Course Project: Final Report

Hamza Aziz
University of Saskatchewan
Saskatoon, Saskatchewan
hamza.aziz@usask.ca

Abstract

A final report for the SoccerNet Tracking competition that is associated with the 9th International Workshop on Computer Vision in Sports(CVsports) at CVPR 2023. The git repository for this project can be found in this link: (https://github.com/Hamza975A/CMPT489-Project).

1. Introduction

Sports have for a long time now played a major role in both society as a whole, as well as how many individuals enjoy their lives. Whether it is playing or spectating, sports have the ability to really grasp the hearts of numerous people. With people's accessibility and desire to watch sports online increasing rapidly in recent years, so does the effort in using technology to enhance the viewing of sports. With the visual displaying for sports becoming more dynamic, it has made it more difficult to track objects (players, balls, etc.) and accurately identify them, in order to collect useful data. This data is important for the benefit of the players and teams, to allow them to review their performances and improve upon themselves to achieve a higher level of gameplay. Not only would this higher level of gameplay enhance the viewing experience for spectators, but the data itself allows for viewers to compare players, teams, and the games, adding another dimension to the viewing experience.

For the previously stated reasons, I wanted to work on a project for the SoccerNet Tracking competition, that is associated with the 9th International Workshop on Computer Vision in Sports(CVsports) at CVPR 2023. The task for this competition is Multiple Object Tracking (MOT), which expects competitors to detect all objects of interest (players, goalkeepers, referees, staff, ball) and make the association.

This competition is difficult as there are several factors that must be dealt with when tracking in soccer. A lot of the players are similar, making it hard to accurately identify them. Throughout all the clips, there are many times where a player will leave and re-enter the scene, every time that requires them to be reidentified. The cameras in soccer games show many individuals who are not meant to be tracked, such as fans, referees, ball boys, players on the bench, as well as coaching and medical staff. Distinguishing who we want to track and who we don't want to track can be difficult at times. The recorded video clips of soccer gameplay include fast motions, as well as scenarios where vision of the objects is obscure, which make both detection and association difficult. The competition has a focus on considering these difficulties, and including them within the training, testing and challenge sets for the competition to ensure the competition is challenging and thus push towards more robust solutions.

2. Methods

The following sections contain descriptions on the dataset, models, and metrics used for this project.

2.1. Dataset

The dataset used for this competition comes from a dataset called SoccerNet-v2, which has data collected from 12 complete soccer games, only using the main camera, which is not easy to find. As well as having the full 12 soccer games recorded (during the 2019 Swiss Super League), there is also 200 clips of 30 seconds each, and a complete halftime, both annotated with tracking data. The 30 second clips in particular are clips of key moments, or

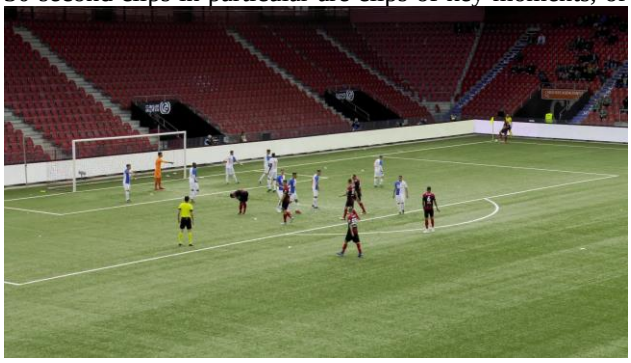

Figure 1: Sample image of a corner kick clip from the SoccerNet-v2 Tracking dataset.

moments that would be harder for tracking to perform well

on, such as corners, fouls, goals, free kicks, penalties, etc. These clips can be difficult to track multiple objects in due to the objects being closer together, sometimes overlapping, or certain fast paced motions of both the objects and the camera itself. “Note that a subset of this data is used in this first challenge. In particular, this accounts for 57 30-seconds clips for the train set, 49 clips for the test set, 58 clips for our first public challenge, and 37 clips for our future challenges, including the entire half-time video in the latter.” [1]. The first public challenge refers to the 2022 iteration of this competition, and so the 2023 version of this competition includes the 37 clips for the challenge set.

