Ball Trajectory Detection for Pickleball

Naman Bharara, Andrew Dettor, Julie Fleischman, Huilin Piao

Group 11, DSC 383

Goergen Institute for Data Science

University of Rochester

I. Introduction

Pickleball is a sport that combines elements of badminton, table tennis, and tennis and has been the fastest growing sport in the United States [1]. Players can play as a team of two or play against a single opponent in tennis style games. This popularity has seen a major spike in the United States with a 85.7% increase in players from 2021 to 2022 [2]. This growth has translated to a playerbase of over 8.9 million players in the United States for 2022 [2]. Thanks to this rapid growth the Pickleball industry is expected to be worth roughly 1.19 billion dollars globally in 2022 and is expected to be worth 2.14 billion dollars by 2027 [3]. While the sport has been known for its reputation as a relaxing pastime, there has been increased interest in the sport's competitive scene with popular athletes such as Tom Brady, LeBron James, and Kevin Durant purchasing ownership stake in professional teams [4]. Due to this, there is a demand for an introduction of a pickleball analytics platform into the sport as is present in nearly every other major sport. The sports analytics industry is worth an estimated 2.45 Billion dollars in 2022, and is expected to have revenues of 22 billion dollars by 2030 [5].

Due to these two rapidly growing markets our project sponsor Fritz Ebner, intends to develop a pickleball analytics model that is able to keep track of player statistics such as shot type, shot velocity, location, and score. A key issue with the current platform is inconsistent ball detection. Problems include failing to track the ball as it approaches the net, picking up balls that are from adjacent courts, or improperly predicting the ball trajectory. In an attempt to remedy these issues, we set to modify an existing tracking model and utilize transfer learning techniques to develop a better ball tracking solution. An existing object tracking system known as Tracknet 2.0 was developed in Taiwan, by a team of researchers working on making a shuttle tracking program for badminton. This model was trained on over 55,000 frames of professional badminton matches [6]. Due to the similar dimensions of court size, and the nature of object tracking this was the ideal model to adapt for pickleball tracking.

II. Literature Review

A. Computer Vision

The paper "Ball tracking and 3D trajectory approximation with applications to tactics analysis from single-camera volleyball sequences" focuses on the development of a ball tracking and 3D trajectory approximation for volleyball sequences using a single camera. In this paper, the authors develop a method that can identify the volleyball, using filtering techniques based on size and shape. A key component of the model we used is the inclusion of deep convolutional neural networks [7]. The advantage of their use is seen in the paper "Going Deeper with Convolutions" which focuses on using convolutional layers to reduce the dimensionality of feature maps to improve model performance [8]. The method of using a CNN for dimensionality reduction has been used in other ball tracking systems such as the real time fault detection developed in "A Study of Automatic and Real-Time Table Tennis Fault Serve Detection System" in which researchers developed a system to determine the validity of serves in table tennis by

using a motion detection algorithm. In this work, the researchers propose the usage of the convolutional neural network in order to determine the position of the ball [9]. The latest work in this field culminated in the creation of TrackNet, a deep learning model which is specifically developed for tracking small, high speed objects in sports like tennis, and badminton. In this work, researchers developed a model that uses an Encoder-Decoder model in order to keep track of the ball via a heatmap [10]. The most recent iteration of this model, TrackNetV2, has been trained on over 55,000 frames of professional badminton games from a variety of courts and camera angles [6].

B. Transfer Learning

A key concept of this project is the use of transfer learning, which is the process of taking a pretrained model, and applying it for a new but similar problem. Transfer learning has been found to be a useful method that allows users to leverage the strength of a model that has already been trained to be used in a new problem without the need of developing and training a new model [11]. Previous attempts to use transfer learning on CNN for visual recognition have been found to be successful [11][12]. In "CNN Features off-the-shelf: an Astounding Baseline for Recognition" the authors used a pretrained CNN feature as a baseline for image recognition and found that the model had achieved high scores for accuracy without fine tuning, demonstrating the validity of using transfer learning for image recognition [11]. Meanwhile, the authors of "A Study on CNN Transfer Learning for Image Classification" found that by implementing additional layers, and adjusting the weights of a CNN model, can significantly improve the accuracy of a pre-trained model. The authors also found that the use of transfer learning is ideal in situations where a pre-trained model that has been trained on a large dataset is used as a starting point for training on a smaller dataset [12].

