ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO

ĐỒ ÁN CUỐI KÌ

NHẬP MÔN HỌC MÁY

Lớp: 21KHDL1

Nhóm 7

Giảng viên lí thuyết: TS. Bùi Tiến Lên

Trần Trung Kiên

Bùi Duy Đăng

Thành phố Hồ Chí Minh - 2024

Mục lục

1 Thông tin nhóm

Thành viên nhóm:

MSSV	Họ và tên
21127050	Trần Nguyên Huân
21127076	Doãn Anh Khoa
21127143	Nguyễn Minh Quân
21127240	Nguyễn Phát Đạt

2 Sơ lược về dự án

2.1 Giới thiệu dự án

Ứng dụng web bằng Streamlit được phát triển để phân tích trận đấu bóng đá. Nó sử dụng các kỹ thuật Học sâu và Thị giác máy tính để nhận diện cầu thủ, đánh giá hiệu suất của họ và tạo ra bản đồ chiến thuật của trận đấu.

Mục tiêu của dự án Football Analytics sử dụng kỹ thuật Deep Learning và Computer Vision là cung cấp một công cụ mạnh mẽ để phân tích chiến thuật và hiệu suất trong trận đấu bóng đá. Dưới đây là các mục tiêu cụ thể của dự án:

- **Nhận diện và phân loại cầu thủ**: Dự án nhằm nhận diện cầu thủ, trọng tài và quả bóng từ video trận đấu, sau đó phân loại cầu thủ vào các đội bóng tương ứng.
- **Tạo bản đồ chiến thuật**: Từ các vị trí của cầu thủ và quả bóng trên sân bóng, dự án tạo ra các bản đồ chiến thuật trực quan giúp người dùng hiểu rõ hơn về cơ cấu tấn công và phòng thủ của đội bóng.
- **Theo dõi quả bóng và chiến thuật**: Dự án giúp theo dõi quả bóng qua các khung hình để phân tích các tình huống trong trận đấu và đánh giá hiệu suất của từng cầu thủ và đội bóng.

- **Cung cấp thông tin hữu ích cho quản lý và HLV**: Dự án cung cấp thông tin phân tích chi tiết và đồ thị thống kê về trận đấu, giúp quản lý và HLV đưa ra các quyết định chiến thuật và tập luyện hiệu quả.
- **Tạo trải nghiệm tốt hơn cho người hâm mộ**: Bằng cách cung cấp thông tin phân tích và bản đồ chiến thuật trực quan, dự án giúp tạo ra trải nghiệm thú vị và sâu sắc hơn cho người hâm mộ bóng đá.

2.2 Cách sử dụng

- **Tải lên video trận đấu**: Người dùng tải lên video từ camera chiến thuật của trận đấu bóng đá thông qua giao diện người dùng của ứng dụng web.
- **Chọn tên đội bóng**: Người dùng nhập tên của hai đội bóng tham gia trận đấu vào các trường văn bản tương ứng trên giao diện.
- **Chọn màu áo đội bóng**: Chọn thứ tự khung hình (frame) mà ở đó cầu thủ và thủ môn cả 2 đội đã được nhận diện. Sau đó chọn màu cho bằng cách click vào hình ảnh cầu thủ và thủ môn tương ứng với mỗi đội. Sau đó có thể tùy ý điều các ô màu sao cho phù hợp với màu mỗi đội nhất.
- **Tùy chỉnh các tham số và tùy chọn**: Người dùng có thể tùy chỉnh các ngưỡng tin cậy cho phát hiện cầu thủ và điểm chính trên sân, cũng như các tham số khác như ngưỡng cho quả bóng và đường đi của nó.
- **Bắt đầu phân tích trận đấu**: Sau khi cài đặt hoàn tất, người dùng nhấn nút "Bắt đầu nhận diện" để bắt đầu quá trình phân tích trận đấu.
- **Theo dõi quá trình phân tích**: Trong quá trình phân tích, người dùng có thể theo dõi tiến trình và kết quả trên giao diện của ứng dụng web. Bản đồ chiến thuật, dự đoán đội, và thông tin khác về trận đấu sẽ được hiển thị.
- **Lưu kết quả (tùy chọn)**: Nếu cần, người dùng có thể chọn tùy chọn "Lưu kết quả" để lưu lại video kết quả của quá trình phân tích. Họ có thể nhập tên tệp để lưu và sau đó tải về video đã phân tích từ ứng dụng web.

- **Dừng và kết thúc**: Khi phân tích hoàn tất, người dùng có thể nhấn nút "Dừng nhận diện" để kết thúc quá trình và đóng video. Sau đó, họ có thể tải xuống video kết quả (nếu đã lưu) hoặc tiếp tục phân tích các video khác.

3 Các khái niệm chính về mô hình và ứng dụng

3.1 Giới thiệu sơ lược về YOLOv8

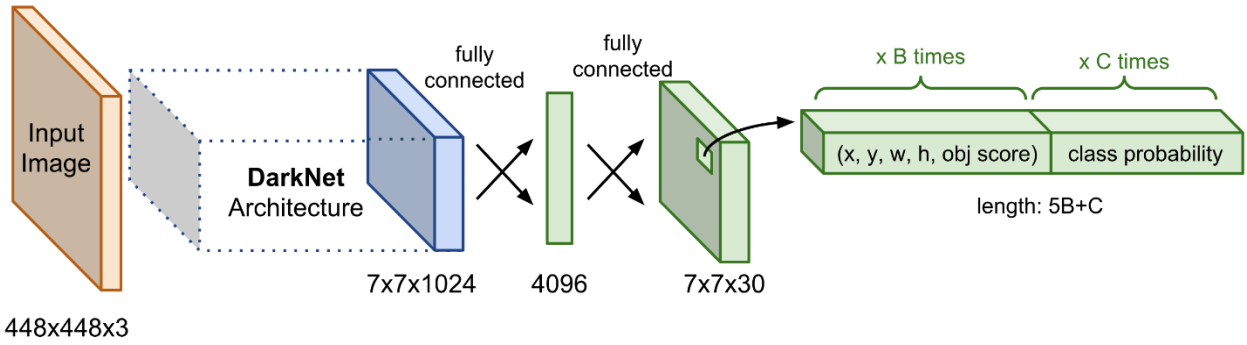
3.1.1 Mạng YOLO là gì?

- YOLO trong bài toán Object Detection có nghĩa là “You only look once”. Tức là chỉ cần nhìn 1 lần là có thể phát hiện ra vật thể. YOLOv8 là một mô hình nhận dạng đối tượng dựa trên mạng convolutional neural network (CNN) được phát triển bởi Joseph Redmon và nhóm nghiên cứu của ông tại Đại học Washington.
- Sử dụng kỹ thuật Attention để cải thiện khả năng nhận dạng đối tượng của mô hình.
- Tích hợp cơ chế tự động điều chỉnh tỷ lệ tăng kích thước của hình ảnh đầu vào (AutoScale).
- Hỗ trợ giám sát bằng video (Video Supervision): Mô hình có khả năng phát hiện và giám sát vật thể trong các video và đưa ra dự đoán liên tục trên toàn bộ video.
- Tích hợp công nghệ Ensemble.
- Tính năng điều chỉnh tỷ lệ tự động (AutoAnchor): Cải thiện việc phát hiện đối tượng với nhiều tỷ lệ khác nhau.
- Thật vậy, về độ chính xác thì YOLO có thể không phải là thuật toán tốt nhất nhưng nó là thuật toán nhanh nhất trong các mô hình Object Detection. Nó có thể đạt được tốc độ gần như real time mà độ chính xác không quá giảm so với các model thuộc top đầu.

