Soccer Player Tracking

Mark-Robin Giolando Agam Singh Kalra Sameer Dedge
Oregon State University
{giolanm, kalraa, dedges}@oregonstate.edu

Abstract

Soccer is a team focused game, where victory relies on utilizing the correct team formation to counter the opposing team's team formation. In order to generate these teams, players on the field must be tracked, a task big clubs will hire individuals to manually do. Our research investigates the usage of PeleeNet to autonomously do this task on a lightweight enough scale to enable fans to deploy the system. A Faster R-CNN algorithm is also trained to provide baseline results. Our PeleeNet was able to achieve a player identification accuracy of 83.6% compared to the 84.3% of the Faster R-CNN, while being 1/9th the size (19MB vs 162MB). These findings demonstrate the potential for light weight networks to be deployed to serve the end-user customer.

1. Introduction

Soccer is a team focused sport reliant on team player formations. A team must adapt their formation to counter the formation of the opposing team. Coaches and big clubs will often have individuals dedicated to manually tracking this information, or will use algorithms to automatically do so. Smaller clubs or individual fans do not have the money or bandwidth to compete. Current work largely utilizes either tracking sensors for the human to wear, or larger networks that may suffer when deployed on consumer performance devices.

Our work aims to empower individual superfans to better understand the formations formed by each team. This team formation problem can be subdivided into four subcomponents: player tracking, ball tracking, pass detection and 2D localization and plotting. For this work, we focused on the problem of player tracking which can be further subdivided into image segmentation, frame-by-frame player tracking and occlusion handling. We have applied a PeleeNet algorithm to the problem of player tracking and compare the results to a Faster R-CNN (Region Based Convolutional Neural Networks) solution. This is a challenging problem due to the small size of the soccer players on a broadcast screen, as well as the many players who must be segmented and the high rate of occlusions that occur when players pass by one another.

Results are generated as a series of bounding boxes representing the players. The results are analyzed to determine if the players are detected (e.g., is there a bounding box near where one is expected), as well as the error for how close the generated bounding boxes are to the ground truth values. These determinations determine if our algorithms are able to detect the players, and with what accuracy they may be detected.

A PeleeNet solution was trained and tested using the SoccerDB dataset, resulting in a player identification rate of 83.6%. Furthermore, we tested a Faster R-CNN solution resulting in a player identification rate of 84.3%. The resultant model for the PeleeNet was 19MB whereas the model for the Faster R-CNN was 162MB.

Section 2 discusses the related work that has been done in these field. We discuss our methodology in Section 3 and our results in Section 4. Finally, our conclusions and future work recommendations are discussed in Section 5.

2. Related Work

Tracking players may be decomposed to three subcomponent tasks: players must be identified with image segmentation, players must be tracked from frame to frame with player tracking, and occlusions must be handled to identify individual players. Occlusions occur when players pass in front of one another, a common issue in soccer games.

2.1. Image Segmentation

A common way to identify the players is to use background subtraction [9]. Background subtraction consists of subtracting the foreground from the background in order to identify the items of interest. Background subtraction has been augmented in the past with manual correction [1]. A downside to background-foreground separation methods is higher levels of error (e.g., 11m [6]).

Hierarchical clustering algorithms may also be used to segment the players, using uniform color and estimated motion via the Lucas-Kanade algorithm [12]. Optical flow may augment the clustering technique by analyzing the large displacements of the players and estimate motion in each frame. A major downside of the particular approach used by Kagalagomb is the requirement for depth data, something that cannot be guaranteed.

Players may also be identified through edge detection algorithms, color detection and algorithms such as the Otsu's algorithm [2]. Otsu's algorithm uses image thresholding to return foreground and background classes for the image.

2.2. Occlusion

Thermal sensors outperform RBG cameras when detecting players. To address occlusion problems, algorithms such as a bagged tree classifier may be used [18]. The objective of this classifier is to separate blobs into the component players, through the shape and orientation of the players.

