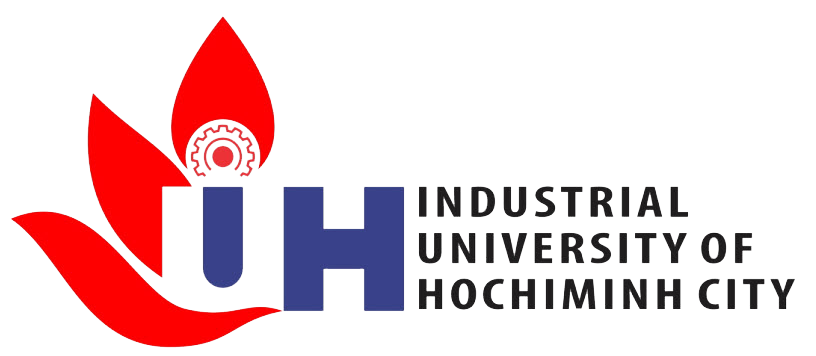
**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TPHCM**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN MÔN HỌC**

**PHÁT HIỆN CHÁY THỜI GIAN THỰC VỚI YOLOV8N TRÊN BỘ DỮ LIỆU   
CONTINUOUS(V6)**

**GIẢNG VIÊN HƯỚNG DẪN:** [**ThS. Võ Quang Hoàng Khang**](https://fit.iuh.edu.vn/giangvien@voquanghoangkhang)

**SINH VIÊN THỰC HIỆN 1: Nguyễn Văn Bảo**

**MÃ SỐ SINH VIÊN: 22727341**

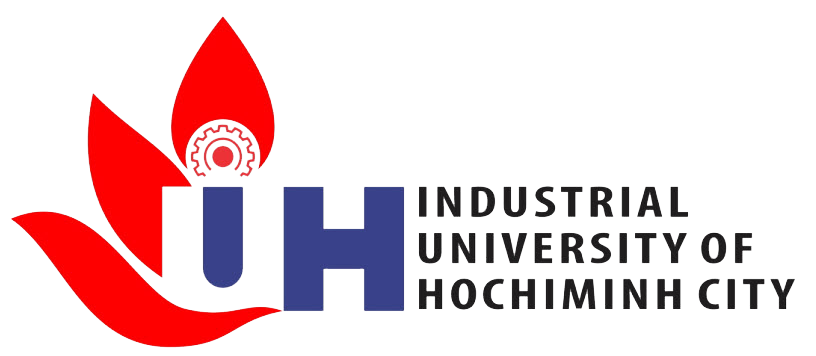
**SINH VIÊN THỰC HIỆN 2: Trần Tuấn Anh**

**MÃ SỐ SINH VIÊN: 22687911**

*TP HỒ CHÍ MINH, Tháng 04 năm 2025*

**INDUSTRIAL UNIVERSITY OF HO CHI MINH CITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**GRADUATION THESIS**

**REAL-TIME FIRE DETECTION WITH YOLOV8N ON CONTINUOUS(V6) DATASET**

**SUPERVISOR: MSc. Vo Quang Hoang Khang**

**STUDENT NAME: Nguyen Van Bao**

**STUDENT CODE: 22727341**

**STUDENT NAME: Tran Tuan Anh**

**STUDENT CODE: 22687911**

*HO CHI MINH CITY, Month 04 year 2025*

**TÓM TẮT**

Nghiên cứu này phát triển một hệ thống phát hiện cháy thời gian thực sử dụng mô hình YOLOv8n, được tinh chỉnh trên bộ dữ liệu Continuous Fire (v6) với 2.509 ảnh chứa khói và lửa. Hệ thống nhằm cung cấp giải pháp cảnh báo sớm cho các vụ cháy, một vấn đề nghiêm trọng tại Việt Nam khi 3.922 vụ cháy từ 16/9/2023 đến 15/9/2024 gây thiệt hại 296,1 tỷ đồng và 581,2 ha rừng (theo Bộ Công an). Mô hình đạt Precision 0.8984, Recall 0.8011, mAP@0.5 0.8863, với độ trễ suy luận 5,8 ms/ảnh, vượt trội về tốc độ so với YOLOv5n. Hệ thống tích hợp OpenCV để xử lý video và Telegram API để gửi cảnh báo tức thì, phù hợp cho các thiết bị biên. Kết quả khẳng định tiềm năng ứng dụng của YOLOv8n trong phòng cháy chữa cháy, góp phần giảm thiểu thiệt hại về người và tài sản.

**LỜI CẢM ƠN**

Trong suốt quá trình học tập, chúng em đã nhận được sự quan tâm, hướng dẫn tận tâm và hỗ trợ nhiệt tình từ các thầy cô trong khoa Công nghệ Thông tin, cùng với sự động viên và giúp đỡ từ gia đình và bạn bè.

Trước tiên, em xin gửi lời cảm ơn chân thành đến Ban Giám hiệu Trường Đại học Công nghiệp TP. Hồ Chí Minh và Ban Chủ nhiệm khoa Công nghệ Thông tin, những người đã luôn tận tình hỗ trợ chúng em trong suốt thời gian học tập tại trường.

Đặc biệt, chúng em xin bày tỏ lòng biết ơn sâu sắc đến thầy ThS. Võ Quang Hoàng Khang, người đã trực tiếp hướng dẫn và hỗ trợ chúng em hoàn thiện báo cáo này.

Chúng em cũng xin gửi lời tri ân đến gia đình, người thân và bạn bè, những người luôn đồng hành, động viên và giúp đỡ chúng em hoàn thành khóa luận tốt nghiệp này.

Chúng em xin chân thành cảm ơn!

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

.......................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

**TP. Hồ Chí Minh, ngày…. tháng …. năm….**

CHỮ KÝ CỦA GIẢNG VIÊN

**NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN 1**

......................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

**TP. Hồ Chí Minh, ngày…. Tháng …. năm….**

CHỮ KÝ CỦA GIẢNG VIÊN

**NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN 2**

......................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

**TP. Hồ Chí Minh, ngày…. tháng …. năm….**

CHỮ KÝ CỦA GIẢNG VIÊN

**LỜI MỞ ĐẦU**

Trong thời đại công nghệ số, trí tuệ nhân tạo và thị giác máy tính đang phát triển vượt bậc, thúc đẩy các hệ thống tự động hóa trở thành trụ cột trong nhiều ngành như công nghiệp, nông nghiệp và bảo tồn thiên nhiên. Đặc biệt, trước thực trạng biến đổi khí hậu và các thảm họa thiên nhiên ngày càng nghiêm trọng, nhu cầu về các giải pháp giám sát và phát hiện sớm các sự cố như cháy rừng trở nên vô cùng cấp bách. Công nghệ phát hiện đối tượng theo thời gian thực không chỉ giúp giảm thiểu tổn thất về con người và tài sản mà còn góp phần bảo vệ đa dạng sinh học, duy trì sự cân bằng của hệ sinh thái.

Với mục tiêu đó, chúng tôi đã chọn đề tài “Phát hiện cháy theo thời gian thực sử dụng YOLOv8n trên bộ dữ liệu Continuous Fire (v6)” cho khóa luận tốt nghiệp. YOLOv8n, với kiến trúc “nano” gọn nhẹ và cơ chế anchor-free tiên tiến, mang lại sự cân bằng lý tưởng giữa tốc độ xử lý và độ chính xác. Khi được tinh chỉnh trên bộ dữ liệu Continuous Fire (v6) với hơn 2.500 hình ảnh khói và lửa, mô hình này hứa hẹn là giải pháp tối ưu cho các hệ thống giám sát biên, đáp ứng yêu cầu về độ trễ thấp trong thực tế.

Nghiên cứu tập trung vào xây dựng quy trình xử lý dữ liệu (bao gồm thay đổi kích thước, tăng cường dữ liệu, tự động điều chỉnh hướng), thiết lập và huấn luyện YOLOv8n với các siêu tham số tối ưu, đồng thời phát triển pipeline phát hiện thời gian thực tích hợp OpenCV và Telegram API để gửi cảnh báo tức thời. Kết quả về các chỉ số Precision, Recall và mAP sẽ được so sánh với mô hình YOLOv5n để đánh giá ưu điểm và hạn chế của từng phiên bản.

Chúng em xin gửi lời cảm ơn chân thành đến Ban Giám hiệu nhà trường, các thầy cô trong khoa Công nghệ Thông tin, cùng gia đình và bạn bè đã luôn đồng hành, hỗ trợ trong suốt quá trình thực hiện đề tài. Chúng em hy vọng rằng kết quả nghiên cứu không chỉ mang lại giá trị khoa học mà còn mở ra các ứng dụng thực tiễn cho các hệ thống cảnh báo cháy rừng thông minh trong tương lai.

