Brief Report: Player Tracking and Re Identification in Sports Video

(-Heemaal Jaglan)

1. Objective

The goal of this project is to detect and track multiple players in a 15 second sports (soccer) video using a YOLO based object detection model. The key requirement is to maintain consistent player IDs even when a player leaves and re-enters the frame demonstrating a real time re-identification and tracking system.

2. Approach and Methodologies

2.1 Tools and Technologies used in project

- **YOLOv5 (Ultralytics)**: For real object detection (players and ball)
- **DeepSORT**: For object tracking and maintaining player identities for tracking
- **OpenCV**: For video reading, processing, and annotation over players.
- **Google Colab**: environment for code execution and development.

2.2 Step by Step Approach:

1. **Model Preparation**
    - A custom trained YOLOv5 model (.pt format) is used, fine tuned to detect players and the ball in real time.
    - The model is uploaded and loaded in a google colab environment.
2. **Video Processing**
    - The 15 second input video is processed frame by frame using OpenCV module.
    - Each frame is passed through the YOLO model to obtain bounding boxes and labels over the real time objects like player and balls.

3. **Object Tracking**
    - The bounding boxes of objects from YOLO detections are then passed to DeepSORT.
    - DeepSORT assigned and maintained unique player Identies across frames, enabling re identification of players.
4. **Output Generation**
    - The annotated frames are then saved into a new video file named (output_detected.mp4) showing bounding boxes and consistent IDs.
    - The output is downloaded for further analysis or sharing.

3. Techniques Tried and Outcomes

Technique	Outcome
YOLOv5 Detection	Accurately detects all players and the ball within the frames of video.
DeepSORT	Successfully tracked players(object) across frames and maintained consistent player identity.
Google Colab	Provided a simple and scalable environment with GPU support for proper execution of code.
OpenCV Video IO	Enabled real time frame annotation and final video export as output.

4.Challenged Encountered

- **Large Model Size:** The .pt model file was heavy that affected upload and load times.
- **Long Processing Time:** High resolution video combined with per frame detection led to significant processing delays in uploading and processing of the video.
- **Manual Upload Requirement:** Both the video and model files had to be manually uploaded in google colab environment.
- **File re upload After Restart:** Every time the Colab notebook session was restarted, the model and video had to be uploaded again and again.

5.Pending Work

- **Game Event Detection:** No mechanisms are currently imposed to detect key game events such as goals, passes, fouls, or substitutions. Event tagging based on tracking data and ball/player interaction could enhance analytical performance.
- **Team Differentiation**: All players are currently treated as one class. Adding classification logic (based on jersey color or team identity) would enable team specific statistics, comparisons, and tactical insights for players.
- **Long Video Support:** The solution is currently optimized for shorter clips. To scale for longer videos or full matches, the technical pipelines would require optimization in terms of memory usage, tracking consistency, and processing speed for enhancement in future.
- **Visual Summaries and Dashboards:** Adding a post processing steps to generate visual reports, heatmaps to study correlation or dashboards would make the output more usable for coaches or analysts.
- **Automation of File Handling:** Automating the file upload and management process (especially in google colab) would improve workflow efficiency and reproducibility by reducing latency.

6. Conclusion

This project successfully demonstrated the ability to detect and consistently track players in a sports video using a combination of YOLOv5 and DeepSORT models. The use of a google colab based workflow made it accessible and cloud executable. While the core tracking objective was achieved, further improvements such as automated file handling and event recognition can be pursued in future iterations to enhance the speedup and efficiency of the model .