2.3. Player Tracking

Convolutional Neural Networks (CNNs) are a common tool for performing player tracking. Examples such as YOLO9000 [17] have been used to identify the locations of players. YOLO9000 uses a single CNN to generate bounding boxes for the various players, along with probability values associated with the various classes (e.g., player, ball or background). Another used CNN is the Visual Geometry Group from Oxford, which is a a very small network of 3 convolutional layers and 2 fully connected layers [13]. Other solutions might use CNNs such as Faster-RCNN with Feature Pyramid Networked trained on images of soccer players using a Resnet150 and Resnet18 network for studen teacher networks respectively [11]. Transfer learning and Deep Convolutional Generative Adversarial Networks can also provide data augmentation for these networks to improve their performance [17]. Online learning methods may also be used to improve player tracking from frame to frame by using spatiotemporal context metrics for the players current position as well as predictions for where they will be in the next frame [21].

Other methods for tracking players are to use wearable sensors such as GPS devices [16] [3] [14], to track players as they move around the field. For institutions with the financial capital, and access to players, this may be a usable system. However, issues with sensor accuracy degrade the results. Additionally, these sensors require the players to actually wear the sensors, and is unusable in scenarios wear the sensors are not worn. Data for player tracking may also be collected with multiple cameras may also be used to track players [9], or to cover the entire field [6]. For our purposes, these functions do not work as we are focusing on using broadcast data which does not cover the entire field.

3. Methodology

To identify players, bounding boxes will be used, rather than identifying the entire player. Identifying the entire player is more computationally expensive, and is unnecessary for merely locating the player in the video. Two algorithms use these bounding boxes: a PeleeNet Algorithm and a Faster R-CNN model.

3.1. Algorithm

For our methodology, we used two algorithms, PeleeNet and Faster R-CNN, to identify soccer players. The PeleeNet was tested with two experiments. The first experiment used transfer learning, where the PeleeNet was trained on a VOC2007 and VOC2012 dataset. The resultant network was then evaluated on the SoccerDB dataset. The second experiment with PeleeNet was to train on the SoccerDB dataset, before being test on elements from the same dataset. The Faster R-CNN was used as a comparison to the PeleeNet, to provide a baseline.

3.1.1 PeleeNet

Pelee [19] is an object detection model similar to an SSD(Single-Shot Detector). As in SSD, which extracts feature maps from a pre-trained VGG16, Pelee extracts feature maps from a pre-trained PeleeNet – a model trained on a classification task. Similar to SSD, Pelee predicts 4 bounding boxes for 4 priors for every point in the feature map [10]. Priors are predefined bounding boxes of aspect ratios and scales based on a priori knowledge. Pelee uses 5 feature maps – [19x19, 10x10, 10x10, 5x5, 1x1] from PeleeNet that go through a Residual block to predict the class label of a bounding box and the offsets of a selected prior in contrast to SSD which uses 6 feature maps with the biggest being 38x38 for efficiency purposes.

We implemented a Pelee network [20] based on the SoccerDB dataset [5] and its class labels. We implemented Pelee with 3 classes namely 'player', 'ball', 'goal' mapping to class labels 0, 1, 2 respectively. PeleeNet uses a Stem block which convolutes the input at different scales to increase the feature expressiveness but without the computational overhead caused by increasing the number of output features [4]. We wanted to but were unsuccessful in implementing a Stem Block on the layer outputting 38x38 feature map before going to 19x19 to try to increase feature expressiveness in layers with higher receptive field as we theorized that it might help in detecting smaller objects like a ball. Limitations including but not limited to not being able to detect the ball are mentioned below.

3.1.2 Faster R-CNN

We also used a Faster R-CNN to compare against the performance of the PeleeNet model. This serves as a baseline to evaluate the quality of the model. Faster R-CNNs evaluate regions of the image, using CNNs to generate region proposals for object detection, drastically reducing the time required. The Faster R-CNN we used contained 4 classes (background, player, ball and goal) mapped to 0, 1, 2, and 3 index values respectively.

