Tracking and Classifying NBA Players in Basketball Video Clips

Ryan Won
Computer Engineering
Drexel University
rhw37@drexel.edu

***Abstract*—While the tracking, identification, and detection of sports players on their respective playing surface is a common endeavor, doing so using only the single camera angle provided in most sports clips and broadcasts – the pan tilt camera angle – is a more challenging and less common task. This project aims to process YouTube clips and game broadcasts recorded in this angle and identify and detect players on the court by implementing the cited research paper. This project also aims to implement the classification of which team a detected player belongs to. Additionally, it will keep track of how many players are on the court for each team according to the detection.**

I. Background

A. Problem Interest

With the rise of media capabilities and interest in the NBA, full game highlights from almost any game played in the last few seasons can be found on YouTube or similar video services. With an increased interest from NBA teams to apply analytics to improve the performance of their teams, these videos serve as a potential data source for sports analytics. However, as these videos are taken from the standard recording and display of NBA games, the pan tilt camera angle, almost all of the data requires processing before becoming remotely useful. The ability to successfully and accurately detect players in this single angle turns innocuous highlight clips into a wealth of data for sports analytics.

II. Related Work

The work for this project was inspired by the paper listed in citation [1], which researches ways to efficiently and accurately track and classify NBA players in this single camera angle. Like the paper mentioned, the goal of this project is to be able to successfully track players on the court, then be able to classify the team that they belong to. However, I slightly diverge from the paper in terms of the methods used to achieve this goal. In the given paper, a deformable part model is used to locate the players in frames, and then a logistic regression classifier is used to train the model to recognize which players belong to each team. I chose to experiment with the Mask Region-based Convolutional Neural Network to identify the players, and also used it to then classify which players belonged to each team.

III. Methodology

A. Data Selection

For this project, I decided to use clips from Kobe Bryant's last game, both for sentimental reasons and also for strategic reasons. To minimize the amount of noise from spectators, referees, and bench players, I looked for a clip that had the players wear different colors from all the other entities. Luckily, in Kobe's last game, the Lakers in the court donned yellow while the majority on the bench wore purple with some yellow. The Jazz wore dark blue uniforms with yellow and white trim, which I felt was distinct enough from the bench attire of black t shirts. One minor issue is the refs, as they wear gray shirts and dark pants, which may be similar to the uniform kit of the Jazz. However, when taking in the fact that I idolized Kobe Bryant when I was growing up as I went to high school in the school district next to his high school, the issue with the refs was moot and I decided that this was as good of a color setting that I was going to get. For use as training data, I took footage of highlights from the second half, and for the test data I took still images from Kobe's last two offensive possessions of the game.

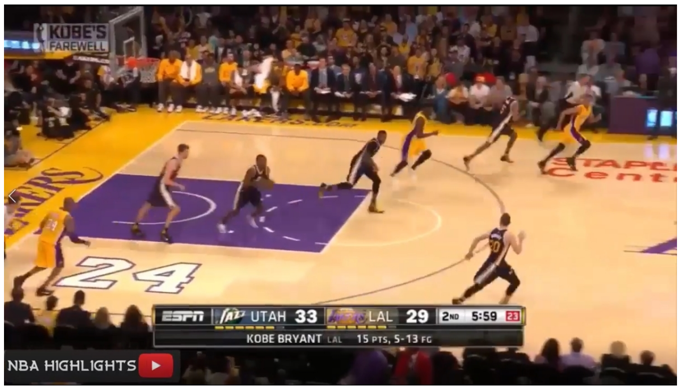

Fig. 1. A frame from a highlight of the Utah Jazz on an offensive transition.

B. Data Pre-Processing

To be more specific, the training data was a 30 second video of highlights from the second quarter, which covers a possession from each team and also footage of the Jazz in transition on offense, shown in the still image of Fig. 1. First, I needed to split this video into frames. I decided on a framerate of 3 FPS for the training, which meant that I would take 3 still images from the video every second. While I contemplated a higher framerate, I decided that this would result in training data where the frames next to each other have barely any movement, resulting in a lot of the same data being given to the training model. Once I split the videos into frames, where one image was equivalent to one frame, I ended up with 91 images for the training set. To use these with the Mask R-CNN model, however, I had to first annotate them.

The Mask R-CNN is a supervised learning model, required labeled training data. This is why we cannot just put in these images into the model. To label the training data, I used a tool called LabelImg, which allows one to place a bounding box around a figure or element in an image, and apply a label to it. An example frame being annotated in LabelImg is shown in Fig. 2. The annotation information is stored in an xml file that corresponds to the original image file. Once all these images are annotated with class labels, they can be used to train the model.