3.1.2 Kiến trúc mạng và các thuật toán được sử dụng YOLO

- Lớp đầu vào: Nhận hình ảnh đầu vào.

- CNN Layers: YOLO sử dụng một mạng nơ-ron tích chập (CNN - mạng nơ ron áp dụng các layer Convolutional kết hợp với Maxpooling) để rút trích đặc trưng của hình ảnh. Các lựa chọn phổ biến có thể bao gồm DarkNet, ResNet, hoặc các biến thể của EfficientNet, đặc biệt là các phiên bản cải tiến như EfficientDet. Trong đó DarkNet-53 là kiến trúc được sử dụng nhiều trong YOLOv8.
- Fully Connected Layers: Sau khi qua các lớp CNN, đặc trưng được đưa vào một hoặc một số lớp fully connected layers để tạo ra các dự đoán về đối tượng có mặt trong hình ảnh.
- Detection Layer: Là lớp cuối cùng của mạng, nơi mà các dự đoán về vị trí và class của các đối tượng được tạo ra. Lớp này sử dụng phương pháp thống kê (ví dụ như non-maximum suppression) để loại bỏ các dự đoán trùng lặp và không chính xác.
- Output: YOLO trả về một tensor chứa các dự đoán về vị trí và class của các đối tượng.



Hình 1: Kiến trúc mô hình YOLO [6]

Output của mô hình YOLO là một véc tơ sẽ bao gồm các thành phần:

$$y^T = [p_0, < t_x, t_y, t_w, t_h >, < p_1, p_2, \dots, p_c >]$$

Trong đó:

- p_0 là xác suất dự báo vật thể xuất hiện trong bounding box.
- $< t_x, t_y, t_w, t_h >$ giúp xác định bounding box. Trong đó t_x, t_y là tọa độ tâm và t_w, t_h là kích thước rộng, dài của bounding box.
- $< p_1, p_2, \dots, p_c >$ là vector phân phối xác suất dự báo của các classes.

Hàm lỗi trong YOLO được tính trên việc dự đoán và nhãn mô hình. Loss Function bao gồm 3 phần:

$$L_{total} = L_{classification} + L_{localization} + L_{confidence}$$

Trong đó:

- Độ lỗi của việc dự đoán loại nhãn của object - Classifycation loss

$$L_{classification} = \sum_{i=0}^{S^2} \mathbb{I}_i^{obj} \sum_{c \in Class} (p_i(c) - \hat{p}_i(c))^2$$

  - $\mathbb{I}_i^{obj}$: bằng 1 nếu ô vuông đang xét có đối tượng ngược lại bằng 0.
  - $\hat{p}_i(c)$: xác xuất có điều kiện của lớp c tại ô vuông tương ứng mà mô hình dự đoán

- Độ lỗi của dự đoán tọa độ tâm, chiều dài, rộng của boundary box (x, y ,w, h) - Localization loss

$$L_{localization} = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y})^2 + \left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2]$$

- Độ lỗi của việc dự đoán bounding box đó chứa object so với nhãn thực tế tại ô vuông đó - Confidence loss

$$L_{confidence} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} (\mathbb{I}_{ij}^{obj} + \lambda_{noobj}(1 - \mathbb{I}_{ij}^{obj}))\ (C_{ij} - \hat{C}_{ij})^2$$

  - C_{ij} : điểm tin cậy của ô thứ i = P(contain object) * IoU (predict bbox, ground truth bbox)
  - $\hat{C}_{ij}$: điểm tự tin dự đoán

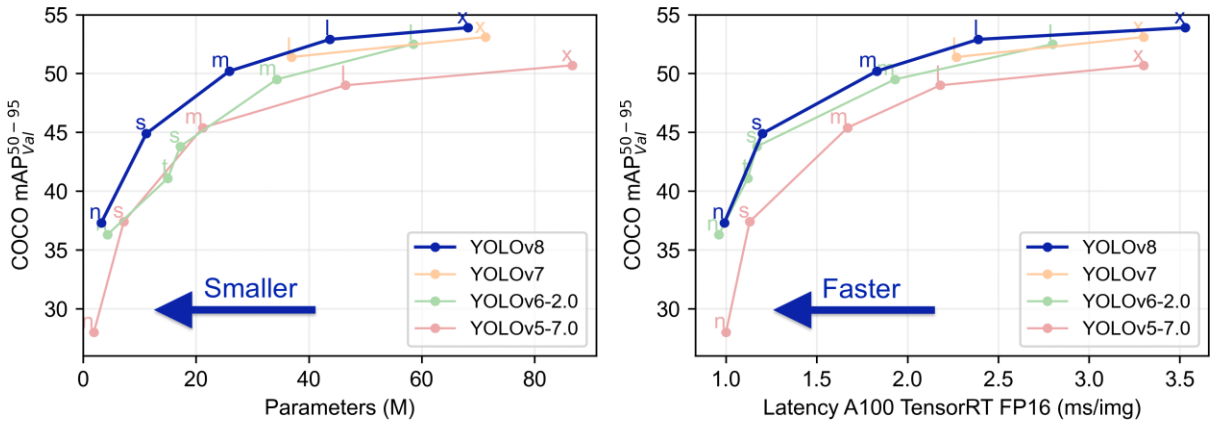
- Các thuật toán tối ưu cho việc huấn luyện tập dữ liệu trong mô hình YOLO: SGD, Adam, AdamW, Nadam, RAdam, RMSProp…
  - Nếu là SGD, learning rate thường là 1e-2

- Nếu là Adam, learning rate thường là 1e-3

- YOLO có sử dụng weight decay (L2 regularization) để tránh overfitting với tham số phạt trọng số mặc định được sử dụng trong mô hình $\lambda = 0.0005$. Ngoài ra, YOLO cũng có thể sử dụng các kỹ thuật để tránh overfitting: data augmentation và dropout (tham số mặc định trong mô hình dropout = 0).

3.1.3 Ưu điểm và hạn chế của YOLOv8

1. **Ưu điểm**
    - **Tốc độ**: YOLOv8 được đánh giá là nhanh chóng và thời gian phản hồi thấp, giúp xử lý các tác vụ nhận diện đối tượng và phân-segment ảnh trong thời gian thực.
    - **Độ chính xác**: YOLOv8 được xây dựng trên các tiến bộ về học sâu và thị giác máy tính, đảm bảo độ chính xác cao trong việc nhận diện đối tượng.
    - **Sự linh hoạt:** YOLOv8 hỗ trợ việc nhận diện đối tượng và phân-segment trên cả GPU và CPU, tận dụng các công nghệ như TensorRT của Nvidia và OpenVino của Intel.