Each 30 second clip/sample is stored as 750 jpg images, which is 25 frames per second. Each jpg image has a dimension of 1920x1080, and the 750 images together take a size of 127 mb. The images do not contain any components usually displayed on TV, like advertisements, details of the score and time, as the data collected is directly from the main camera used for these games, as shown in Figure 1. Along with all the pictures, each sample also has two INI files called “gameinfo” and “seqinfo”. The seqinfo files have information stored that define attributes of the 30 second clip, such as framerate, sequence length, image size measurements, and image data type. Within the gameinfo files, labels for the sample are stored. Some of the labels are the name of the sample, gameID, both the game time start and end, visibility, and so on. One of the labels is called “actionClass”, which defines what the action within the 30 second clip is. For example, the actionClass might be a foul, or a corner, or a penalty kick, etc. The gameinfo file also contains a label for the number of unique tracklets throughout the 30 second clip, and a label exists for each tracklet ID going from 1 to x, x being the number of tracklets. The gameinfo file is only present in samples within the train and test set, and not within the challenge set.

Each clip is also stored along with the ground truth and detections for the bounding boxes of the multiple objects, both being stored in comma-separate csv files. These csv files have 10 columns, “These values correspond in order to: frame ID, track ID, top left coordinate of the bounding box, top y coordinate, width, height, confidence score for the detection (always 1. for the ground truth) and the remaining values are set to -1 as they are not used in our dataset, but are needed to comply with the MOT20 requirements.” [1]. The ground truth and detections data is only present with samples in the train and test sets, not in the challenge set. The 2022 dataset kept the detections data in the challenge set, as it is essentially all the bounding boxes ground truth without identifying the tracklets, allowing for competitors to focus on association. Using this data and focusing purely on association is also an acceptable task for the competition this year, as it was the main task of the previous year’s competition for CVsports at CVPR 2022, but this project will also be focusing on detecting the multiple objects on its own as well as associating them, as the new version of the SoccerNet Tracking competition states. The SoccerNet-v2 dataset allows for a large variety of use, but there are also other dataset’s out there with the purpose of soccer video understanding.

One of the similar datasets to SoccerNet-v2, is “SoccerTrack” a Dataset for soccer footage recorded with drones and fish-eye cameras [2]. The purpose of SoccerTrack is a solution to more accurately track objects on a soccer field and collect data. Since they use a camera with a fish-lens attached to a drone above the soccer field, it allows them to always view the entire pitch and all of the multiple objects. Another similar dataset is called SoccerDB which contains 171,191 video segments from 346 different high-quality soccer games [3]. This competition uses the second version of SoccerNet which has added more classes of actions (original SoccerNet only had goal, yellow/red card, and substitution), more annotations and has made a huge focus public tasks with reproduceable benchmarks for each one, to allow for events (like this competition) which would allow the public to help improve both the dataset and the technology of this area. There is also a third version of SoccerNet (SoccerNet-v3 which is not used for these events yet [4]. SoccerNet-v3 tries to connect the replay sequence which comes at a different camera angle, to the original live action frame from the main camera, which includes each of the two having the lines and goals annotated and players framed within a bounding box, and then having the all the

