Player Re-Identification in Sports Footage: Brief Report

1. Introduction

This report details the approach and methodology employed to address the player re-identification assignment for Liat.ai. The core challenge involves maintaining consistent player identities across video frames, particularly when players temporarily exit and re-enter the field of view. This capability is crucial for advanced sports analytics, enabling more accurate tracking of individual player performance, tactical analysis, and automated highlight generation. The assignment specifically focused on Option 2: Re-Identification in a Single Feed, which required identifying each player within a 15-second video clip (`15sec_input_720p.mp4`) and ensuring that players who go out of frame and reappear are assigned the same identity as before. The solution aims to simulate real-time re-identification and player tracking, leveraging modern computer vision techniques.

2. Approach and Methodology

The chosen approach for player re-identification in a single video feed is built upon a combination of object detection and a custom tracking logic. The primary tools and libraries utilized are Ultralytics YOLOv8 for robust object detection and OpenCV for video processing and visualization.

2.1. Object Detection with YOLOv8

At the heart of the solution is the YOLOv8 (You Only Look Once, version 8) model. YOLOv8 is a state-of-the-art, real-time object detection model known for its speed and accuracy. For this assignment, a pre-trained `yolov8n.pt` model (the nano version, optimized for performance) was used. This model is capable of detecting various objects, and for our specific task, it was configured to identify 'person' objects (class 0 in the COCO dataset, which YOLOv8 is typically trained on). The choice of YOLOv8 was

driven by its efficiency, which is critical for simulating real-time tracking, and its ability to provide bounding box coordinates for detected players in each frame.

2.2. Player Tracking and Re-Identification Logic

While YOLOv8 provides object detection, the `model.track()` function from the Ultralytics library was leveraged for its built-in tracking capabilities. This function is designed to assign unique `track_id` values to detected objects and persist these IDs across frames. This persistence is a fundamental component of re-identification. The `persist=True` argument ensures that the tracker attempts to maintain the identity of objects from one frame to the next.

To address the specific requirement of re-identifying players who go out of frame and reappear, a custom re-identification logic was implemented. The `model.track()` function assigns a new `track_id` if a player is lost and then reappears. To maintain a consistent `player_id` for such instances, the following methodology was adopted:

1. **Initial ID Assignment:** When a new `track_id` is encountered from the YOLOv8 tracker, it is mapped to a unique `player_id` from a continuously incrementing counter (`next_player_id`). This `player_id` is the consistent identifier we aim to maintain.

2. **Tracking Active Players:** A dictionary (`active_players`) is maintained to keep track of players currently in the frame, storing their assigned `player_id`, their last known bounding box, and their status (active or lost).

3. **Handling Lost Players:** If a `player_id` that was previously active is not detected in the current frame, its status in `active_players` is updated to 'lost'. This signifies that the player has temporarily gone out of view.

4. **Re-identification of Re-appearing Players:** When the YOLOv8 tracker assigns a *new* `track_id` (meaning it's a new detection or a re-appearance), the system checks if this new detection corresponds to a previously 'lost' player. Currently, this re-identification is a basic check: if a new `track_id` appears, and there are 'lost' players, it assumes one of the 'lost' players has re-appeared and re-assigns the existing `player_id`. This is a simplified approach for the purpose of this assignment. In a more advanced system, this would involve:

    - **Feature Matching:** Comparing visual features (e.g., appearance embeddings) of the newly detected player with those of 'lost' players.
    - **Motion Prediction:** Using the last known trajectory of 'lost' players to predict their re-entry point and match with new detections.
    - **Spatial Proximity:** Considering the proximity of the new detection to the last known position of 'lost' players.
5. **Visualizing Results:** The processed video frames are displayed using OpenCV, with the consistent `player_id` overlaid on each detected player's bounding box. This provides a real-time visual representation of the re-identification process.