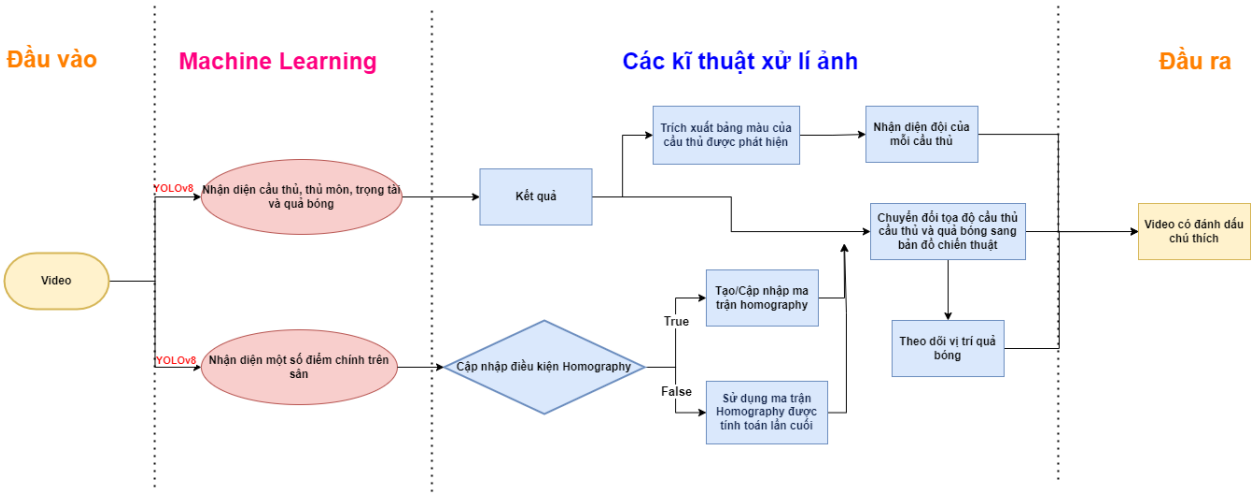


Hình 2: So sánh hiệu suất mô hình [9]

2. **Hạn chế**
    - Cần phải được huấn luyện trên một tập dữ liệu đủ lớn và đa dạng để đạt được hiệu quả cao nhất.

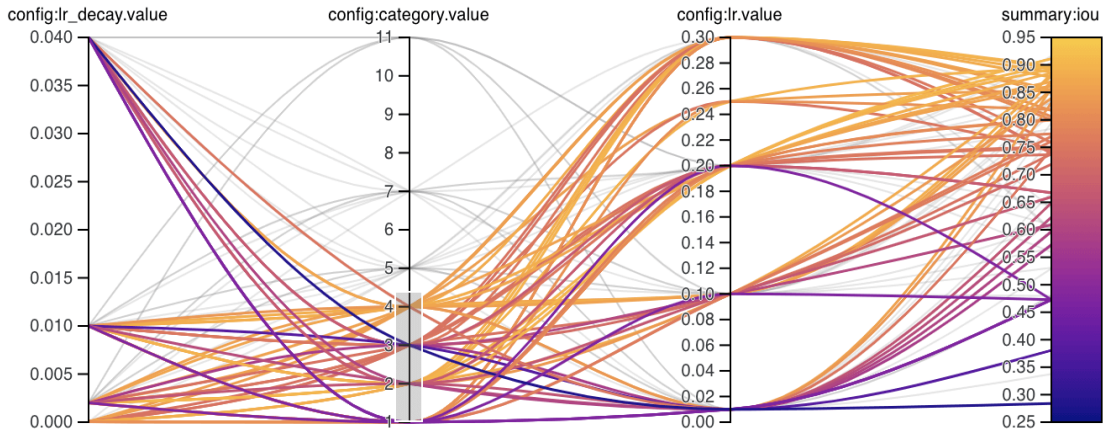
- Yêu cầu các tài nguyên tính toán cao để đạt được tốc độ xử lý nhanh và chính xác.

3.2 Phân tích Flow Chart



Mô tả sơ lược các bước trong Flow Chart:

1. Bắt đầu với video đầu vào: Video đầu vào được chuyển đến hai mô hình nhận diện đối tượng.
2. Phát hiện các đối tượng quan trọng trong video đầu vào: Mục tiêu chính của ứng dụng web này là phát hiện các đối tượng quan trọng trong video đầu vào, bao gồm cầu thủ, trọng tài và quả bóng. Được thực hiện bằng hai mô hình YOLOv8 được huấn luyện trên một tập dữ liệu đã được tinh chỉnh và được gán nhãn thủ công trên Roboflow. Mỗi bộ dữ liệu chứa khoảng 500 ảnh. Chúng tôi đã sử dụng các video góc quay chiến thuật trong bóng đá để tạo ra bộ dữ liệu 500 ảnh này. Những model YOLOv8 này đã được tinh chỉnh bộ tham số để tối ưu kết quả đạt được trên tập dữ liệu tương đối nhỏ.

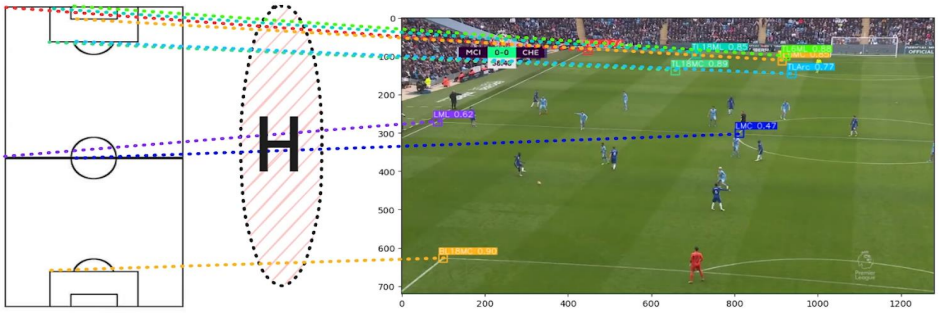


Hình 3: So sánh các siêu tham số khác nhau [7]

3. Kết quả từ mô hình nhận diện cầu thủ, trọng tài và quả bóng được sử dụng để trích xuất bảng màu cho mỗi cầu thủ được phát hiện. Bảng màu được trích xuất này sau đó được tính toán khoảng cách so với màu được định nghĩa ban đầu của mỗi đội được sử dụng để dự đoán đội bóng của cầu thủ được phát hiện.
4. Kết quả từ mô hình phát hiện điểm chính trên sân được sử dụng để tính toán một ma trận homography, là một ma trận ánh xạ vị trí các điểm chính trên bản đồ chiến thuật với vị trí các điểm chính được phát hiện trên khung hình (thuật toán Homography).

