UNIVERSITY OF BUCHAREST

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

DISSERTATION

SCIENTIFIC COORDINATOR

Conf. Dr. Alexe Bogdan

STUDENT

Petrișor-Ionuț Calofir

BUCHAREST

2020

UNIVERSITY OF BUCHAREST

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

Video Action Recognition In Synthetic Soccer Games

SCIENTIFIC COORDINATOR

Conf. Dr. Alexe Bogdan

STUDENT

Petrișor-Ionuț Calofir

BUCHAREST

2020

Contents

Chapter 1

Introduction

In recent years, significant progress has been made in video recognition. A notable mention is [3] in which the authors introduced a new architecture for videos called Inflated 3D ConvNet (I3D) and a new dataset called Kinetics [4] containing 400 human action classes.

Due to these advances, researchers also began to investigate video recognition on sports videos. This task aims to recognize the activity from a group of players interacting with each other and has many practical applications such as helping to better understand the performance of the players or summarizing a long video showing only the important parts.

In this work we chose to investigate video recognition in soccer and to see if the goal probability of a shot can be predicted using images or videos. Our motivation for choosing this topic was the fact that soccer is one of the most watched sport in the world and we wanted to contribute and improve the work in this topic. One of the problem with this topic is that there are not many datasets available, so because of this we generated a synthetic data.

More specifically, the contributions of our work are: 1) introducing a new synthetic dataset for soccer; 2) investigating if the video recognition methods work for soccer videos and if it is possible to extract specific events from a match; 3) investigating if predicting the goal probability can be achieved only from images or videos. The previous mentioned things come with different challenges that we will highlight in the following chapters.

The structure of this work is as follows: in Chapter 2 we will present how we generated the synthetic soccer dataset. The created dataset contains three classes (no action, pass and shot) and is built using a football engine; in Chapter 3 we will

present how video recognition can be applied on short soccer videos successfully, the main goal being to summarize an entire soccer match; in Chapter 4 we will present and compare two methods used for predicting the goal probability of a shot. The first method uses only images and the second one uses short videos.

1.1 Related Work

The most notable works for video recognition are [3][6]. [6] won the first place at International Challenge on Activity Recognition (ActvityNet) from 2019 and introduced a new architecture called SlowFast which is currently used as backbone in state-of-the-art approaches for video recognition.

There are few studies about video recognition in soccer which are related to our current work. In [7] the authors introduce a new dataset called SoccerDB that contains multiple events from a soccer match such as shots, goals, corners, free kicks and substitutions. However the dataset is not currently available, but the authors said they will publish it in the near future. In [10] the authors build a synthetic dataset using the Google Football engine and use some heuristics to recognize different events. In [11] the authors investigate the usage of video recognition methods on soccer videos.

Currently, expected goals are computed from raw features like players position, the type of the situation (open play, set piece). There are currently no works for predicting the goal probability from raw images or videos.

Chapter 2

Dataset

In [8] the researches from Google Brain introduced a football engine written in C++ and Python which is based on the engine from [5]. The engine simulates an entire football match and contains all the events that may occur during the match (e.g. corner kick, free kick, penalty kick, etc.).

The authors released this engine to be used as a reinforcement learning environment. However, because the engine is open-source it can be modified to be used for various tasks.

Figure 2.1: *The Google Research Football Environment. This image was taken from [8].*

2.1 Dataset Creation

Similar to [10], we modified the engine to create a synthetic dataset that can be used for Video Recognition and Expected Goals tasks. At every time frame we extracted the status of the ball (whether it is in the possession of a player or not), what action is performed by each player and the screenshot of the frame. Also, as can be seen in Figure [2.1] the original engine shows different information such as scoreboard, radar and the name of the current player. For creating the dataset we removed these information so only the players and the ball are visible.

In order to analyze the video recognition in soccer we created three classes: pass action, shot action and no action as described in the below subsections based on the information retrieved from the engine.

2.1.1 Pass Action

The pass action occurs when a player performs a pass (information retrieved from the engine) and the ball reaches another player from his team or from the opposing team or it goes out of play. An example can be seen in Figure [2.2].

Figure 2.2: *Example of pass action.*

2.1.2 Shot Action

The shot action occurs when a player performs a shot (information retrieved from the engine) and the ball travels a certain distance until is blocked by another player or

goes out of play. An example can be seen in Figure [2.3].

Figure 2.3: *Example of shot action.*

2.1.3 No Action

The no action class contains videos of different lengths randomly chosen from the match which are different from the ones present in the above two classes. An example can be seen in the Figure [2.4].

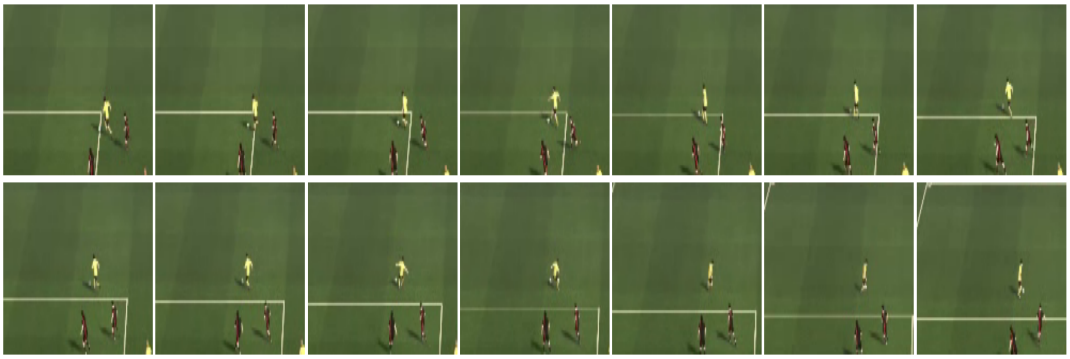
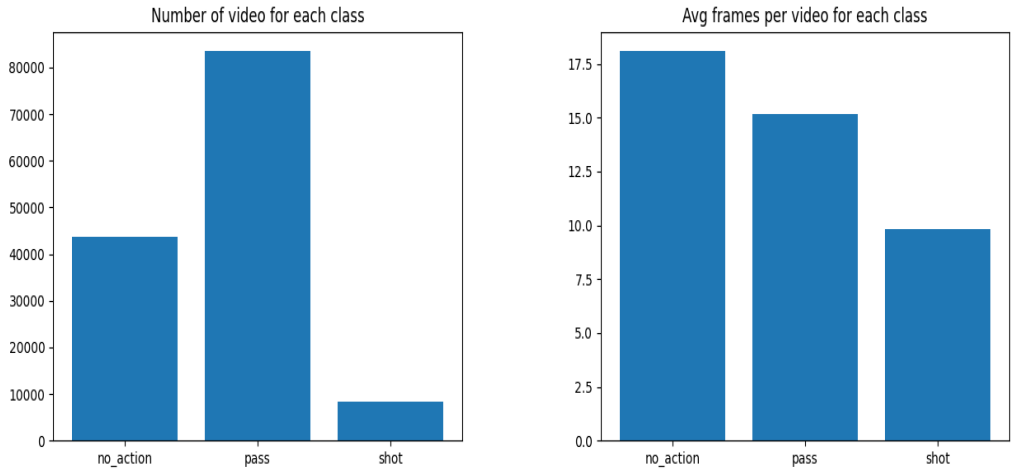


Figure 2.4: *Example of no action.*

2.1.4 Dataset Description

The dataset was created from 1049 simulated games. Each game has approximately 3000 frames and is displayed at 10 frames per second meaning its duration is 5 minutes. There are in total 135591 generated videos having a duration of 59 hours. As can be seen from Figure [2.5a] the classes are imbalanced. Another observation is

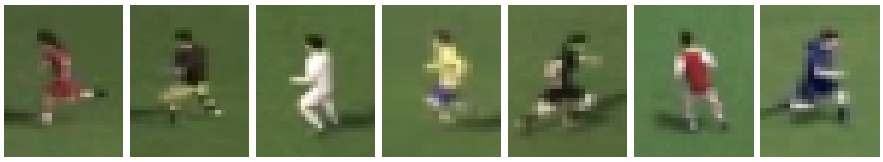
that the average number of frames for shots is lower than the average number of frames for passes (Figure [2.5b]).



(a) **(b)**

Figure 2.5: *Dataset description.*

There are 12 teams. Each team has an unique kit as can be seen in Figure [2.6]. During the generation of the dataset, the teams were chosen randomly.



(a) **(b)** **(c)** **(d)** **(e)** **(f)** **(g)**

Figure 2.6: *Different teams.*

Chapter 3

Video Recognition

3.1 Experiments

For this topic we chose to investigate only two actions: passes and shots. However, during a football match there are multiple actions like: corner kick, penalty kick, foul, throw-in or free kick. In our case, these action are treated as being part of the no action class. The purpose of this topic was to see if video recognition can be applied successfully on synthetic soccer games, but the engine and the models used can be easily extended to include the previous mentioned actions.

3.1.1 Dataset

The dataset used is described in Chapter 2, we split the dataset as follows: 70% of videos are used for training, 15% of videos are used for validation and 15% of videos are used for testing. Because the classes are imbalanced, the split was made in a stratified fashion meaning that the percentage of samples for each class is kept for the three sets (train/validation/test). The actual number of samples for each set can be seen in Table [3.1].

Class	Train	Validation	Test
No Action	30694	6577	6578
Pass	58680	12574	12574
Shot	5919	1269	1268

Table 3.1: *Number of samples for each class for the three sets.*

3.1.2 Architecture

We used the I3D architecture introduced in [3] with inflated ResNet-50 [1]. Because a lot of data is required to train such models for video recognition we used a pre-trained model trained on Kinetics-400 dataset and finetuned it on our dataset. The pretrained model can be found at [9].