III. Data

A. Dataset

The dataset developed for this analysis was based on a fifty-four-minutes recording of a Pro Senior Mixed Doubles Gold Medal Match in September, 2021. The dimension of the video is 1920 x 1080, and the video was recorded at a 60 frames per second rate.

B. Preprocessing

The preprocessing includes three steps: Video Editing, Labeling with programming, and file converting.

i. Video Editing

To avoid overfitting and reduce dataset size, we screen out unrelated frames and segments for game related clips by trimming videos to moments starting from ball serving to score. A seven-minute video with 12810 frames in total is trimmed with recording all the complete plays.

ii. Python Labeling

We utilize an adapted labeling tool in Python to label the ball status for the first 12,000 frames of the play. Each frame attaches the following attributes: "Frame", "Visibility", "X", "Y". Frame records the number of frames. Visibility is a dummy variable indicating whether the ball is visible in the frame. Visibility = 0

implies that the ball is not within the frame. “X” and “Y” present the x and y coordinates of the detected ball in pixels. Coordinates for balls with visibility = 0 are marked as (-1, -1).

iii. File converting

We obtain a csv file containing labeling attributes. We then create a dataset in numpy format combining the attributes with frames, which also enables us to produce heatmap for frames with labeled coordinates for pickleball in frames.

C. Exploratory Data Analysis

The dataset after preprocessing includes 12,000 observations. Each frame in the video of in-play pickleball has a corresponding visibility status and cartesian coordinates of the ball, if applicable. Although the data is based on in-play clips, the ball may be marked not visible if it is obscured by a person, by the solid part of the net, or if it is outside the scope of the camera.

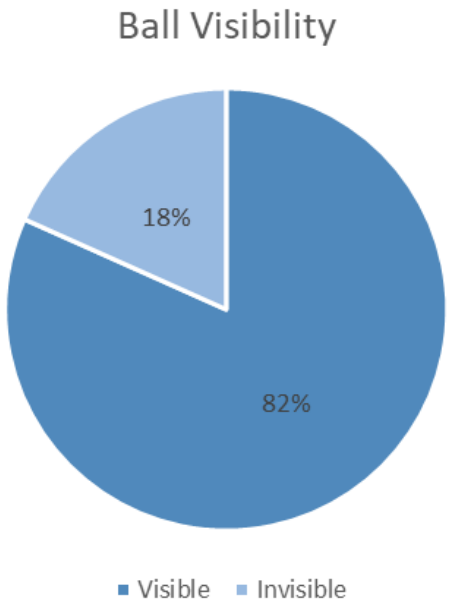

Figure 1. Distribution of Ball Visibility Among 12,000 Frames

As Figure 1 states, nearly one-fifth of the data in the dataset is not visible. The majority of these instances are when players on the front half of the court, closer to the camera, are obscuring the ball, which is why future works may benefit from utilizing different camera angles for the same pickleball game.