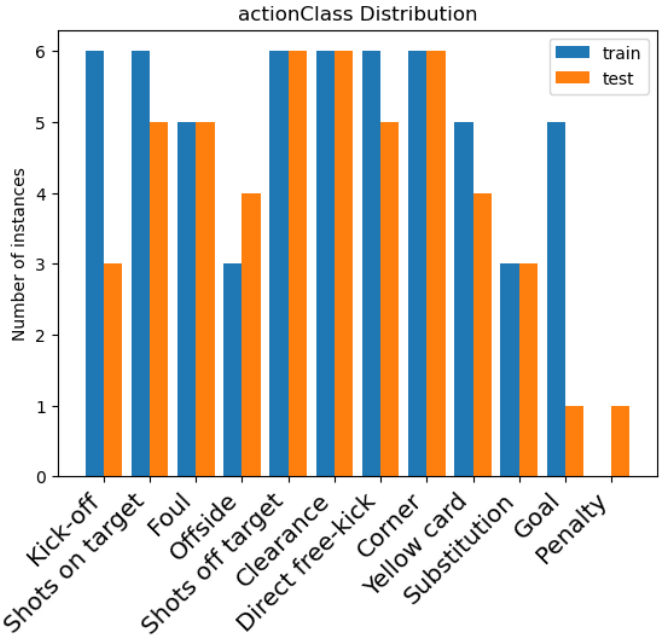

Figure 2: Bar chart representing the distribution of actionClass labels for the train and test sets.

annotations and bounding boxes in the two different views linked. It is still a work in progress, but it is trying to utilize the improved techniques from people attempting the tasks for SoccerNet-v2.

2.1.1. Visualization of The Data Labels

When analyzing the labels for the dataset, one of the first things that was considered was the "actionClass" label. One of SoccerNet-v2's biggest improvements was the increase of action labels, originally only having classes for goals, yellow/red cards, and substitutions. I wanted to investigate both what the new classes were in SoccerNet-v2, as well as the distribution of these actionClass labels in the testing and training sets given for this tracking competition. This is important to better understand what actions are occurring in the samples given for the competition, as well as to ensure there is a relatively even distribution between the testing and training sets. As displayed in Figure 2, the total number of unique actionClass labels went from 3 in the old SoccerNet dataset, to 12 in the given dataset for this competition from SoccerNet-v2. The distribution of labels within the train and test set's are fairly evenly distributed, with the goal and penalty actionClass labels being the exception. The test set only has one goal and one penalty sample, and the train set does not have any penalty samples. However since the focus of the competition is tracking and not action spotting, this is fine.

Another label that was considered was the "visibility" label. At first what this label suggested was unclear, and the initial assumption made was that some of the samples might not have as clear visibility of the pitch and objects as other samples, causing these samples to be difficult to deal with when tracking multiple objects. Upon further investigation, out of the 57 samples in the train set and the 49 samples in the test set, almost all the samples had a visibility label equal to "visible". Two samples in the train set and two samples in the test set had their visibility label equal to "not shown". Once these samples were investigated, it showed that the visibility refers to the main action of the clip. For example, in a clip where the main action is an offside call, if the player is not visible from the camera when the incident occurs, the action is "not shown" and thus the visibility label is defined. This consideration of when the objects are in the clip and how many lead to the analysis of the "num_tracklets" label. In the "gameinfo.ini" file, the num_trcklets label defines how many unique objects ever entered the cameras view during the clip, and each of these tracklets also have an ID stored in the file. On average throughout the test and train sets, 24.714 unique tracklets entered the camera's view in the samples, with a median value of 25. The clip with the least amount of tracklets had 20, and the most tracklets in any of the clips was 31. Finally, the last statistical analysis performed on the dataset related to the labels, was the analysis of the distribution of bounding box shapes.

Using the ground truth files given with each sample within the testing and training set, Figure 3 was created and represents the average size for all bounding boxes, for both sets individually, with the bounding boxes for the balls separated from all other bounding boxes. The figure shows that the bounding boxes for the balls are significantly smaller, which suggests that the bounding boxes are being properly wrapped around the detected object. The bounding boxes in the test set seem to be on average larger than the train set's bounding boxes. This can make sense because the train set has more clips than the test set, and the objects(players) spend most of their time is a standing position, meaning their bounding boxes will mostly be closer to being the largest they will ever be, with incidents like slide tackles or crouching being less common, where the bounding boxes would get smaller. Thus, I conclude that there are no apparent issues with the database related to the labels.