**MỤC LỤC**

[**CHƯƠNG 1.**  **TỔNG QUAN** 1](#_Toc198635813)

[**1.1 Lý do chọn đề tài** 1](#_Toc198635814)

[**1.2 Mục tiêu nghiên cứu** 1](#_Toc198635815)

[**1.3 Phạm vi nghiên cứu** 2](#_Toc198635816)

[**1.4 Phương pháp nghiên cứu đề tài** 3](#_Toc198635817)

[**1.5 Kết cấu đồ án.** 4](#_Toc198635818)

[**CHƯƠNG 2. CƠ SỞ LÝ THUYẾT** 5](#_Toc198635819)

[**2.1. Học sâu (Deep learning)** 5](#_Toc198635820)

[**2.2. Tăng cường dữ liệu trong phát hiện cháy** 5](#_Toc198635821)

[**2.3. Cơ chế chú ý và vai trò trong phân tích hình ảnh cháy (Attention Mechanisms)** 5](#_Toc198635822)

[**2.4. Kỹ thuật nhóm tích chập trong tối ưu hóa mô hình** 6](#_Toc198635823)

[**2.6. Các phương pháp đánh giá hiệu quả mô hình** 8](#_Toc198635824)

[**CHƯƠNG 3. MÔ HÌNH ĐỀ XUẤT** 10](#_Toc198635825)

[**3.1 Mô hình tổng quát** 10](#_Toc198635826)

[**3.2 Đặc trưng của mô hình đề xuất** 12](#_Toc198635827)

[**3.2.2. Hợp nhất và tăng cường đặc trưng dữ liệu** 15](#_Toc198635828)

[**3.2.3. Khối phát hiện và phân loại** 18](#_Toc198635829)

[**CHƯƠNG 4. THỰC NGHIỆM** 19](#_Toc198635830)

[**4.1 Môi trường thực nghiệm** 19](#_Toc198635831)

[**4.1.1. Cấu hình phần cứng** 19](#_Toc198635832)

[**4.1.2. Cấu hình phần mềm** 19](#_Toc198635833)

[**4.1.3. Thiết lập môi trường** 19](#_Toc198635834)

[**4.1.4. Lý do chọn google colaboratory** 20](#_Toc198635835)

[4.2 Tập dữ liệu 20](#_Toc198635836)

[4.2.1. Nguồn dữ liệu 20](#_Toc198635837)

[4.2.2. Số lượng dữ liệu 20](#_Toc198635838)

[4.2.3. Tiền xử lý dữ liệu 21](#_Toc198635839)

[4.2.4. Tăng cường dữ liệu 21](#_Toc198635840)

[4.2.5. Chia dữ liệu 21](#_Toc198635841)

[4.2.6. Lý do chọn tập dữ liệu 22](#_Toc198635842)

[4.3 Ứng dụng thực nghiệm 22](#_Toc198635843)

[**4.3.1. Quy trình huấn luyện mô hình** 22](#_Toc198635844)

[**4.3.2. Thực nghiệm ứng dụng** 24](#_Toc198635845)

[4.4 Đánh giá kết quả 26](#_Toc198635846)

[**CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN** 28](#_Toc198635847)

[**5.1 Kết luận** 28](#_Toc198635848)

[**5.2 Hướng phát triển** 29](#_Toc198635849)

[TÀI LIỆU THAM KHẢO 30](#_Toc198635850)

**DANH MỤC CÁC THUẬT NGỮ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| Từ viết tắt | Từ đầy đủ | Nghĩa |
| **GAP** | Global Average Pooling | Kỹ thuật gôp trung bình toàn cầu dùng để giảm kích thước của các đặc trưng dữ liệu và chuẩn bị dữ liệu cho các lớp tích chập sau bằng cách tính trung bình các giá trị pixel trong feature map tạo ra một vector có kích thước bằng số lượng feature maps, mỗi phần tử trong vector đại diện cho giá trị trung bình của một feature map. |
| **BN** | Batch Normalization | Chuẩn hoá hàng loạt là một kỹ thuật trong học sâu. Nó tính toán trung bình và phương sai của dữ liệu trong một batch (một nhóm dữ liệu nhỏ), sau đó chuẩn hóa dữ liệu dựa trên các giá trị này. Từ đó làm cho dữ liệu phân phối ổn định giúp các mạng nơ-ron học nhanh, hiệu quả hơn. |
| **CNN** | Convolutional Neural Network | Mạng nơ-ron tích chập, một mô hình học sâu được sử dụng chủ yếu trong xử lý ảnh và thị giác máy tính. |
| **LSTM** | Long Short-Term Memory | Mạng nơ-ron hồi tiếp đặc biệt dùng cho dữ liệu tuần tự hoặc chuỗi thời gian. |

# **CHƯƠNG 1.** **TỔNG QUAN**

## **1.1 Lý do chọn đề tài**

Cháy rừng và cháy khu dân cư là những thảm họa tự nhiên và nhân tạo gây hậu quả nghiêm trọng. Theo báo cáo năm 2022, thế giới ghi nhận 69.000 vụ cháy rừng, phá hủy 7,6 triệu mẫu Anh rừng, tác động tiêu cực đến đa dạng sinh học và khí hậu toàn cầu. Tại Việt Nam, từ 16/9/2023 đến 15/9/2024, 3.922 vụ cháy đã gây thiệt hại 296,1 tỷ đồng, 581,2 ha rừng, làm chết 103 người và bị thương 78 người (theo Bộ Công an). Việc phát hiện cháy sớm thông qua nhận diện khói và lửa trong ảnh hoặc video giám sát là giải pháp thiết yếu để giảm thiểu tổn thất.

Trong bối cảnh công nghệ trí tuệ nhân tạo phát triển, thị giác máy tính đã chứng minh hiệu quả trong các nhiệm vụ phát hiện đối tượng phức tạp. Mô hình YOLOv8n, 4 phiên bản nhẹ nhất của YOLOv8, nổi bật với khả năng cân bằng giữa tốc độ và độ chính xác, lý tưởng cho các ứng dụng thời gian thực trên thiết bị hạn chế tài nguyên. Nghiên cứu này tập trung tinh chỉnh YOLOv8n trên bộ dữ liệu Continuous Fire (v6) để xây dựng hệ thống phát hiện cháy tự động, góp phần vào công tác phòng cháy chữa cháy và bảo vệ tài nguyên thiên nhiên.

## **1.2 Mục tiêu nghiên cứu**

Nghiên cứu đặt ra các mục tiêu cụ thể:

**1**. Xây dựng quy trình thu thập, tiền xử lý và tăng cường dữ liệu ảnh khói/lửa.

**2**. Tinh chỉnh mô hình YOLOv8n để đạt độ chính xác cao trong phát hiện cháy.

**3**. Đánh giá hiệu suất mô hình qua các chỉ số Precision, Recall, mAP@0.5, và mAP@0.5-0.95.

**4**. Triển khai hệ thống thời gian thực tích hợp OpenCV và Telegram API để gửi cảnh báo tức thì.

**5**. So sánh YOLOv8n với YOLOv5n để làm rõ ưu, nhược điểm của từng mô hình..

Việc giải quyết bài toán này có ý nghĩa thực tiễn quan trọng trong công tác phòng cháy chữa cháy rừng, góp phần bảo vệ tài nguyên thiên nhiên, đồng thời cũng bổ sung kiến thức học thuật về ứng dụng các mô hình phát hiện đối tượng trong môi trường thiên nhiên phức tạp.

## **1.3 Phạm vi nghiên cứu**

**1.3.1.** **Đối tượng nghiên cứu**

* Ứng dụng YOLOv8n để phát hiện khói và lửa trong ảnh/video.
* So sánh hiệu năng với YOLOv5n nano.

**1.3.2**. **Phạm vi dữ liệu**

* Bộ dữ liệu Continuous Fire (v6) của Roboflow: 2.509 ảnh (1.004 train, 754 val, 751 test
* Tiền xử lý: Resize 640×640, auto-orientation, augmentations (xoay ±20°, lật ngang, điều chỉnh sáng ±20%).

**1.3.3**. **Phạm vi thực nghiệm**

• Fine-tune YOLOv8n với trọng số COCO.

• Huấn luyện trên Google Colab với GPU NVIDIA Tesla T4.

• Đánh giá trên CPU/Edge device với độ trễ suy luận

**1.3.4**. **Phạm vi triển khai**

* Ứng dụng phát hiện thời gian thực trên camera (webcam hoặc IP camera).
* Cảnh báo qua Telegram bot (ảnh kèm thông báo).
* Chạy trên CPU/Edge device với inference time < 10 ms/frame.