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$\mathbf{p'}$ $\mathbf{H}$ $\mathbf{p}$



Hình 4: Mô tả thuật toán Homography [3]

5. Tạo bản đồ chiến thuật: Kết quả từ mô hình phát hiện điểm chính trên sân được sử dụng để tính toán ma trận homography, từ đó chuyển đổi vị trí của cầu thủ trên khung hình thành vị trí trên bản đồ chiến thuật.
6. Dự đoán đội bóng của cầu thủ: Sử dụng một giải thuật đơn giản mà không cần mô hình học máy phức tạp, dự đoán đội bóng của cầu thủ bằng cách so sánh bảng màu trích xuất với màu áo đội bóng được xác định từ trước.
7. Chuyển đổi homography: Chuyển đổi tọa độ của các cầu thủ từ mặt phẳng khung hình sang mặt phẳng bản đồ chiến thuật sử dụng ma trận homography.
8. Theo dõi quả bóng: Theo dõi chuyển động của quả bóng trên bản đồ chiến thuật, với giả định rằng chỉ có một quả bóng trong sân và chỉ cần theo dõi vị trí của nó qua các khung hình.
9. Hiển thị kết quả: Hiển thị khung hình đã được chú thích, với các cầu thủ, quả bóng và các thông tin tương ứng trên bản đồ chiến thuật.

Hình 5: Kết quả sau khi chạy mô hình

3.3 Thuật toán sử dụng để dự đoán đội

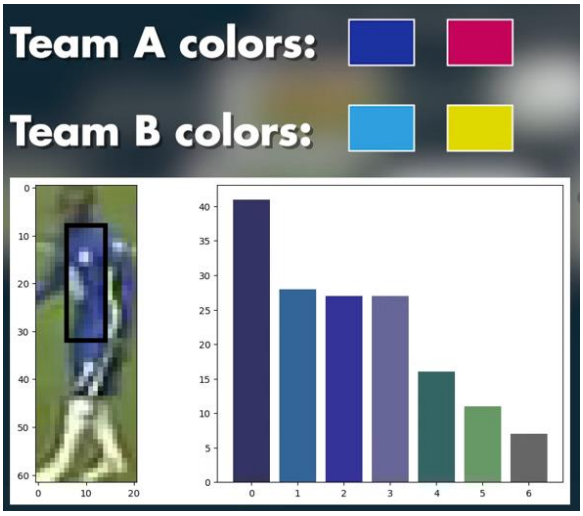
Thuật toán được sử dụng cho việc dự đoán đội bóng là một giải pháp trực tiếp mà không cần mô hình học máy. Bằng cách xác định từ đầu màu sắc của mỗi đội và mỗi lần

chúng ta phát hiện một cầu thủ, chúng ta có thể trích xuất bảng màu của cầu thủ đó và sau đó tính toán khoảng cách giữa các màu này và màu đã được xác định trước của từng đội.

Như vậy, chúng ta có thể gán mỗi cầu thủ vào đội của mình một cách chính xác hơn. Tuy nhiên, vì chúng ta thường có màu xanh lá cây xuất hiện trong hầu hết các hình ảnh của các cầu thủ được phát hiện (màu của sân bóng), chúng ta có thể mong đợi rằng màu xanh này sẽ luôn hiện diện trong bảng màu chúng ta trích xuất, điều này không hữu ích cho việc dự đoán đội bóng.

Đó là lý do tại sao chúng tôi quyết định sử dụng một hình chữ nhật nhỏ hơn ở trung tâm để trích xuất bảng màu, điều này thực sự cải thiện đáng kể độ chính xác của thuật toán dự đoán đội bóng.

Lưu ý: Khoảng cách giữa bảng màu trích xuất và các màu đã được xác định trước được tính toán trong không gian chứ không phải là không gian RGB, điều này bởi vì được báo cáo rằng các khoảng cách trong không gian này ý nghĩa hơn và gần với cảm nhận của mắt người.

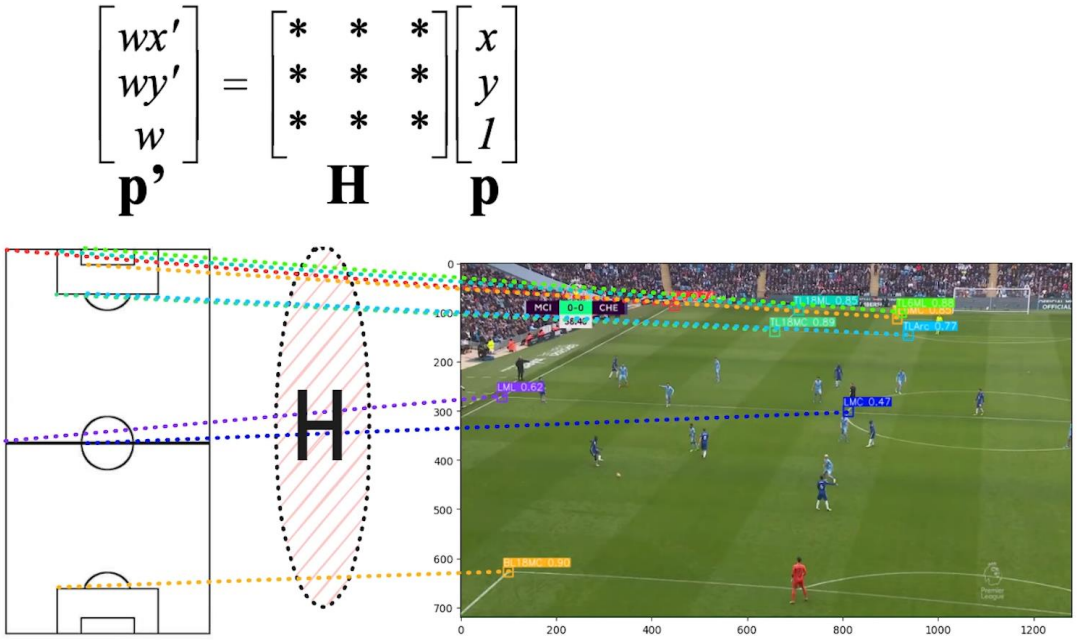


Hình 6: Trích xuất màu chủ đạo của cầu thủ [3]

Hình 7: Tỉ lệ màu [3]

3.4 Thuật toán sử dụng để trực quan trên bản đồ chiến thuật (phép biến đổi Homography)

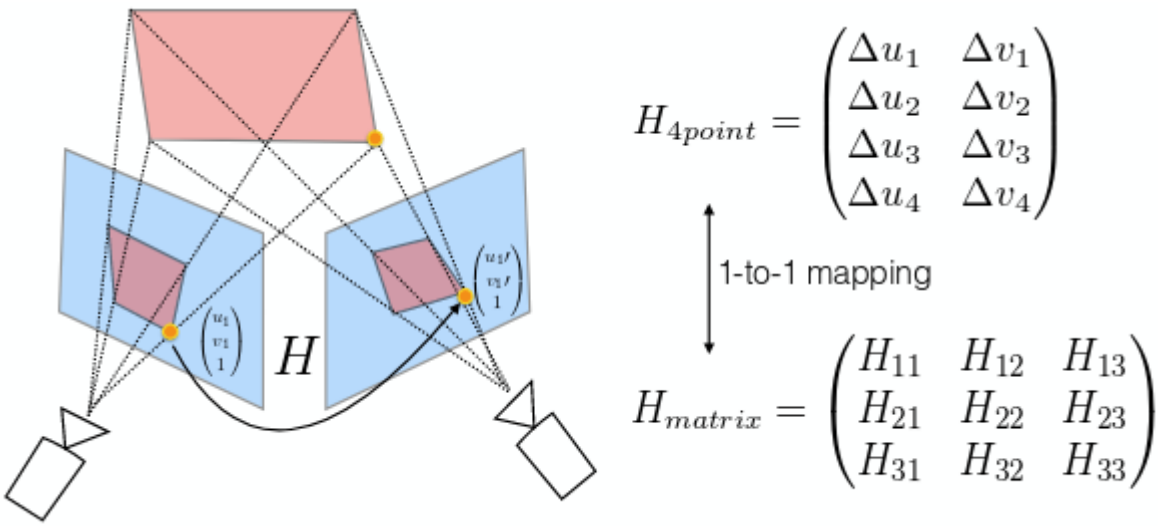
Phép biến đổi homography, là phép biến đổi tọa độ của các cầu thủ được phát hiện từ mặt phẳng khung hình sang mặt phẳng bản đồ chiến thuật. Xem hình dưới:



Hình 8: Mô tả thuật toán Homography [3]

Để thực hiện phép biến đổi này, chúng ta sẽ sử dụng ma trận homography, một ma trận được sử dụng phổ biến để thay đổi góc nhìn. Như đã minh họa trong hình ảnh này, chúng ta có thể sử dụng nó để nhìn vào cùng một cảnh từ các góc nhìn khác nhau và phép biến đổi từ một góc nhìn này sang góc nhìn khác được thực hiện thông qua ma trận homography.

Để thực hiện điều đó, chúng ta cần ánh xạ ít nhất bốn điểm từ góc nhìn đầu tiên sang góc nhìn thứ hai. Trong trường hợp của chúng ta, chúng ta có một bản đồ chiến thuật trong đó chúng ta xác định một số lượng nhất định các điểm chính và lưu lại tọa độ của chúng, bây giờ chúng ta phát hiện ít nhất bốn điểm chính trên bất kỳ khung hình nào, chúng ta có thể tính toán một ma trận homography.



Hình 9: Mô tả thuật toán Homography [10]

Qua chạy thực nghiệm, một điều tôi nhận thấy là việc ánh xạ bốn điểm chính hoàn toàn có thể vẽ bản đồ chiến thuật, chúng ta có thể có độ chính xác 100% của việc di chuyển góc nhìn sử dụng phép biến đổi homography, đặc biệt là ở gần các điểm này.

Tuy nhiên, khi chúng ta tăng số lượng điểm chính, độ chính xác sẽ giảm và điều này có thể do một sự biến dạng nhỏ trong quá trình ghi hình của camera hoặc trong trường hợp các điểm không nằm trong cùng một mặt phẳng 3D.

Sai số với nhiều điểm trên sân từ phép biến đổi homography không quá lớn nên chúng ta có thể tiến hành phân tích nhưng vấn đề chính là sự chênh lệch nhẹ của các tọa độ trên bản đồ chiến thuật, điều này chủ yếu là do chúng ta đang tính toán ma trận homography trong mỗi khung hình nơi chúng ta phát hiện nhiều hơn bốn điểm chính và đó là lý do nên cập nhập một điều kiện cập nhật ma trận homography được sử dụng để xác định trong mỗi khung hình xem có cần cập nhật ma trận homography hay không hoặc sử dụng ma trận homography cuối cùng và điều này rõ ràng sẽ giảm chi phí tính toán và làm tăng chút ít tốc độ khung hình và cũng làm cho biểu diễn bản đồ chiến thuật mượt mà hơn nhiều.

Điều kiện mà chúng tôi đặt là một điều khá đơn giản, trong mỗi khung hình nếu chúng ta phát hiện nhiều hơn một số điểm chính nhất định đã được phát hiện trong khung hình trước đó, chúng ta tính toán xem những điểm phát hiện trên sân đó chung đó đã di chuyển trung bình vượt quá một ngưỡng nhất định hay không và nếu đúng, chúng ta cần cập nhật ma trận homography và nếu không, chúng ta sẽ tiếp tục sử dụng ma trận homography được tính toán cuối cùng. Số lượng điểm trên sân cũng như ngưỡng khoảng cách là các siêu tham số cần được thiết lập cẩn thận.

Theo dõi di chuyển của quả bóng: Khi chúng ta có được tọa độ của cầu thủ và quả bóng trên bản đồ chiến thuật, chúng ta có thể dễ dang theo dõi sự chuyển động của qua bóng vì chúng ta chỉ có một quả bóng hiện diện trên sân.

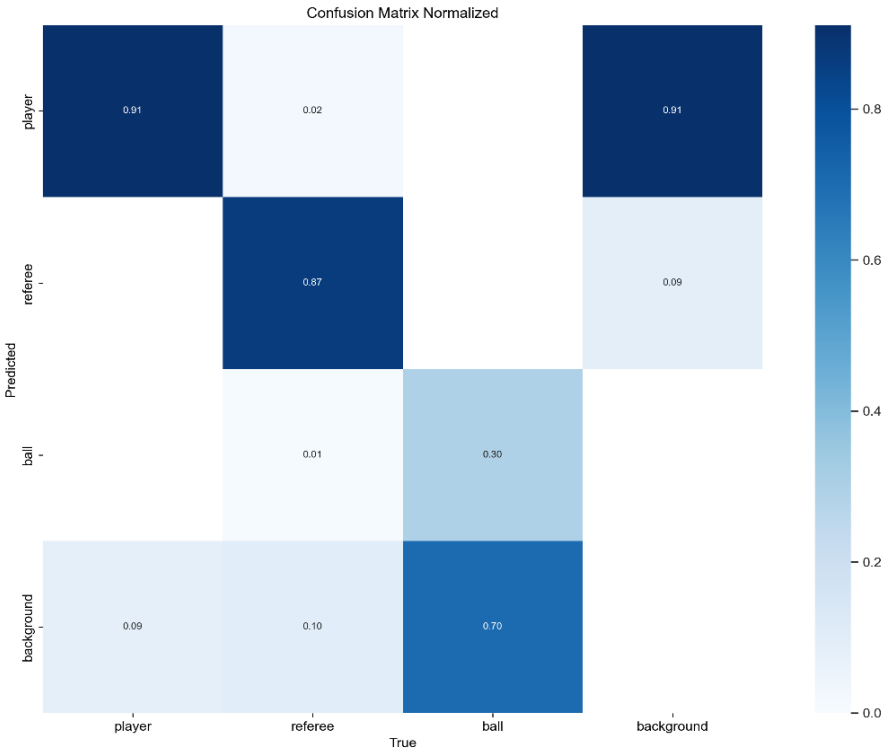
4 Huấn luyện mô hình và khó khăn gặp phải

4.1 Sơ lược về tập dữ liệu và mô hình sử dụng

- 2 mô hình: Pre-train YOLOv8, đã được tinh chỉnh các siêu tham số với bộ dữ liệu trên Roboflow.
    - Nhận diện cầu thủ, trọng tài, và quả bóng.
    - Nhận diện một số điểm chính trên sân.
- Bộ dữ liệu gồm các ảnh được trích ra từ video này.
    - Classes: Cầu thủ, trọng tài, quả bóng, và các keypoints trên sân