3.1.3 Training

The inputs of the network are videos of size 3 x T x H x W. The T represents the number of frames sampled from each video. The H and W represent the height and width of the frames. The number 3 represents the number of channels, in this case representing the RGB color model.

The input videos are resized to 224 x 224 and T is set to 8. The sampling of the frames for a video is done in the following way for each selected frame, skip the next 3 frames. If the video is smaller then the last frame is sampled until the number of selected frames is equal to 8, and if the video is bigger, then a starting frame is randomly chosen.

Other hyperparameters that we used during training are: BATCH_SIZE = 8, LEARNING_RATE = 0.01, OPTIMIZER = sgd and LOSS FUNCTION = $cross-entropy$. For the loss function we used a weight for each class because they are imbalanced. Also as data augmentation, at training time, random horizontal flip is performed for a video with probability $p = 0.5$

Another idea is to have the crops around the players and the ball as additional inputs to the network. The authors from [11] tried this idea, but they obtained a lower performance when adding the additional crops. They obtained the best performance when they used only the raw frames so for our experiments we did the same and focused on only the raw frames.

3.1.4 Evaluation

Because the classes are imbalanced, accuracy metric can not be used because it does not offer relevant information. Instead, for evaluating the performance of the model we used the f1 score. Because there are 3 classes and also they are imbalanced we used a modified version of the f1 score. We computed the f1 score for each class and then computed the average considering the percentage of each class.

3.2 Results and Discussion

In Table [3.2] and Figure [3.1] are the results on the test set. Also, it can be seen that on all classes the recall is quite good and the model is able to identify the majority of the videos from the respective classes. However, looking at at the confusion matrix, we can see that there are two notable difficulties: the network predicts some pass videos as no action and some no action videos as shots.

Class	Precision	Recall	F1
No Action	0.8819	0.9338	0.9071
Pass	0.9751	0.9303	0.9522
Shot	0.8428	0.9659	0.9001

Table 3.2: *Results on the test set.*

3.2.1 Grad-CAM Visualization

To better understand and visualize what the network learned, we used Grad-CAM introduced in [2] which is a method for visualizing the activations from a specific layer when predicting a certain class. All figures from this section represent some videos from the test set (the frames are from left to right).

In Figures [3.2] and [3.3] are present two videos from the no action class. We see that the activations are present on the entire frame which a strong indication that there is no event happening in that video.

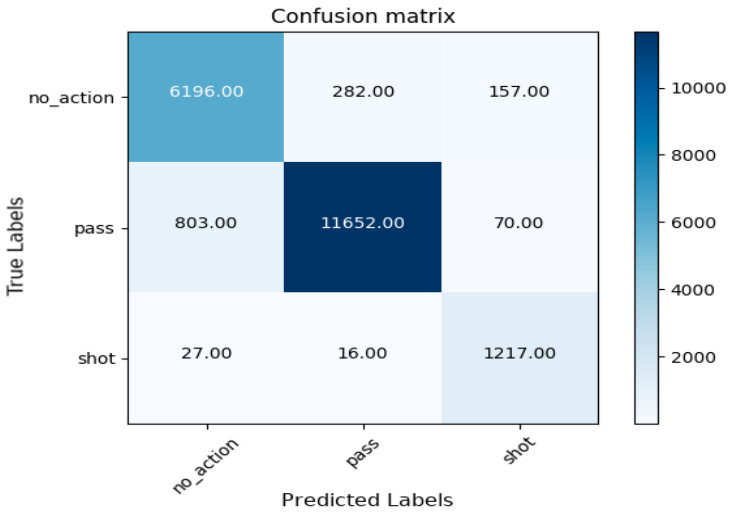


Figure 3.1: *Confusion matrix for the test set.*

In Figures [3.4] and [3.5] are present two videos from the pass class. We see that the network focuses especially on the ball and the players involved in this action.

In Figures [3.6] and [3.7] are present two videos from the shot class. Also we see that the network focuses on the ball. In Figure [3.7] there is another interesting observation because we can see that the network focuses on the region around the goal and we believe it is because the shots happen only in that region so besides the ball, the network looks at the current location.

Overall, from all figures we can see the that the network is able to recognize the actions and to focus on the important things such as the ball, the players involved in the action and the location of the video.