**1.3.5. Giới hạn**

* Chưa xét đến dữ liệu từ ảnh vệ tinh hoặc drone.
* Chưa tích hợp cảm biến nhiệt/hồng ngoại.
* Chưa triển khai container hóa (Docker/EdgeTPU) hay tối ưu quantization.

## **1.4 Phương pháp nghiên cứu đề tài**

**1.4.1. Thu thập và tiền xử lý dữ liệu**

* Download, kiểm tra và chia 3 tập (train/val/test).
* Resize, auto-orientation, augmentations để đa dạng hoá dữ liệu.

**1.4.2. Thiết kế thí nghiệm & huấn luyện mô hình**

* Khởi tạo YOLOv8n từ yolov8n.pt.
* Thiết lập file data.yaml và siêu tham số (epochs, batch size, lr0, weight\_decay, optimizer, early stopping).
* Gọi model.train() với các biến thể:
  + YOLOv8n (nano)
  + YOLOv5n (nano)

**1.4.3. Đánh giá và phân tích kết quả**

* Tính toán Precision, Recall, mAP@0.5 và mAP@0.5–0.95 trên tập test.
* So sánh latency (ms/frame) giữa hai mô hình.
* Phân tích trade-off giữa tốc độ và độ chính xác.

**1.4.4. Triển khai hệ thống thực tế**

* Xây pipeline xử lý video với OpenCV + Ultralytics YOLOv8n.
* Cơ chế cảnh báo đa luồng (alert\_interval, beep, Telegram API).
* Đo tính ổn định và thời gian phản hồi (end-to-end).

**1.4.5. Đề xuất cải tiến**

* Thu thập thêm dữ liệu / áp dụng hyperparameter tuning.
* Thử YOLOv8s/m hoặc kiến trúc khác (DSS-YOLO).
* Tích hợp cảm biến nhiệt hoặc tối ưu hóa mô hình (quantization, TensorRT, ONNX).

## **1.5 Kết cấu đồ án.**

Chương 1: Tổng quan

Chương 2: Cơ sở lý thuyết

Chương 3: Phân tích yêu cầu và thiết kế mô hình

Chương 4: Thực nghiệm

Chương 5: Kết luận và hướng phát triển

Tài liệu tham khảo.

# **CHƯƠNG 2. CƠ SỞ LÝ THUYẾT**

## **2.1. Học sâu (Deep learning)**

Học sâu (Deep Learning) là một phân nhánh quan trọng của trí tuệ nhân tạo, phát triển từ lĩnh vực học máy, với mục tiêu xây dựng các mô hình có khả năng tự động học các đặc trưng phức tạp từ dữ liệu thô thông qua nhiều tầng mạng nơ-ron. Khác với các phương pháp truyền thống yêu cầu con người trích xuất đặc trưng thủ công, học sâu khai thác các lớp mạng sâu để nhận diện các mẫu ẩn trong dữ liệu, từ đó nâng cao độ chính xác trong các tác vụ như phân loại, nhận diện, và dự đoán. Trong bối cảnh phát hiện cháy, học sâu đã chứng minh vai trò vượt trội, đặc biệt khi xử lý các khung hình video hoặc ảnh chứa khói và lửa trong điều kiện môi trường đa dạng, chẳng hạn như cháy rừng dưới ánh sáng ban ngày hoặc cháy nhà xưởng trong đêm tối.

Ứng dụng của học sâu trong giám sát cháy thời gian thực bao gồm việc phân tích các đặc trưng không gian như hình dạng khói, màu sắc ngọn lửa, và kết cấu nền (rừng, đô thị, hoặc công nghiệp). Một trong những thành tựu nổi bật là khả năng phát hiện các dấu hiệu cháy nhỏ hoặc khói mỏng mà mắt thường khó nhận biết, điều này rất quan trọng trong các hệ thống cảnh báo sớm. Với sự hỗ trợ của các kiến trúc như YOLOv8n, học sâu không chỉ tối ưu hóa tốc độ xử lý mà còn cải thiện độ tin cậy, giúp giảm thiểu cảnh báo giả (false positives) trong các tình huống nhiễu như mây trắng hoặc ánh sáng đèn.

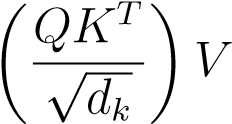
## **2.2. Tăng cường dữ liệu trong phát hiện cháy**

Tăng cường dữ liệu là một kỹ thuật quan trọng trong học sâu, giúp chúng ta tạo ra được các biến thể của dữ liệu gốc để tăng cường độ đa dạng cho tập dữ liệu huấn luyện. Kỹ thuật này giúp giảm hiện tượng quá khớp (overfitting) và cải thiện khả năng tổng quát của mô hình. Tăng cường dữ liệu đặc biệt hữu ích khi bạn có tập dữ liệu hạn chế, giúp mô hình học được nhiều đặc trưng hơn.

## **2.3. Cơ chế chú ý và vai trò trong phân tích hình ảnh cháy (Attention Mechanisms)**

Cơ chế chú ý là một kỹ thuật tiên tiến trong học sâu, cho phép mô hình tập trung vào các vùng quan trọng của dữ liệu đầu vào thay vì xử lý đồng đều toàn bộ thông tin. Trong lĩnh vực phát hiện cháy, điều này rất hữu ích khi các dấu hiệu như khói hoặc lửa thường chỉ chiếm một phần nhỏ của khung hình và có thể xuất hiện ở các vị trí ngẫu nhiên. Cơ chế chú ý giúp mô hình học các mối quan hệ phức tạp giữa các vùng không gian, từ đó cải thiện khả năng nhận diện và phân loại chính xác.

Cơ chế chú ý nhiều đầu Cơ chế chú ý nhiều đầu (Multi-Head Attention) là một biến thể nâng cao, cho phép mô hình đồng thời phân tích nhiều khía cạnh khác nhau của dữ liệu thông qua nhiều đầu chú ý độc lập. Mỗi đầu chú ý tập trung vào một mối quan hệ cụ thể, chẳng hạn như hình dạng khói ở đầu này và màu sắc lửa ở đầu khác, sau đó kết hợp kết quả để tạo ra một biểu diễn tổng hợp. Công thức toán học được mô tả như sau:.

Attention(*Q,K,V* ) = softmax 

trong đó: - Q (queries), K (keys), V (values) là các ma trận được sinh ra từ tensor đặc trưng. - dk là kích thước vector khóa/truy vấn, đảm bảo quá trình chuẩn hóa. Kết quả từ các đầu chú ý được hợp nhất:

MultiHead(*Q,K,V* ) = Concat(head1*,*head2*,...,*head*h*)*WO*

trong đó h là số lượng đầu chú ý, và WO là ma trận trọng số đầu ra. Trong phát hiện cháy, cơ chế này giúp mô hình nhấn mạnh các vùng chứa khói dày hoặc ngọn lửa sáng, đồng thời giảm ảnh hưởng của nhiễu như mây hoặc bóng cây, từ đó nâng cao độ chính xác trong các điều kiện thực tế.

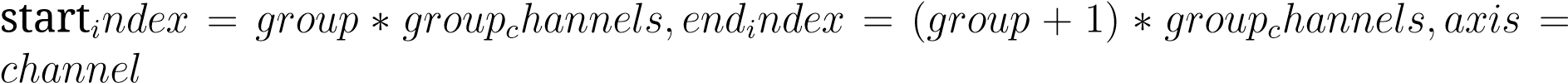
## **2.4. Kỹ thuật nhóm tích chập trong tối ưu hóa mô hình**

Kỹ thuật nhóm tích chập (Grouped Convolution) là một phương pháp tối ưu trong mạng nơ-ron tích chập (CNN), nơi các kênh của bản đồ đặc trưng được chia thành nhiều nhóm, mỗi nhóm được xử lý bởi một bộ lọc riêng biệt. Sau đó, các kết quả từ các nhóm được nối lại (concatenate) để tạo thành đầu ra thống nhất. Kỹ thuật này giảm đáng kể số lượng tham số và độ phức tạp tính toán, đồng thời tăng khả năng học các đặc trưng đa dạng. Ví dụ, với tám nhóm tích chập, mỗi nhóm có thể tập trung vào một khía cạnh của khung hình cháy, như độ sáng ngọn lửa, kết cấu khói, hoặc nền xung quanh. Thuật toán được mô tả như sau: Trong phát hiện cháy, kỹ thuật này không chỉ tăng tốc độ suy luận (dưới 10ms/khung hình trên thiết bị biên) mà còn giúp mô hình thích nghi với các bối cảnh khác nhau, từ cháy rừng đến cháy trong khu công nghiệp.