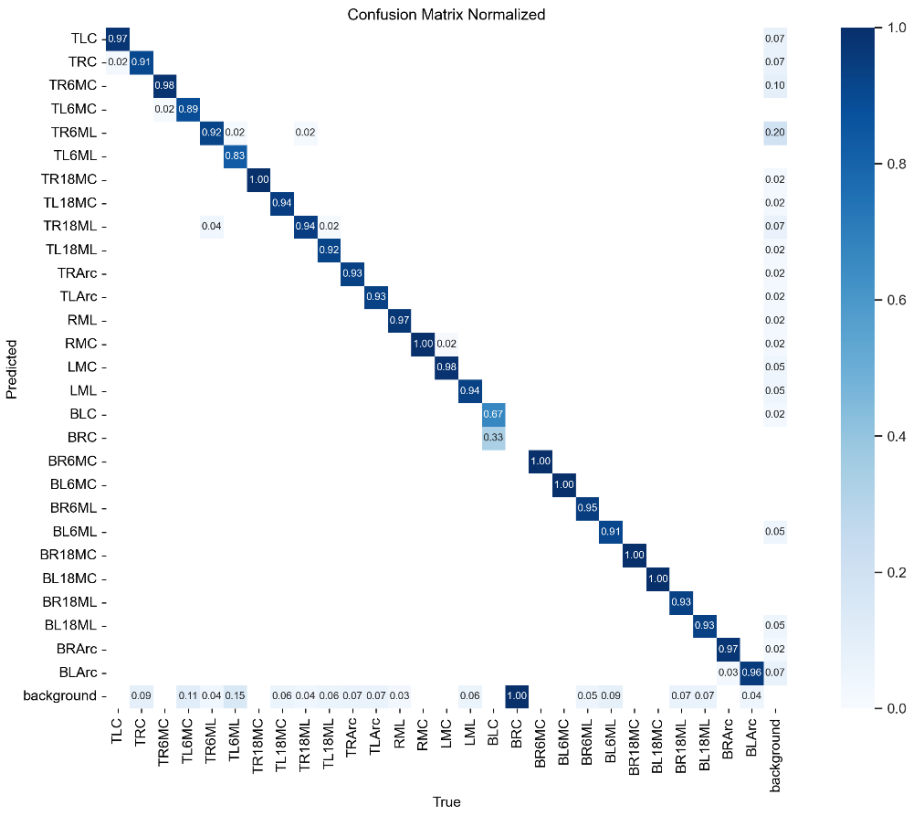
4.2 Khám phá hiệu suất mô hình và phân bổ kết quả của các Class

Đầu tiên chúng ta hãy đánh giá hiệu suất mô hình đầu tiên (nhận diện cầu thủ, trọng tài và quả bóng) thông qua confusion matrix đã được chuẩn hóa.



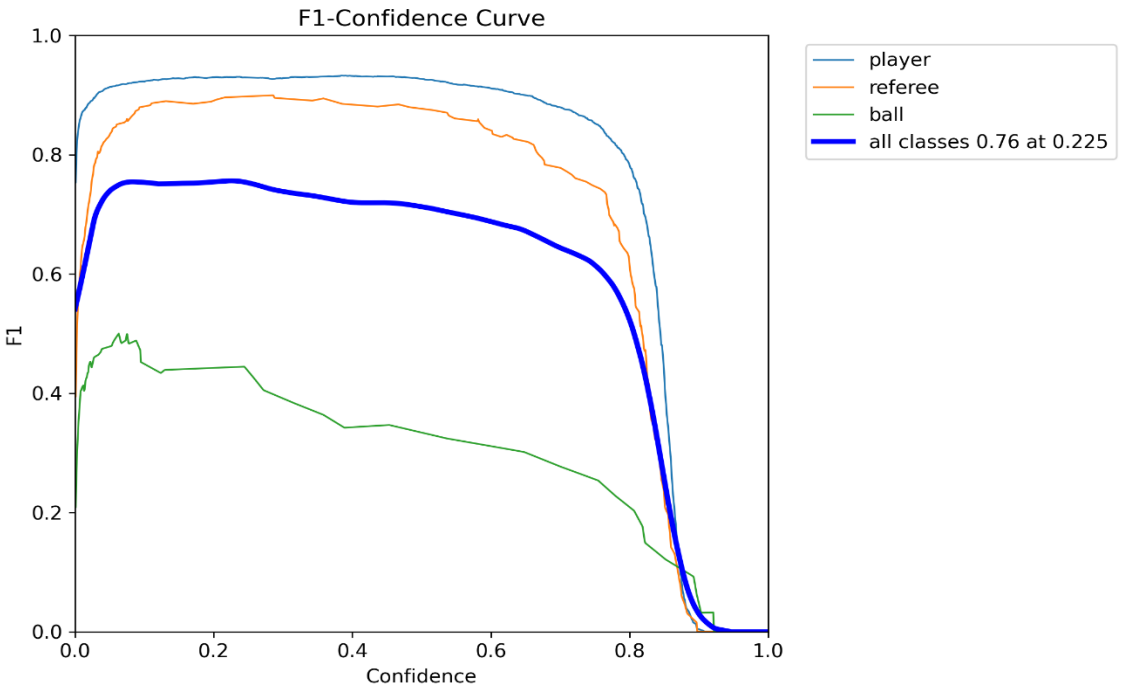
- **Nhận xét:**
  - Chúng ta có thể thấy thông qua confusion matrix của mô hình thứ nhất kết quả tương đối tốt cho tất cả các lớp khác nhau.
  - Ngoại trừ lớp bóng nơi chỉ có 25% của số lượng bóng được dự đoán có mặt trong tập validation dẫn đến chỉ số Recall cho lớp bóng thấp
  - Điều này cũng khá dễ hiểu vì dữ liệu về lớp bóng bị thiếu khá nhiều. Đây là vấn đề của mất cân bằng dữ liệu.