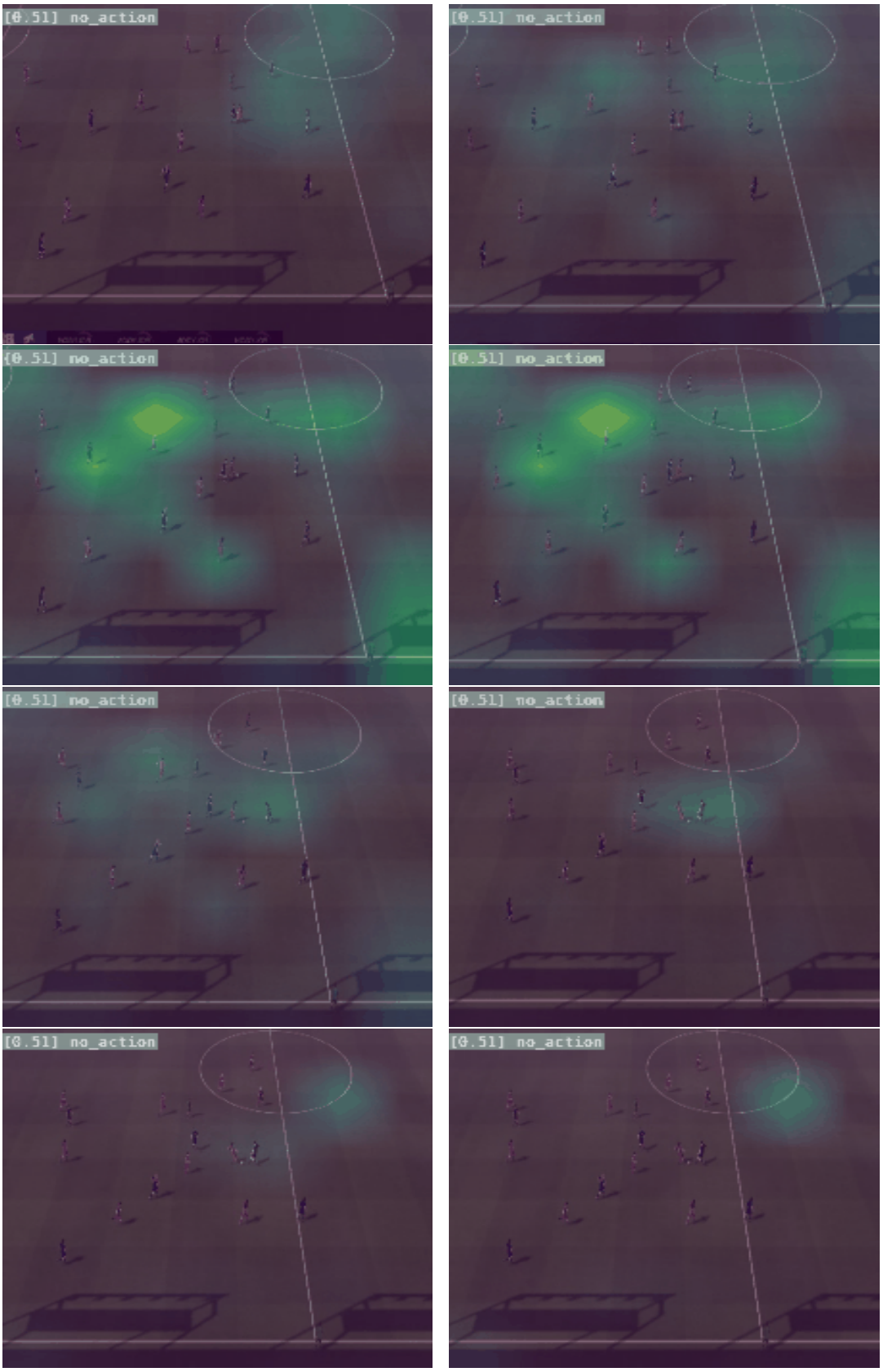


Figure 3.2: *Example of no action class.*

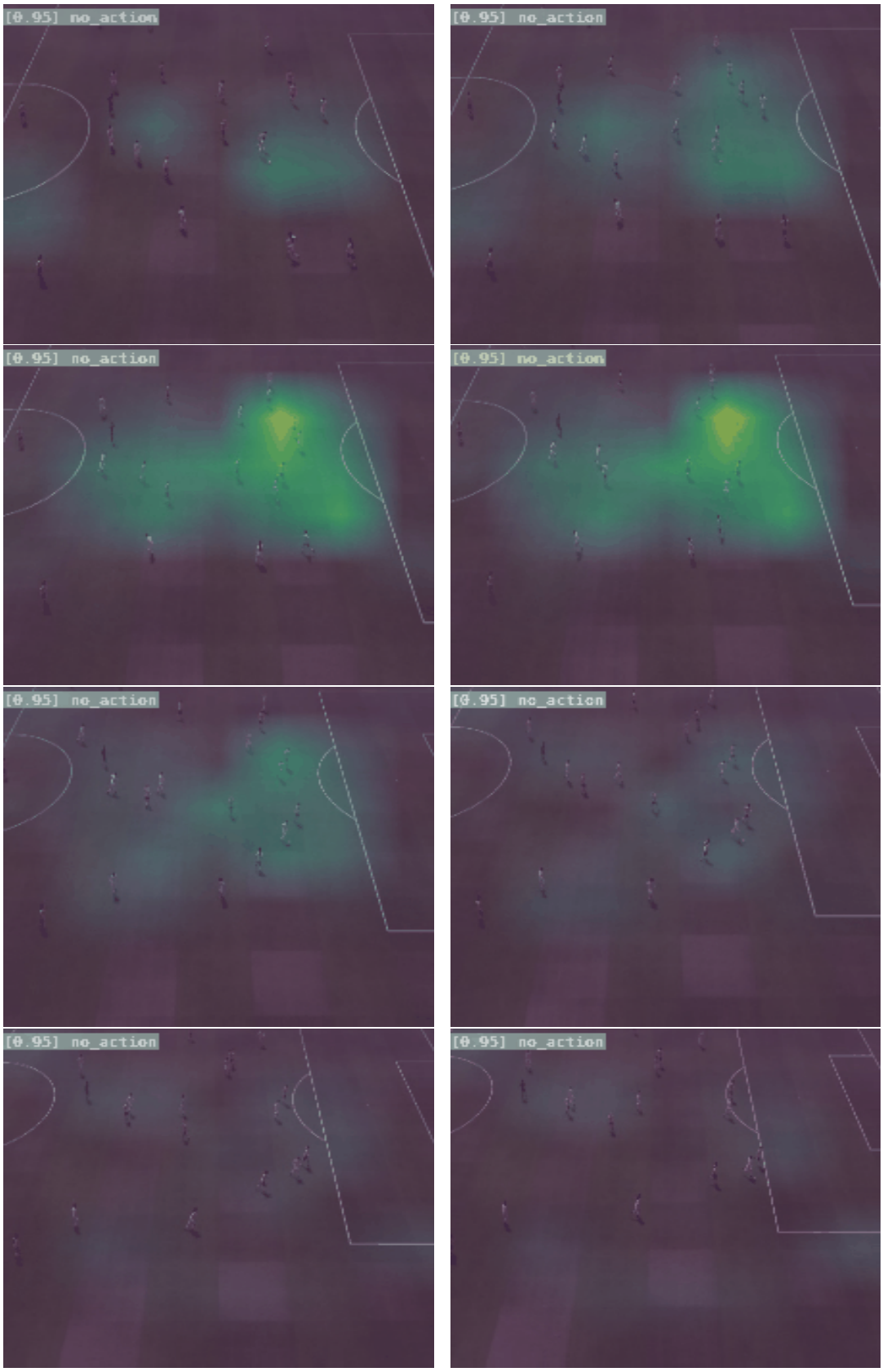


Figure 3.3: *Example of no action class.*

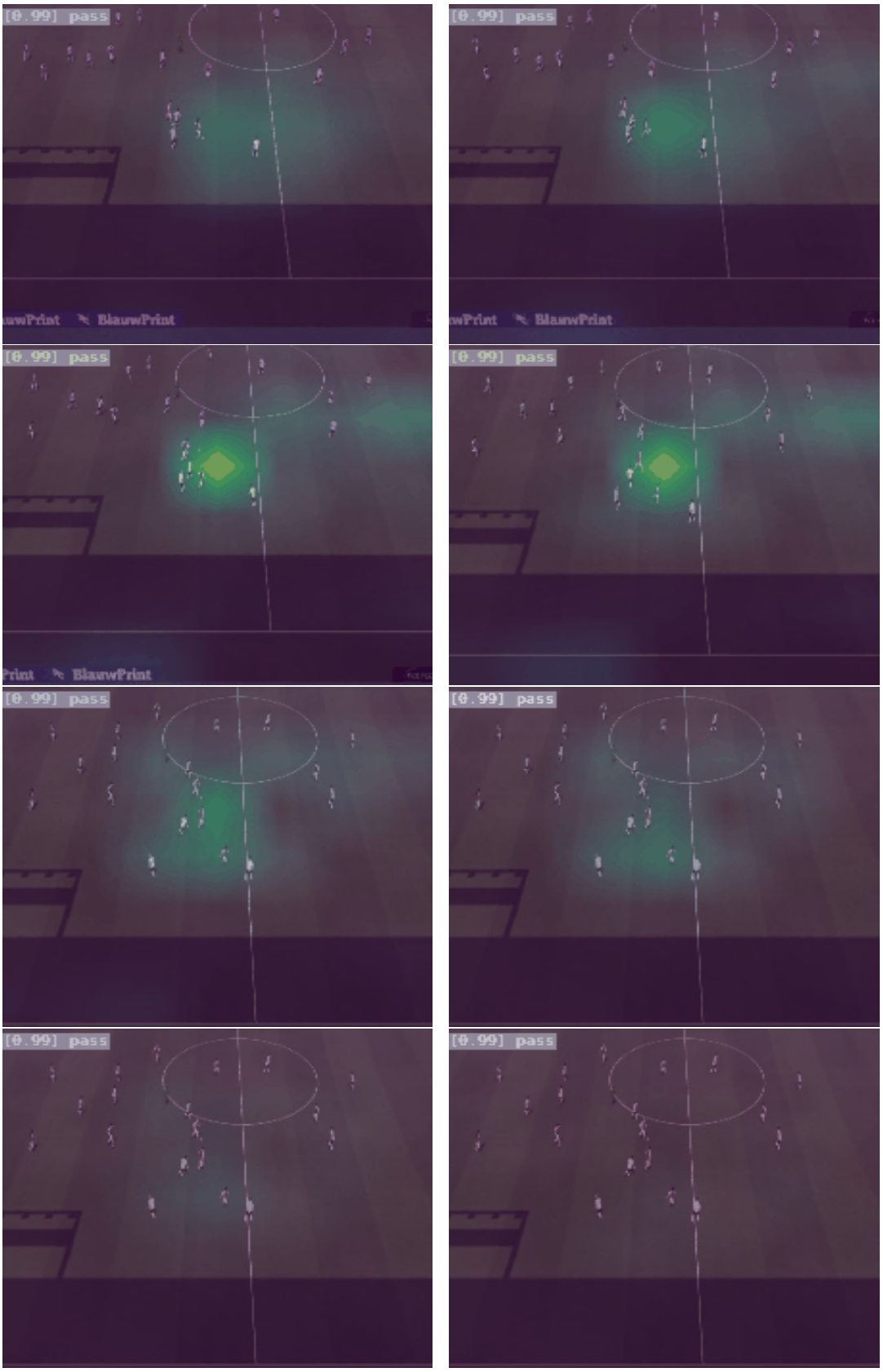


Figure 3.4: *Example of pass class.*

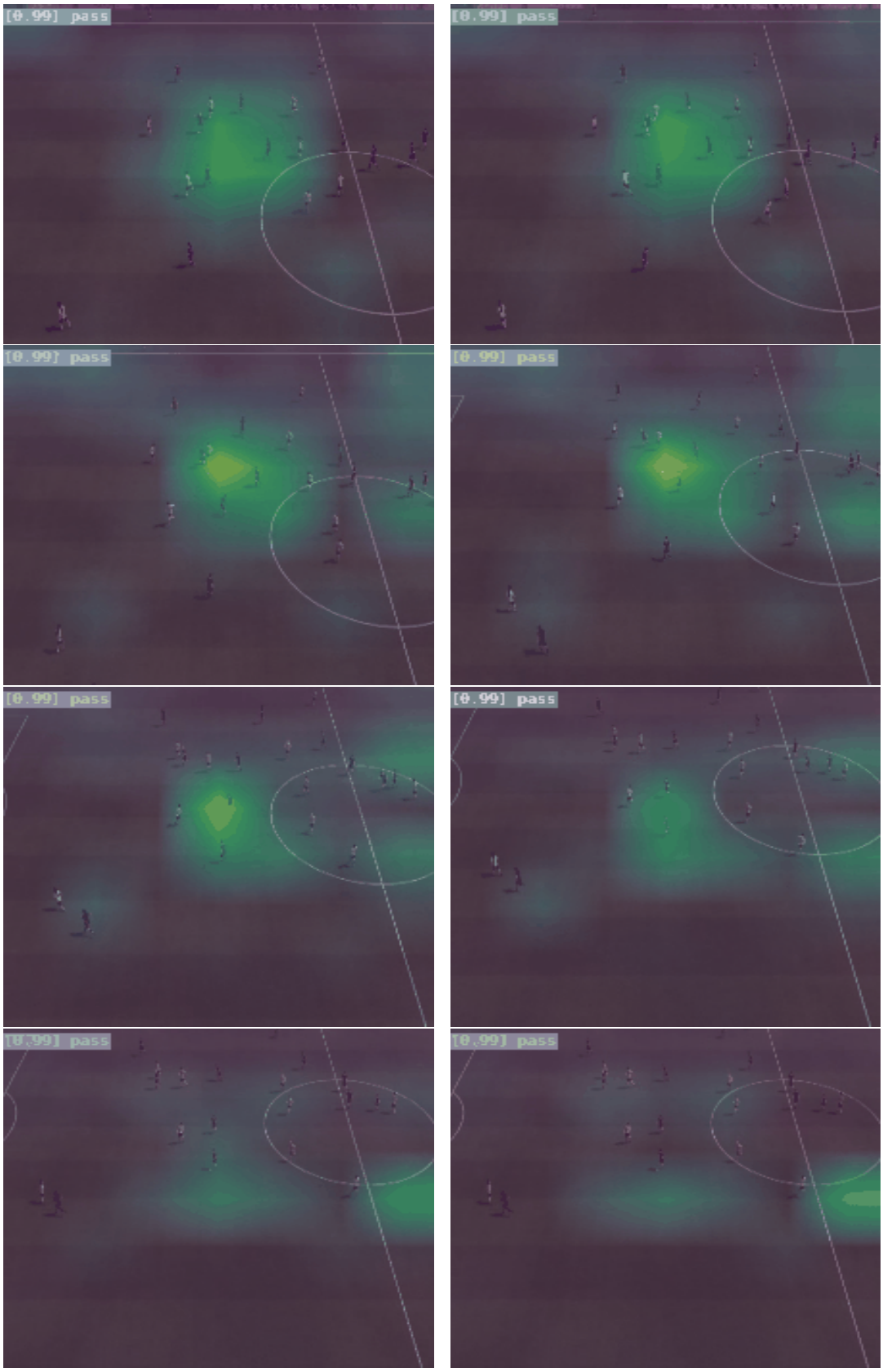


Figure 3.5: *Example of pass class.*

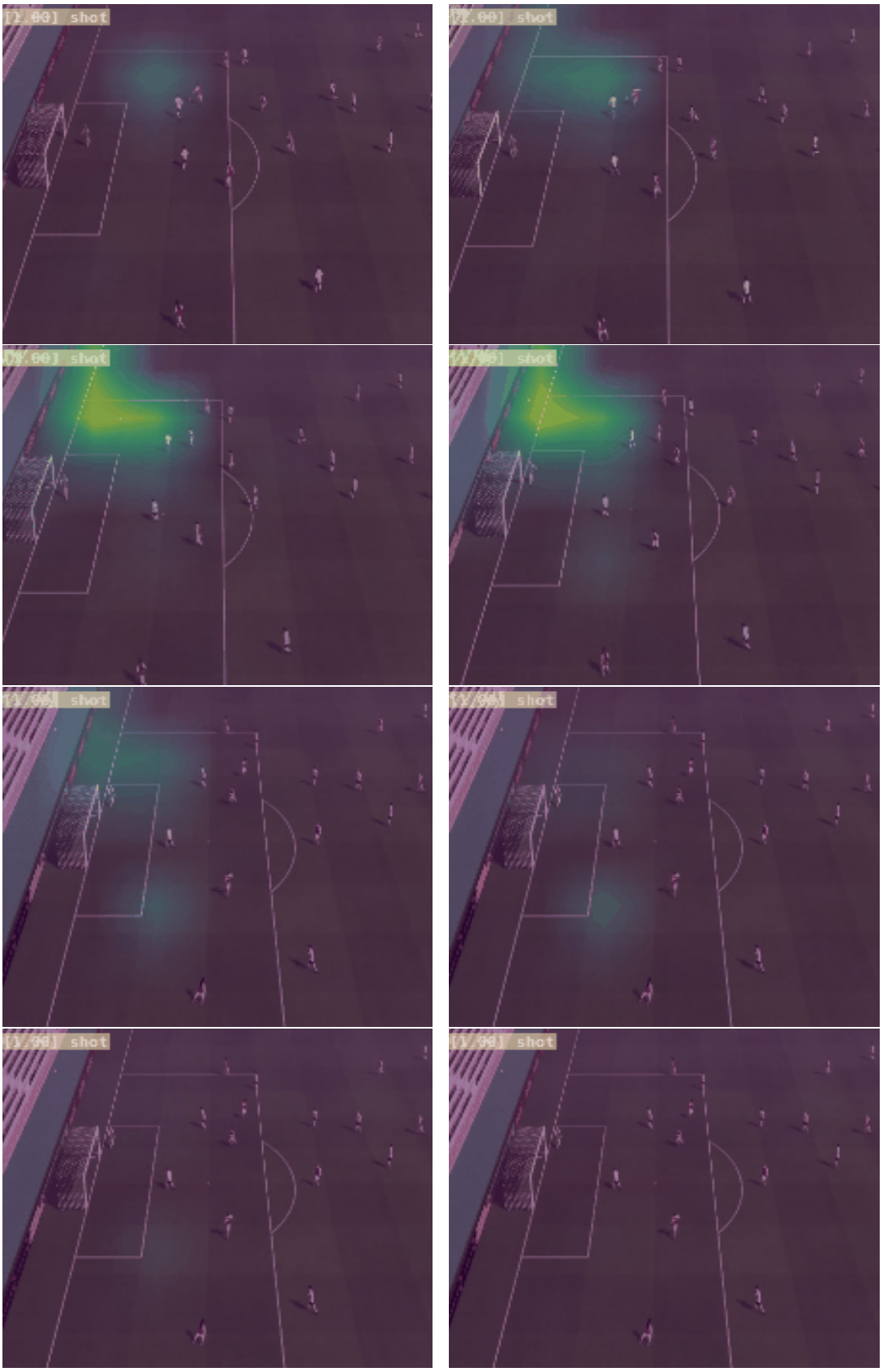


Figure 3.6: *Example of shot class.*

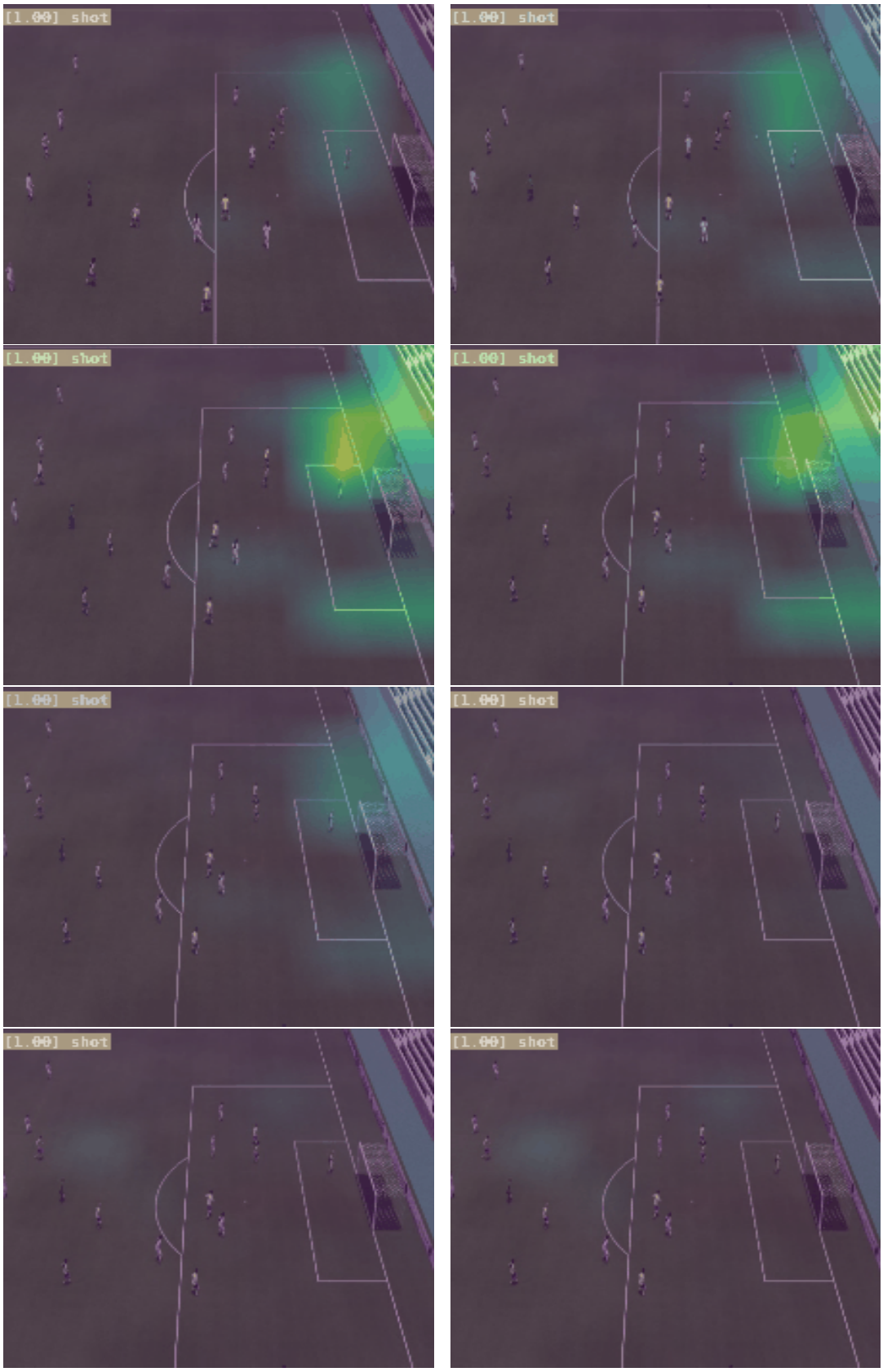


Figure 3.7: *Example of shot class.*

3.3 Highlight Detection.

The ultimate goal of video recognition is to be able to extract the events from an entire football match.

3.3.1 Experiments

To extract all events from an entire match we used a sliding window with stride 1 on frames. For each sliding window we predicted the event (pass or shot). Because this approach predicts the event for each frame we used a non-maximum suppression (NMS) technique to eliminate multiple detections of a single event. Also, we used a threshold to eliminate those predictions with low confidence score.

First of all, we defined some variables. W_{len} represents the length of a sliding window. As can be seen in Figure [2.5b] the average number of frames per video is different for each class. So for the W_{len} we chose the values $[10, 15, 20]$. W_{NMS_class} represents the length of the window used for NMS. For a fixed frame f_i we apply NMS in range $[f_i - W_{NMS\_class}, f_i + W_{NMS\_class},]$. TH_{class} represents the threshold used for each class to eliminate the predictions with low confidence score. $W_{assign\_to\_gts}$ represents the length of the window to assign a prediction to a ground truth event. In table [3.3] are present all values used for the above defined variables.

To evaluate the performance of the above approach using different values for the variables we computed the precision, recall and f1 scores. A prediction is counted as TP (true positive) if it is assigned to a ground truth given the $W_{assign\_to\_gts}$ variable, otherwise it will be counted as FP (false positive). A ground truth event is counted as FN (false negative) if it was not assigned to a prediction given the $W_{assign\_to\_gts}$. If there are multiple predictions candidates to a ground truth event, only the nearest one is assigned.