2.3. Development Environment Setup

The development environment was set up in a sandboxed Ubuntu 22.04 Linux environment. The following key libraries were installed:

- **`ultralytics`**: For YOLOv8 model loading, object detection, and tracking functionalities.
- **`opencv-python`**: For video capture, frame processing, and displaying the output.

All necessary video files (`15sec_input_720p.mp4`, `broadcast.mp4`, `tacticam.mp4`) were provided by the user and placed in the working directory for direct access by the scripts.

3. Techniques Tried and Their Outcomes

Initially, the approach relied solely on the `model.track()` function of Ultralytics YOLOv8. However, it was observed that when a player was occluded or left the frame and then reappeared, the `track_id` assigned by YOLOv8 would often change. This behavior, while expected for a basic tracker, did not meet the assignment's requirement for *consistent* player re-identification across disappearances and reappearances. To address this, the custom `player_id_map` and `active_players` logic was introduced. This allowed for a more robust assignment of a persistent `player_id` that transcends the transient `track_id` values generated by the YOLOv8 tracker.

4. Challenges Encountered

Several challenges were encountered during the development process:

1. **Broken Download Links:** The initial Google Drive links provided in the assignment document for both the assignment materials and the pre-trained model were broken (resulting in 404 errors). This required manual intervention and web searches to locate alternative sources for the YOLOv11 model (though YOLOv8 was ultimately used due to easier integration and availability of pre-trained weights for general object detection) and the video files. The user's timely provision of the video files was crucial in overcoming this hurdle.

2. **Dependency Management:** Ensuring all necessary Python packages (`ultralytics`, `opencv-python`) were correctly installed and accessible within the sandboxed environment required some troubleshooting, including re-installing packages and verifying their presence.

3. **Re-identification Logic Complexity:** The core challenge of player re-identification is inherently complex. The current implementation uses a simplified re-identification logic based on the assumption that a new `track_id` appearing after a player was 'lost' corresponds to that same player. While this provides a basic solution for the assignment, a production-ready system would necessitate more sophisticated algorithms that incorporate:

    - **Appearance Embeddings:** Using deep learning models to generate unique feature vectors for each player's appearance, allowing for robust matching even after long occlusions or changes in pose/lighting.
    - **Kalman Filters/Motion Models:** Predicting player trajectories to anticipate re-entry points and improve association accuracy.
    - **Global Optimization:** Considering all tracks simultaneously to resolve ambiguities and minimize identity switches over the entire video sequence.

5. Future Improvements

Given more time and resources, the following improvements would be implemented:

- **Advanced Re-identification Algorithms:** Integrate more sophisticated re-identification techniques, such as those based on Siamese networks or re-

ranking algorithms, to improve accuracy and robustness.

- **Performance Optimization:** Explore techniques like model quantization, pruning, or using more optimized inference engines (e.g., TensorRT) to further enhance real-time performance.
- **User Interface:** Develop a more interactive user interface (e.g., using Streamlit, as per user preference) to allow for easier configuration, visualization of tracking results, and analysis of re-identification metrics.
- **Comprehensive Evaluation Metrics:** Implement more rigorous evaluation metrics beyond simple re-identification counts, such as Multiple Object Tracking Accuracy (MOTA) and Identity F1 Score, to quantitatively assess the solution's performance.
- **Handling Edge Cases:** Address challenging scenarios like multiple players entering/exiting simultaneously, extreme occlusions, and changes in player attire.

6. Conclusion

This assignment provided a valuable opportunity to apply computer vision principles to a real-world sports analytics problem. The developed solution, while foundational, successfully demonstrates the core concept of player re-identification in a single video feed using YOLOv8 for detection and a custom logic for maintaining consistent player identities. The challenges encountered highlighted the complexities of robust tracking and re-identification in dynamic environments, paving the way for future enhancements and more sophisticated implementations.

7. References

[1] Ultralytics YOLOv8 Documentation. Available at: https://docs.ultralytics.com/ [2] OpenCV Documentation. Available at: https://docs.opencv.org/