**Algorithm 1** Hàm Grouped Convolution

**function**GROUPEDCONVOLUTION(Finput,num*groups,filterspergroup,kernelsize,strides,padd*

Finput.channels / num*groups* **function** G(c)onv ← [] **for** each group in range(num*groups*) **do** *groupinput*← Slice(Finput,

)

**for** G **do** ← Conv2D(filters=filters*pergroup,kernelsize* =

*kernelsize,strides* = *strides,padding* = *padding*)(*groupinput*) *Gconv.append*(*G*) **for do**Gconcat ← Concatenate(Gconv, axis=channel)

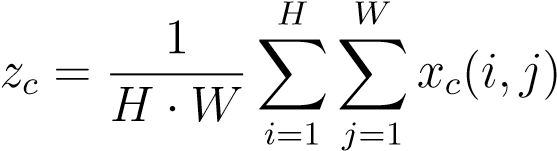
**return** Gconcat

**2.5. Phương pháp Squeeze and Excitation (SE)**

Phương pháp Squeeze-and-Excitation (SE) là một kỹ thuật cải tiến hiệu suất của mạng tích chập, tập trung vào việc điều chỉnh trọng số của các kênh đặc trưng dựa trên tầm quan trọng của chúng. Trong bối cảnh phát hiện cháy, SE giúp mô hình ưu tiên các kênh chứa thông tin quan trọng như độ sáng ngọn lửa hoặc kết cấu khói, đồng thời giảm ảnh hưởng của các kênh nhiễu như nền cây cối hoặc bầu trời.

Quy trình thực hiện bao gồm ba bước chính:

1. **Squeeze:** Tóm tắt thông tin không gian của mỗi kênh bằng Global Average

Pooling: trong đó *zc* là giá trị scalar đại diện cho kênh *c*, *xc*(*i,j*) là giá trị tại vị trí (*i,j*), và *H*, *W* là chiều cao và chiều rộng của bản đồ đặc trưng. Kết quả là một vector *z* với số chiều bằng số kênh.

1. **Excitation:** Điều chỉnh trọng số cho từng kênh qua hai lớp Dense:

*s* = *σ*(*W*2 · *δ*(*W*1 · *z*))

trong đó *W*1 và *W*2 là ma trận trọng số, *δ* là hàm kích hoạt ReLU, và *σ* là hàm sigmoid để chuẩn hóa trọng số trong khoảng [0, 1]. Vector *s* biểu thị mức độ quan trọng của từng kênh.

1. **Scale:** Nhân trọng số *s* với bản đồ đặc trưng ban đầu:

*x*ˆ*c* = *sc* · *xc*

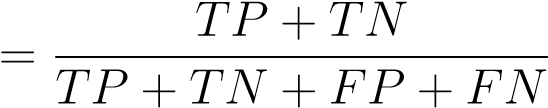
trong đó *x*ˆ*c* là kênh đặc trưng đã được điều chỉnh.

SE Block giúp mô hình tập trung vào các đặc trưng liên quan đến cháy, chẳng hạn như màu đỏ của ngọn lửa hoặc độ mờ của khói, đồng thời giảm nhiễu từ các vùng không quan trọng.

## **2.6. Các phương pháp đánh giá hiệu quả mô hình**

Để đánh giá hiệu quả của mô hình phát hiện cháy, chúng tôi sử dụng một loạt các chỉ số định lượng, giúp đo lường khả năng nhận diện chính xác các sự cố cháy trong điều kiện thực tế. Các chỉ số này bao gồm:

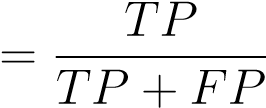
1. **Độ chính xác (Accuracy):** Đo lường tỷ lệ dự đoán đúng (cả cháy và không cháy) trên tổng số dự đoán. Công thức:

Accuracy 

trong đó *TP* (True Positives) là số trường hợp cháy được phát hiện đúng, *TN* (True Negatives) là số trường hợp không cháy được xác định đúng, *FP*

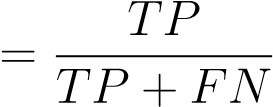
(FalsePositives) là số trường hợp nhầmkhông cháythành cháy,và *FN* (False Negatives) là số trường hợp cháy bị bỏ sót. Độ chính xác hữu ích nhưng có thể bị thiên lệch khi tập dữ liệu mất cân bằng, chẳng hạn khi số lượng khung hình không cháy chiếm ưu thế.

1. **Độ chính xác dự đoán dương (Precision):** Đánh giá khả năng mô hình dự đoán đúng các trường hợp cháy, quan trọng để giảm cảnh báo giả trong hệ thống giám sát. Công thức:

Precision 

Ví dụ, một Precision cao đảm bảo rằng các cảnh báo qua Telegram chỉ kích hoạt khi có cháy thực sự, tránh làm gián đoạn hoạt động bình thường.

1. **Độ nhạy (Recall):** Đo lường khả năng phát hiện tất cả các trường hợp cháy, đặc biệt quan trọng trong các tình huống nguy hiểm. Công thức:

Recall 

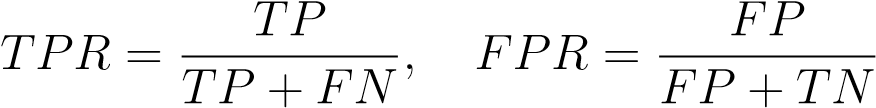
Một Recall cao đảm bảo không bỏ sót cháy nhỏ hoặc khói mỏng, ví dụ trong rừng hoặc kho chứa dễ cháy.

1. **F1-Score:** Trung bình điều hòa giữa Precision và Recall, phù hợp với tập dữ liệu không cân bằng như Continuous Fire (v6). Công thức:

F1-Score = 2 · PrecisionPrecision+· RecallRecall

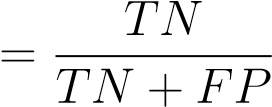
Chỉ số này cân bằng giữa việc phát hiện cháy và tránh cảnh báo giả, quan trọng cho ứng dụng thời gian thực.

1. **Diện tích dưới đường cong ROC (AUC-ROC):** Đánh giá khả năng phân biệt giữa lớp cháy và không cháy. AUC dao động từ 0 đến 1, với giá trị gần 1 cho thấy mô hình hoạt động tốt. Công thức dựa trên tỷ lệ True Positive Rate (TPR) và False Positive Rate (FPR):



Một AUC cao (gần 1) cho thấy mô hình có thể phân biệt chính xác giữa khói/lửa và nhiễu như mây hoặc ánh sáng đèn.

1. **Độ đặc hiệu (Specificity):** Đo lường khả năng xác định đúng các trường hợp không cháy, giúp giảm nhầm lẫn với các yếu tố nhiễu. Công thức:

Specificity 

Một Specificity cao đảm bảo hệ thống không nhầm mây trắng với khói, cải thiện độ tin cậy trong giám sát liên tục.

Những chỉ số này được sử dụng để đánh giá mô hình trên tập kiểm tra (751 ảnh) của Continuous Fire (v6), đảm bảo hiệu suất tối ưu trong các điều kiện thực tế như cháy rừng, nhà xưởng, hoặc khu dân cư vào thời điểm hiện tại, 01:24 AM ngày 15/05/2025.