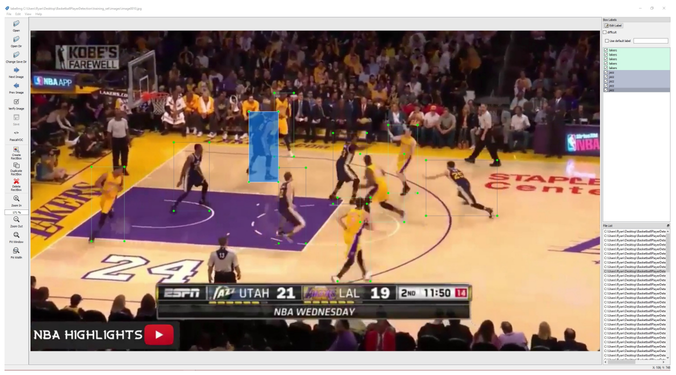

Fig. 2. Example of training data classification in LabelImg.

C. Model Training

To be able to take the raw frames and xml files and build moldable testing data, I wrote a Dataset class with aid from Machine Learning Mastery [2]. This class creates a dataset based off of the raw images and coordinates from the corresponding xml files, and assigns the labeled coordinates to one of two classes, Lakers or Jazz. Using the architecture and weights for the pre-fit Mask R-CNN model, Once the training data was loaded, I decided to created 5 heads for the neural network, where one model file is output for each.

Based on the loss values, I saw that the fifth output model was the best one to work with, so that model was set as the model for testing. An interesting thing to note is that the total loss, classification loss, and bounding box loss all decrease with continuing epochs. This can indicate that more training epochs could lead to a more accurate model. However, I was unable to do more in the interest of time. This was because the first training models I created were from tensorflow CPU, which took an hour to train a single model compared to the five minutes it took the GPU version of tensorflow. The statistics for the training model are shown in Table 1.

Epoch.	**Training (s)**	**Total Loss**	**Classification Loss**	**Bounding Box Loss**
1	294 s	1.5546	0.4717	0.3663
2	285 s	1.0954	0.3357	0.2290
3	283 s	0.9743	0.2877	0.1946
4	283 s	0.9060	0.2518	0.1772
5	280 s	0.8286	0.2114	0.1577

TABLE I
DATA FROM MASK R-CNN TRAINING SET.

D. Model Testing

Using the fifth model from the training neural network, I then applied that model to the testing set. The testing set is comprised of a 17 second video that contains two clips of Kobe's last possessions. Similar to the training set, this video is broken down into frames and saved as images. For this test set, however, I chose to use 30 FPS so that when reconstructed, the video would still be fairly smooth. This resulted in 579 frames of test data. Each frame was run through the test model, which produced its predicted bounding boxes and predicted class labels for each bounding box. A comparison of an actual frame from the testing set to the classification and bounding box prediction made by the model is shown in Fig. 3. The bounding boxes outlined in red represent predicted Lakers players and the green outlined bounding boxes are predicted Jazz players.

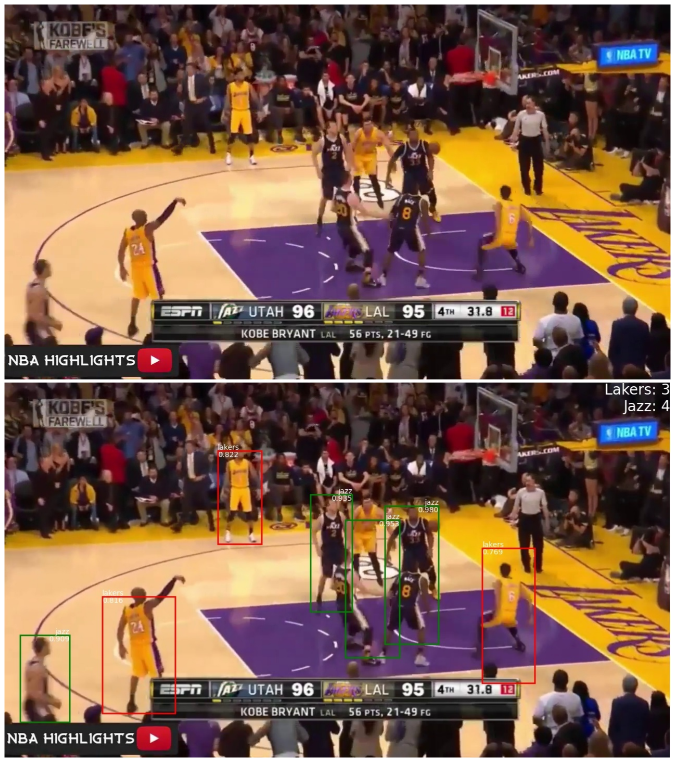

Fig. 3. A prediction made by the CNN model on a single frame.