For this task we generated 60 matches that are different from the ones used in Chapter 2. We kept 30 matches for validation and 30 matches for testing the final performance.

To choose the best set of variables we performed a grid search on all values

from Table [3.3] on the validation matches and chose the ones that achieved the best weighted f1 score.

Variables	values
W_{len}	[10, 15, 20]
W_{NMS_class}	[10, 15, 20]
TH_{class}	[0.85, 0.90, 0.95]
$W_{assign\_to\_gts}$	[10, 15, 20]

Table 3.3: *Values used for grid search.*

After performing the grid search, we obtained the following values for the variables: $W_{len}* = 15$, $W_{assign\_to\_gts}* = 20$, $W_{NMS\_class}* = [10,20]$, $TH_{class}* = [0.85,0.85]$ which achieved a $f1weighted$ score of 0.8307 on the validation set.

3.3.2 Results and Discussion

The results obtained on the test set can be seen in Table [3.4],

Class	Avg Precision	Avg Recall	Avg F1
pass	0.8543	0.8005	0.8253
shot	0.8417	0.7536	0.7859

Table 3.4: *Results on the test set. The weighted f1 score is* 0.8217.

In Figure [3.8] we plotted the results from one match taken from the test set. On OX axis are the frames of the entire match and OY axis represent the probability of each event to occur. The points on the dotted line are the ground truths, there are 73 passes and 7 shots. On average, for the pass class, the difference between the predicted event and the ground truth is about 176 frames and for the shot class is about 15 frames. The shots detected automatically with this method can be then used to predict the expected goals. But there are some limitations: the difference between the ground truth and the predicted event in frames should be small so there is no offset because for expected goals we need the exact frame where the player performs the shot. We think that for detecting the exact frame another approach is needed. The second limitation is that the precision must be high to be sure that the predicted event is really a shot.

Also, we see that the model is able to identify most of the events that occurred during the match.

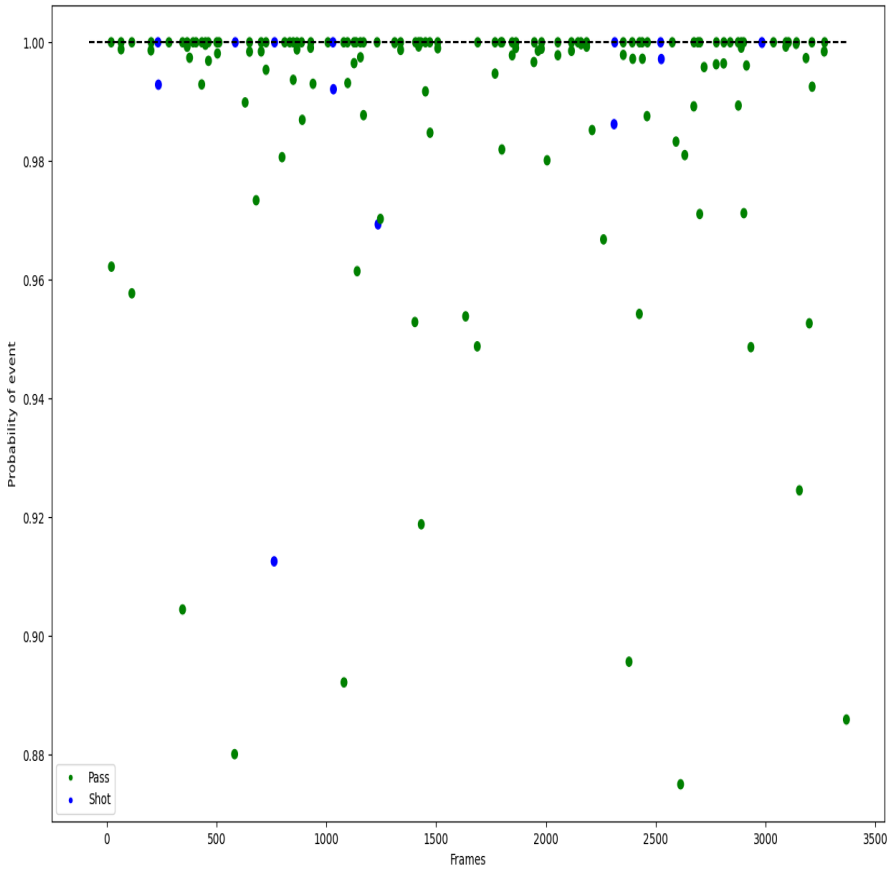


Figure 3.8: *Example of highlight detection for an entire match.*

Chapter 4

Expected Goals

Expected goals (xG) measures the quality of a shot to result in a goal (i.e. the probability of scoring a goal for a given shot). In many cases, the final score of a match is not a good indicator of how well a team played. For example a team can have only one shot and score a goal, while the opposing team can have ten shots, all of them being a miss chance, so expected goals provides more information about the match and the chances of the teams.

Current approaches for expected goals use features like the angle of the ball with the center of the goal line, the distance of the ball to the goal line, the type of situation (open play, set piece), etc. The current models are trained on over 100000 shots, each shot example having over 10 features like the ones described earlier.

There are no studies on expected goals only using images or short videos so in the following sections we will discuss my experiments and results.

4.1 Experiments

Because obtaining a large dataset annotated manually is very challenging we decided to investigate if the probability of scoring a goal for a given shot can be determined only using image or short videos on our synthetic dataset.

4.1.1 Dataset

Video. To predict the probability of a shot resulting in a goal we created the following dataset. Having the shot videos from Chapter 2, we computed the exact frame where the player touches the ball to perform the shot. Given this frame, we went

back 20 frames to create the video, so the video starts with 20 frames before the shot and ends exactly when the player touches the ball to shoot. The ground truths of these videos are 0 meaning that the shot did not result in a goal and 1 otherwise.

Image. For the image approach, we used the same method described for Video in creating the dataset, but instead of using a short video of 20 frames we only took the last frame, where the player touches the ball to perform the shot.

In total there are 8457 shots (Figure [4.1]) which are much smaller than the number of shots used for the current approaches for expected goals. Also, because the classes are imbalanced, the split for the train/validation/test sets was made in a stratified fashion, keeping the percentage of examples per class (Table [4.1]).

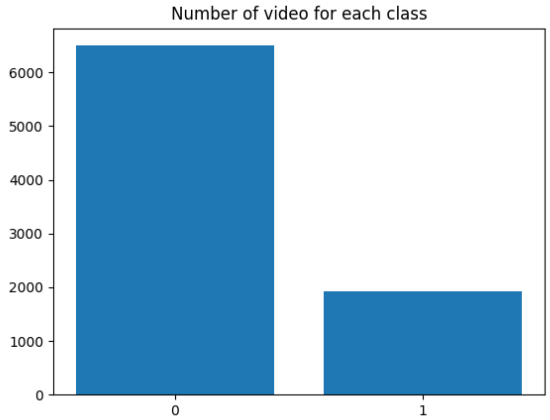


Figure 4.1: *Number of examples per class.*

Class	Train	Validation	Test
0	4563	978	979
1	1356	291	290

Table 4.1: *Split of the dataset in train/validation/test.*

4.1.2 Architecture

Video. We used the same architecture used in Section 3.1.2.

Image. We used a ResNet-50 network. The weights of this network were pre-trained on ImageNet.

4.1.3 Training

Video. We used the same setup as in Section 3.1.3, except for the learning rate. In this case we used $LEARNING\_RATE = 0.001$

Image. The inputs of the network are images of size 3 *x H x W*. The H and W represent the height and width of the image. The number 3 represents the number of channels, in this case represenging the RGB color model.

The images are resized to 224 *x* 224. Other hyperparameters that we used during training are: $BATCH\_SIZE = 32$, $LEARNIGN\_RATE = 0.0001$, $OPTIMIZER = sgd$ and $LOSSFUNCTION = cross - entropy$. For the loss function we used a weight for each class because they are imbalanced. Also as data augmentation, at training time, random horizontal flip is performed for an image with probability $p = 0.5$.

4.1.4 Evaluation

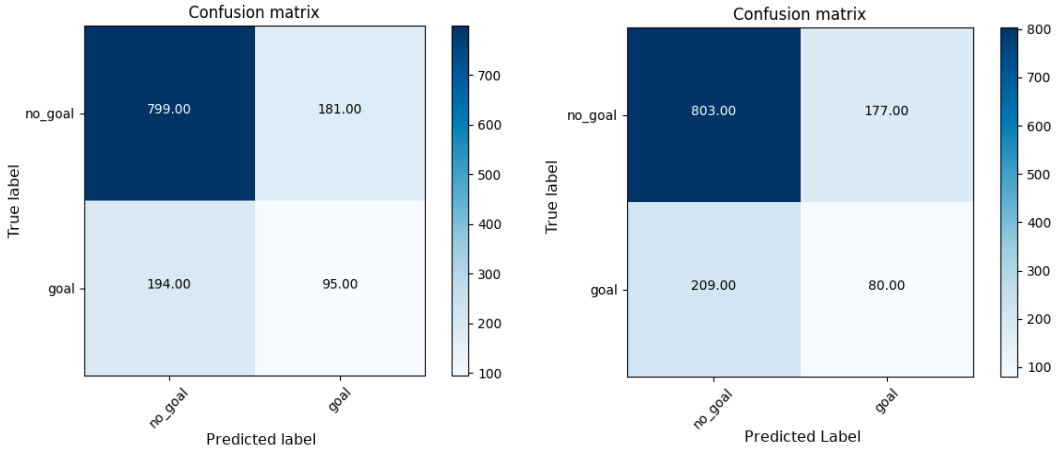
In order to evaluate the performance of the models for expected goals we analyzed two metrics. The first one is Area Under the Receiver Operating Characteristic Curve (AUC-ROC). The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) ($TPR = \frac{TP}{TP+FN}$, where TP - true positive and FN - false negative and $FPR = \frac{FP}{FP+TN}$, where FP - false positive and TN - true negative). Because the ROC curve uses FPR it is not useful to be used in the context where the dataset is imbalanced as in our case because the number of TN examples is very big. The second metric is the Area Under the Precision-Recall Curve (AUC-PR). This metric solves the previous problem because it plots the precision vs recall. For choosing the best model on the validation set and to compare the model trained on images vs the model trained on videos we used the AUC-PR metric.

