition and prediction: A survey. *CoRR* abs/1806.11230.

[15] Kong, Y.; Gao, S.; Sun, B.; and Fu, Y. 2018. Action prediction from videos via memorizing hard-to-predict samples. In *AAAI Conference on Artificial Intelligence*.

[16] Kong, Y.; Kit, D.; and Fu, Y. 2014. A discriminative model with multiple temporal scales for action prediction. In *European conference on Computer Vision (ECCV)*.

[17] Kong, Y.; Tao, Z.; and Fu, Y. 2017. Deep sequential context networks for action prediction. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3662–3670.

[18] Laptev, I. 2005. On space-time interest points. *International Journal of Comput Vision* 64:107–123.

[19] Li, K., and Fu, Y. 2014. Prediction of human activity by discovering temporal sequence patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36(8):1644–1657.

[20] Li, K.; Hu, J.; and Fu, Y. 2012. Modeling complex temporal composition of actionlets for activity prediction. In *European conference on Computer Vision*.

[21] Liu, J.; Tong, X.; Li, W.; Wang, T.; Zhang, Y.; and Wang, H. 2009. Automatic player detection, labeling and tracking in broadcast soccer video. *Pattern Recognition Letters* 30(2):103 – 113. Video-based Object and Event Analysis.

[22] Lu, W.; Ting, J.; Little, J. J.; and Murphy, K. P. 2013. Learning to track and identify players from broadcast sports videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35(7):1704–1716.

[23] Lukac, M.; Kameyama, M.; and Migranova, Y. 2017. Live-feeling communication: Multi-algorithm approach to the estimation of human intentions,. In *Proceedings of IEEE SMC*.

[24] Mazzeo, P. L.; Spagnolo, P.; Leo, M.; and D'Orazio, T. 2008. Visual players detection and tracking in soccer matches. In *2008 IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance*, 326–333.

[25] Ng, J. Y.; Hausknecht, M. J.; Vijayanarasimhan, S.; Vinyals, O.; Monga, R.; and Toderici, G. 2015. Beyond short snippets: Deep networks for video classification. *CoRR* abs/1503.08909.

[26] Pei, M.; Yunde Jia; and Zhu, S. 2011. Parsing video events with goal inference and intent prediction. In *2011 International Conference on Computer Vision*, 487–494.

[27] Redmon, J., and Farhadi, A. 2018. Yolov3: An incremental improvement. *arXiv*.

[28] Ryoo, M. S. 2011. Human activity prediction: Early recognition of ongoing activities from streaming videos. In *2011 International Conference on Computer Vision*, 1036–1043.

[29] Simonyan, K., and Zisserman, A. 2014. Two-stream convolutional networks for action recognition in videos. *CoRR* abs/1406.2199.

[30] Soomro, K.; Idrees, H.; and Shah, M. 2019. Online localization and prediction of actions and interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41(2):459–472.

[31] Soomro, K.; Zamir, A. R.; and Shah, M. 2012. UCF101: A dataset of 101 human actions classes from videos in the wild. *CoRR* abs/1212.0402.

[32] Sun, C.; Myers, A.; Vondrick, C.; Murphy, K.; and Schmid, C. 2019. Videobert: A joint model for video and language representation learning. *CoRR* abs/1904.01766.

[33] Sun, D.; Roth, S.; and Black, M. J. 2010. Secrets of optical flow estimation and their principles. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2432–2439.

[34] Varol, G.; Laptev, I.; and Schmid, C. 2018. Long-term temporal convolutions for action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40(6):1510–1517.

[35] Wang, H.; Kläser, A.; Schmid, C.; and Liu, C. 2011. Action recognition by dense trajectories. In *CVPR 2011*, 3169–3176.

[36] Xu, M.; Lowey, L.; and Orwell, J. 2004. Architecture and algorithms for tracking football players with multiple cameras. In *IEE Intelligent Distributed Surveilliance Systems*, 51–55.