E. Team Classification

The model selects which team it believes it is based on a confidence score metric, which goes from zero to one. Any object or part of the image is scanned and rated for the confidence that it is under the classification of Lakers or Jazz. If an object crosses the threshold for a classification, it is assigned that classification and a bounding box is assigned to the object. Looking closer at the output prediction model shown in Fig. 4., we can see that the bounding box as the name of the class at the top of the bounding box and the confidence score underneath the top line of the bounding box.

Additionally, the current amount of players on each team that the model predicts is shown in the top right corner. This is a good indicator of how well the model is performing, as we know there should be a maximum of 5 players from one team represented, with a possibly of less if players are not in the camera angle.

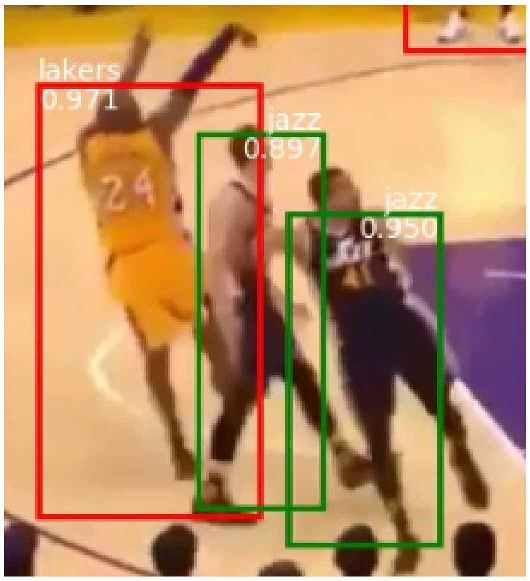

Fig. 4. Classification and confidence score predictions.

IV. Method Performance

Unfortunately, my work is not as accurate as I would hope. Looking through the output video, output.avi under the videos directory in the code, one can see that there are many instances of incorrect classifications. Oftentimes, the referees are classified as Jazz players, and Lakers players are often classified as Jazz players, leading the output video to show that there are more than 5 Jazz players on the court for many sections of the video. Additionally, the method struggles with displaying players that are very close to each other or behind another person, often combining two players into one classification or ignoring the hidden player. There are also instances when a single player is sometimes given both classifications. Overall, the mean absolute precision score, or mAP, was at a pretty low score of 40.7%.

V. Future Work

In order to increase performance, I think that having more training models could lead to better performance when it comes to the test models. Additionally, adding a way to have player tracking between frames could increase accuracy, as many times players would go back and forth between the team classifications in the video. Finally, I think increasing the amount of training data as well could be beneficial, as we only really used two possessions worth of data in our training data, which is a miniscule amount of data compared to how much there is in a full basketball game.

VI. Acknowledgements

I would like to thank Dr. Louis Kratz for his tremendous efforts in teaching everything related to Computer Vision, especially amidst the current online climate we find ourselves in due to the coronavirus. Learning should be much more difficult in such a situation, but Dr. Kratz was able to make the subject matter incredibly interesting and the material easy to comprehend.

I would also like to dedicate this project to the memory of Kobe Bryant. I will never forget hearing the shocking news that Kobe had passed away, only moments before I played in a basketball tournament myself. Kobe was a big inspiration to me, and is one of the reasons I am constantly trying to meld my love of sports with my interests in computer science and computer engineering.

VII. Code and Documentation

GitHub Link to source code.

References

[1] W. Lu, J. Ting, J. J. Little and K. P. Murphy, "Learning to Track and Identify Players from Broadcast Sports Videos," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 7, pp. 1704-1716, July 2013, doi: 10.1109/TPAMI.2012.242.

[2] J. Brownlee, "How to Train an Object Detection Model with Keras," Machine Learning Mastery, 03-Oct-2019. [Online]. Available: https://machinelearningmastery.com/how-to-train-an-object-detection-model-with-keras/.