Understat. Understat is a website that contains different information about the matches from several football leagues like EPL (England), La Liga (Spain), Bundesliga (Germany), Serie A (Italy) and Ligue 1 (France). It contains information

about all matches from 2014-2020. Moreover, those who run the site trained a model for expected goals on over 100000 shots, each shot containing more than 10 features. So in order to compare our models with the model from Understat we scraped the website and gahtered all shots from 2014-2020 from all leagues. For each shot we have its xG and whether or not it was a goal.

Model	AUC-PR	Random AUC-PR
Understat*	0.4853	0.1050
Video	0.3375	0.2277
Image	0.2858	0.2277

Table 4.2: *Results on the test set. For the understat model, the results are on the matches scraped from their website.*



(a) *Confusion matrix for the Video model.* **(b)** *Confusion matrix for the Image model.*

Figure 4.2: *Confusion matrices for both models.*

From Figure [4.2] we can see that both models perform similar. The model trained on videos has a better recall and a slightly lower precision. However, from Table [4.2] the model trained on videos achieves a better AUC-PR so in the following chapters we will further analyze this model in more detail. The model used in Understat achieves the best performance, but this model cannot be compared directly with our models because the datasets and the number of examples differ quite a lot.

4.2 Results and Discussion

4.2.1 Expected Goals Graph

Another method to test that the model performs well is to plot the expected goals against the actual number of goals. In Figure [4.3] taken from the Understat website we can see that the graph of xG follows the trend of the goals graph which is a good sign. We also plotted the xG graph versus the goals graph over 30 matches in Figure [4.4]. The trend of xG is quite similar to the goals graph, but for example on the last few matches there is a bigger difference.

In Figure [4.5] we plotted the number of actual goals versus the sum of expected goals for all shots. In Figure [4.5a] is the plot for the understat model. We used all the gathered shots from their website mentioned in Section 4.1.4. In Figure [4.5b] is the plot for our model evaluated on the 30 matches.

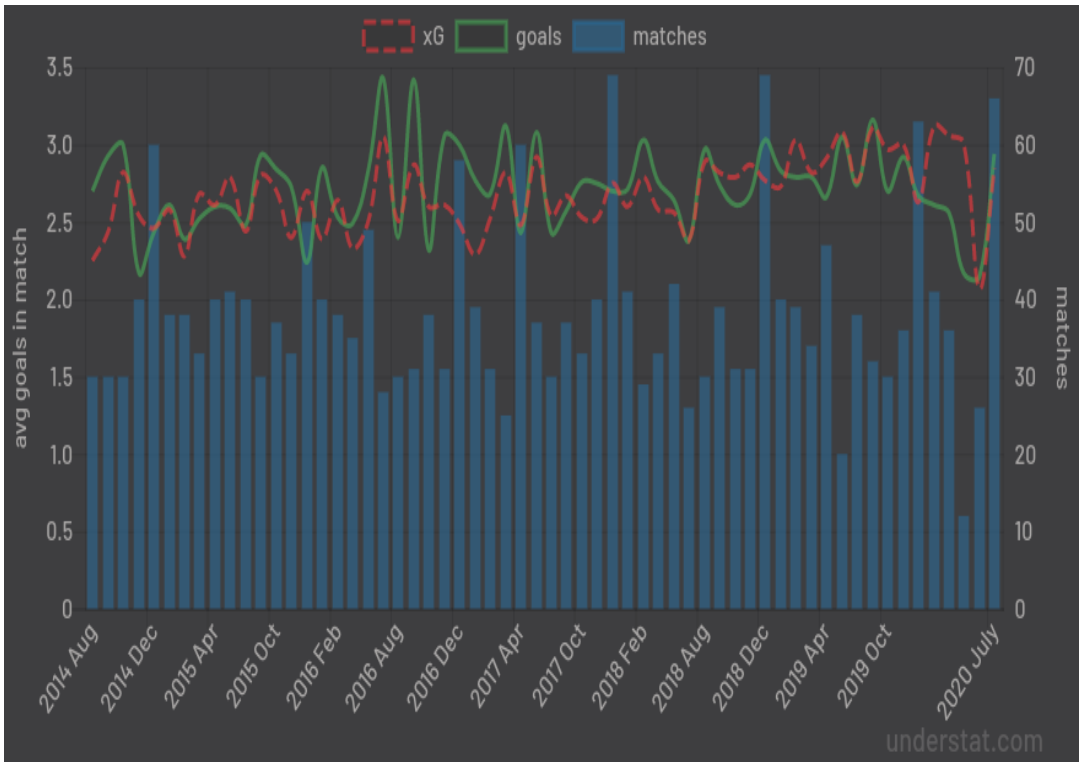


Figure 4.3: *Graph of expected goals vs actual goals taken from Understat.*

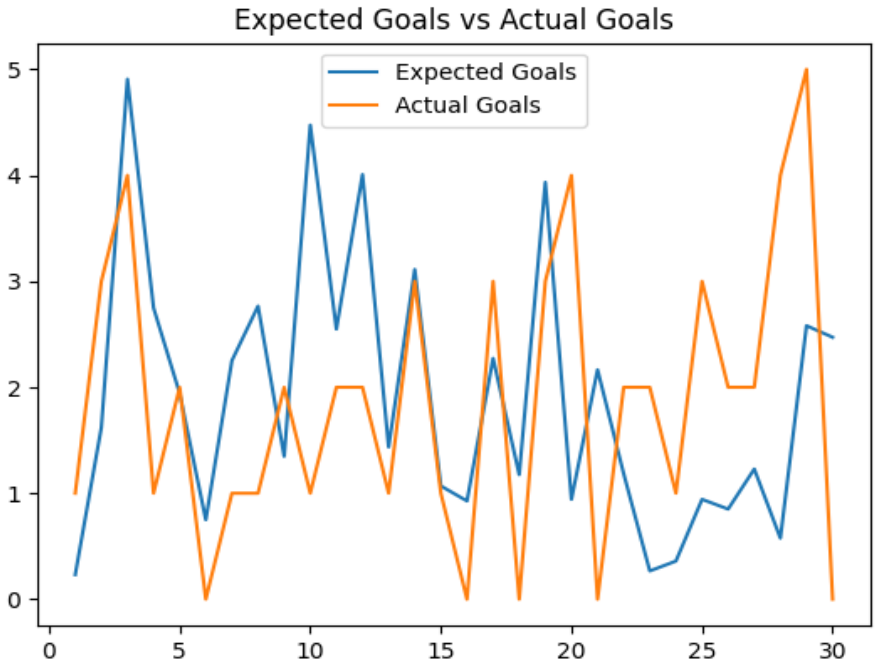
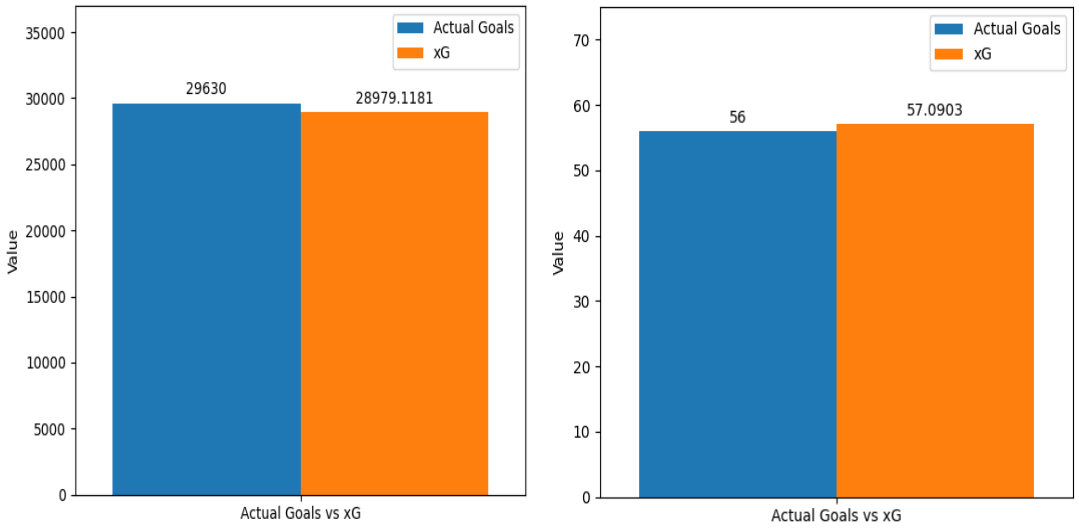


Figure 4.4: *Graph of expected goals vs actual goals. On OX there are the 30 matches used for testing.*



(a) *Understat model.* **(b)** *Our model.*

Figure 4.5: *Actual Goals vs Expected Goals.*

4.2.2 Eye Test and Grad-CAM Visualization

Also, to verify the performance of the model a good idea is to visualize the videos with their corresponding probabilities and see if they make sense. For this, we plotted some videos and we also used the Grad-CAM visualization.

