Fluendo Jr. CV Technical Test – Mar Coch Alcina

Q1

1. What do you think it occurred during this model development (training & evaluation)?

Since we assume the system was built with a deep learning model, the issue lies in the data. Not having proper data makes the model unreliable and perform poorly. In this case, the model was not prepared to deal with the situation, in which the lightning and camera angle made the lineman's head frequently match with the model's "idea" of a football better than the actual football.

This model's performance is most likely due to the lack of a labeled, high-quality, varied (reflecting all types of situations) dataset. It is possible that the data used was low-quality (low-definition footage, poorly labeled…) or wasn't extensive enough to reflect most situations.

In the footage we can see that when the camera follows the lineman's head it's because it has lost the ball, whether it is from occlusion, going out of bounds, or too high in the sky for the camera to track.

One of the biggest issues with computer vision is that it lacks the "knowledge" to follow the ball when the camera can no longer properly see it, such as when it goes too fast or players step in front. That's why the model was retreating to the most "ball-like" figure it could find: the lineman's head.

If the model had been trained considering the numerous occasions that the ball won't be seen perfectly, it could take into consideration the circumstances and predicting the path, instead of finding the most visually alike object (to the model's concept of a ball) every few frames, thus performing correctly.

2. How would you fix this behavior? Please provide at least 2 options explaining their pros and drawbacks.

To fix the dataset it was trained on, we need to obtain a high-quality, comprehensive dataset (lightning conditions, camera angles, occlusions, etc.). If not enough HD footage can be obtained, or it is too expensive and/or time-consuming, we could opt for adding synthetic data, or maybe even studying the possibility of using video-game-generated footage, since it has reached an ultra-high level of realism and a variety of situations that would be hard to come by in real life.

Pros: Synthetic data has been proven to improve the performance of models when mixed with real data, it is easy to create and bypasses privacy and copyright issues which are a problem when obtaining footage.

Cons: As we generate it ourselves, the quality of synthetic data relies on the quality of the model used to generate it, thus needing an additional step of verifying its quality before using it to train the model.

Secondly, we could try to make the model understand the game better, incorporating contextual information into the tracking to allow motion prediction. This could be achieved using a CNN (or RNN or a combination of both) to predict the trajectory and train it using datasets for scene understanding such as COCO.

Pros: by using a CNN we get a more robust and adaptable model that should be able to generalize and perform correctly in most situations.

Cons: the model would become more complex and need more training data than before to understand the context (COCO or similar datasets could be used to alleviate the data-gathering efforts to begin training the model). This results in longer training times and computational resources.

As an "out-of-the-box" solution, leaving the vision domain, audio cues from the audience could be used to better track the ball, such as which side of the stadium (relating to the camera) is cheering louder, or certain crowd reactions. This, however, is no longer computer vision, and I don't know exactly how much it would improve performance in relation to the additional data costs.

3. Extra: Do you know any tracking algos (Deep learning based) that could be used here?

As mentioned before, I know of RNN and CNN networks and some of their implementations, being Faster R-CNN one of the best suited for object detection, but I haven't used them, I have only retrained AlexNet to understand how a CNN and the training process works.

I have also heard of other algorithms that detect and classify objects, such as Deep SORT and YOLO. I have read that Deep SORT is one of the best multi-object, tracking algorithms (subject to input video quality) but it is very complex to implement.