BÁO CÁO NGHIÊN CỨU XÂY DỰNG MÔ HÌNH PHÁT HIỆN CHÁY

• 1. Giới thiệu (Introduction)

- **Bối cảnh vấn đề**:
  Việc nhận diện cháy đóng vai trò vô cùng quan trọng trong các hoạt động liên quan đến an toàn công cộng, quản lý môi trường và phòng chống cháy nổ. Cháy có thể gây ra thiệt hại lớn về tài sản, môi trường và thậm chí cả tính mạng con người. Việc phát hiện sớm các đám cháy giúp giảm thiểu nguy cơ lan rộng và tăng cường khả năng ứng phó kịp thời.
- **Ứng dụng cụ thể**:
  Trong phòng cháy chữa cháy, việc nhận diện cháy sớm thông qua các hệ thống camera và cảm biến giúp tăng cường hiệu quả trong việc phát hiện và dập tắt đám cháy, giảm thiểu thiệt hại.
- **Vai trò của AI và Machine Learning**:
  AI và Machine Learning đã và đang cách mạng hóa các lĩnh vực liên quan đến phát hiện và nhận diện tự động. Đặc biệt trong bài toán nhận diện cháy, AI có thể sử dụng các thuật toán học sâu (Deep Learning) để phân tích hình ảnh hoặc dữ liệu cảm biến nhằm phát hiện ra các tình huống bất thường. Những mô hình học máy này có thể học từ dữ liệu lớn, cho phép chúng đưa ra dự đoán chính xác và kịp thời hơn so với các phương pháp truyền thống.
- **Mục tiêu báo cáo**:
  Báo cáo này nhằm nghiên cứu và đánh giá các phương pháp nhận diện cháy thông qua việc ứng dụng AI và Machine Learning. Cụ thể, báo cáo sẽ trình bày các thuật toán hiện có, so sánh hiệu suất của các mô hình, và đưa ra những đề xuất nhằm tối ưu hóa khả năng phát hiện cháy trong các hệ thống tự động.

• 2. Cơ sở lý thuyết (Literature Review)

- **Nhận diện cháy**:
  Phương pháp truyền thống để nhận diện cháy chủ yếu dựa trên các hệ thống cảm biến nhiệt, khói, hoặc các phương tiện phát hiện hồng ngoại. Tuy nhiên, các phương pháp này thường phản ứng khi đám cháy đã bùng phát và có thể có độ trễ trong việc phát hiện sớm. Với sự phát triển của công nghệ xử lý hình ảnh và AI, các hệ thống nhận diện cháy dựa trên hình ảnh và video đã ra đời, giúp nhận diện đám cháy sớm hơn nhờ phân tích các yếu tố như khói, lửa, và sự thay đổi nhiệt độ trong không gian. Các mô hình học máy như Convolutional Neural Networks (CNN) được áp dụng để phát hiện các đặc trưng hình ảnh liên quan đến cháy, giúp tăng cường khả năng nhận diện tự động và chính xác.

- **Các thuật toán Machine Learning**:
  Trong lĩnh vực nhận diện hình ảnh, một số thuật toán học máy phổ biến như Convolutional Neural Networks (CNN), Support Vector Machines (SVM), và Random Forest được sử dụng rộng rãi. CNN là một phương pháp học sâu mạnh mẽ trong việc trích xuất đặc trưng hình ảnh và nhận diện đối tượng trong ảnh, nhờ vào khả năng học tự động từ dữ liệu đầu vào mà không cần phải lập trình các đặc trưng thủ công. SVM và Random Forest cũng là các phương pháp học máy truyền thống hiệu quả trong việc phân loại hình ảnh, tuy nhiên chúng yêu cầu các kỹ thuật xử lý ảnh thủ công để trích xuất các đặc trưng trước khi thực hiện phân loại.
- **Công nghệ liên quan**:
  Xử lý hình ảnh và tín hiệu đóng vai trò nền tảng trong các hệ thống nhận diện cháy và rác. Ngoài ra, sự phát triển của IoT (Internet of Things) cũng tạo điều kiện cho việc thu thập dữ liệu từ các cảm biến và camera đặt tại nhiều vị trí khác nhau, giúp nâng cao khả năng giám sát và phát hiện tự động. Những công nghệ này kết hợp với AI và Machine Learning tạo thành một hệ thống phát hiện thông minh, có khả năng tự động hóa và hoạt động liên tục, không cần sự can thiệp của con người.

3. Phương pháp nghiên cứu (Methodology)

3.1 Thu thập dữ liệu:
Dữ liệu phục vụ cho việc nhận diện cháy được thu thập từ nhiều nguồn khác nhau, nhằm đảm bảo tính đa dạng và phong phú của bộ dữ liệu. Các nền tảng nổi bật như Kaggle, Roboflow, Huggingface, Paperwithcode, ScienceDB, và Mivia cung cấp các bộ dữ liệu hình ảnh và video đã được gắn thẻ (labeled) phục vụ cho việc huấn luyện các mô hình AI. Cụ thể:

- **Kaggle**: Là một nguồn cung cấp bộ dữ liệu lớn với nhiều bộ dữ liệu về hình ảnh cháy được cộng đồng chia sẻ.
- **Roboflow**: Tập trung vào dữ liệu hình ảnh, cung cấp nhiều bộ dữ liệu về phân loại và phát hiện vật thể, phù hợp với bài toán nhận diện cháy.
- **Huggingface và Paperwithcode**: Là các nền tảng cung cấp không chỉ bộ dữ liệu mà còn các mô hình Machine Learning đã được huấn luyện trước (pre-trained models).
- **Mivia**: Cung cấp các bộ dữ liệu video về các tình huống cháy trong môi trường thực tế.
- **ScienceDB**: Là một cơ sở dữ liệu khoa học, cung cấp dữ liệu được thu thập và nghiên cứu kỹ lưỡng, bao gồm cả dữ liệu về môi trường và cháy rừng.