Therefore, in Figures [4.6] and [4.7] the are two videos containing a penalty kick. In the first figure the probability of the goal is 0.75 which can be a good sign because in real world, the xG of a penalty kick is about 0.76. But looking at the second image we see that the probability is 0.27 which is small. So the probability for penalties varies quite a lot, a reason could be the small number of examples present in the dataset.

Another example in which the predictions of the network vary is present in Figures [4.8] and [4.9]. There are two similar chances, but for the first one the probability is 0.96 and for the second is 0.57.

We put the Figures [4.10] and [4.11] because they contain interesting videos. In the first figure, one player does a long pass and the other player remains 1v1 with the goalkeeper. The network focuses on the first frames, but not on the ball and then the focus is changed to the player which received the ball. The probability of scoring for this video is 1.0 which we think is a bit too optimistic. In the second video however, the network focuses on the players present in the box for the first frames, but on the last frames it fails to focus the player with the ball. In this case, the probability of scoring is 0.27.

Overall, from all videos we see that in some cases the network performs as expected, while on other cases the network fails. We think a solution to improve the performance is to increase the dataset because only 8457 shots that were used for training are not enough.

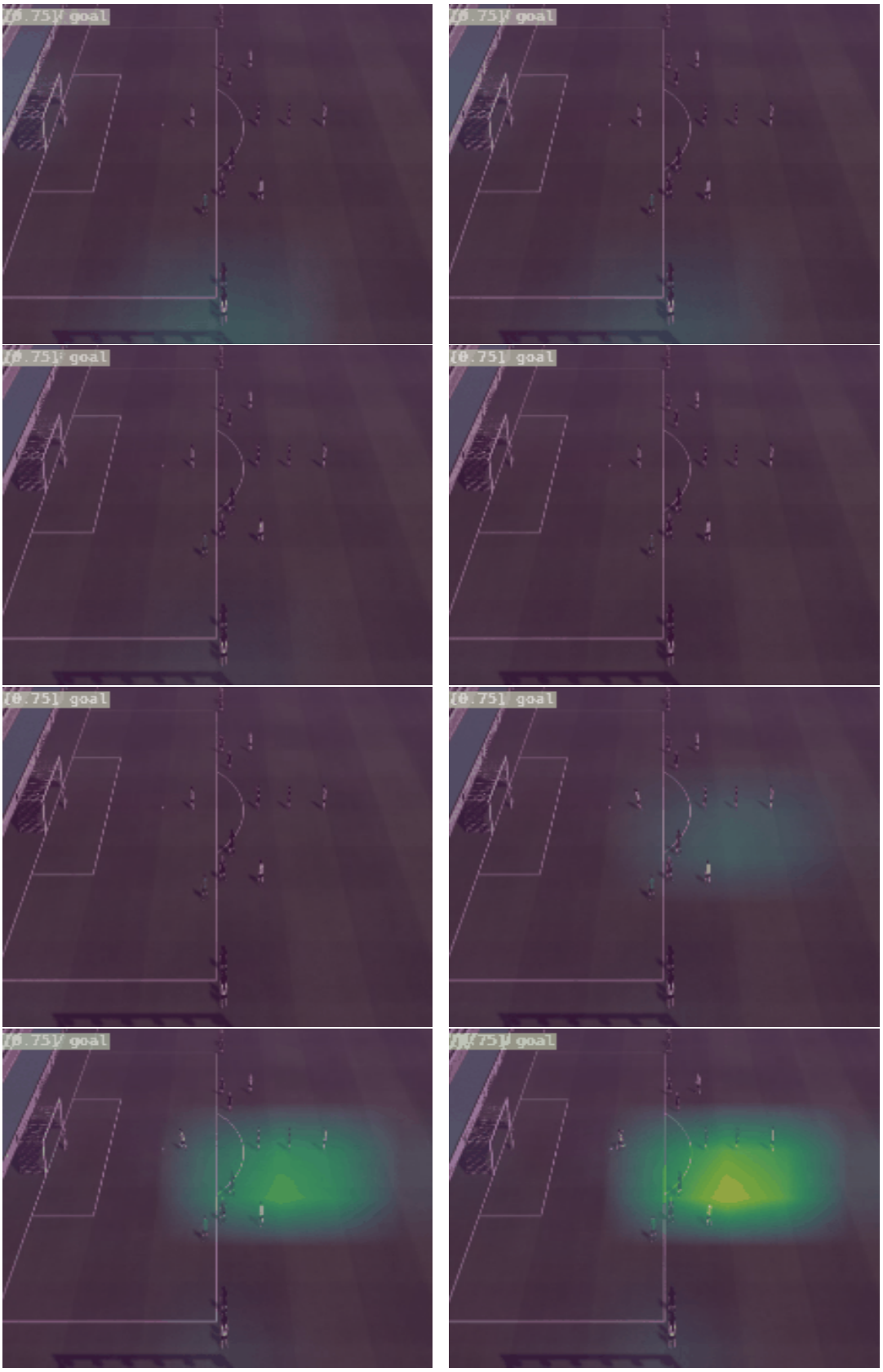


Figure 4.6: *Penalty kick..*

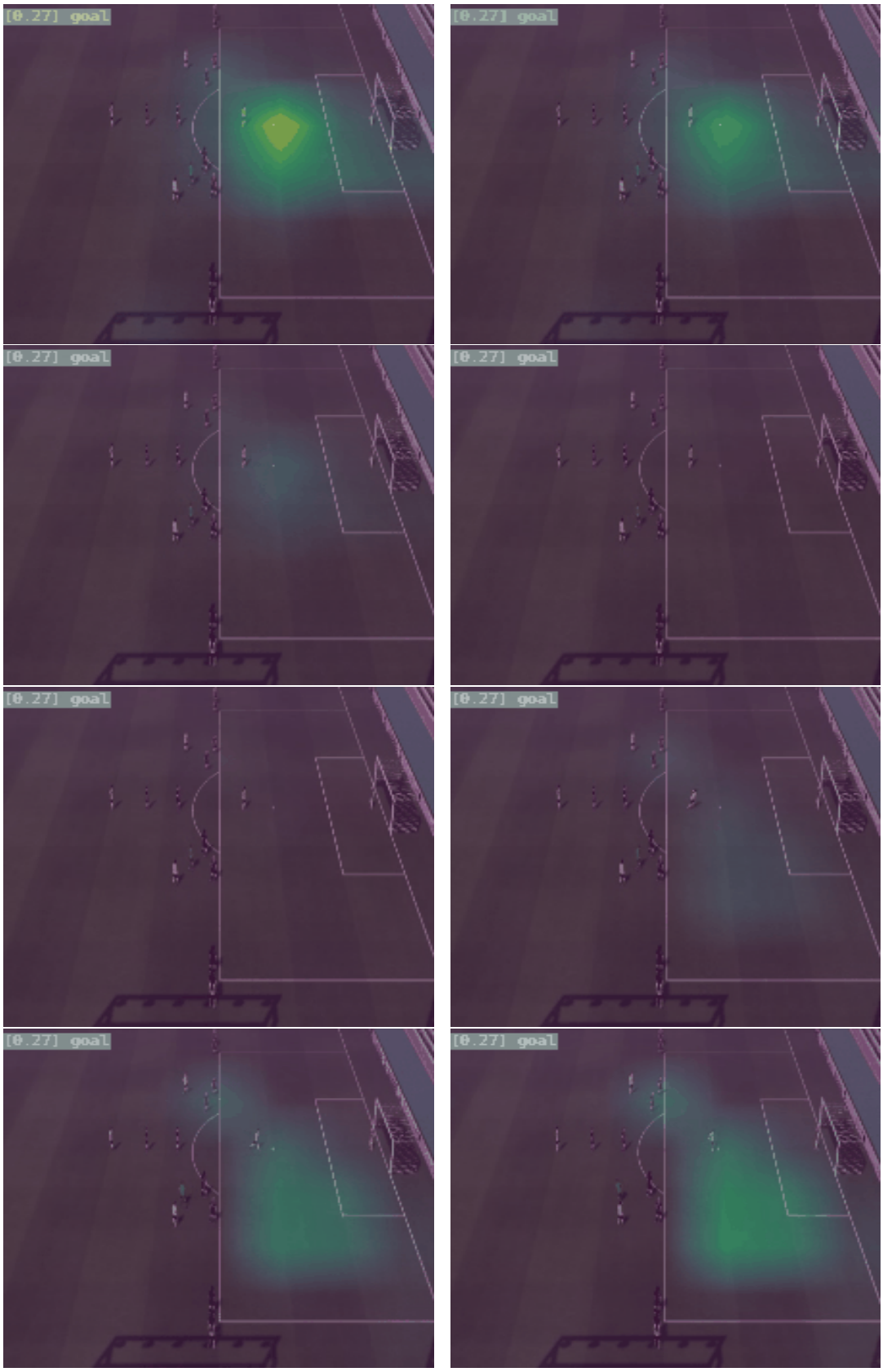


Figure 4.7: *Penalty kick.*

Figure 4.8: *Open play.*

Figure 4.9: *Open play.*

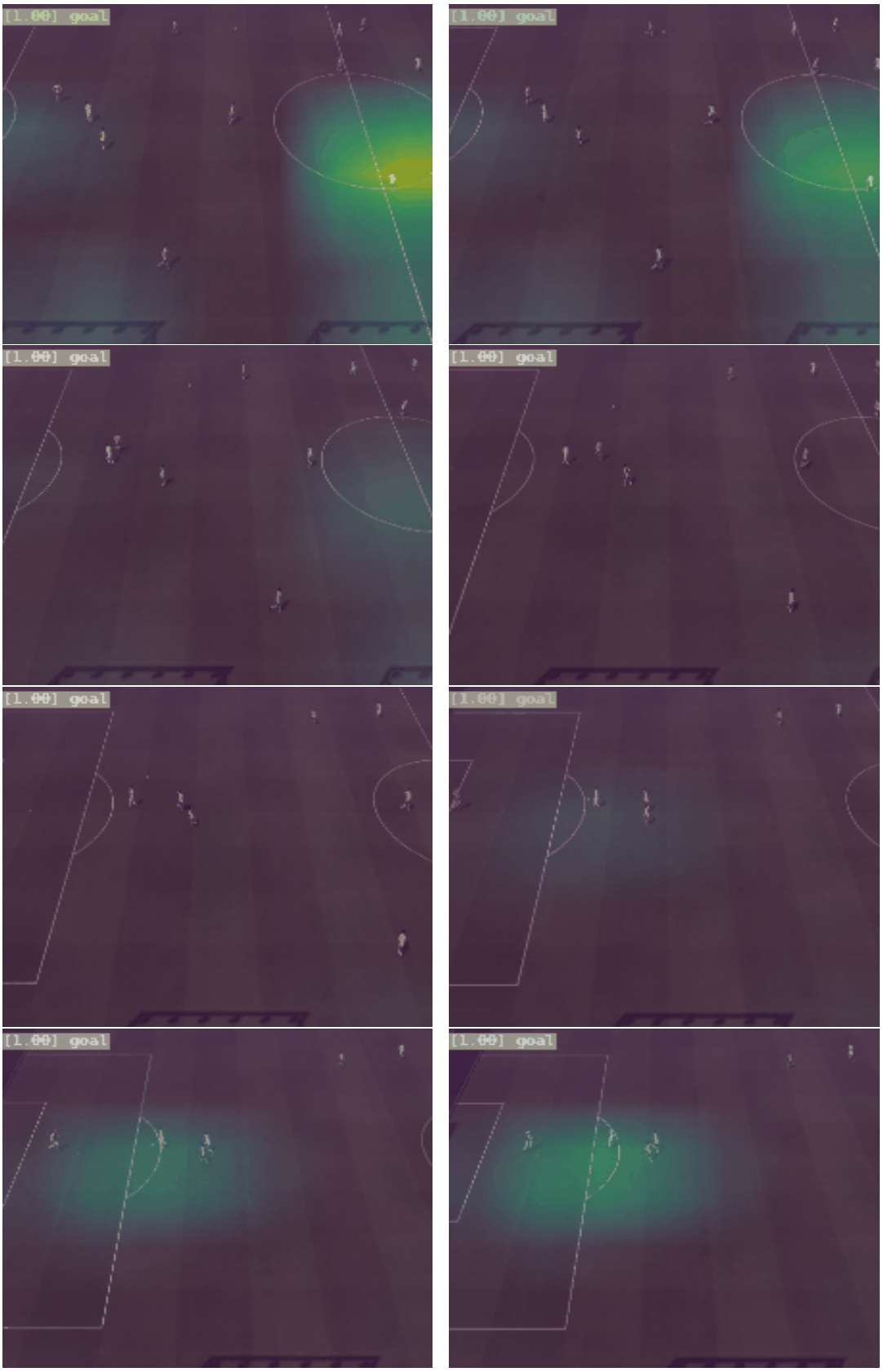


Figure 4.10: *Open play.*

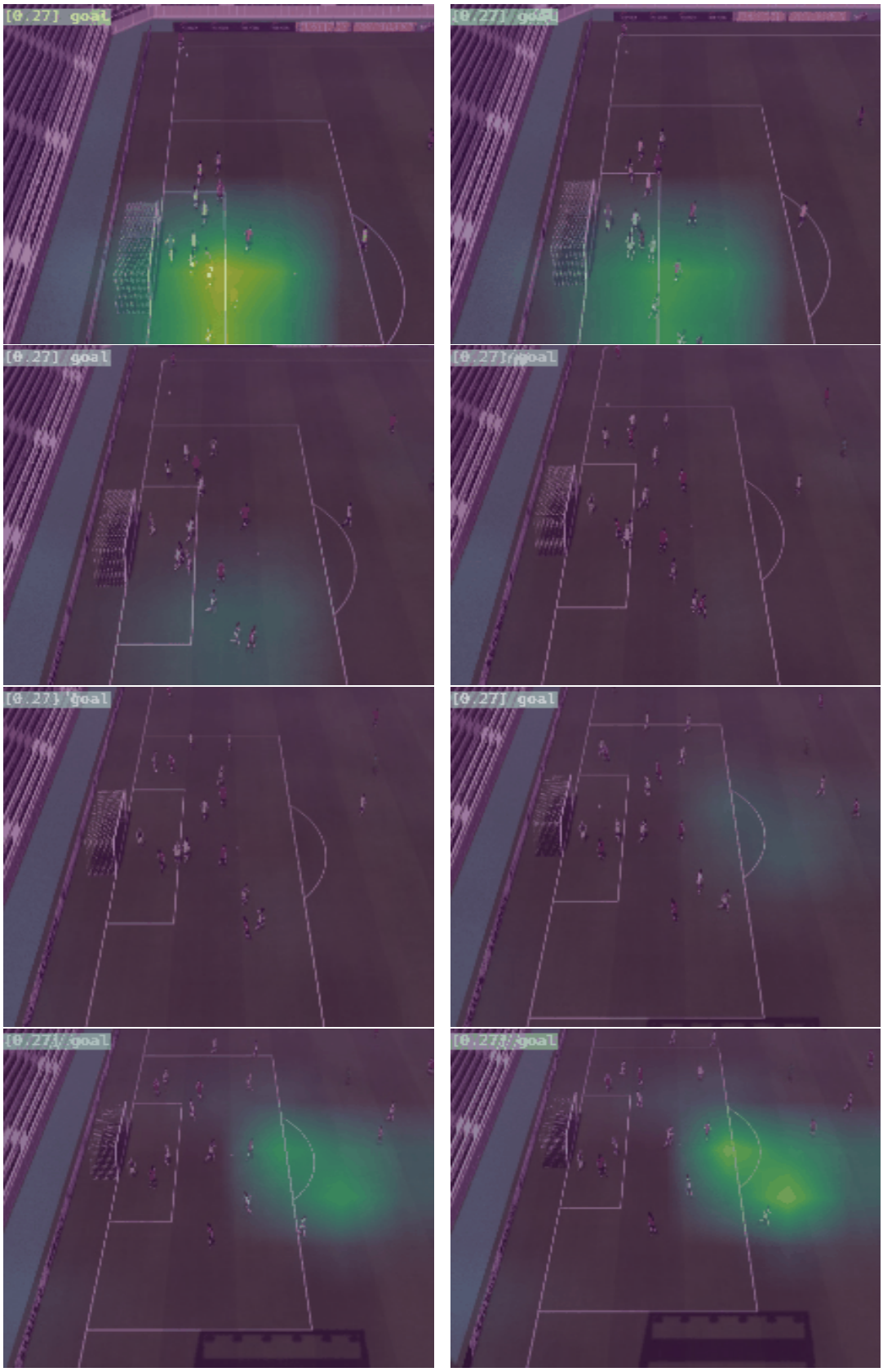


Figure 4.11: *Corner kick.*

Chapter 5

Conclusions

In this work we presented a way to generate a synthetic dataset for soccer. Even though the videos are not very photorealistic we think it still can be used for future research.

Also, we showed that video recognition methods performs well on sports videos, in this case soccer. We presented a method to summarize an entire football match having a model trained on short clips by using a sliding window approach followed by a non-maximum suppression technique and showed that this method achieves good results.