To implement this network, we began with a pretrained Resnet50 based Faster R-CNN [15]. We then fine-tuned the network. A series of experiments were also run to attempt to replace the backbone of the Faster R-CNN with other networks such as MobileNetV2 to further improve the results. Ultimately, the backbone attempts failed to generate noteworthy results.

3.2. Dataset

For our dataset, we made use of the SoccerDB dataset [5]. This dataset contains 171,191 video segments from 346 soccer matches. 702,096 bounding boxes for players are also included in this dataset, representing ground truth data. From these video segments, we used 1774 of the annotated frames for our experiments with 70% of the frames forming the training set, 20% forming the validation set and 10% formed the test set. The SoccerDB dataset labels are formatted as: label, center x, center y, width, height. SoccerDB contains three classes, human, ball and goal represented by the values 0, 1 and 2 respectively. For the Faster R-CNN, modifications had to be made to the dataset. Faster R-CNNs however use class 0 to represent the background, requiring the class representations to be adjusted to 1, 2 and 3 for human, ball and goal respectively.

We also used the Pascal VOC 2007 [7] and 2012 datasets [8] for experiments with transfer learning. These two datasets identified 20 classes in an image, including humans. Our research attempted to use this data set to use the human identification portion of this dataset in order to identify the players on the field.

4. Results and Discussion

4.1. PeleeNet

In our first approach, peleenet was trained using VOC train dataset 2007 and 2012 for 100 epochs with a batch size of 64. Testing was done with 100 samples from the SoccerDB dataset. During the data processing a confidence of 0.5 was applied to filter the results. Any confidence below 0.5 was discarded. Peleenet trained on VOC dataset resulted in poor result with player detection less than 5% with an IOU of 0.054. Intersection Over Union (IOU) is an evaluation matrix calculated using area of intersection of ground truth bounding boxes and Predicted bounded boxes divided by area of union of ground truth bounding boxed and Predicted bounded boxes. An IOU above 0.5 is considered as a good object detection model.

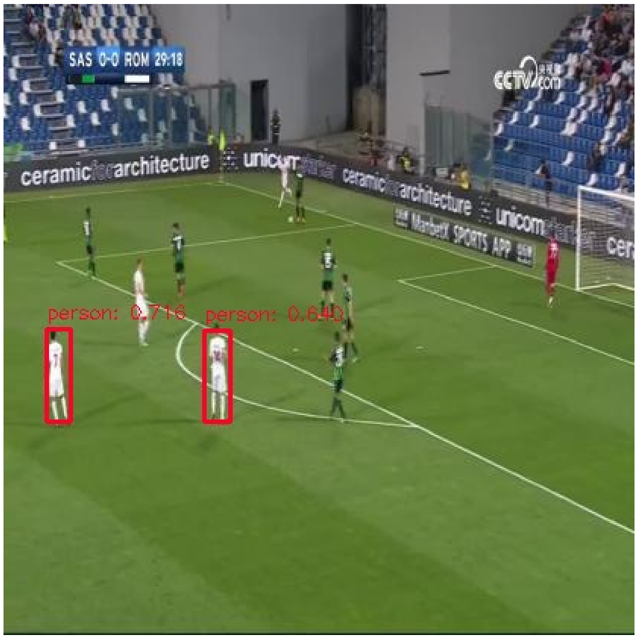

Figure 1. Performing Transfer Learning with the VOC dataset resulted in poor performance, with only a few players being detected.

The poor result is achieved because the VOC dataset contains facial closeup pictures of persons and the SoccerDB dataset contains a lot of players in one picture with distance taken in account. Figure 1 shows how poorly Peleenet detects players. In some of the outputs it detects 0 players in the frames. Hence, this approach failed.