3.2 Xử lý dữ liệu:
Trước khi sử dụng dữ liệu cho việc huấn luyện mô hình, các bước tiền xử lý được thực hiện để đảm bảo dữ liệu chất lượng:

- **Lọc nhiễu**: Các ảnh hoặc video cháy bị mờ, nhiễu hoặc chứa dữ liệu không liên quan sẽ được loại bỏ hoặc làm sạch. Điều này giúp mô hình tập trung vào các đặc trưng quan trọng.
- **Lọc các nhãn không cần thiết:** Các bộ dữ liệu thu thập từ nhiều nguồn khác nhau có thể chứa nhiều lớp nhãn như: fire, smoke, solid, explosion, other,.... Mục tiêu là lọc các lớp không cần thiết và chỉ giữ lại lớp "fire."
- **Gán nhãn dữ liệu:** Một số bộ dữ liệu chỉ chứa ảnh về cháy mà không có nhãn sẵn, do đó cần thực hiện quá trình gán nhãn dữ liệu để sử dụng cho việc huấn luyện mô hình.
- **Tăng cường dữ liệu**: Kỹ thuật tăng cường dữ liệu (data augmentation) như xoay, thay đổi độ sáng, cắt ảnh, hoặc lật ảnh được sử dụng để tạo ra thêm nhiều biến thể của dữ liệu, giúp mô hình có khả năng tổng quát tốt hơn trên dữ liệu mới.
- **Tổng hợp bộ dữ liệu:** Bộ dữ liệu sẽ được tổng hợp từ nhiều nguồn và được hợp nhất thành một bộ dữ liệu hoàn chỉnh.

- **Tạo bộ dữ liệu:** Bộ dữ liệu có hơn 14.000 ảnh về cháy sẽ được chia thành ba phần: train, valid, và test với tỷ lệ 7:2:1. Mỗi phần sẽ gồm hai thư mục: image và label. Phần nhãn sẽ được biến đổi phù hợp với từng mô hình. Đối với mô hình YOLO, sẽ có một file `data.yaml` tương ứng.

3.3 Xây dựng mô hình:

a. **Tổng quan về model**
Ba mô hình chính được sử dụng trong nghiên cứu này bao gồm YOLOv10 và YOLOv11 Đây là các mô hình mạnh mẽ trong việc phát hiện đối tượng (object detection) với khả năng nhận diện các đặc trưng của cháy từ dữ liệu hình ảnh:

- **YOLOv10 và YOLOv11**: Đây là các phiên bản cải tiến của mô hình YOLO (You Only Look Once), nổi tiếng với tốc độ xử lý nhanh và khả năng phát hiện đối tượng trong thời gian thực. Cả hai mô hình đều có cấu trúc mạng CNN được thiết kế tối ưu để phát hiện các đối tượng nhỏ và lớn trong hình ảnh.

  Sử dụng mô hình **YOLO** để thực hiện nhiệm vụ phát hiện cháy trong ảnh. Nó kế thừa từ lớp `BaseTrainer` và được mở rộng để xây dựng quy trình huấn luyện mô hình phát hiện đối tượng (Object Detection). Trong quá trình này, dữ liệu được xử lý, mô hình được khởi tạo, và các chức năng huấn luyện, đánh giá, cũng như trực quan hóa kết quả được cung cấp.

b. **Các thành phần chính**

Thư viện sử dụng:

- `torch`: sử dụng cho deep learning, đặc biệt là `torch.nn` cho xây dựng và huấn luyện các mô hình mạng nơ-ron.
- `numpy`: để xử lý các mảng dữ liệu.
- `ultralytics`: Đây là một framework hỗ trợ YOLO để xây dựng và huấn luyện mô hình phát hiện đối tượng.

Class `DetectionTrainer`: Đây là lớp chính chịu trách nhiệm cho quá trình huấn luyện mô hình phát hiện đối tượng. Các phương thức quan trọng bao gồm:

- `build_dataset`: Xây dựng dữ liệu YOLO từ đường dẫn tới tập dữ liệu hình ảnh.
- `get_dataloader`: Trả về `dataloader` từ tập dữ liệu, giúp chia nhỏ dữ liệu thành các batch để huấn luyện và đánh giá.
- `preprocess_batch`: Tiền xử lý các batch dữ liệu, bao gồm việc chuyển đổi kích thước hình ảnh và chuẩn hóa chúng.
- `set_model_attributes`: Cấu hình các thuộc tính của mô hình như số lớp, tên các lớp và siêu tham số.
- `get_model`: Khởi tạo mô hình YOLO cho quá trình huấn luyện.
- `get_validator`: Trả về đối tượng `DetectionValidator` dùng để đánh giá mô hình.

- `label_loss_items`: Gán nhãn cho các giá trị tổn thất (loss) trong quá trình huấn luyện.
- `plot_training_samples`: Trực quan hóa mẫu huấn luyện với các chú thích (annotations).
- `plot_metrics`: Vẽ đồ thị các chỉ số hiệu suất (performance metrics) từ quá trình huấn luyện.
- `plot_training_labels`: Trực quan hóa các nhãn trong quá trình huấn luyện.

c. Quy trình huấn luyện mô hình phát hiện đối tượng cháy

Để huấn luyện mô hình phát hiện đối tượng cháy, các bước cần thực hiện bao gồm:

- **Bước 1: Chuẩn bị dữ liệu**
  - Tập dữ liệu bao gồm hình ảnh và các chú thích tương ứng (bounding boxes) được lưu trong một tệp định dạng YOLO.
  - Các hình ảnh chứa đối tượng cần phát hiện (lửa) phải được chú thích đầy đủ với nhãn và tọa độ hộp giới hạn.
- **Bước 2: Xây dựng tập dữ liệu**
  - Sử dụng phương thức `build_dataset`, dữ liệu hình ảnh được nạp vào từ đường dẫn chỉ định.
  - Phương thức này sẽ xác định xem dữ liệu đang ở chế độ huấn luyện (`train`) hay đánh giá (`val`) và thực hiện các augmentations (biến đổi dữ liệu) phù hợp.
- **Bước 3: Khởi tạo `Dataloader`**
  - `get_dataloader` sẽ tạo ra các dataloader để xử lý và nạp dữ liệu trong quá trình huấn luyện và đánh giá.
  - Với các batch hình ảnh được tạo ra, các hình ảnh sẽ được tiền xử lý bởi `preprocess_batch` như chuẩn hóa (chia giá trị pixel cho 255) và thay đổi kích thước phù hợp với siêu tham số (hyperparameters) của mô hình.
- **Bước 4: Thiết lập và khởi tạo mô hình**
  - Phương thức `get_model` khởi tạo mô hình YOLO dựa trên các cấu hình (`cfg`) và tải các trọng số (weights) nếu cần.
  - Mô hình cũng được gán các thuộc tính như số lượng lớp (`nc`) và tên các lớp (`names`).
- **Bước 5: Quá trình huấn luyện**
  - Trong quá trình huấn luyện, loss function sẽ được tính toán để đo lường độ sai lệch của dự đoán mô hình so với nhãn thực tế.
  - Các giá trị loss bao gồm `box_loss` (tổn thất hộp giới hạn), `cls_loss` (tổn thất phân loại), và `dfl_loss` (tổn thất phân phối phân cấp).
- **Bước 6: Đánh giá và trực quan hóa**
  - Sau khi huấn luyện, mô hình được đánh giá trên tập dữ liệu kiểm tra (validation). `get_validator` trả về validator để tính toán các chỉ số hiệu suất như mAP (mean Average Precision).