Finally, we investigated two approaches for predicting the goal probability from a shot using images or short videos. We showed what things do work and what things do not work. As far as we know, there are no studies that investigated this problem based on images/videos.

5.1 Future Work

In the future we have in plan more ideas:

- Generating more data. Especially for the expected goals task we plan to generate much more data and see if this will improve the performance because from the preliminary results it looks promising.

- We plan to try domain adaptation and see if the models trained on our synthetic dataset outperforms those who are trained only on real data. At the moment there are no datasets available so we are thinking of annotating about 100 real games to try this idea.

- Try different approaches like using a custom loss for soccer videos or find a way to improve the sliding window method.

References

[1] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *arXiv preprint arXiv:1512.03385v1* (2015).

[2] Ramprasaath R. Selvaraju et al. “Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization”. In: *arXiv preprint arXiv:1610.02391v4* (2016).

[3] João Carreira and Andrew Zisserman. “Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset”. In: *arXiv preprint arXiv:1705.07750v3* (2017).

[4] Will Kay et al. “The Kinetics Human Action Video Dataset”. In: *arXiv preprint arXiv:1705.06950v1* (2017).

[5] Bastiaan Konings Schuiling. *GameplayFootball*. `github.com/BazkieBumpercar/GameplayFootball`. 2017.

[6] Christoph Feichtenhofer et al. “SlowFast Networks for Video Recognition”. In: *arXiv preprint arXiv:1812.03982v3* (2018).

[7] Yudong Jiang et al. “Comprehensive Soccer Video Understanding: Towards Human-comparable Video Understanding System in Constrained Environment”. In: *arXiv preprint arXiv:1912.04465v2* (2019).

[8] Karol Kurach et al. “Google Research Football: A Novel Reinforcement Learning Environment”. In: *arXiv preprint arXiv:1907.11180v2* (2019).

[9] Haoqi Fan et al. *PySlowFast*. `https://github.com/facebookresearch/slowfast`. 2020.

[10] Lia Morra et al. “Slicing and dicing soccer: automatic detection of complex events from spatio-temporal data”. In: *arXiv preprint arXiv:2004.04147v2* (2020).

[11] Ryan Sanford et al. “Group Activity Detection from Trajectory and Video Data in Soccer”. In: *arXiv preprint arXiv:2004.10299v1* (2020).