In our second approach, Peleenet was trained using 3814 training samples, 1367 validation samples from the SoccerDB dataset, the Peleenet was trained for 110 epoch with a batch size of 16. Testing was done with 512 samples from the SoccerDB dataset. Again, confidence of 0.5 was applied to filter the results. Any confidence below 0.5 was discarded.

When trained on SoccerDB, we managed to achieve 0.671985 for the evaluation metric IoU when the prediction for a ball is not considered and 0.6252335 when the prediction for the case where the ball is considered. As the IOU is greater than 0.5 we can say the model is good. This model detected 83.56% of the players in all images, with an example show in Figure 2. The trained model is of size 19MB which is very small compared to the Faster-RCNN model which makes it a probable solution for mobile applications. The small size is due to the smaller feature maps used during Object detection and the cleverness of computing the Stem block.

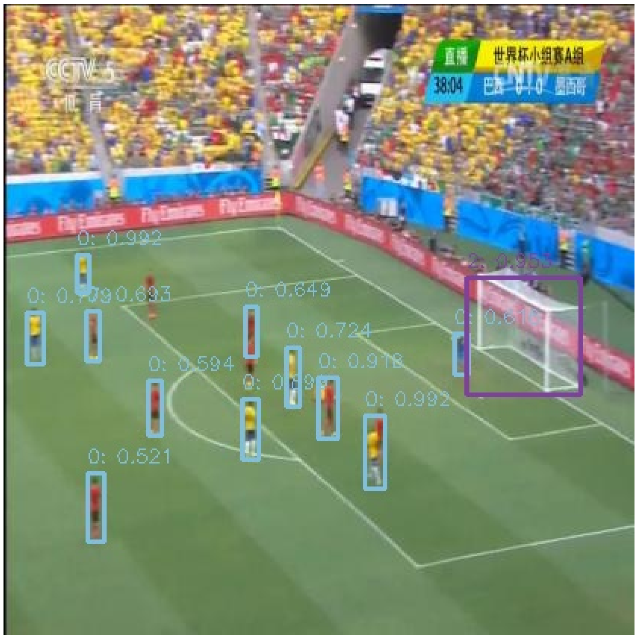

Figure 2. After training the PeeleNet on the SoccerDB, performance greatly increased and players could be detected.

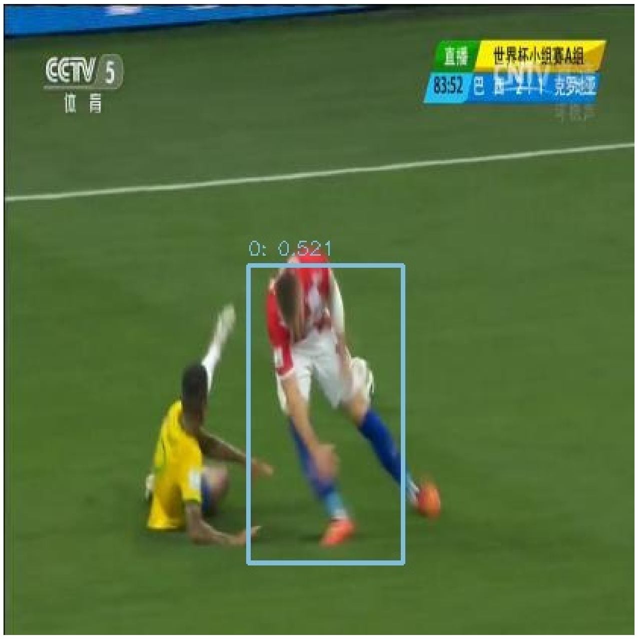

Figure 3. Pelee struggles to detect players that are in a horizontal position.