- Kết quả huấn luyện và các nhãn được trực quan hóa qua các phương thức `plot_training_samples`, `plot_metrics` và `plot_training_labels`.

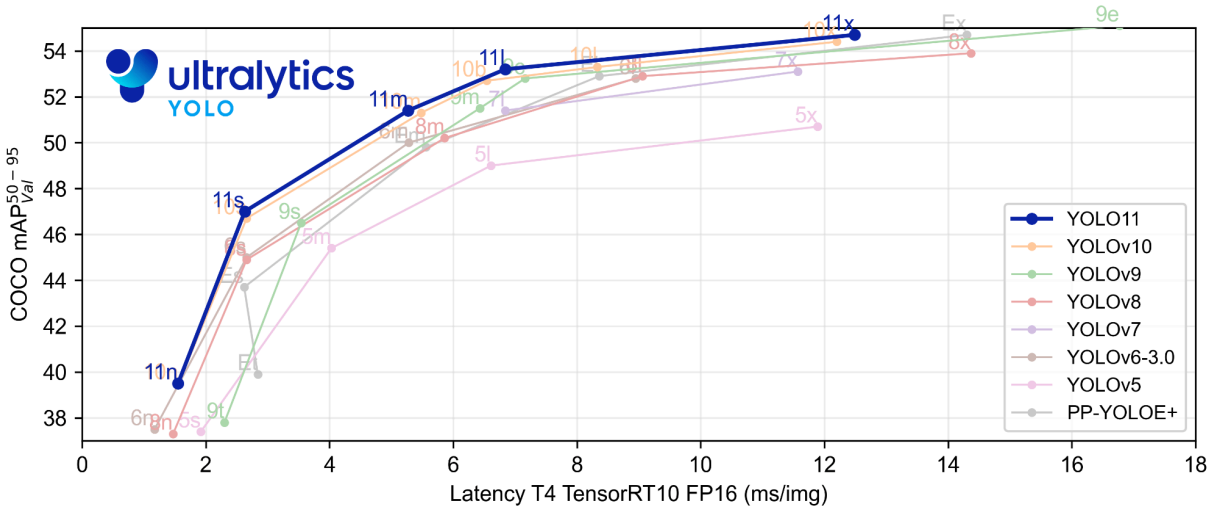
4. Thực nghiệm (Experiments)

4.1 Thiết lập nghiên cứu:
Thí nghiệm được thực hiện trên môi trường phần cứng và phần mềm hiện đại, đảm bảo khả năng xử lý tốc độ cao và hiệu quả:

- **Phần cứng**: Các nghiên cứu được thực hiện trên máy chủ có GPU (Graphics Processing Unit) mạnh như T4GPU hoặc NVIDIA A100, giúp tăng tốc độ huấn luyện và tối ưu hóa mô hình trong thời gian thực.
- **Phần mềm**: Các framework như Utralytics, TensorFlow, PyTorch và OpenCV được sử dụng để xây dựng, huấn luyện và kiểm thử mô hình. TensorFlow và PyTorch được sử dụng để triển khai các kiến trúc mạng YOLOv10, YOLOv11 và RT-DETR, trong khi OpenCV hỗ trợ việc xử lý ảnh và video.
- **Môi trường phát triển**: Hệ thống được xây dựng trên nền tảng Colab với môi trường Linux để đảm bảo tính nhất quán trong quá trình huấn luyện và kiểm thử.

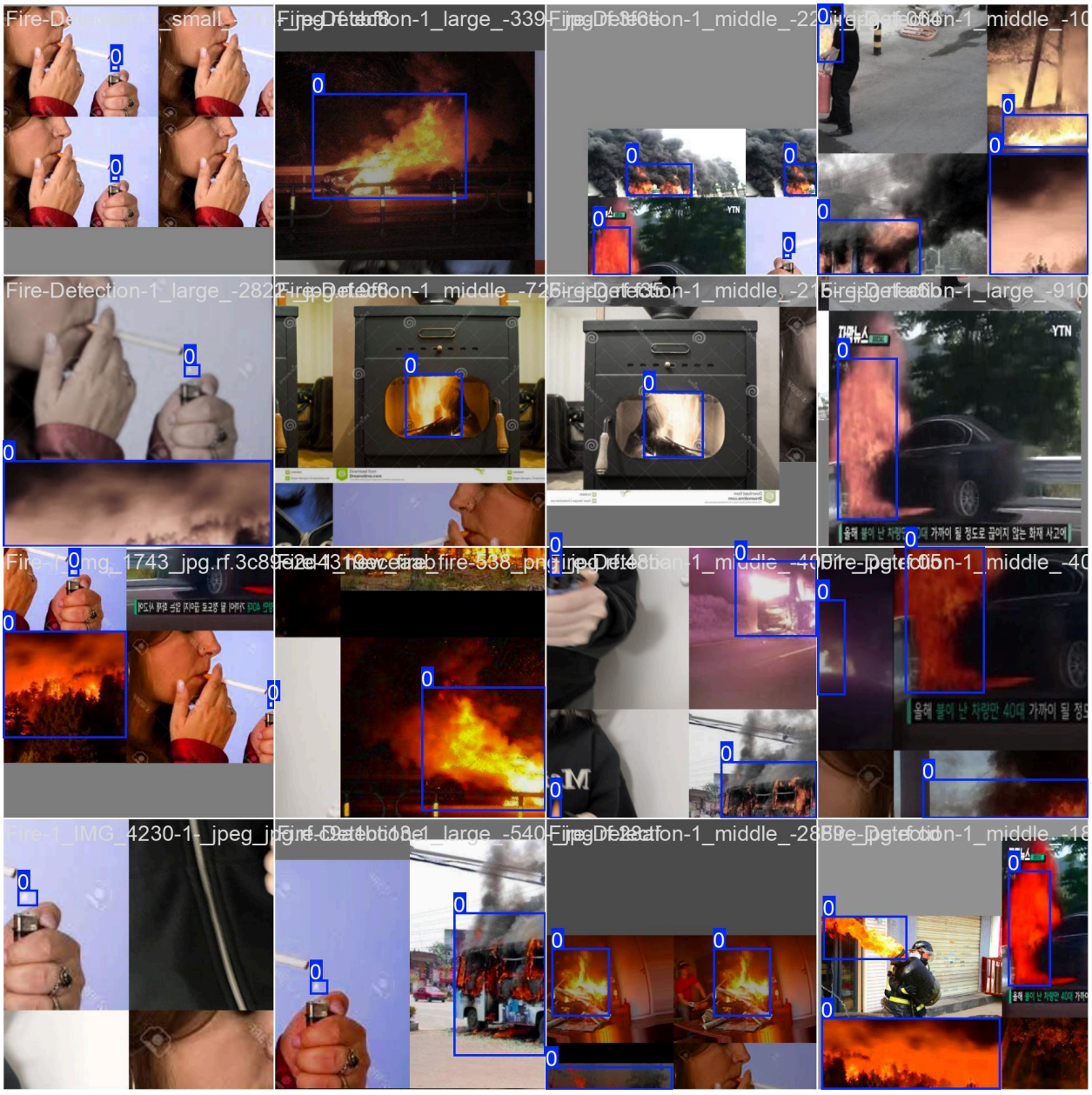
4.2 Kết quả nghiên cứu:
Kết quả của các nghiên cứu được thực hiện với các mô hình YOLOv10 và YOLOv11 trên các bộ dữ liệu nhận diện cháy và rác:

- **YOLOv10**: Đạt được kết quả tốt với tốc độ nhanh, tuy nhiên độ chính xác giảm trong việc phát hiện các đối tượng nhỏ.
- **YOLOv11**: Cải thiện hơn về độ chính xác và mAP50 và mAP50-95 so với YOLOv10, đặc biệt trong bài toán phát hiện cháy với các đám cháy nhỏ.



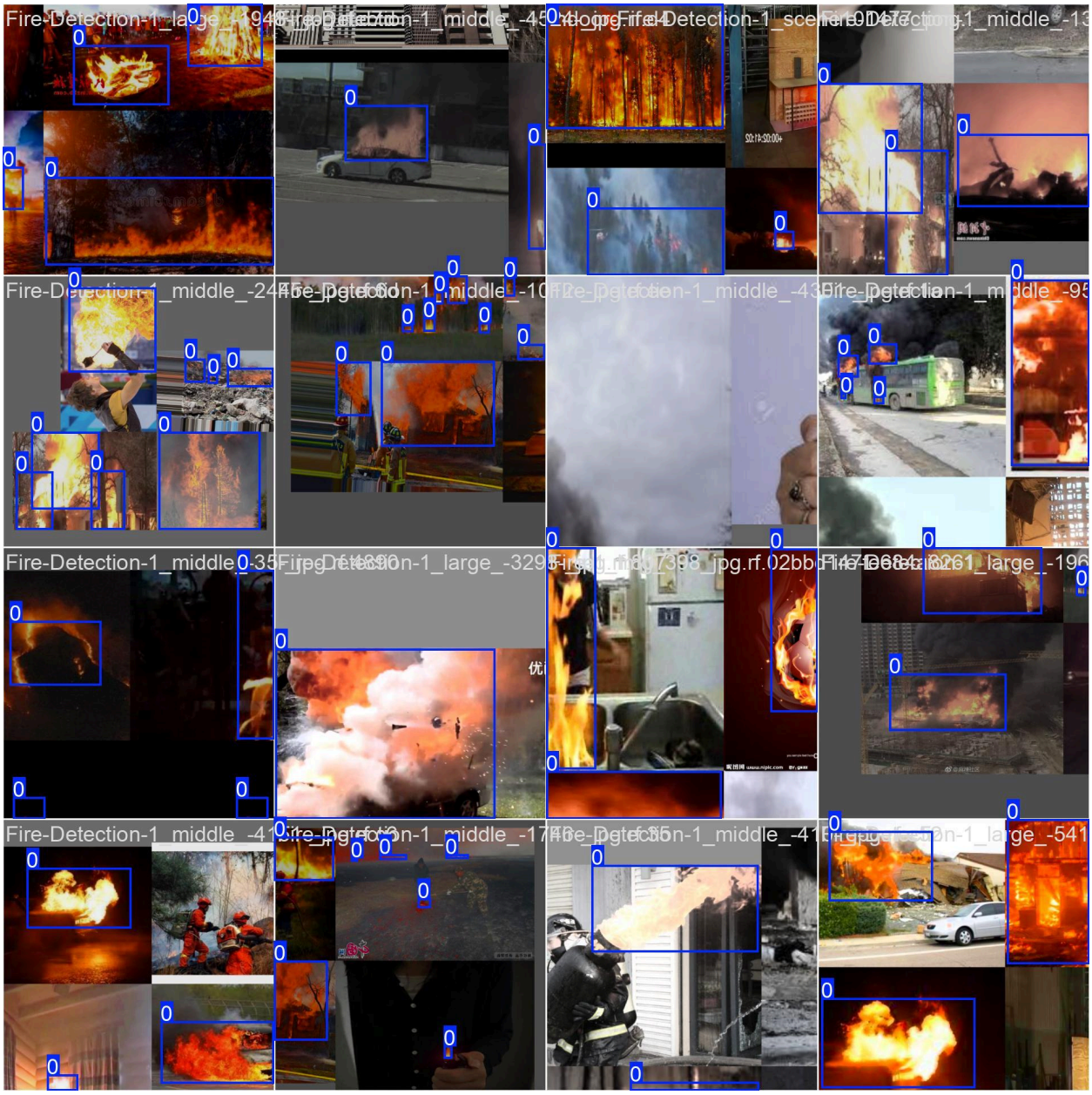
Hình 1. So sánh độ chính xác và độ trễ của các model trên bộ dữ liệu COCO

Sau đây là kết quả được thực hiện với model YOLOv11.