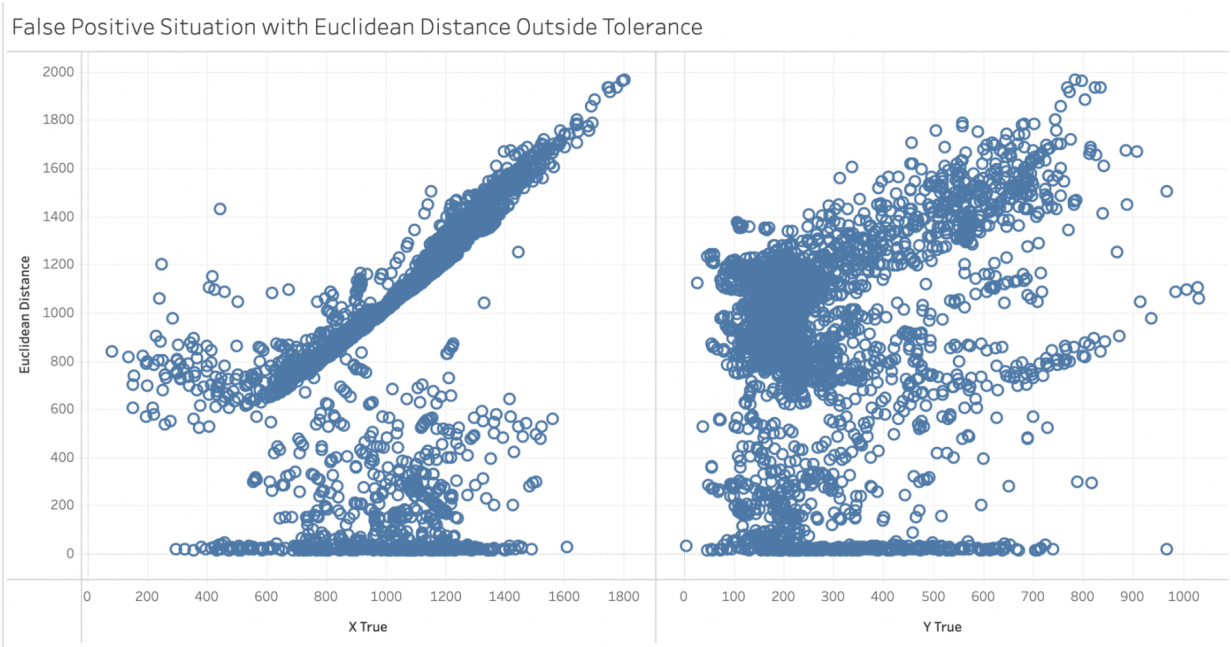

Figure 2. Euclidean Distance between prediction and ground truth

Due to the singular camera view, the density of points is clusters towards y-coordinates that are smaller, which corresponds to the top of the screen. This is a result of the far side of the court appearing smaller in the camera frame than the close side of the court, however there is no obvious pattern aside from that. This randomization, along with randomization evident in Figure 2 on x-coordinates of the dataset, indicates that the dataset is normally distributed for where on the court the ball is being hit. Because this analysis is being conducted to improve original weights of the TrackNetV2 model, the other aspect of data exploration is to establish the baseline and patterns in the original model.

IV. Hypothesis

We hypothesize that using transfer learning techniques, we can achieve a true positive rate of 65% for pickleball ball detection. The true positive rate, which is the percentage of balls that are correctly recognized and labeled by the model, is used as the basis for developing a quantified goal. This is because the ball detection model is going to be implemented in a more expansive piece of software for analyzing pickleball playing, and therefore it is more important to capture a high percentage of balls when they are visible than overall accuracy, which includes both correctly capturing balls when they are visible and not predicting locations when the ball is not visible. Because this model will be used to develop higher statistics about ball velocity, spin, and other information, it is most important to reach a level of true positive rate (TPR) at which these statistics can be reliably produced.

V. Model Development

TrackNetV2 Background

The model being used in this analysis is TrackNetV2. This was developed by researchers from Taiwan National Chiao Tung University [6] for badminton. It was developed as an extension of TrackNetV1 which was a ball tracking program trained on over twenty thousand frames of tennis play. TrackNetV1

was also developed at Taiwan University, by Huang et. al in 2019. TrackNetV2 was trained on 55,563 frames from 18 professional and amateur badminton matches.

A. Model Training Environment

In this project, the model was trained on an environment set up on Kaggle.com, which is a free cloud GPU provider. The model was trained on an NVIDIA P100 GPU with 16GB VRAM, an Intel Xeon CPU with 12GB RAM.

The limitations of Kaggle, however, are a storage limit of 100GB, however for the purposes of this project that was sufficient, as our 7500 labeled frames were 66.5GB and the MP4 that frames came from was .5 GB. There are also GPU limitations of 30 GPU-hours per week per person, which in aggregate was about 120 GPU-hours in total. While this was enough to train the model, further discussion of project expansions and future works may include the usage of a more advanced training environment, which would permit more epochs and larger batch sizes for more efficient training. The final model for this project was based on 20 epochs and it ran in approximately 11 hours.

B. Model Architecture

The TrackNetV2 model is a 63 layer encoder-decoder neural network. It contains 18 convolutional layers. In aggregate, this corresponds to 11,334,531 trainable parameters and 8,064 non-trainable parameters. An encoder-decoder model is a neural network which, as figure 3 shows, scales down dimensionality in the first half of the model through an encoding process, and then uses a decoding process to scale it back to the input frame size.

An encoder uses convolutional layers to extract features from input frames, however they lower the dimensionality of frames. Likewise, deconvolutional layers bring dimensions back up to that of the original frames, and therefore they make the output of the network the same size as the input frames. This scaling-up process occurs through U-net connections, which are the red bands in the figure. U-net connections concatenate output from earlier convolutional layers to their mirror deconvolutional layers on the other side of the network. This combines lower-level features from early layers with higher-level features from later layers, therefore retaining maximal feature information for use in the model training.