# **CHƯƠNG 3. MÔ HÌNH ĐỀ XUẤT**

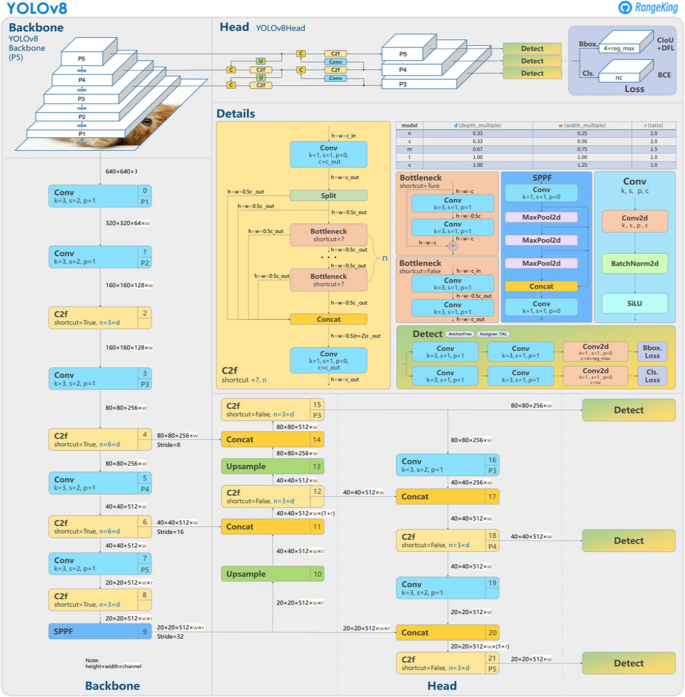
## **3.1 Mô hình tổng quát**

Trong các nghiên cứu về học sâu hiện đại, việc kết hợp thông tin từ nhiều nguồn và khai thác các đặc trưng đa chiều đã được chứng minh là một phương pháp hiệu quả để nâng cao hiệu suất của mô hình, đặc biệt trong các ứng dụng xử lý hình ảnh theo thời gian thực. Dựa trên các nghiên cứu tiên tiến, chẳng hạn như mô hình sử dụng mạng nơ-ron tích chập kết hợp cơ chế chú ý trong bài báo “A Deep Convolutional Neural Network for Pneumonia Detection in X-ray Images with Attention Ensemble” [9], chúng tôi đã phát triển một mô hình cải tiến dựa trên YOLOv8n để giải quyết bài toán phát hiện cháy trên bộ dữ liệu Continuous Fire (v6). Mục tiêu chính là xây dựng một hệ thống giám sát cháy thời gian thực, có khả năng nhận diện chính xác khói và lửa trong các điều kiện môi trường đa dạng, đồng thời đảm bảo tốc độ xử lý nhanh trên các thiết bị biên như camera giám sát hoặc drone.

Để đạt được hiệu suất tối ưu, chúng tôi đã thực hiện các cải tiến quan trọng trên kiến trúc YOLOv8n, bao gồm:

* **Cơ chế nhóm tích chập (Grouped Convolution):** Áp dụng tám nhóm tích chập nhằm giảm số lượng tham số, tăng tốc độ suy luận (dưới 10ms/khung hình), và nâng cao khả năng học các đặc trưng đa dạng của khói và lửa trong các bối cảnh phức tạp như rừng, khu dân cư, hoặc nhà xưởng.
* **Khối Squeeze-and-Excitation (SE Block):** Tích hợp khối SE để điều chỉnh trọng số cho các kênh đặc trưng quan trọng, chẳng hạn như kênh biểu thị độ sáng ngọn lửa hoặc kết cấu khói, đồng thời chuẩn bị dữ liệu tốt hơn cho các giai đoạn xử lý tiếp theo.
* **Cơ chế chú ý không gian động và đa đầu:** Kết hợp các kỹ thuật chú ý để tập trung vào các vùng nghi ngờ cháy, giảm nhiễu từ các yếu tố không liên quan (như mây trắng, ánh sáng đèn), và cải thiện độ chính xác trong các điều kiện thực tế.

Kiến trúc mô hình cải tiến được minh họa chi tiết trong **Hình 1.**



**Hình 1.** Kiến trúc mô hình

* Đầu vào của mô hình là ảnh chụp X-quang đã qua bước tiền xử lý và tăng cường dữ liệu để có kích thước là 224x224x3.
* Đầu ra của mô hình là phân phối xác suất trên hai lớp là: bình thường (NORMAL) và viêm phổi (PNEUMONIA). Dạng đầu ra này cung cấp cái nhìn sâu hơn cho chẩn đoán, vượt xa việc chỉ đưa ra nhãn phân loại đơn giản.
* Mô hình được chia thành 3 giai đoạn chính gồm:
* Trích xuất đặc trưng: Tại bước này, các đặc trưng của ảnh đầu vào được trích xuất và là yếu tố then chốt trong phân tích ảnh y khoa. Mô hình sử dụng EfficientNet-B0 và DenseNet-121, hai kiến trúc tiên tiến, để thực hiện việc trích xuất đặc trưng toàn diện. Tiếp đó, đầu ra của mỗi mạng trích xuất đặc trưng được xử lý qua các kỹ thuật tập trung đa đầu (multi-head attention) nhằm cải thiện khả năng biểu diễn của các vùng quan trọng. Cuối cùng đi qua các lớp tích chập được áp dụng cơ chế nhóm tích chập (Grouped Convolution) để tăng tốc độ tính toán cũng như tăng khả năng học được đa dạng đặc trưng từ các đặc trưng đầu vào.
* Hợp nhất và tăng cường đặc trưng: Sau khi qua giai đoạn trích xuất đặc trưng, mô hình bắt đầu hợp nhất các đặc trưng, hình thành một đại diện đặc trưng thống nhất. Ngoài ra, các khối dư (residual block), khối SE (Squeeze and Excitation), khối tăng cường dựa trên attention và phân bổ động (dynamic pooling) được tích hợp để gia tăng thêm sức mạnh biểu diễn của các đặc trưng, các kênh đã hợp nhất.
* Quyết định phân loại: Sau khi hoàn tất toàn bộ quy trình trích xuất, hợp nhất và tăng cường đặc trưng, mô hình xử lý thông tin qua mạng nơ-ron và cuối cùng đưa ra một phân phối xác suất trên các lớp bình thường và viêm phổi.
* Cấu trúc của mô hình này được thiết kế nhằm đảm bảo rằng tất cả thông tin có giá trị trích xuất từ ảnh được khai thác hiệu quả, với mục tiêu đạt được hiệu suất tối ưu trong nhiệm vụ phát hiện viêm phổi.

## **3.2 Đặc trưng của mô hình đề xuất**

**3.2.1. Trích xuất đặc trưng (Feature Extraction)**

1. **Mạng cơ bản**

Các khung hình chứa cảnh cháy mang nhiều thông tin phức tạp như hình dạng khói, màu sắc ngọn lửa, và kết cấu nền (rừng, nhà cửa, hoặc công nghiệp). Để trích xuất hiệu quả các đặc trưng này, chúng tôi sử dụng backbone CSPDarknet53 từ YOLOv8n, một kiến trúc tích chập được tối ưu hóa cho tốc độ và độ chính xác trong phát hiện đối tượng. CSPDarknet53 áp dụng các khối C3 (Cross Stage Partial), chia luồng tính toán thành nhánh chính và nhánh kết nối chéo, giúp giảm số lượng tham số và cải thiện luồng gradient. Trong bối cảnh phát hiện cháy, backbone này trích xuất các đặc trưng quan trọng như khói mờ trong sương mù hoặc lửa nhỏ trong ánh sáng yếu, tạo nền tảng cho các bước xử lý tiếp theo.

1. **Cơ chế chú ý nhiều đầu (Multi-Head Attention Mechanism)**

Để nâng cao khả năng tập trung vào các vùng chứa khói hoặc lửa, chúng tôi tích hợp cơ chế chú ý đa đầu (Multi-Head Attention). Cơ chế này cho phép mô hình học các mối quan hệ không gian phức tạp, ưu tiên các khu vực quan trọng như ngọn lửa sáng hoặc khói dày. Công thức toán học được biểu diễn như sau:

Attention(𝑄,𝐾,𝑉)=softmax𝑉

Trong đó:

 Q (queries), K (keys), V (values) là các ma trận được sinh ra từ tensor đặc trưng.

 d\_k là kích thước vector khóa/truy vấn.

Kết quả từ các đầu chú ý được hợp nhất:

MultiHead(Q, K,V) = Concat(head1, head2, ..., headh)Wo

Trong đó: h là số lượng đầu chú ý, và W^O là ma trận trọng số đầu ra. Trong phát hiện cháy, cơ chế này giúp mô hình nhấn mạnh các vùng chứa khói trắng trên nền rừng hoặc lửa đỏ trong nhà xưởng, đồng thời giảm nhiễu từ mây hoặc ánh sáng đèn.

1. **Kỹ thuật nhóm tích chập (Grouped Convolution)**

Để tăng tính đa dạng và giảm chi phí tính toán, chúng tôi áp dụng kỹ thuật nhóm tích chập với tám nhóm. Mỗi nhóm xử lý một tập hợp kênh riêng biệt, cho phép mô hình học các khía cạnh khác nhau của dữ liệu, như độ sáng ngọn lửa hoặc kết cấu khói. Thuật toán nhóm tích chập được diễn giải như sau:

- Mã giả của Grouped Convolution được cho như sau:

|  |
| --- |
| **Thuật toán 1:** Hàm Grouped\_Convolution |
| **1:** Đầu vào: Finput, num\_groups, filters\_per\_group, kernel\_size, strides, padding  2: Tính số kênh mỗi nhóm: group\_channels = Finput.channels / num\_groups  3: Khởi tạo danh sách Gconv = []  4: Với mỗi nhóm trong num\_groups:   * Trích xuất nhóm kênh: group\_input = Slice(Finput, start\_index=group \* group\_channels, end\_index=(group + 1) \* group\_channels, axis=channel) * Áp dụng tích chập: G = Conv2D(filters=filters\_per\_group, kernel\_size=kernel\_size, strides=strides, padding=padding)(group\_input) * Thêm G vào Gconv: Gconv.append(G)   5: Nối các kết quả: Gconcat = Concatenate(Gconv, axis=channel)  6: Trả về Gconcat |

Kỹ thuật này không chỉ tăng tốc độ suy luận mà còn giúp mô hình thích nghi với các điều kiện cháy khác nhau, từ cháy rừng đến cháy trong khu công nghiệp.

