Real-Time Player Re-Identification in Football Broadcast Video

Author:Sanju K

🎯 Objective

The goal of this assignment is to build a real-time player detection and re-identification system for a 15-second football broadcast video (15sec_input_720p.mp4). The system must:

- Detect football players using a pre-trained YOLOv11 model.
- Assign unique IDs to players in the initial frames.
- Ensure that players who leave and later re-enter the frame retain their original identity (no ID switching).
- Simulate a real-time inference and tracking pipeline.

This task reflects the practical challenges of maintaining consistent identity across occlusions, camera motion, and scene transitions in live sports broadcasts.

🧠 Approach and Methodology

To solve this problem, the following steps were executed:

1. Frame Extraction

- The 15-second input video was sampled using OpenCV.
- Extracted frames were used for annotation and dataset generation.

2. Dataset Creation

- Created via Roboflow with 4 custom classes:
  - Football players
  - Referee
  - Football

- Miscellaneous (goalposts, etc.)

- Data augmentation was applied: flipping, brightness, blur.

Final dataset split:

- Total: 238 images
    - Train: 222
    - Val: 11
    - Test: 5

📎 Dataset hosted at: [Roboflow Link]

---

3. Model Training

- Selected YOLOv11 (Ultralytics) for real-time object detection.
- Training was done in Football_Player_Reidentification.ipynb:
    - 150 epochs
    - Early stopping enabled
    - Best checkpoint saved as best.pt

---

4. Inference & Tracking Pipelines

- Two scripts were developed:
    - yolo_inference.py: for baseline detection using pretrained weights.
    - main.py: for detection + tracking with custom weights (best.pt).
- Player tracking was implemented using:
    - Consistent ID assignment in initial frames.
    - Re-identification logic using a custom tracker.
    - Results saved to: tracker_stubs/player_detection.pkl

🧪 Techniques Tried and Outcomes

Technique	Outcome
Pretrained YOLOv11 (yolo_inference.py)	Detected players and ball but lacked consistent identity across frames.
Custom YOLOv11 Training	Achieved strong results on validation data. High precision/recall.
Re-ID and Tracking (main.py)	Successfully assigned and retained player identities across frames—even during occlusion or re-entry.

Re-ID simulation was done in a frame-by-frame pipeline, mimicking real-time video playback.

⚠ Challenges Encountered

1. Identity Gaps After Frame Exit:
   Players returning to the scene were occasionally misidentified or assigned new IDs due to visual appearance changes.

2. Uniform Confusion Between Players:
   Players from the same team (same kit) were sometimes assigned the same ID or swapped.

3. Blur/Occlusion:
   High-speed scenes or partial occlusion affected the quality of embeddings, hurting re-identification accuracy.

📊 Model Performance and Results

Class	Precision	Recall	mAP@50	mAP@50–95
Football	0.811	0.778	0.747	0.373
Football Players	0.984	0.922	0.953	0.469
Referee	0.918	0.929	0.963	0.445
Overall	**0.904**	**0.876**	**0.888**	**0.429**

Speed Benchmarks:

- Preprocessing: 0.2 ms/image
- Inference: 17.4 ms/image
- Postprocessing: 1.1 ms/image

```
                 all         11        152       0.87      0.883      0.868      0.373

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
    147/150      10.4G     0.8305     0.3913      1.048        156        640: 100% 14/14 [00:10<00:00,  1.39it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  4.31it/s]
                   all         11        152      0.872      0.883      0.864      0.371

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
    148/150      10.4G     0.8166     0.3901      1.045        170        640: 100% 14/14 [00:10<00:00,  1.38it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  4.69it/s]
                   all         11        152      0.876      0.878       0.86      0.364
EarlyStopping: Training stopped early as no improvement observed in last 100 epochs. Best results observed at epoch 48, best
model saved as best.pt.
To update EarlyStopping(patience=100) pass a new patience value, i.e. `patience=300` or use `patience=0` to disable EarlyStop
ping.

148 epochs completed in 0.492 hours.
Optimizer stripped from runs/detect/train2/weights/last.pt, 51.2MB
Optimizer stripped from runs/detect/train2/weights/best.pt, 51.2MB

Validating runs/detect/train2/weights/best.pt...
Ultralytics 8.3.160 🚀 Python-3.11.13 torch-2.6.0+cu124 CUDA:0 (Tesla T4, 15095MiB)
YOLO11l summary (fused): 190 layers, 25,282,396 parameters, 0 gradients, 86.6 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  4.16it/s]
                   all         11        152      0.904      0.876      0.888      0.429
              football          9          9      0.811      0.778      0.747      0.373
      football players         11        131      0.984      0.922      0.953      0.469
                refree          7         12      0.918      0.929      0.963      0.445
Speed: 0.2ms preprocess, 17.4ms inference, 0.0ms loss, 1.1ms postprocess per image
Results saved to runs/detect/train2
💡 Learn more at https://docs.ultralytics.com/modes/train
```

The system ran consistently on standard video input and produced annotated output saved in output_videos/.

🔭 Future Work and Possible Extensions

To improve real-world applicability and robustness:

- Temporal Matching for Re-ID:
  Use temporal embeddings or similarity learning to retain identity across long gaps.
- Custom Feature Extraction:
  Incorporate player jersey numbers, logos, or pose estimation for personalized ID tracking.
- Motion Forecasting:
  Use Kalman filters or LSTM-based prediction for better continuity during occlusions.
- Model Optimization:
  Deploy the model using ONNX or TensorRT to improve frame-per-second performance for live settings.
- Multi-Camera Cross-View Tracking:
  Extend the pipeline to support multiple viewpoints and synchronize player identities across cameras.

☑️ Conclusion

This project achieved the goal of re-identifying football players in a single video feed, simulating real-time tracking using YOLOv11. By combining fine-tuned detection with a custom tracking pipeline, the system successfully:

- Detected key entities (players, referee, ball)
- Maintained unique identities across the 15-second input clip
- Outputted annotated video and tracking metadata