2.1.2. Visualization of The Data Inputs

The input data for the Soccernet-v2 dataset is very consistent according to the publishers. In particular, the input data for the SoccerNet Tracking competition's dataset, has consistent image characteristics. All samples have the same .jpg data type, a frame rate of 25, and an image size of 1920x1080. This data comes from the

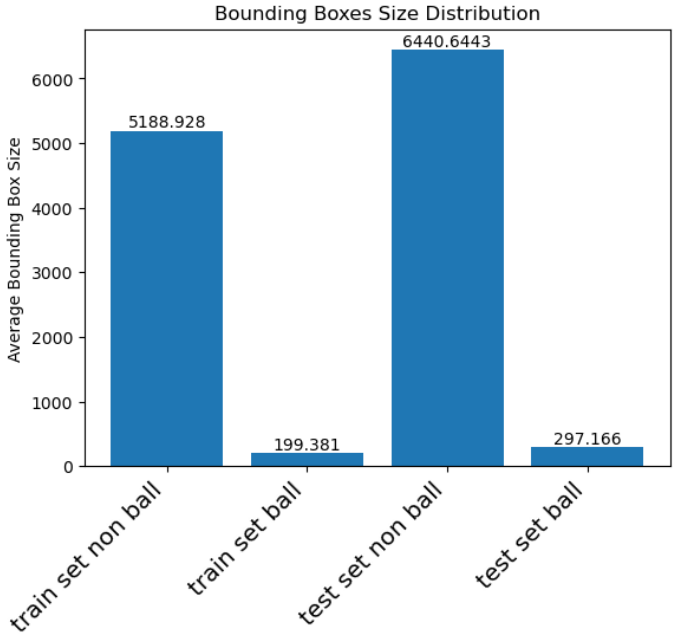

Figure 3: Bar chart representing the distribution of the Bounding Boxes for both the training set and testing set. Bounding Boxes for the ball are separated. The average sizes are calculated via width * height.

"seqinfo.ini" file as mentioned before, so I decided to perform an analysis to confirm that all the input data was of the correct and consistent format as the publishers stated. Figure 4 shows that all the images are size 1920x1080, and also are colour images. It even shows that the frame rate for the clips are correct, since the train, test and challenge set have a total of 143 clips, and 30 seconds each with a 25 frame rate, equals a total of 107250 images, which is the amount I found in the sets. The image size was also confirmed directly from the images, instead of the "seqinfo.ini" files, and the files were required to have ",jpg" type, so that also confirms all the image files were the correct data type.

I also decided to analyze the distribution of samples among the 12 complete soccer games in the SoccerNet-v2 dataset. Figures 4 and 5 show that within both the test and training set, the games which the samples originate from(indicated by the gameID) are evenly distributed, however the training set has samples from game's 4,6 and 9, while the test set has samples from games 7,8 and 11. This means none of the samples from the training set come from the same game as any of the samples in the test set. This makes me assume that the case is the same for the challenge set as well. This is likely intended, so that the processes between the training, testing, and challenge stages are all properly separated.

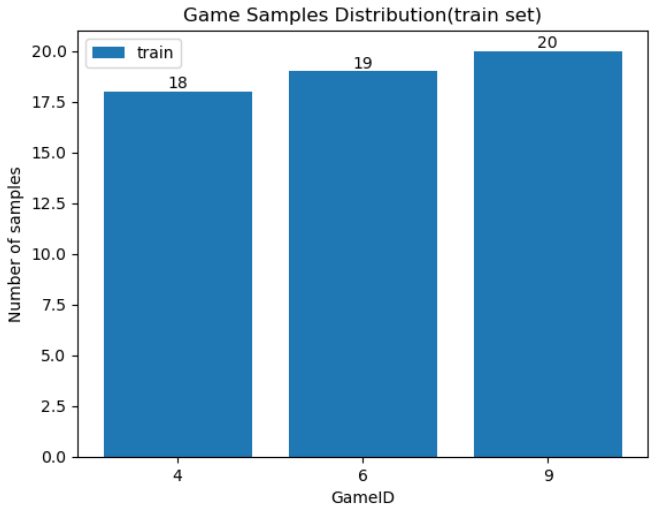

Figure 4: Bar chart representing the distribution of the gameID's for the samples of the train set.