For this analysis, only the last layers of the network were set to be unfrozen for retraining purposes. Due to the large quantity of trainable parameters, a full training of the model from scratch would have required significantly more resources and time, whereas transfer learning, as demonstrated earlier, has been shown

to perform similarly in many fewer training epochs.

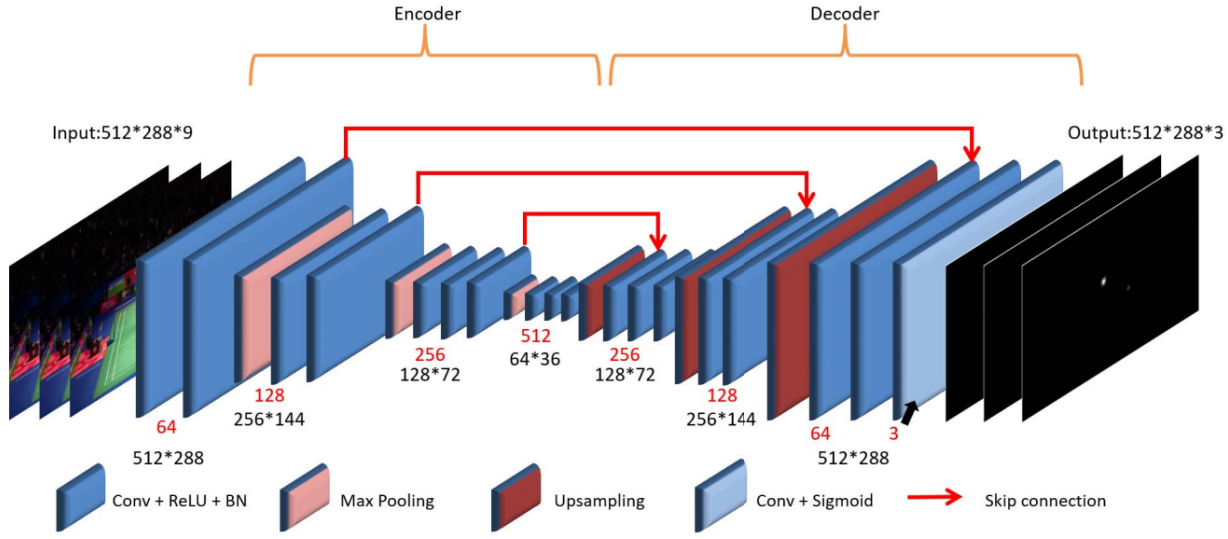

Figure 3. The architecture of TrackNetV2. The encoder-decoder structure basically follows the design of TrackNet. Compared with the original TrackNet, the innovation includes the downsizing of input images, skip connections enlightened by U-net, concise heatmap representation, and MIMO network design. [6]

C. How Model Learns

The TrackNetV2 model reads in numpy files which contain a gaussian heatmap of the ball location for every frame of a video. Ground-truth ball location is encoded as a black and white gaussian heat map centered at the labeled location of the ball in each frame. Figure 4 represents this heatmap and process. Because there are 256 colors in grayscale, this a 256-class classification problem. Different iterations of the model have been designed to read in a certain number of consecutive frames and output a similar number. The better performing models for badminton and tennis read multiple frames at a time so that prior knowledge of the ball location can be used to inform subsequent predictions.
The model trained on in this analysis is a 3-in and 3-out model, meaning that it reads 3 consecutive frames from a pickleball video and outputs 3 predicted heatmaps for those frames. Neural network parameters are learned through backpropagation using multi-label cross-entropy loss on ground-truth heatmap and predicted heatmap.

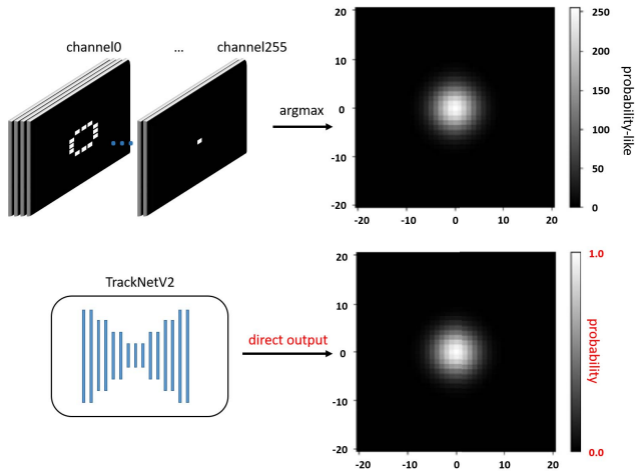

Figure 4. TrackNet generates a 256-channel boolean array for the ball detection.
Instead, TrackNetV2 directly generates a real-valued array to have the heatmap. [6]