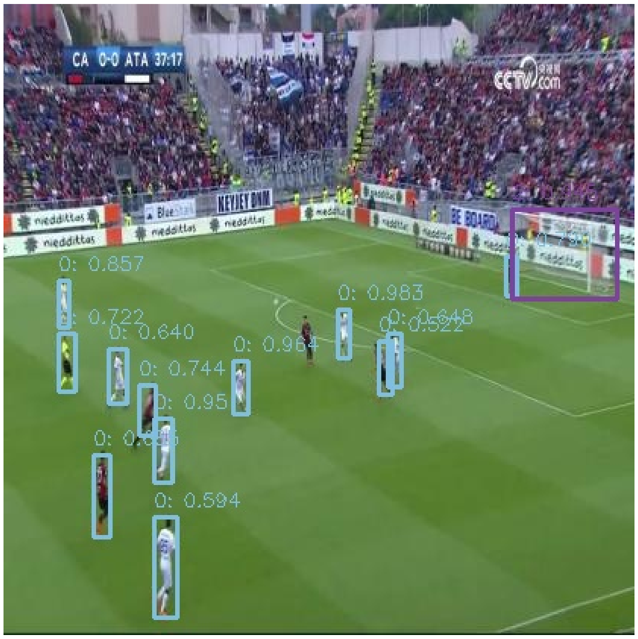

Figure 4. PeleeNet is able to detect occlusions caused during high camera angles.

The smaller feature maps lose information about small objects and thus the model was unable to detect the ball in the images. There are 2 different behaviors the model exhibits when it encounters occlusion in the image. When the players occlude each the the model has a difficulty in detecting them but it performs visibly better on occlusion caused due to camera angle than on images where occlusion is caused by proximity of players (as shown in Figure 4). Another peculiar behavior of the model is that it does not detect players that have fallen as seen from Figure 3. This behavior we believe is caused due to underrepresentation of fallen players in the images of the dataset as compared to players standing which made the model to not learn to select priors that are horizontal bounding boxes instead of vertical bounding boxes.

4.2. Faster R-CNN

Using 502 training samples, and 75 validation samples from the SoccerDB dataset, the Faster R-CNN network was trained over 10 epochs with a batch size of 1. Testing was done with 150 samples from the SoccerDB dataset. Experiments to expand the batch size ran into hardware issues where our computer ran out of CPU memory. The R-CNN produced bound boxes, labels and values representing the confidence of the bounding box. During the data processing a confidence threshold of 0.8 was applied to filter the results. Any confidence below 0.8 was discarded.

Figure 5 and Figure 6 show two sample frames with bounding boxes identified by the R-CNN trained on the SoccerDB training dataset. The “1” labels indicate players, and the “3” represents the goal. Figure 5 demonstrates the capabilities of the model to identify the players on the field. A noteworthy behavior is that the model is able to handle player occlusion. Additionally, the model is able to handle inter-class occlusions, as shown in Figure 6.

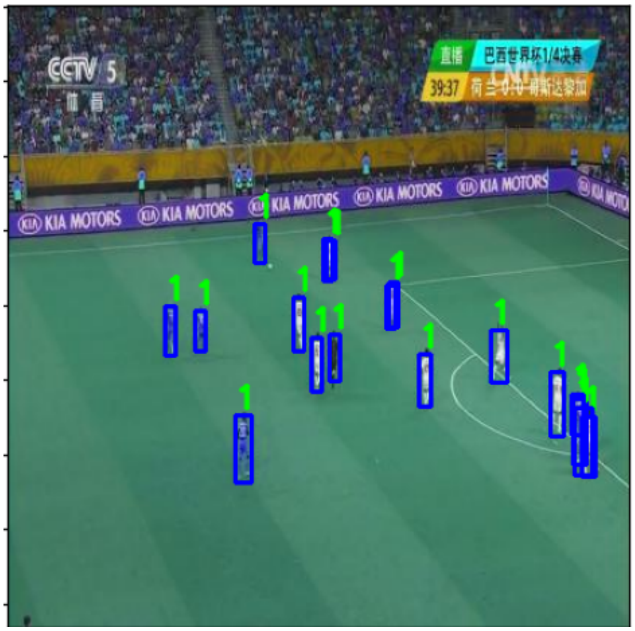

Figure 5. Faster R-CNN is able to detect the majority of players, as well as handle player occlusion.