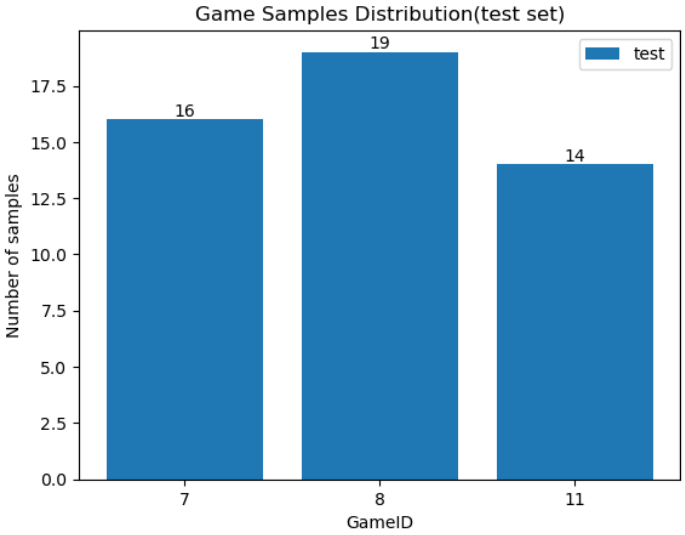

Figure 5: Bar chart representing the distribution of the gameID's for the samples of the test set.

2.2. Models

The two models I have selected are called DeepSORT and ByteTrack, which are baselines models provided for the competition. The DeepSORT method is a method built on top of a SORT method, which is a simple online and realtime tracking method, and DeepSORT uses a convolutional network that extracts appearance information and is pretrained on a large-scale person re-id dataset. ByteTrack on the other hand, is comprised of a strong object detector called YOLOX, and has a insightful data association method called BYTE, that first associates the high score detection boxes, and then associates the low score boxes with the remaining tracklets. A major difference between the two are that ByteTrack has better baseline performances in the metrics considered for this competition (thanks to its YOLOX object detector being very strong), however because ByteTrack is pretrained on a large dataset, more training is not as effective as more training on other models is.

2.3. Metrics

For both tasks, the performance would be measured by comparing the output with a predefined ground truth (stored in comma separated csv files) to calculate a HOTA (Higher Order Tracking Accuracy) value. HOTA is a newer but widely used metric that is used to evaluate object tracking systems. HOTA is a performance measurement that aims to determine the balance between sub metrics, specifically DetA (Detection Performance) and AssA (Association Performance).

3. Results

The results for this project are limited due to difficulties getting the models to work and achieving the minimum baseline results. Ultimately the final issues were related to the installation of CUDA and CUDNN. These installations are listed as requirements and not given with instructions, so when after I did manage to install them, when running the command "python3 setup.py develop", which is a step for setting up the ByteTrack model after cloning the git repository, it has issues detecting the CUDA installation. Attempts were made to resolve this through reinstalling CUDA multiple times (the same version or different versions), completely restarting this project's git repo, and attempting to set the project up on a different device with a fresh Ubuntu, and double checking related environmental variables, but ultimately nothing worked. Much debugging was done with the rest of the setup for the baseline models to set them up for this project, and so if another had CUDA and CUDNN installations working as intended, then the git repository should work correctly to get the baseline results. For this section I will mainly talk about my planned experiments, why I wanted to do these experiments and what I expected the results to be.