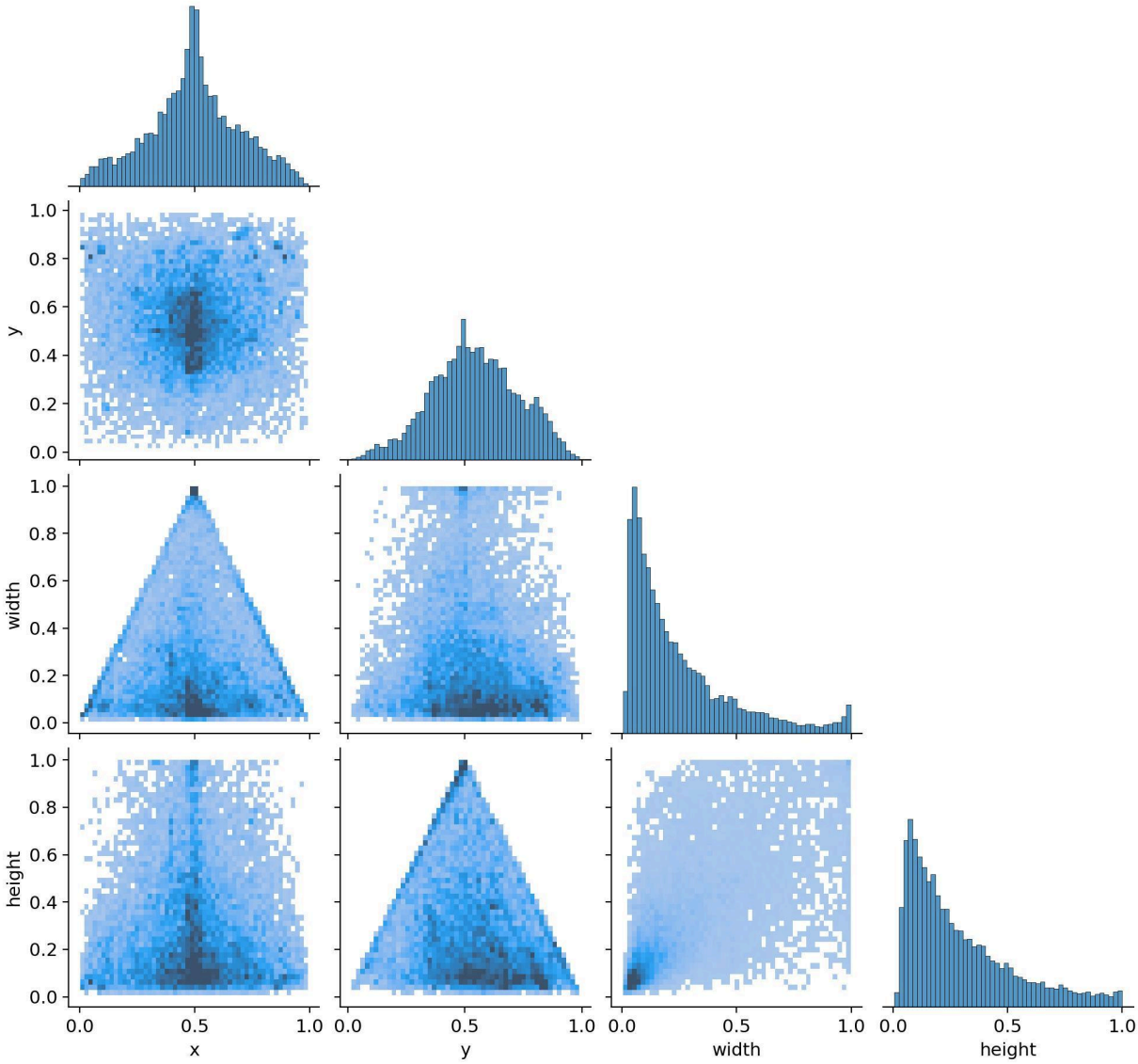


Hình 2: Train batch 0

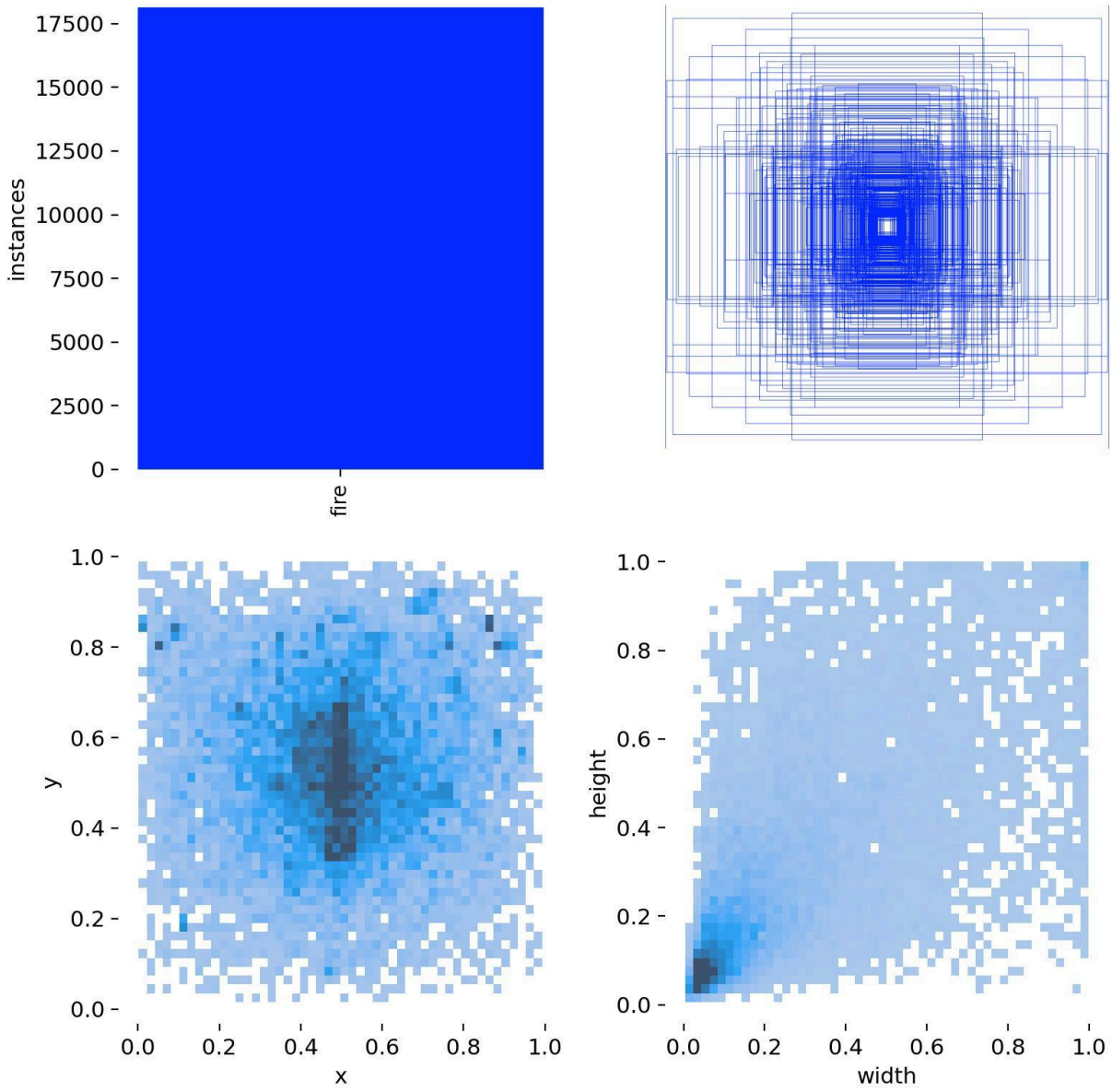
Hình 3: Train batch 1



Hình 4. Train batch 3

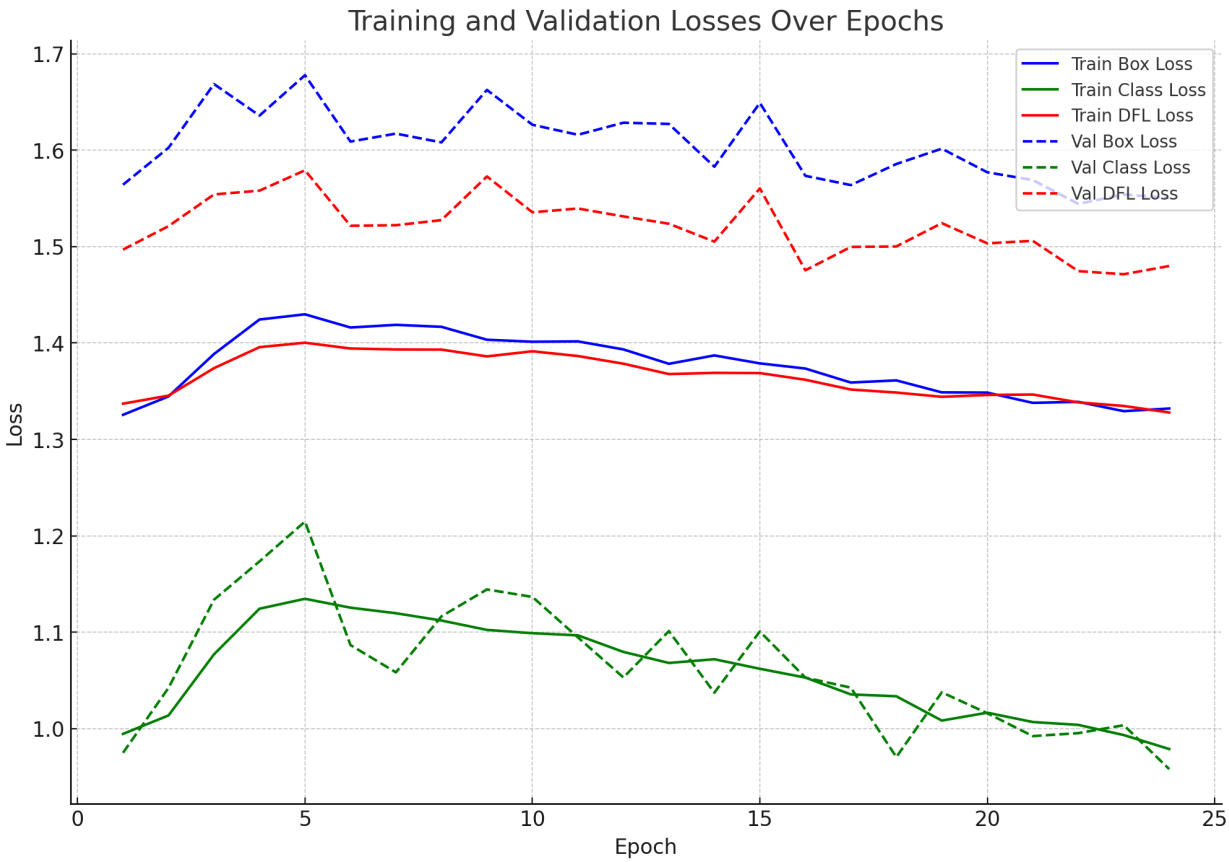


Hình 5. Label correlogram

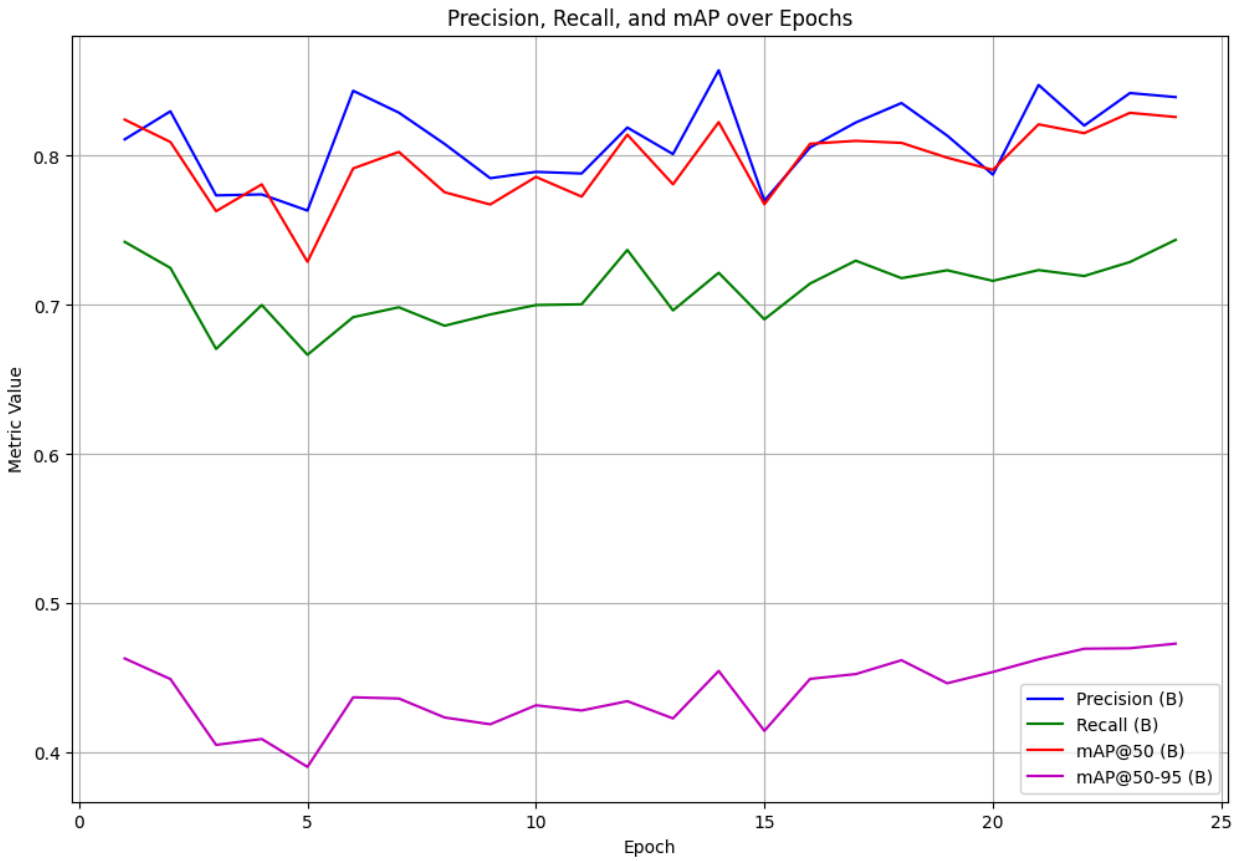


Hình 6: Labels

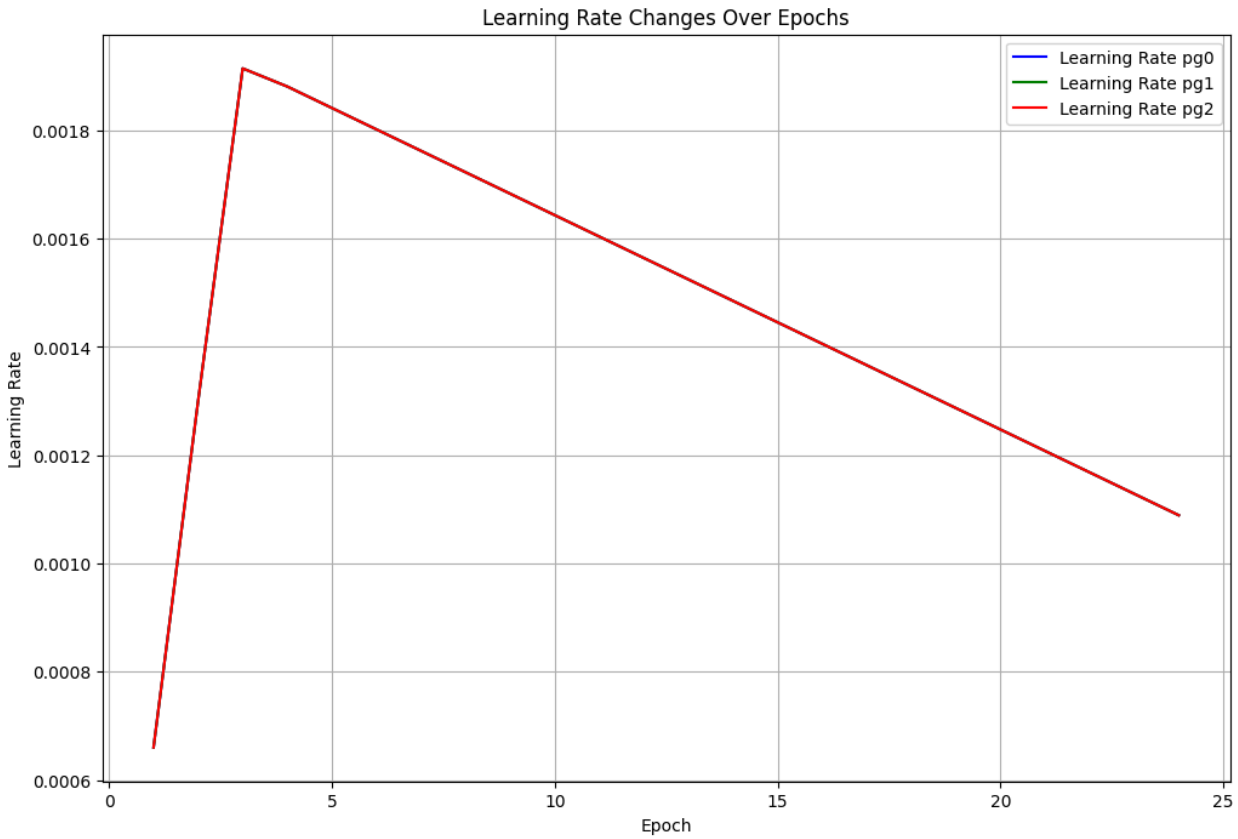
Sau khi train model khoảng 83 epoch thì mAP90-95 của model đạt khoảng 0.48, sau khi test qua bộ text thì cho thấy model khá ổn định. Tôi tiếp tục train model qua 24 epoch nữa thì có kết quả như sau:



Hình 7. Biểu đồ về Training and Validation Losses trên 24 epoch cuối



Hình 8. Biểu đồ về độ chính xác, recall và mAP metrics qua 24 epoch cuối.



Hình 9. Biểu đồ về sự thay đổi của Learning Rate qua 24 epoch cuối.

4.3 Phân tích kết quả:

- Phân tích Training and Validation Losses qua 26 epoch cuối (Hình 7):

  - **Xu hướng giảm dần của các Loss**: Cả ba loại loss (box, class, DFL) trong quá trình huấn luyện đều giảm dần, điều này cho thấy rằng mô hình đang học tốt dần theo thời gian. Tuy nhiên, sự giảm dần không phải lúc nào cũng đều đặn, có một số dao động nhẹ trong các giá trị.