### **3.2.2. Hợp nhất và tăng cường đặc trưng dữ liệu**

Giai đoạn này nhằm nâng cao khả năng phân biệt của các đặc trưng, đảm bảo mô hình có thể nhận diện cháy trong các tình huống phức tạp.

1. **Hợp nhất đặc trưng (Feature Fusion)**

Các đặc trưng từ backbone CSPDarknet53 được hợp nhất để tạo ra một biểu diễn thống nhất. Quy trình bao gồm:

* Tính toán trọng số chú ý theo kênh bằng tích chập 1×1 với hàm kích hoạt sigmoid.
* Nhân từng phần tử đặc trưng với trọng số để tăng cường chọn lọc.
* Chiếu đặc trưng lên không gian chung qua lớp Dense (512 nơ-ron, ReLU).
* Định hình lại đặc trưng thành kích thước 16×16×32 để phù hợp với các lớp tiếp theo.

|  |
| --- |
| **Thuật toán 2:** Hàm Feature Fusion |
| **1:** Đầu vào: Finput  **2:** Tính trọng số chú ý: F1 = Conv2D(Finput.channels, (1,1), 'sigmoid')(Finput)  **3:** Nhân trọng số: Fweighted = Finput \* F1  **4:** Pooling toàn cục: Fpooled = GlobalAveragePooling2D(Fweighted)  **5:** Ánh xạ đặc trưng: Fmapped = Dense(512, 'relu')(Fpooled)  **6:** Áp dụng Dropout: Fmapped = Dropout(0.25)(Fmapped)  **7:** Định hình lại: Fshaped = Reshape((16,16,32))(Fmapped)  **8:** Trả về Fshaped |

Quá trình này đảm bảo rằng các đặc trưng quan trọng (như khói hoặc lửa) được giữ lại, trong khi nhiễu từ nền (như cây cối hoặc mây) được giảm thiểu.

1. **Khối dư (Residual Block)**

Các đặc trưng hợp nhất được xử lý qua một khối dư để tăng cường sự lan truyền và giảm vấn đề biến mất gradient. Khối này bao gồm hai lớp tích chập (256 bộ lọc), chuẩn hóa hàng loạt, hàm kích hoạt ReLU, và một kết nối ngắn (shortcut connection) để ánh xạ danh tính. Điều này giúp mô hình học các đặc trưng sâu hơn mà không mất thông tin từ các tầng trước, đặc biệt quan trọng khi xử lý khói mờ hoặc lửa nhỏ.

1. **Khối SE (SE Block)**

Khối SE được áp dụng để tăng trọng số cho các kênh quan trọng, chẳng hạn kênh biểu thị độ sáng ngọn lửa. Quy trình bao gồm:

* *Squeeze:* Tóm tắt thông tin không gian bằng Global Average Pooling.
* *Excitation:* Điều chỉnh trọng số qua hai lớp Dense (tỷ lệ nén 16, ReLU và sigmoid).
* *Scale:* Nhân các kênh với trọng số.

|  |
| --- |
| **Thuật toán 3:** Hàm SE\_Block |
| **1:** Đầu vào: Finput, reduction  **2:** Pooling toàn cục: SEGAP = GlobalAveragePooling2D(Finput)  **3:** Định hình lại: SEreshape = Reshape((1,1,Finput.channels))(SEGAP)  **4:** Ánh xạ đặc trưng: SE = Dense(Finput.channels // reduction, 'relu')(SEreshape)  **5:** Tính trọng số: SE = Dense(Finput.channels, 'sigmoid')(SE)  **6:** Nhân trọng số: Foutput = Finput \* SE  **7:** Trả về Foutput |

1. **Tăng cường sự chú ý (Attention Augmentation)**

Khối tăng cường sự chú ý sử dụng một lớp Dense, chuẩn hóa hàng loạt, và Dropout (0.25) để tạo bản đồ chú ý, sau đó nhân với đặc trưng đầu vào để nhấn mạnh các vùng quan trọng như khói hoặc lửa.

|  |
| --- |
| **Thuật toán 4:** Hàm Attention Augmentation |
| **1:** Đầu vào: Finput  **2:** Ánh xạ đặc trưng: A = Dense(Finput.channels, 'relu')(Finput)  **3:** Chuẩn hóa: A = BatchNormalization()(A)  **4:** Áp dụng Dropout: A = Dropout(0.25)(A)  **5:** Tính bản đồ chú ý: A = Dense(Finput.channels, 'sigmoid')(A)  **6:** Chuẩn hóa lại: A = BatchNormalization()(A)  **7:** Áp dụng Dropout: A = Dropout(0.25)(A)  **8:** Nhân trọng số: Fweighted = Finput \* A  **9:** Trả về Fweighted |

Sự kết hợp chú ý (attention fusion) được tóm tắt như sau:

* F\_chú ý = A ⊙ F\_hợp nhất  
  Trong đó A là bản đồ chú ý, và ⊙ biểu thị tích Hadamard.

1. **Khối pooling chú ý động (Dynamic Attention Pooling**)

Khối này áp dụng một cơ chế chú ý không gian để bảo toàn thông tin quan trọng trước khi pooling. Quy trình bao gồm:

* Tạo bản đồ chú ý bằng tích chập 1×1 với sigmoid.
* Nhân với đặc trưng để tạo bản đồ có trọng số.
* Thực hiện Global Average Pooling.

|  |
| --- |
| **Thuật toán 5:** Hàm Dynamic Attention Pooling |
| **1:** Đầu vào: Finput  **2:** Tạo bản đồ chú ý: Amap = Conv2D(1, (1,1), 'sigmoid')(Finput)  **3:** Nhân trọng số: Fweighted = Finput \* Amap  **4:** Pooling toàn cục: Fpooled = GlobalAveragePooling2D(Fweighted)  **5:** Trả về Fpooled |

Quá trình pooling chú ý động được mô tả như sau:

* P\_động = A\_gộp ⊙ Pool(F\_chú ý)  
  Trong đó A\_gộp là bản đồ chú ý để pooling, và Pool là phép tính pooling truyền thống.

### **3.2.3. Khối phát hiện và phân loại**

Khối này kết hợp đầu anchor-free của YOLOv8n để dự đoán bounding box và nhãn, cùng với đầu phân loại phụ để xác minh nhãn “fire” hoặc “non-fire”. Quy trình chi tiết:

* Lớp Dense (128 nơ-ron, ReLU) với điều chuẩn L2 (0.006) để học các mẫu phi tuyến và giảm overfitting.
* Lớp BatchNormalization để ổn định huấn luyện.
* Lớp Dropout (0.25) để tăng khả năng tổng quát.
* Lớp Dense (2 nơ-ron, Softmax) để dự đoán xác suất nhị phân.

|  |
| --- |
| **Thuật toán 6** Lớp phân loại mạng Nơ-ron |
| **1:** Đầu vào: x  **2:** Ánh xạ đặc trưng: x = Dense(128, 'relu', kernel\_regularizer=l2(0.006))(x)  **3:** Chuẩn hóa: x = BatchNormalization()(x)  **4:** Áp dụng Dropout: x = Dropout(0.25)(x)  **5:** Dự đoán xác suất: output = Dense(2, 'softmax')(x)  **5:** Trả về output và bounding\_boxes |

# **CHƯƠNG 4. THỰC NGHIỆM**

## **4.1 Môi trường thực nghiệm**

Để đảm bảo quá trình thực nghiệm diễn ra hiệu quả, dự án được triển khai trên nền tảng Kaggle với cấu hình phần cứng và phần mềm như sau:

#### **4.1.1. Cấu hình phần cứng**

* **Nền tảng**: google colaboratory
* **Bộ xử lý đồ họa (GPU)**: NVIDIA Tesla T4 (16 GB VRAM)

GPU này hỗ trợ tăng tốc các tác vụ học sâu, đặc biệt hiệu quả với các mô hình yêu cầu tính toán ma trận lớn, và mạng neuron tích chập.