D. Getting Model Performance Metrics

The goal of the analysis is to obtain a true positive rate of 65%, however this requires identifying the different results of the model and comparing to ground truth in order to obtain information about how well the transfer learning is performing. The model produces heatmaps for frames it believes contain a ball, and TrackNetV2 uses traditional computer vision algorithms to find circles within the heatmap. The model output with the predicted ball location is the center of the largest circle.

There are five categories a frame may be categorized into. The first category is the goal category, true positive (TP). A true positive is that the model predicts a ball, there's a ball, and it's at the right location. A true negative (TN) result is that the model predicts no ball and there's no ball. False positives are categorized into two distinct categories. In the first case, FP1, the model predicts a ball, and there is a ball, but it's too far away. In the second, FP2, the model predicts a ball, but there is no ball in the frame.

To quantify the difference between TP and FP1, where the model is making a prediction but it must be identified as an accurate or inaccurate prediction, we set a tolerance of 0.0075. Thus, if distance between predicted location and actual is less than tolerance value, it's a true positive even if prediction is not precisely the same as the ground truth.

The tolerance value, 0.0075, is normalized so that it can be applied to any video, regardless of dimensions. Based on the training data, the ratio of the ball diameter to the whole horizontal axis was 0.006 on average. There is some variation in this diameter as the ball moves closer and further from the camera, but 0.006 was the average value. Thus the radius of ball was .003 in rescaled frames, and because the predicted location of ball should be at least tangential to actual location of ball, which means that it can be up to 0.0075 from the center of the ground truth. When finding performance metrics, we scale tolerance value to match dimensions of input video, so the scaled tolerance is calculated as

$$\text{scaled\_tol} = .0075 * \sqrt{(video\ width)^2 + (video\ height)^2}$$

VI. Transfer Learning Process

The intention of using transfer learning is to use TrackNetV2 as a baseline and improve with new data, as generated by the labeling process. Appending a dense layer is one approach to transfer learning. Typically, convolutions are used to learn features and dense layers make predictions using those learned features. The original TrackNet V2 model goes straight from convolution layers to output without a dense layer between. Therefore, using a dense layer could allow the model to better use the extracted features from the convolutional layers.

The second approach to transfer learning is to freeze training of early layers in the network and only adjust a fixed number of final layers of the network. Since early layers learn lower-level features, and pickleball low-level features are similar to badminton low-level features, there is no need to adjust weights from those early layers. It has been demonstrated that this often gives high performance with a small amount of additional data.

The data for this model included 12000 labeled frames, of which 7500 are part of the training dataset. In order to feed into the model, the data was split into batches of 250 frames. Within each batch of 250, the

first 175 frames (70%) were used as training data, and the last 75 frames (30%) were used as validation data to determine best hyperparameters.

The first approach of appending a dense layer had extremely poor performance, however there were more significant results with the layer freezing method. The goal with layer freezing is to find optimal hyperparameters, including the number of epochs, learning rate, and number of convolutional layers to improve starting from the end of the network, which were optimized in that order.

Due to demonstrated convergence after about five iterations, a limit of five epochs was set on training attempts in order to see a trend in model metrics within a reasonable runtime. Then, learning rates of .00001, .0001, .001, .003, and .01 were all tested, and it was found that 0.01 is a quick and reliable learning rate that seems to extend well to validation and test data. Figure 5 demonstrates the performance of these learning rates.