  - **Sự phân kỳ giữa Training Loss và Validation Loss**: Từ giữa quá trình huấn luyện, có sự chênh lệch nhỏ giữa tổn thất huấn luyện và tổn thất kiểm tra (validation loss). Trong khi tổn thất huấn luyện tiếp tục giảm, tổn thất kiểm tra dường như không giảm mạnh tương ứng. Đây có thể là dấu hiệu của việc **overfitting**, tức là mô hình có thể học quá mức dữ liệu huấn luyện và không còn tổng quát tốt trên tập dữ liệu kiểm tra.

- **Validation Loss khá ổn định**: Mặc dù có sự dao động nhẹ, validation loss (đường nét đứt) không tăng lên nhiều, điều này cho thấy mô hình chưa gặp vấn đề nghiêm trọng với overfitting và vẫn đang hoạt động tương đối ổn định trên dữ liệu kiểm tra.

- Phân tích Precision, Recall, and mAP qua 26 epoch cuối (Hình 8):
    - **Precision và Recall dao động**: Precision và Recall có sự dao động qua các epoch, đặc biệt là Recall có xu hướng giảm nhẹ ở các epoch sau. Precision duy trì ở mức khá cao, nhưng vẫn có những biến động, cho thấy mô hình có sự nhạy cả m với dữ liệu khác nhau giữa các epoch.