For the main experiment, I wanted to improve upon the baseline results by training and to record how the performance of the models scaled with training. I expected both to improve and achieve better HOTA scores when tested, but for DeepSORT model to improve more than the ByteTrack model. The reason for this is because as stated in the model section, ByteTrack is already pretrained on a large dataset for detecting objects, so more training with the relevant data for this project/competition may not be as effective. DeepSORT is not pretrained, and thus would scale better with training since the only data it will have seen would be more relevant training data. However, since ByteTrack's pretraining allows for it to have better object detection skills, it may improve a lot at first and then stop improving much, while DeepSORT will keep learning but slower. Another thing I wanted to investigate for the main experiment was adjusting the buffer space for the bounding boxes and watching the effects.

For adjusting the hyper-parameters for the ByteTrack model, the first parameter I considered was the maximum number of epoch's, a single epoch is where the dataset is fully passed through the model during training. The default maximum number of epochs for the model is 80, which seems excessive to me. Since I felt that 80 may be a waste, I wanted to test different values for the maximum number of epochs. The first two numbers I considered are 5, 10 and 15 to try and see if lower epoch values still get similar results, meaning we can save training time/resources by doing less epochs for training. In case these results were not as good, I wanted to test the numbers, 50, 70, 100 and 115, to see if other larger numbers near the original improved or got worse results.

For the ablation experiment I wanted to both test removing and adding layers. Adding layers would allow me to help improve the model's ability to learn features and improve with training, whereas removing layers would help me avoid overfitting and make the model more optimal when considering time and money. I decided I wanted to focus on adding layers first, as I felt it was not too likely that the model as is was struggling due to overfitting, but rather that it needed to improve it's performances when detecting and identifying objects. If I were able to add layers, I would have started with trying out layers with the RELU activation function first, as this activation function is simple yet effective, and commonly used for similar deep learning models. I might not be able to retrain the ByteTrack model with this architectural modification due to the model being pretrained to begin with on a very large database, which I cannot retrain, so I would be able to try this approach with the DeepSORT model instead. I would expect that adding these layers to the DeepSORT method would allow the model to obtain a higher HOTA score and perform better when tracking multiple objects.

4. Discussion

Due to not having the models working for my testing's, there are not any results to discuss.

5. Conclusions

The toolkit's provided had many adjustments that needed to be made due to the publishers updating some content for the 2023 version of the competition where detecting and identifying is part of the task, while other files and competition information pages were not updated. The specific CUDA and CUDNN versions they suggested to be used with UBUNTU 20.04 were not options to be downloaded via the NVIDIA developer site, and so I had to get versions as close as possible. I spent a week trying to resolve several issues, with the CUDA and CUDNN issues being the (potentially) final roadblock I could not resolve to get the baselines working. If I could go back in time, I would have used their discord discussion forums at an earlier point to see if I could get clarification on what I needed related to the CUDA and CUDNN prerequisites with the 2023 version of this competition. The competition due date is April 25th, 2023, and I am looking forward to seeing how other competitors did and reading any of the published papers. I still believe that Deep Learning in sports is an amazing research field, and as a fellow sports fan myself, I am looking forward to future innovations.

References

[1] Anthony Cioppa, Silvio Giancola, kangle, mars. (2022). SoccerNet - Tracking. https://github.com/SoccerNet/sn-tracking

[2] Atom Scott, Ikuma Uchida, Masaki Onishi, Yoshinari Kameda, Kazuhiro Fukui, Keisuke Fujii. (2022). SoccerTrack: A Dataset and Tracking Algorithm for Soccer With Fish-Eye and Drone Videos https://github.com/AtomScott/SoccerTrack

[3] Yudong Jiang, Kaixu Cui, Leilei Chen, Canjin Wang, Changliang Xu. (2020). SoccerDB: A Large-Scale Database for Comprehensive Video Understanding. https://arxiv.org/pdf/1912.04465.pdf

[4] Anthony Cioppa, Adrien Deliège, Silvio Giancola. (2022). SoccerNet-v3 https://github.com/SoccerNet/SoccerNet-v3