Tương tự đánh giá cho mô hình thứ hai thông qua confusion matrix



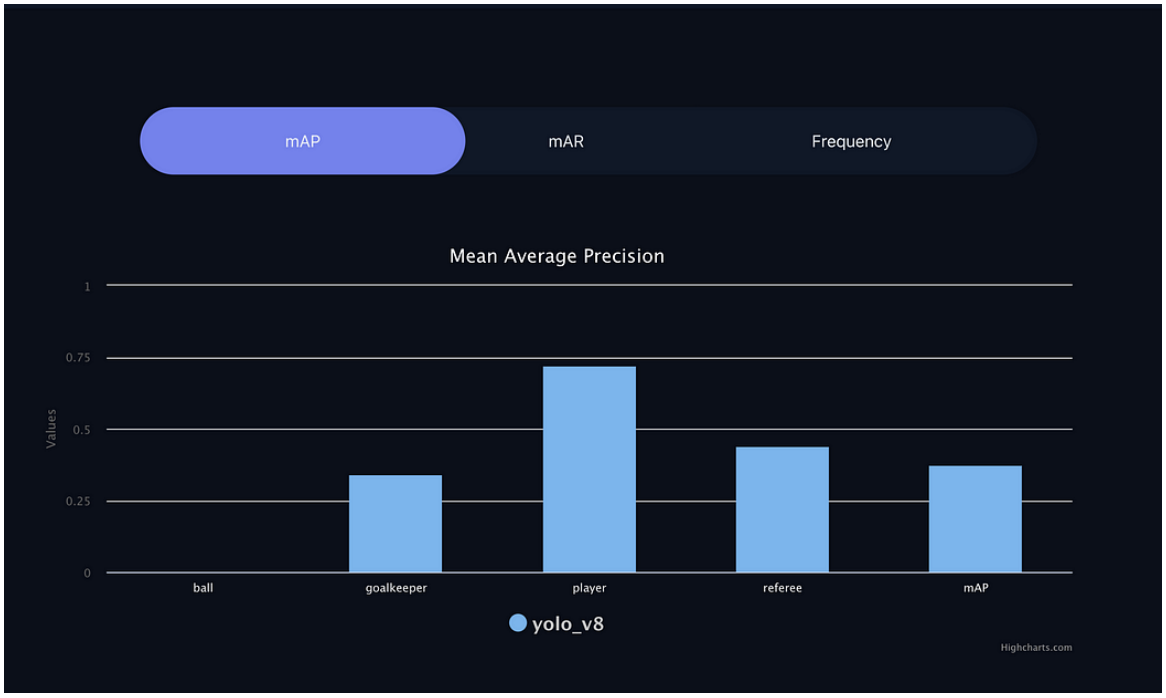
- **Nhận xét:**
    - Đối với confusion matrix của mô hình 2 ta có thể thấy kết quả tương đối tốt trên hầu hết các điểm.
    - Tuy vậy vẫn có chút sai lệch nhỏ nhưng không quá đáng kể. Mô hình tương đối tốt để có thể sử dụng.

Tiếp đến chúng ta hãy phân thêm về đường cong F1-Confidence để hiểu thêm vấn đề của mô hình thứ nhất.



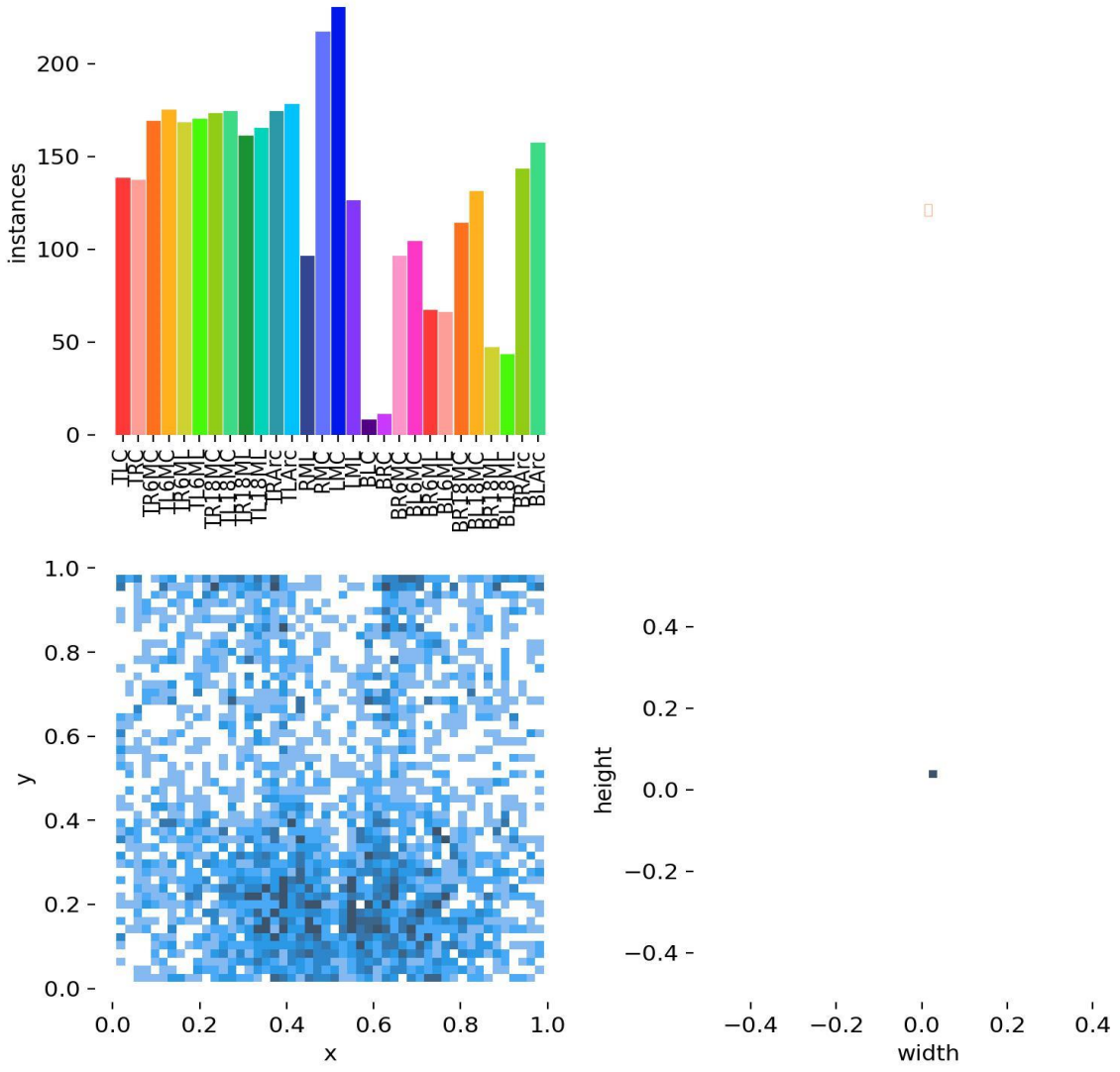
- **Nhận xét:**
  - Qua đường cong F1-Confidence biểu diễn điểm F1 so với mức độ tự tin của các lớp khác nhau.
  - Ta có thể thấy lớp “Player” và lớp “Referee” có điểm F1 cao và ổn định qua các mức độ tự tin.
  - Lớp “Ball” có điểm F1 thấp và biến đổi đáng kể qua các mức độ tự tin khác nhau.

Map bằng 0 cho Lớp Ball có thể có hai ý nghĩa: Nó gợi ý rằng lớp đó đã bị bỏ sót hoàn toàn (False Negatives) hoặc tất cả các phát hiện đều không chính xác (False Positives).



Thứ hai, chúng tôi phân tích sự phân bố đối tượng theo từng lớp trong dữ liệu của chúng tôi. Hình dưới đây minh họa rằng Lớp Ball có 273 đối tượng được chú thích, tuy nhiên các đối tượng của lớp này không được dự đoán.

Tiếp theo, hãy thử phân tích phân bổ của các điểm chính trên sân. Qua đó thấy được kết quả tương đối tốt ở hầu hết các lớp chỉ có 2 lớp BLC và BRC bị thiếu khá nhiều nhưng điều này không ảnh hưởng quá nhiều đến kết quả vì chúng ta có tới tổng cộng 28 điểm trên sân đôi lúc sai 1 tới 2 điểm thì lúc tính toán homography matrix vẫn có thể ánh xạ để trực quan lên bản đồ chiến thuật tương đối tốt (vì chỉ cần 4 điểm thì đã có thể vẽ được bản đồ trực quan).