    - **mAP@50 giảm nhẹ**: mAP@50, một chỉ số quan trọng để đánh giá độ chính xác tổng thể, có xu hướng giảm nhẹ ở các epoch cuối, điều này có thể cho thấy mô hình không còn cải thiện nhiều về khả năng phát hiện chính xác các đối tượng.
    - **mAP@50-95 ổn định nhưng thấp hơn**: Chỉ số mAP@50-95 (phản ánh độ chính xác của mô hình với các mức Intersection over Union - IoU khác nhau) có giá trị thấp hơn so với mAP@50 và không tăng lên rõ rệt. Điều này có nghĩa là mặc dù mô hình có thể phát hiện đối tượng ở mức IoU thấp, nhưng nó không thực sự mạnh khi đòi hỏi mức IoU cao hơn (yêu cầu dự đoán chính xác hơn).
- Phân tích biểu đồ Learning Rate Changes qua 24 Epochs cuối:
    - **Tăng nhanh ở những epoch đầu**: Từ epoch 0 đến khoảng epoch 4, learning rate tăng dần từ giá trị rất nhỏ (khoảng 0.0006) lên đến đỉnh khoảng 0.0018. Giai đoạn tăng này là một phần của chiến lược "warm-up", giúp mô hình từ từ học trước khi tăng tốc, tránh cập nhật trọng số quá nhanh ban đầu có thể dẫn đến việc mô hình dao động không ổn định.
    - **Giảm dần sau khi đạt đỉnh**: Sau epoch 4, learning rate bắt đầu giảm dần một cách đều đặn. Điều này cho thấy mô hình đang sử dụng chiến lược giảm tốc