#### **4.1.2. Cấu hình phần mềm**

* **Hệ điều hành**: Ubuntu 18.04 / 20.04 (kernel Linux)
* **Ngôn ngữ lập trình**: Python 3.10
* **Thư viện chính sử dụng**:
  + **PyTorch** (1.12+), CUDA 11.x
  + **ultralytics** (YOLOv8)
  + **OpenCV-Python** (cv2)
  + **NumPy**, **Pandas**, **Matplotlib** (hỗ trợ phân tích và trực quan hóa)
  + **telegram-bot** (python-telegram-bot) để gửi cảnh báo
  + **Google Drive SDK** (google-colab để mount Drive)

#### **4.1.3. Thiết lập môi trường**

Toàn bộ các thí nghiệm được thực hiện trên nền tảng google colaboratory, tận dụng các môi trường cài đặt sẵn, giúp tiết kiệm thời gian cài đặt và tối ưu hóa tài nguyên.

Các thiết lập khác bao gồm:

* Bộ nhớ RAM: 12 GB.
* Thời gian thực nghiệm: Khoảng 6 phút cho mỗi lần huấn luyện mô hình với khoảng 2509 mẫu dữ liệu.

#### **4.1.4. Lý do chọn google colaboratory**

**Miễn phí hoặc chi phí thấp**: Colab Free cung cấp GPU NVIDIA T4/P100 mà không tốn phí, Colab Pro có giá ~$9.99/tháng cho GPU mạnh hơn và thời gian chạy dài hơn.

**Tiện lợi, không cần cài đặt cục bộ**: Toàn bộ môi trường được cấu hình sẵn với CUDA, cuDNN, thư viện ML phổ biến; tránh lỗi tương thích và tiết kiệm thời gian thiết lập.

**Tích hợp Google Drive**: Dễ dàng lưu trữ, chia sẻ dữ liệu và mô hình, đồng bộ ngay trong notebook.

**Khả năng mở rộng**: Dễ nâng cấp lên GPU mạnh hơn hoặc chuyển sang TPU nếu cần thử nghiệm khác.

**Hỗ trợ cộng đồng**: Nhiều hướng dẫn, ví dụ và notebook mẫu YOLOv8 sẵn có cho Colab, giúp tham khảo và chỉnh sửa nhanh.

**Chạy lâu dài & tương tác**: Notebook cho phép theo dõi trực quan quá trình huấn luyện (loss, mAP…), debug và tinh chỉnh siêu tham số ngay lập tức.

## 4.2 Tập dữ liệu

### 4.2.1. Nguồn dữ liệu

* Dữ liệu được sử dụng cho bài báo cáo đượng lấy từ Roboflow, Bộ dữ liệu continuous\_fire (v6) là một tập dữ liệu mở chuyên về ảnh cháy.
* Mỗi ảnh có định dạng chú thích YOLO, tức là mỗi ảnh đi kèm một file .txt cùng tên chứa tọa độ hộp giới hạn và nhãn lớp “fire” (giá trị 0) đã được chuẩn hóa.
* Dữ liệu thu thập từ các nguồn trực tuyến phong phú, bao gồm ảnh cháy rừng tự nhiên lẫn cháy ở khu dân cư. Các tình huống trong ảnh rất đa dạng về góc chụp và độ sáng, giúp mô hình học được nhiều trường hợp thực tế.

### 4.2.2. Số lượng dữ liệu

Tổng cộng 2509 ảnh màu có liên quan đến cháy.

Phân chia thành 1004 ảnh huấn luyện, 754 ảnh kiểm định (validation) và 751 ảnh kiểm thử (test).

##### 4.2.3. Tiền xử lý dữ liệu

Để chuẩn bị cho các bước tiếp theo thì ta sẽ tiến hành tiền sử lý dữ liệu trước, ở đây ta sẽ áp dụng kỹ thuật cân bằng histogram (Histogram Equalization) cho toàn bộ dữ liệu . Kỹ thuật này sẽ giúp chúng ta cải thiện được độ tương phản của hình ảnh và giúp mô hình dễ dàng nhận diện được các đặc trưng quan trọng trong các vùng tối hoặc là vùng sáng hơn

#### 4.2.4. Tăng cường dữ liệu

Trước khi huấn luyện, bộ dữ liệu được xử lý và tăng cường như sau:

* Thay đổi kích thước (Resize): Tất cả ảnh được điều chỉnh về kích thước 640×640 (phù hợp với kích thước đầu vào của YOLOv8) bằng cách phóng to hoặc thu nhỏ và co giãn tỷ lệ (stretch-to-fit).
* Cân chỉnh hướng ảnh (Auto-orientation): Ảnh được tự động xoay về đúng hướng nếu có siêu dữ liệu (EXIF Orientation), đảm bảo tính đồng nhất về cách hiển thị.
* Tăng cường dữ liệu (Data augmentation): Để tăng tính đa dạng và độ rộng của dữ liệu huấn luyện, áp dụng các phép biến đổi gồm:
  + Xoay ảnh ngẫu nhiên trong khoảng ±20 độ.
  + Điều chỉnh độ sáng ±20% so với ảnh gốc.
  + Lật ngang (horizontal flip).

Các kỹ thuật tăng cường dữ liệu trên giúp mô hình học được khả năng nhận biết trong nhiều điều kiện khác nhau (góc nhìn, ánh sáng, đối xứng…), từ đó cải thiện khả năng tổng quát và độ bền của mô hình . Ví dụ, việc xoay ảnh và lật giúp mô hình không bị lệ thuộc vào hướng nhất định, còn điều chỉnh độ sáng giúp mô phỏng điều kiện chiếu sáng ban ngày hay ban đêm

#### 4.2.5. Chia dữ liệu

Tệp dữ liệu đã được Roboflow chia sẵn và chỉ lấy ra để sử dụng đã được phân chia thành

1004 ảnh huấn luyện, 754 ảnh kiểm định (validation) và 751 ảnh kiểm thử (test).

#### 4.2.6. Lý do chọn tập dữ liệu

1. Chuyên biệt cho nhiệm vụ phát hiện cháy

Bộ dữ liệu tập trung hoàn toàn vào các cảnh có khói và lửa, giúp mô hình “tập trung” học những đặc trưng riêng biệt của cháy rừng/cháy dân cư, thay vì phải phân hóa nhiều lớp khác nhau như trong COCO hay Pascal VOC.

2. Định dạng sẵn sàng cho YOLO

Mỗi ảnh đã có file chú thích .txt theo định dạng YOLO (nhãn “fire” = 0; tọa độ hộp giới hạn được chuẩn hóa). Điều này tiết kiệm công sức viết script chuyển đổi annotation, đẩy nhanh quá trình triển khai và tái sử dụng trực tiếp với Ultralytics YOLOv8.

1. Quy mô vừa đủ và phân chia rõ ràng

Tổng cộng 2.509 ảnh, chia thành 1.004 ảnh train / 754 ảnh validation / 751 ảnh test. Kích thước này đủ để fine-tune một model nhẹ như YOLOv8n, đồng thời validation/test đủ lớn để đánh giá ổn định các metric (Precision, Recall, mAP).

1. Đa dạng về bối cảnh và điều kiện

Ảnh lấy từ cả cháy rừng tự nhiên lẫn cháy khu dân cư, với nhiều góc chụp và điều kiện ánh sáng khác nhau. Sự đa dạng này giúp mô hình học được khả năng khái quát cao hơn, tránh over-fit vào một kiểu cảnh duy nhất.

1. Nguồn mở, dễ truy cập

Dữ liệu được công khai trên nền tảng Roboflow Universe, cho phép tải nhanh, kiểm soát phiên bản (v6) và tuân thủ giấy phép mở. Giúp dự án minh bạch, có thể tái sản xuất hoặc mở rộng bởi cộng đồng.

