Player Re-Identification in Sports Footage - Submission

Overview

Hello, this is my submission for the Liat.al Player Re-Identification assignment. I picked **Option 2: Re-Identification in a Single Feed**. The goal was to track players in a 15-second video (15sec_input_720p.mp4) and make sure they keep the same ID even if they leave and come back into the frame. I used the YOLOv11 model you provided and a tracking tool to get it done

---

How to Set Up and Run the Code

Prerequisites

You need Python installed in your system. I used Google Colab to run my code, but you can run it locally if you have the required libraries.

Dependencies

To run the code, you need to install the following Python libraries:

- `ultralytics` (for YOLO modell)
- `opencv-python`
- `numpy`
- `scipy`
- `Tqdm`
- `matplotlib`

You can install them using this command:

pip install ultralytics opencv-python numpy scipy tqdm matplotlib

Files Needed

1. **Video File**: `15sec_input_720p.mp4`
2. **YOLO Model**: `best.pt`
3. **Tracker Config**: `botsort.yaml`
4. **Main Script**: `player_reid.py`

Make sure these files are in the same folder as the script or update the file paths in the code to match their locations.

Running the Code

1. Run the ipynb notebook as is or, if you are running it locally,
2. Place `15sec_input_720p.mp4` and `best.pt` in your working directory
3. Use the botsort.yaml file that was created before
4. Run the Python script (`player_reid.py`):
   – python player_reid.py
5. The script will:
   - Load the YOLOv11 model and process the video.
   - Output a tracked video named `output_tracked_video.mp4` with player IDs displayed.

Output

The output is a video file (`output_tracked_video.mp4`) where each player has a bounding box with a consistent ID, even if they leave and re-enter the frame.

Approach and Methodology

What I Did

I used the provided YOLOv11 model to detect players in the video and applied a tracking algorithm called BoT-SORT (provided by Ultralytics) to assign and maintain player IDs.

1. **Loading the Model and Video**:
   - I loaded the YOLOv11 model (`best.pt`) to detect players in each frame of the video.
   - I used OpenCV to read the input video and set up a video writer to save the output.
2. **Tracking with BoT-SORT**:
   - I configured BoT-SORT, a tracking algorithm that supports Re-Identification (ReID), to keep track of players across frames.
   - I created a `botsort.yaml` file with settings like `with_reid: True` to enable appearance-based matching, which helps re-identify players who leave and re-enter the frame.
   - I set a confidence threshold (`conf=0.7`) to ensure only reliable detections are tracked.

    - I used `classes=[2]` because, after checking `model.names`, I found that class 2 corresponds to players

3. **Processing and Saving**:

    - The model processes the video frame by frame, drawing bounding boxes and IDs on each player.
    - The processed frames are saved into a new video (`output_tracked_video.mp4`).

Techniques Tried

- **BoT-SORT Tracker**: I chose BoT-SORT because it combines motion-based tracking (using Kalman filtering) and appearance-based ReID, which is good for keeping consistent IDs when players reappear.
- **Confidence Threshold**: I tested different confidence thresholds (0.5, 0.6, 0.7) and found that 0.7 gave fewer false detections.
- **Tracker Parameters**: I adjusted parameters like `track_buffer` (set to 400) to remember players for longer when they leave the frame, and `appearance_thresh` (set to 0.5) to balance appearance-based matching.

Outcomes

- The code successfully detects players in the video and assigns IDs.
- Most players keep the same ID when they re-enter the frame, specially near the goal event, as BoT-SORT uses appearance features to match them.
- The output video shows bounding boxes with IDs and conf score (players) ,making it easy to see which player is which.

What Worked Well

- The YOLOv11 model detects players accurately in all frames.
- BoT-SORT does a good job of maintaining IDs for players who leave and re-enter, especially when their appearance is clear.

Challenges Encountered

1. **Tuning Tracker Parameters**:
   - Setting up `botsort.yaml` was tricky. I had to read Ultralytics documentation to understand parameters like `track_buffer` and `appearance_thresh`.
   - I tried different values, for better accuracy.
2. **Video Processing**:
   - The video processing took some time on CPU (few minutes on Google Colab) but if you are using a GPU it will only take few seconds, even with ReID enabled.
3. **Re-Identification Accuracy**:
   - When players moved quickly or were partially visible, the tracker sometimes assigned new IDs instead of keeping the old ones.

Next Steps (If I Wanted to improve upon the current implementation)

- **Try Other Trackers**: I'd experiment with trackers like ByteTrack as it may give me more freedom to customize feature extraction and parameter tuning, although I did try Deep SORT, while it performed well for tracking it did not give optimal results for re identification.
- **Improve ReID Accuracy**: I could fine-tune the ReID model or add features like jersey color and number detection to make player matching more robust..
- **Handle Crowded Scenes**: I'd explore techniques like multi-object tracking with occlusion handling to improve tracking when players overlap.