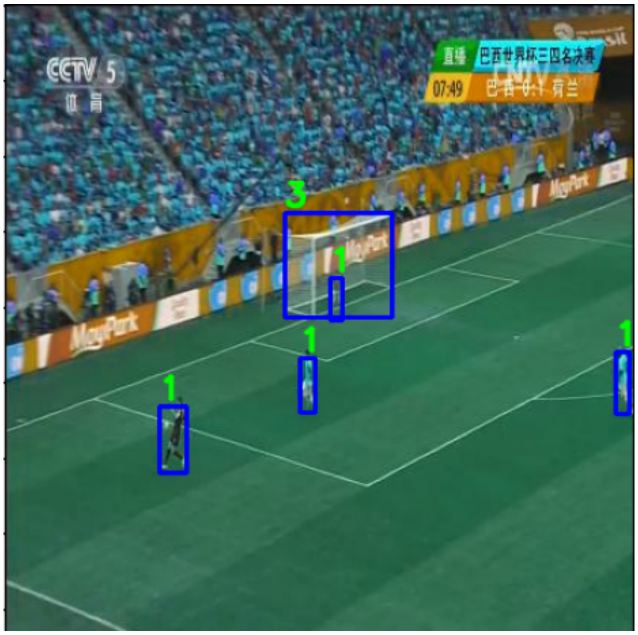

Figure 6. Faster R-CNN is able to detect objects even when other objects from separate classes are in front of them.

For a more detailed evaluation of the results, we examined the center point values of the bounding boxes of the players. We compared the center point of the test box labels to the center points of the player positions in the dataset. To match a model generated bounding box with a bounding box from the dataset, the center points were compared, allowing for a 1% error mismatch between the centers of the bounding boxes. If the distances between the two center points was greater than 1% of the image width/height, the segmentation was deemed to be in error.

Over each of the images in the testing dataset, we found a detection accuracy rate of 84.3%, where the trained model was able to detect 84.3% of the players. Additionally, the average error was 1.60 pixels. We found that the determination threshold may be set too low, as we found a False Positive rate of 4.2%. The total size of the trained model was 162 MB.

5. Conclusion

In this work, we were able to identify soccer players in video frames through the usage of PeleeNet and Faster R-CNN networks. The Faster R-CNN was able to outperform the PeleeNet, despite having less training data and running for less epochs. However, the PeleeNet was able to achieve comparable results (within 1 percent accuracy), and required only 11% of the space as the Faster R-CNN model. We also demonstrated issues that arise when attempting to use transfer learning for human subjects. Identifying smaller images of humans is difficult when the network is trained on larger images of humans.

A major limitation on our experiments was hardware. We were forced to use a batch size of 1 for training the Faster R-CNN network. A second major constraint was time. With more time to train the networks, rather than using a slice of the SoccerDB dataset, the entire dataset could be used in the train, validate, test process. This increased dataset size would greatly improve the accuracy of the PeleeNet and the Faster R-CNN.

The Faster R-CNN would also have been improved if we had been able to replace its backbone with a lighter weight solution. Implementing a MobilenetV2 backbone may have maintained the superior performance of the solution, while lowering the size of the model. Additionally, adding temporal features to the networks would improve the solutions to identify all of the players.

5.1. Future Work

This work has demonstrated the capabilities of lightweight networks to locate players in video frames and to handle occlusions between objects in the same class, as well as between objects of different classes. The next steps to take are to track players between frames and to perform location and mapping of the players so that the player locations within the soccer field can be tracked. Once these steps have been completed, a lightweight system will enable fans to track player formations in real time from their mobile devices.