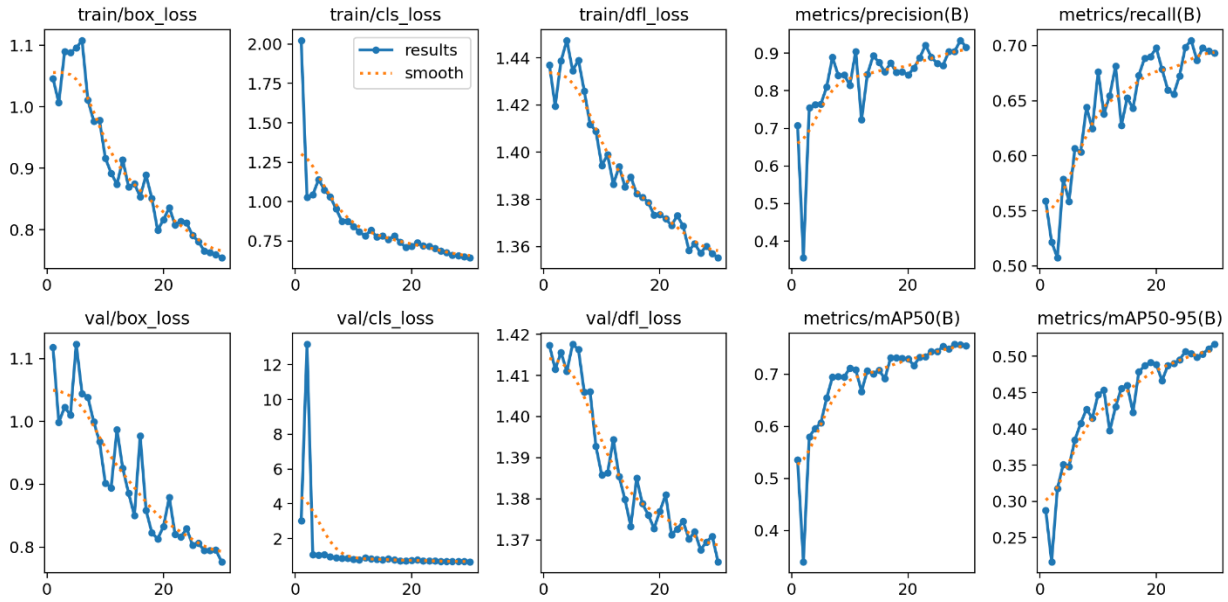
4.3 Hướng khắc phục

Vấn đề này có thể được giải quyết bằng cách sử dụng tập dữ liệu lớn hơn hoặc thông qua các kỹ thuật tăng cường dữ liệu, nhưng đối với thời điểm hiện tại vì lí do thời gian có hạn nên chúng tôi tạm hài lòng với kết quả của các mô hình hiện tại.

Dưới đây là một số cách được chuyên gia khuyến nghị để cải thiện:

- ✓ Ảnh mỗi lớp nên lớn hơn hoặc bằng 1500 ảnh mỗi lớp.
- ✓ Thể hiện mỗi lớp nên lớn hơn hoặc bằng 100000 trường hợp (đối tượng đã được gán nhãn) mỗi lớp.
- ✓ Đa dạng ảnh: Đối với các trường hợp sử dụng thực tế, nên sử dụng các ảnh từ các thời điểm khác nhau trong ngày, các mùa khác nhau, thời tiết khác nhau, ánh sáng khác nhau, góc nhìn khác nhau, nguồn khác nhau (lấy từ trực tuyến, thu thập tại địa phương, các máy ảnh khác nhau) v.v.
- ✓ Nhãn nhất quán: Tất cả các thể hiện của tất cả các lớp trong tất cả các ảnh phải được gán nhãn. Gán nhãn chỉ một phần đối tượng có thể mô hình sẽ không hoạt động tốt.
- ✓ Độ chính xác nhãn: Nhãn phải được gán sát với mỗi đối tượng. Không có khoảng trống nào nên tồn tại giữa một đối tượng và khung giới hạn của nó. Không nên có đối tượng nào thiếu nhãn.
- ✓ Xác minh nhãn: Xem train_batch*.jpg khi bắt đầu huấn luyện để xác minh nhãn của bạn xuất hiện chính xác.
- ✓ Hình ảnh nền: Nên huấn luyện thêm dữ liệu có hình ảnh nền - các hình ảnh không có đối tượng được nên được thêm vào để giảm False Positives (FP). Nên sử dụng khoảng 0 - 10% hình ảnh nền để giúp giảm False Positives (COCO có 1000 hình ảnh nền để tham khảo, 1% tổng số). Lưu ý: Không cần nhãn cho hình ảnh nền.

4.4 Đánh giá chung về kết quả mô hình



Hình 10: Kết quả huấn luyện [3]

- Nhận xét:
    - Qua biểu đồ huấn luyện của mô hình có thể thấy mô hình có vẻ đã cải thiện hiệu suất theo thời gian trong cả giai đoạn huấn luyện và xác thực:
    - Box_loss và idf_loss giảm theo thời gian trong cả hai giai đoạn, cho thấy mô hình đang học và cải thiện theo dữ liệu khá tốt.
    - Độ chính xác (precision), độ nhớ (recall) và mAP50 tăng lên, cho thấy mô hình đang cải thiện khả năng dự đoán của nó
- Nhìn chung với kết quả trên, mô hình có thể chạy thực nghiệm tương đối tốt nên chúng ta có thể áp dụng mô hình vào ứng dụng để tiến hành phân tích.

5 Nguồn tham khảo

[1] Mô hình nhận dạng vật thể YOLOv8: https://hackmd.io/@58ZC49ZfS86wYX--LRGGOg/Viewperm

[2] Tìm hiểu về YOLO trong bài toán real time object detection: https://viblo.asia/p/tim-hieu-ve-yolo-trong-bai-toan-real-time-object-detection-

[3] Football-Analytics-with-Deep-Learning-and-Computer-Vision (Hmzbo): https://github.com/Hmzbo/Football-Analytics-with-Deep-Learning-and-Computer-Vision/tree/master

[4] Sports Analytics — A Data-Centric Approach to Computer Vision: https://medium.com/@tenyks_blogger/sports-analytics-a-data-centric-approach-to-computer-vision

[5] Train YOLOv8 Object Detection on a Custom Dataset: https://blog.roboflow.com/how-to-train-yolov8-on-a-custom-dataset/

[6] Kiến trục mạng YOLO: https://phamdinhkhanh.github.io/2020/03/09/DarknetAlgorithm

[7] Cấu hình trong mô hình YOLO: https://docs.ultralytics.com/vi/guides/hyperparameter-tuning/

[8] Maximizing Object Detection: https://keylabs.ai/blog/maximizing-object-detection-yolov8-performance

[9] Hiệu suất của các mô hình YOLO: https://pytorch.org/hub/ultralytics_yolov5/

[10] Biến đổi ma trận homography: https://www.mdpi.com/2079-9292/12/23/4738#B22-electronics-12-04738