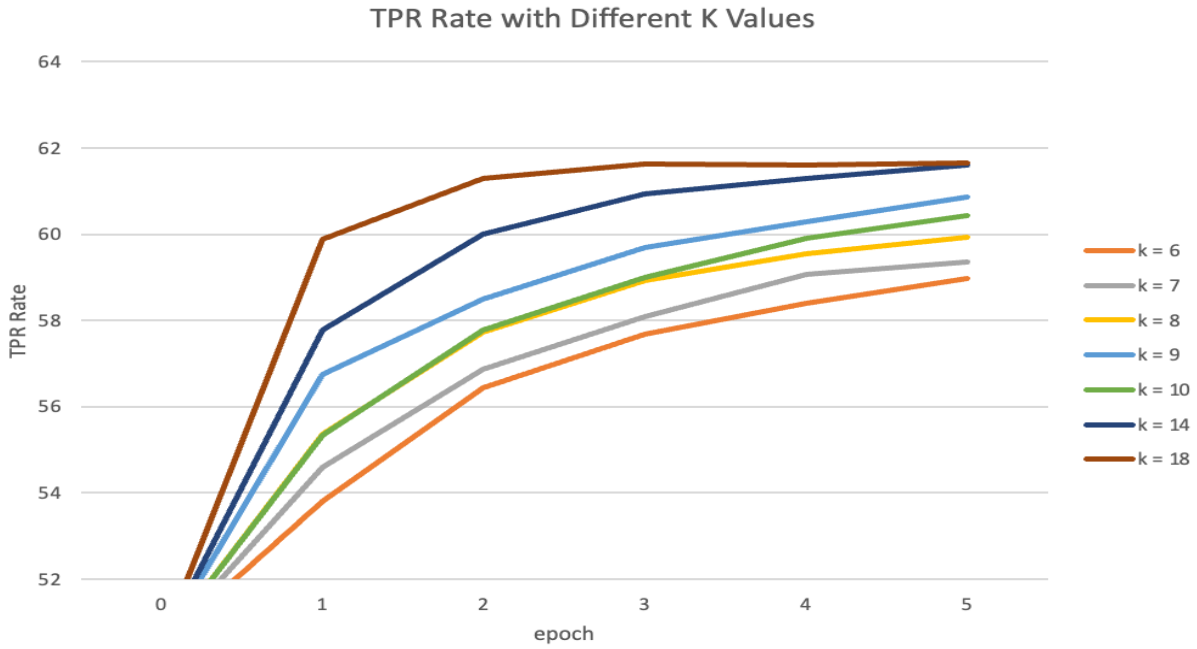

Figure 5. True Positive Rate under different K values

After determining optimal learning rate, the next task was to optimize the number of layers to unfreeze. There are 18 convolutional layers in total, so the first transfer learning attempts were done on very small values, and then the figure shows a larger range of layer values that were unfrozen and trained on with 5 epochs and a learning rate of 0.01. As is demonstrated by figure 5, both 14 and 18 layers seem to converge to the same value after only 5 epochs, and so given bias/variance trade offs, 14 seems like the optimal number of layers to unfreeze. The final model included here was trained with final hyperparameters until performance plateaued,which took approximately 20 epochs.

VII. Performance and Results

A. Baseline

Before beginning any transfer learning, baseline performance metrics had to be established. The set goal is to attain a 65% true positive rate, and initial metrics follow.

Baseline Metrics			
Accuracy	64.27	True Positive Rate	50.47
Recall	63.73	True Negative Rate	13.79
Precision	87.81	False Positive 1	4.46
		False Positive 2	2.55
		False Negative	28.73

Table 1. Baseline Accuracy and Performance Metrics

B. Dense Layer Approach

The dense layer approach was explored as it has previously been shown to perform well under transfer learning, however the results had extremely low accuracy and performance metrics, and so it was not used further.

This is likely due to pixel distortion from the implementation of the dense layer instead of SoftMax, which is what was previously the transition from the convolutional layers to the output layer. As a result, this method was disregarded in subsequent iterations of the model training process.

C. Layer Freezing Approach

The layer freezing approach ultimately was based on hyperparameters:

Learning Rate = 0.01 Epochs = 20 Unfrozen Layers = 14

From this, the model had the best performance at epoch 19, at which the metrics had reached a maximum true positive rate of 62%, which was the best true positive rate reached in the transfer learning process. The final performance was

Performance Metrics			
Accuracy	75.59	True Positive Rate	61.64
Recall	77.04	True Negative Rate	13.94
Precision	91.07	False Positive 1	3.64
		False Positive 2	2.40
		False Negative	18.37

Table 2. Layer Freezing Accuracy and Performance Metrics

VIII. Conclusion

Clearly, our model is able to track the pickleball in a video of a match better than TrackNetV2, a model designed for badminton videos. This improved performance allows our sponsor to better understand the ball's location in footage and utilize our output for downstream analytics to improve player performance. TrackNetV2 had a 50% True Positive Rate and our model used transfer learning to improve that to 62% TPR. The increase in TPR came mostly from decreasing the False Negative Rate from 29% down to 18%. One main issue with TrackNetV2 was it failed to predict many frames where it should have. By decreasing FNR, our model has predictions for more frames where it should. With more predictions, our sponsor is able to get more conclusive match performance results. These additional predictions come at no cost; our model predicts more accurately than TrackNetV2.

