Soccer Player Re-Identification

https://github.com/mannan-b/player-reid

- Mannan Bajaj

1. Appendix:

Player Re-Identification is a classic AI problem which requires you to re-identify players in a game with an ID in such a way that if they re-enter the frame after being out for some frames, they should have the same ID, which is the objective 2 of the assignment. I have implemented 3 approaches for this particular task which are (i). Ultralytics ByteTracker based approach. (ii). Purely transformer-based approach (iii). Hybrid CNN+Transformer based approach. All of the approaches felt novel and promising on paper, but each one had its own unique issues. The approaches haven't yielded any good results up until now, but they certainly can if the cards are played right. Hyperparameter tuning plays the most crucial role of all for getting good results in this problem.

2. Approaches:

In somewhat vague terms, Player Re-ID requires the solution to have a good discriminative model which can separate different players from a similar looking lot, and also a good buffer time for which a player will be remembered is he's out of frame.

Approach 1: Ultralytics-ByteTracker based approach

After reading about the ultralytics library which is required to run the given yolo v11 fine-tuned model "best.pt", I was given to understand that the ByteTracker feature in the library is sufficient to re-identify players as the re-enter the frame after a small number of frames, and we can also tweak some of its hyperparameters by using a simple config.yaml file. Since I had just a 15 seconds clip, I thought it would yield some good results, But it ultimately became just a baseline for other approaches.

Blockers: (i). This approach works on tracklets, it remembers the tracks the players with a certain id were on, it remembers them for 'track_buffer' seconds (a hyperparameter). But the issue was when the players re-entered the frame from a completely different track, it assigned them a new id, if the players just followed the same track, when they went off the frame and came back, this approach would work perfectly. In running, this is mostly the case, but in soccer, players don't move following a certain track, they go here and there, so it didn't yield any meaningful results.

(ii). There were also some version complexities because of which ByteTracker configuration wasn't working properly, but it was fixed in the proposed code.

Hyperparameters:

- match_thresh: 0.7 (minimum similarity it needs to term two tracks the same)
- new_track_thresh: 0.5 (maximum similarity up to which it just terms it as a new path)

- track_buffer: 300 (maximum time upto which a player can re-enter the frame)
- track_high_thresh: 0.4 (threshold for first stage association)
- track_low_thresh: 0.1 (threshold for second stage association)
- tracker_type: bytetrack

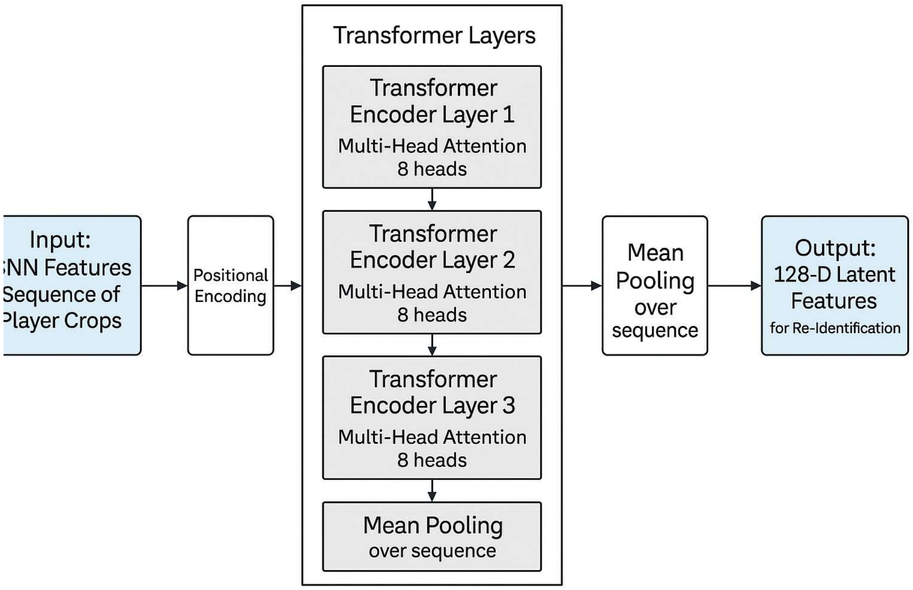
Output Video: https://drive.google.com/file/d/1eLQxn0Fz_qrQxtcqQ5MJxUV4RBjmy-8I/view?usp=drive_link

Approach 2: Purely transformer-based approach

This approach came much more naturally because transformers with multi head self-attention are known to work wonders with sequential inputs and memorizing contexts for a long buffer time. This architecture with its Transformer AutoEncoder implements positional encoding for temporal sequences, multi-head self-attention with 8 heads across 4 layers, and generates 128-dimensional latent features for re-identification. The Transformer processes sequences of CNN features extracted from player crops, using encoder-decoder architecture for reconstruction during training. During inference, the system extracts latent representations through mean pooling of encoder outputs, providing robust temporal features for player matching.