References

[1] Ricardo ML Barros, Milton S Misuta, Rafael P Menezes, Pascual J Figueroa, Felipe A Moura, Sergio A Cunha, Ricardo Anido, and Neucimar J Leite. Analysis of the distances covered by first division brazilian soccer players obtained with an automatic tracking method. *Journal of sports science & medicine*, 6(2):233, 2007. 1

[2] Azam Bastanfard, Sajjad Jafari, and Dariush Amirkhani. Improving tracking soccer players in shaded playfield video. In *2019 5th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS)*, pages 1–8. IEEE, 2019. 2

[3] Alejandro Bastida-Castillo, Carlos D Gómez-Carmona, Ernesto De La Cruz Sánchez, and José Pino-Ortega. Comparing accuracy between global positioning systems and ultra-wideband-based position tracking systems used for tactical analyses in soccer. *European Journal of Sport Science*, 19(9):1157–1165, 2019. 2

[4] Bibek Chaudhary. Pelee: Real-time object detection system on mobile devices. Medium, Aug. 18, 2019 [Online]. 2

[5] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 2, 3

[6] Gabor Csanalosi, Gergely Dobreff, Alija Pasic, Marton Molnar, and László Toka. Low-cost optical tracking of soccer players. In *International Workshop on Machine Learning and Data Mining for Sports Analytics*, pages 28–39. Springer, 2020. 1, 2

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html. 3

[8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 3

[9] Pascual J Figueroa, Neucimar J Leite, and Ricardo ML Barros. Tracking soccer players aiming their kinematical motion analysis. *Computer Vision and Image Understanding*, 101(2):122–135, 2006. 1, 2

[10] Hao Gao. Understand single shot multibox detector (ssd) and implement it in pytorch. Medium, Jun. 6, 2018 [Online]. 2

[11] Samuel Hurault, Coloma Ballester, and Gloria Haro. Self-supervised small soccer player detection and tracking. In *Proceedings of the 3rd international workshop on multimedia content analysis in sports*, pages 9–18, 2020. 2

[12] Chetan G Kagalagomb and Sunanda Dixit. Tracking of soccer players using optical flow. In *International Conference on Innovative Computing and Communications*, pages 117–129. Springer, 2021. 2

[13] Paresh R Kamble, Avinash G Keskar, and Kishor M Bhurchandi. A deep learning ball tracking system in soccer videos. *Opto-Electronics Review*, 27(1):58–69, 2019. 2

[14] Hyunsung Kim, Jihun Kim, Dongwook Chung, Jonghyun Lee, Jinsung Yoon, and Sang-Ki Ko. 6mapnet: Representing soccer players from tracking data by a triplet network. In *International Workshop on Machine Learning and Data Mining for Sports Analytics*, pages 3–14. Springer, 2022. 2

[15] Sovit Ranjan Rath. Faster rcnn object detection with pytorch. Debugger Cafe, Sept. 7, 2020 [Online]. 3

[16] Lars Reinhardt, René Schwesig, Andreas Lauenroth, Stephan Schulze, and Eduard Kurz. Enhanced sprint performance analysis in soccer: New insights from a gps-based tracking system. *PloS one*, 14(5):e0217782, 2019. 2

[17] Rajkumar Theagarajan, Federico Pala, Xiu Zhang, and Bir Bhanu. Soccer: Who has the ball? generating visual analytics and player statistics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1749–1757, 2018. 2

[18] Noor Ul Huda, Kasper H Jensen, Rikke Gade, and Thomas B Moeslund. Estimating the number of soccer players using simulation-based occlusion handling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1824–1833, 2018. 2

[19] Robert J Wang, Xiang Li, and Charles X Ling. Pelee: A real-time object detection system on mobile devices. *Advances in neural information processing systems*, 31, 2018. 2

[20] yxlijun. Pelee.pytorch, 2019. 2

[21] Pei Zhang, Linghan Zheng, Yan Jiang, Lijuan Mao, Zhen Li, and Bin Sheng. Tracking soccer players using spatio-temporal context learning under multiple views. *Multimedia Tools and Applications*, 77(15):18935–18955, 2018. 2