Our model's performance could be improved in many ways. Regarding the data, if we had videos with different camera angles on different courts, our model would be able to generalize better to new videos and new circumstances. Furthermore, as with any deep learning model, if we had more labeled frames we could train the model for more epochs at a potentially lower learning rate and achieve higher performance. With enough additional frames, we could possibly train the model from scratch instead of using transfer learning. The TrackNetV2 paper labeled around 55,000 frames to train their model, while we only had about 7,500 frames to work with. We could also improve the model using the data we already have. Fine-tuning the model with a lower learning rate, a higher number of epochs, and a greater number of unfrozen convolutional layers would allow the model to reach above the ~62% TPR plateau and reach our goal of 65% TPR. A fourth way the model could be improved is if we altered the model's architecture. Potentially, if we added more skip connections or more convolutional layers, the model would improve in performance.

IX. References

[1] J. Golden, "Pickleball, a paddle sport with a whimsical name, is becoming big business," CNBC, Apr. 24, 2022. https://www.cnbc.com/2022/04/24/pickleball-translates-into-big-business-with-tournaments-investments.html (accessed Apr. 30, 2023).

[2] "Pickleball Statistics: America's Fastest Growing Sport in 2023," Pickleheads. https://www.pickleheads.com/blog/pickleball-statistics (accessed Apr. 30, 2023).

[3] "Pickleball Market 2023-2027 | Future Investment, Expansion Plan, Market Dynamics, Key Players | Opportunities, Challenges, Risks Factors Analysis, Sales, Price, Revenue, Gross Margin," Nov. 09, 2022. https://www.yahoo.com/now/pickleball-market-2023-2027-future-124600555.html (accessed Apr. 30, 2023).

[4] D. Saul, "Kevin Durant Buys Pro Pickleball Team—Joining Brady And LeBron As Owners," Forbes. https://www.forbes.com/sites/dereksaul/2022/10/20/kevin-durant-buys-pro-pickleball-team-joining-brady-and-lebron-as-owners/ (accessed Apr. 30, 2023).

[5] E. R. https://www.emergenresearch.com, "Sports Analytics Market Size, Share | Industry Forecast by 2030." https://www.emergenresearch.com/industry-report/sports-analytics-market (accessed Apr. 30, 2023).

[6] N.-E. Sun et al., “TrackNetV2: Efficient Shuttlecock Tracking Network,” 2020 International Conference on Pervasive Artificial Intelligence (ICPAI), pp. 86–91, Dec. 2020, doi: 10.1109/ICPAI51961.2020.00023.

[7] H.-T. Chen, W.-J. Tsai, S.-Y. Lee and J.-Y. Yu, "Ball tracking and 3D trajectory approximation with applications to tactics analysis from single-camera volleyball sequences", Multimedia Tools and Applications, vol. 60, no. 3, pp. 641-667, October 2012.

[8] C. Szegedy, et al., "Going deeper with convolutions," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015 pp. 1-9. doi: 10.1109/CVPR.2015.7298594

[9] Hung CH. A Study of Automatic and Real-Time Table Tennis Fault Serve Detection System. Sports (Basel). 2018 Nov 28;6(4):158. doi: 10.3390/sports6040158. PMID: 30487405; PMCID: PMC6316619.

[10] Y. -C. Huang, I. -N. Liao, C. -H. Chen, T. -U. İk and W. -C. Peng, "TrackNet: A Deep Learning Network for Tracking High-speed and Tiny Objects in Sports Applications," 2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Taipei, Taiwan, 2019, pp. 1-8, doi: 10.1109/AVSS.2019.8909871.

[11] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, “CNN Features off-the-shelf: an Astounding Baseline for Recognition.” arXiv, May 12, 2014. doi: 10.48550/arXiv.1403.6382.

[12] M. Hussain, J. J. Bird, and D. R. Faria, “A Study on CNN Transfer Learning for Image Classification,” in Advances in Computational Intelligence Systems, Cham, 2019, pp. 191–202. doi: 10.1007/978-3-319-97982-3_16.