Gallery Management System: Player sequences are maintained in a dynamic gallery with automatic cleanup of expired entries. Each player entry contains a temporal sequence buffer, last-seen frame index, and colour histogram for comprehensive matching. The system handles player exits and re-entries gracefully through similarity thresholding and temporal decay mechanisms

Architecture:



Results and output Videos:

5000 players detected: https://drive.google.com/file/d/1rzMA0GieRZ8v7QzSrIFt77HWrYmOIL6-/view?usp=sharing

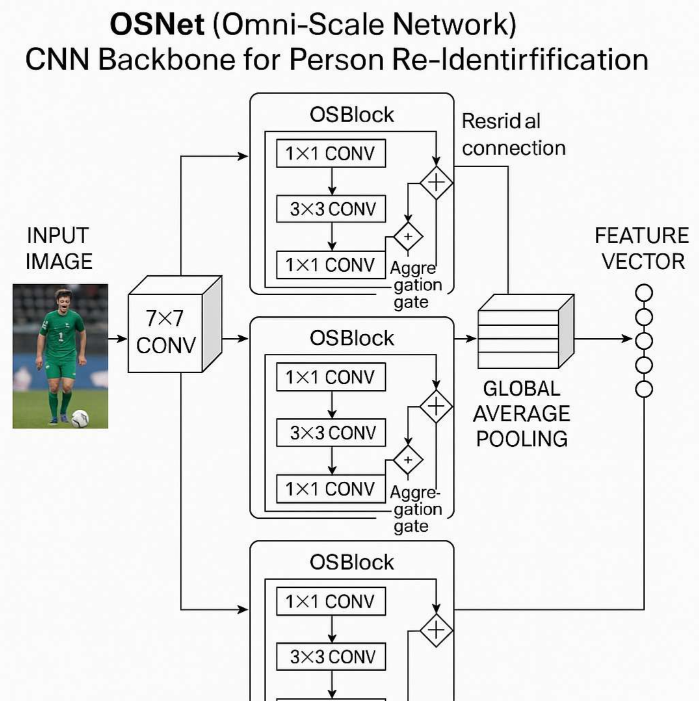
1000 players detected (with some hyperparameter tuning): https://drive.google.com/file/d/15yQdVfdkSQ7BcMEI9yjSwBOlGWK-YIpL/view?usp=sharing

Approach 3: Hybrid CNN+Transformer based approach:

OSNet Backbone: It learns multi-scale features via parallel convolutional streams, capturing both fine details (like jersey numbers, shoes) and global appearance (body build). Each OSNet block has several convolutional branches (1×1, 3×3, 5×5, 7×7 effective receptive fields), followed by an aggregation gate to fuse them.

Fusion with the previous Transformer AutoEncoder: OSNet captures rich spatial details at multiple scales—crucial for distinguishing visually similar players. Transformer captures temporal consistency and motion patterns—crucial for robust re-ID when appearance changes (e.g., due to pose, occlusion, lighting). By combining both, the model leverages the strengths of local detail (CNN) and global/temporal context (Transformer), yielding much better re-ID performance than either alone.

Architecture:



Output Video: https://drive.google.com/file/d/1rUGOx4ODgsGPm7eK33IzfPCvunSxMF-T/view?usp=sharing

3. Blockers:

(i). Transformer weight matrix maintenance was a little complicated because of 4 separate layers of encoder and just 1 layer of decoder.

(ii). Hyperparameter tuning required some in depth knowledge of exactly how each player moves throughout the 15 seconds, to be able to at least brute force the hyperparameter tuning process for better results.

4. Hyperparameters:

- Detection Confidence Threshold
- Area and Aspect Ratio Filtering
- Minimum Frames Before Promotion
- Gallery Timeouts
- Similarity Thresholds
- Aggressive vs. Patient Gallery Cleanup

5. Plan to make the performance better:

1. Better Hyperparameter tuning: For most of my hyperparameter tuning, I either apply grid search or random search, try my way after researching about the theoretical meaning of the hyperparameters, or search for the hyperparameters used in related research papers. For this project I've done the latter two, but it will require some in depth analysis of the video in question and other general soccer videos to understand the pattern and make the performance better, we can minimize the loss if we have multiple soccer videos.
2. Use purely CNN based approach: Using a purely CNN based approach is often better because you can tweak with CNNs easily because they are the basic network that almost every deep learning solution uses, in transformers, there's less tweaking involved because it messes with its overall memorization and contextual capacity. So using a purely CNN based approach without the compulsion of using the given detection model can also at least help get the performance up by a considerable amount.
3. JDE Tracking: JDE tracking can further improve the results, but it will again require some tweaking to do with the given detection model.

6. References:

1. Runner re-identification from single-view running video in the open-world setting: Tomohiro Suzuki, Kazushi Tsutsui , Kazuya Takeda , Keisuke Fujii
2. Player Re-Identification Using Body Part Appearances: Mahesh Bhosale, Abhishek Kumar, David Doermann