(learning rate decay) khi quá trình huấn luyện tiến triển, nhằm tránh việc mô hình cập nhật quá nhanh và vượt qua điểm hội tụ tốt nhất.

- Từ epoch 5 trở đi, learning rate giảm dần một cách tuyến tính, cho thấy quá trình giảm tốc ổn định. Điều này có lợi vì khi learning rate giảm dần, mô hình sẽ điều chỉnh các trọng số một cách tinh tế hơn, cải thiện độ chính xác mà không làm mô hình dao động mạnh.
- **pg2 có giá trị lớn nhất**: Biểu đồ chỉ hiển thị rõ ràng learning rate của `pg2`, điều này có thể cho thấy `pg2` có vai trò quan trọng hơn trong việc cập nhật các trọng số của mô hình hoặc được cấu hình để có tốc độ học khác biệt so với `pg0` và `pg1`.

5. Thảo luận (Discussion)

Kết quả của quá trình huấn luyện cho thấy mô hình đang học tốt qua từng epoch, thể hiện qua việc giảm dần các loại tổn thất (box, class, DFL) và xu hướng ổn định của tổn thất kiểm tra (validation loss). Mặc dù có sự phân kỳ giữa training loss và validation loss, sự chênh lệch không quá lớn và validation loss vẫn giữ ổn định, chỉ ra rằng mô hình chưa gặp vấn đề nghiêm trọng về overfitting. Tuy nhiên, có dấu hiệu nhẹ của overfitting khi tổn thất huấn luyện giảm trong khi validation loss không giảm tương ứng.

Phân tích các chỉ số Precision, Recall, và mAP cho thấy một số điểm đáng chú ý. Precision vẫn giữ ở mức khá cao nhưng dao động qua các epoch, cho thấy mô hình có độ nhạy cao với dữ liệu khác nhau trong từng epoch. Ngược lại, Recall có xu hướng giảm nhẹ về các epoch sau, cho thấy khả năng phát hiện đúng đối tượng của mô hình có phần suy giảm. mAP@50 cũng giảm nhẹ, cho thấy hiệu suất tổng thể không cải thiện nhiều ở giai đoạn sau của quá trình huấn luyện. Chỉ số mAP@50-95 ổn định nhưng thấp hơn, cho thấy mô hình có thể hoạt động tốt với các mức IoU thấp nhưng không mạnh ở các mức yêu cầu IoU cao hơn.

Việc thay đổi learning rate qua các epoch cũng phản ánh chiến lược tối ưu hóa hiệu quả. Sự tăng nhanh learning rate trong các epoch đầu theo chiến lược "warm-up" giúp mô hình học dần mà không gây ra dao động quá mức. Sau đó, việc giảm dần learning rate giúp mô hình điều chỉnh các trọng số một cách tinh tế, tránh cập nhật quá nhanh và làm mô hình dao động. Đặc biệt, learning rate của pg2 có vai trò quan trọng hơn trong

việc cập nhật các trọng số, có thể do sự khác biệt về cách thức hoặc tốc độ học của các nhóm trọng số khác nhau.

So với các nghiên cứu trước đây, kết quả này thể hiện sự tương đồng về chiến lược học và tối ưu hóa, tuy nhiên vẫn còn một số hạn chế cần khắc phục. Một trong số đó là sự dao động của Recall và mAP@50-95, cho thấy mô hình cần cải thiện thêm về khả năng phát hiện đối tượng chính xác hơn, đặc biệt với những đối tượng có mức IoU cao.

Lợi ích và thách thức
Mô hình đề xuất có khả năng học tốt qua từng epoch, ổn định trong việc huấn luyện và kiểm tra. Tuy nhiên, việc Precision và Recall dao động, cùng với sự suy giảm nhẹ của mAP@50 và mAP@50-95, là những thách thức cần lưu ý. Hơn nữa, mặc dù chiến lược giảm learning rate là hiệu quả, nhưng việc tối ưu hóa learning rate của từng phần (pg2, pg0, pg1) có thể cần được nghiên cứu thêm để cải thiện mô hình.

Các yếu tố cải thiện
Để cải thiện mô hình, có thể áp dụng các phương pháp như:

- Tinh chỉnh thêm learning rate, đặc biệt là giữa các nhóm trọng số khác nhau (pg0, pg1, pg2) để tối ưu quá trình học.
- Sử dụng thêm các kỹ thuật như regularization hoặc dropout để giảm thiểu overfitting.
- Thử nghiệm với các mô hình phức tạp hơn hoặc cải thiện việc xử lý dữ liệu, nhằm nâng cao chỉ số mAP@50-95, giúp mô hình hoạt động tốt hơn với các mức IoU cao..

6. Kết luận (Conclusion)

- Quá trình huấn luyện đã cho thấy mô hình học tốt qua thời gian, với các tổn thất giảm dần và validation loss ổn định, cho thấy mô hình không bị overfitting nghiêm trọng. Precision và Recall dao động, nhưng nhìn chung vẫn duy trì ở mức tương đối tốt. Tuy nhiên, mAP@50-95 thấp hơn so với mAP@50 cho thấy mô hình cần cải thiện khi đối phó với những đối tượng yêu cầu độ chính xác cao hơn.
- Để tránh việc model bị overfitting tôi sẽ chọn model dừng lại ở epoch 83 với mAP90-95 đạt khoảng 0.48.
- **Hướng phát triển tương lai**
  Trong tương lai, có thể mở rộng quy mô dữ liệu, tinh chỉnh lại các tham số huấn luyện (đặc biệt là learning rate) và thử nghiệm thêm các kỹ thuật nâng cao nhằm cải thiện hiệu suất của mô hình. Việc tăng cường sử dụng các công nghệ mới như augmentation hoặc các mô hình học sâu tiên tiến hơn có thể giúp nâng cao khả năng tổng quát hóa và phát hiện đối tượng chính xác hơn