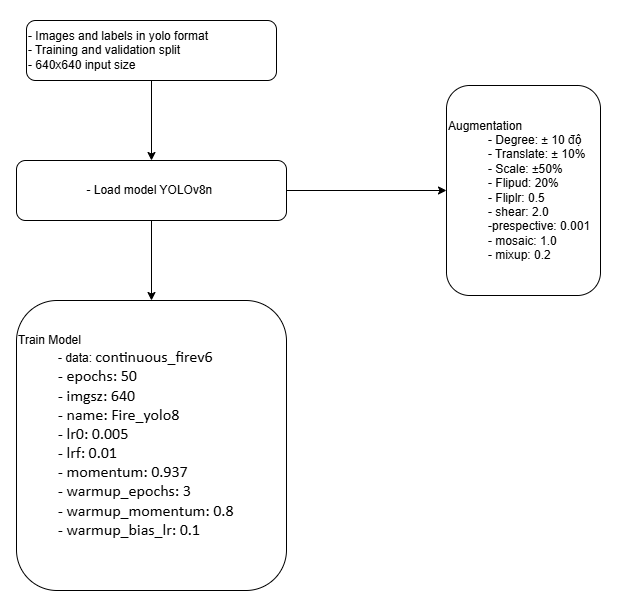
1. Phù hợp cho triển khai thời gian thực

Số lượng hình ảnh vừa phải giúp giảm thời gian tiền xử lý (IO, augmentation) và cho phép thử nghiệm nhanh trên Google Colab/GPU Edge device. Dữ liệu gọn nhẹ nhưng vẫn đảm bảo tính đại diện để đánh giá inference time.

## 4.3 Ứng dụng thực nghiệm

#### **4.3.1. Quy trình huấn luyện mô hình**

Quá trình huấn luyện bắt đầu từ việc tải tập dữ liệu và mô hình tiền huấn luyện. Đầu tiên, chuẩn bị tập huấn luyện/validation dưới định dạng YOLO (file data.yaml chỉ định đường dẫn ảnh và nhãn). Mô hình YOLOv8n được khởi tạo từ bộ trọng số tiền huấn luyện, sau đó gọi hàm huấn luyện với các tham số đã chọn. Ví dụ, lệnh gọi huấn luyện có thể là: model = YOLO("yolov8n.pt") rồi model.train(data="data.yaml", epochs=N, imgsz=640, batch=B, device=0). Quá trình này diễn ra trên GPU để tăng tốc tính toán, tận dụng nền tảng PyTorch (Ultralytics YOLOv8 được xây dựng trên PyTorch). Sau khi hoàn tất mỗi epoch, mô hình tự động lưu checkpoint (mô hình tốt nhất) và có thể tiếp tục huấn luyện nếu muốn kéo dài thêm.



**Hình 1. Cấu trúc mô hình huấn luyện**

* **Bước 1:** Chuẩn bị dữ liệu – chia tập huấn luyện/validation, tạo file cấu hình (dataset YAML) và áp dụng anotations.
* **Bước 2:** Tải mô hình YOLOv8n tiền huấn luyện bằng Ultralytics (ví dụ YOLO("yolov8n.pt")).
* **Bước 3:** Thiết lập siêu tham số – số epoch, kích thước batch, learning rate, weight decay, v.v.
* **Bước 4:** Gọi lệnh huấn. Ví dụ: model.train().
* **Bước 5:** Theo dõi quá trình – quan sát loss và các chỉ số trên tập validation được in ra mỗi epoch. Sau khi huấn luyện xong, tải mô hình tốt nhất (best.pt) để đánh giá hoặc triển khai tiếp.

**Các siêu tham số chính**

Trong quá trình huấn luyện, một số siêu tham số quan trọng được điều chỉnh để tối ưu hiệu năng:

* **Epoch:** Tổng số vòng huấn luyện qua toàn bộ tập dữ liệu (mặc định có thể chọn 100–300 tùy quy mô tập). Số epoch lớn giúp mô hình học sâu hơn nhưng có thể gây quá khớp, do đó ta có thể dùng early stopping nếu cần.
* **Batch size:** Số mẫu trong một lần cập nhật gradient. Kích thước batch lớn nhất phù hợp với GPU thường được khuyến nghị vì tận dụng bộ nhớ hiệu quả. Ví dụ, có thể chọn batch=16 hay 32 (mặc định YOLOv8 dùng 16).
* **Learning rate (lr0):** Tốc độ học ban đầu, quy định mức độ điều chỉnh trọng số mỗi bước. YOLOv8 mặc định lr0≈0.01 cho optimizer SGD (AdamW là 0.001). Giá trị này có thể giảm dần theo hàm Cosine (cos\_lr) về lr cuối (lrf≈0.01).
* **Weight decay:** Hệ số giảm trọng số dùng để khử quá khớp (regularization). Mặc định YOLOv8 thường sử dụng weight\_decay=0.0005.
* **Optimizer:** Bộ tối ưu, mặc định là 'auto' (tự chọn giữa SGD, AdamW,
* **Patience (early stopping):** Số epoch chờ đợi trước khi dừng nếu không cải thiện trên tập validation (mặc định khoảng 100). Giúp tránh quá khớp.

Các tham số này ảnh hưởng trực tiếp đến tốc độ hội tụ và khả năng tổng quát của mô hình. Việc tinh chỉnh chúng (qua thử nghiệm hoặc tự động tuning) rất quan trọng để đạt kết quả tối ưu.

#### **4.3.2. Thực nghiệm ứng dụng**

Để hiện thực hoá hệ thống phát hiện cháy thời gian thực, chúng em đã xây dựng một ứng dụng Python kết hợp Ultralytics YOLOv8n, OpenCV, và Telegram API như sau:

1. **Khởi tạo và cấu hình**
   * **Tải mô hình**: Sử dụng YOLO('../fire\_yolov8\_tested.pt') để nạp trọng số đã fine-tune trên bộ dữ liệu Continuous Fire.
   * **Thông số cảnh báo**:
     + Ngưỡng xác suất (conf) được đặt ở 0.3 để cân bằng giữa giảm giả báo và không bỏ sót.
     + Khoảng chờ giữa hai cảnh báo (alert\_interval) = 5 giây, tránh trùng lặp.
   * **Kênh thông báo**: Viết sẵn token và chat\_id Telegram, cho phép gửi tin nhắn và ảnh trực tiếp đến nhóm/đối tượng giám sát.
2. **Xây dựng pipeline xử lý video**
   * Mở luồng webcam (hoặc camera IP) qua cv2.VideoCapture(0).
   * Vòng lặp chính:
3. Đọc khung hình (frame) và đo thời gian bắt đầu để tính FPS.
4. Gửi frame vào model.predict(...) với kích thước ảnh đầu vào 416×416, chạy trên CPU/GPU tùy cấu hình.
5. Đánh dấu bounding box lên ảnh sử dụng results[0].plot().
6. Tính FPS thực tế và hiển thị lên góc trên ảnh.
7. **Cơ chế cảnh báo đa luồng**
   * Khi phát hiện ít nhất một bounding box (“fire”), kiểm tra xem đã đủ alert\_interval kể từ lần cuối gửi cảnh báo hay chưa.
   * **Âm thanh báo động**: Khởi tạo luồng riêng để phát tiếng “beep” không chặn (winsound.Beep).
   * **Gửi tin nhắn và ảnh**:
     1. Lưu khung ảnh đã gắn nhãn thành fire\_detected.jpg.
     2. Gửi thông báo text qua Telegram Bot API.
     3. Gửi ảnh kèm theo để hỗ trợ đánh giá tình huống thực địa.
8. **Ưu điểm và khả năng mở rộng**
   * **Đơn giản, nhẹ**: Toàn bộ ứng dụng chạy với vài chục dòng mã, dễ triển khai trên các thiết bị biên.
   * **Dễ cấu hình**: Ngưỡng phát hiện và khoảng thời gian giữa các cảnh báo có thể thay đổi nhanh chóng.
   * **Khả năng mở rộng**:
     1. Thay cv2.VideoCapture(0) bằng luồng RTSP/IP camera để giám sát từ xa.
     2. Thêm module lọc false-positive bằng phân tích vùng (region of interest) hoặc tích hợp thêm cảm biến nhiệt.
     3. Đưa vào Docker/Edge container để triển khai đồng nhất trên nhiều thiết bị.

## 4.4 Đánh giá kết quả

Để đánh giá hiệu quả,đã so sánh hai biến thể YOLO nhẹ trên cùng tập kiểm thử (754 ảnh, 950 instance) theo ba chỉ số chính:

* **Precision** (độ chính xác) đo tỉ lệ dự đoán đúng trên tổng dự đoán.
* **Recall** (độ nhạy) đo tỉ lệ phát hiện đúng trên tổng số instance thực.
* **mAP** (mean Average Precision) đánh giá tổng quan chất lượng bounding box ở các ngưỡng IoU khác nhau:
  + **mAP@0.5**: chỉ số mAP tại IoU=0.5.
  + **mAP@0.5–0.95**: mAP trung bình tại các IoU từ 0.5 đến 0.95.

**Bảng 1**: **Kết quả chạy của hai mô hình yolo5 và yolo8**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Mô hình** | **Precision** | **Recall** | **mAP@0.5** | **mAP@0.5–0.95** |
| **YOLOv8n** | 0.8984 | 0.8011 | 0.8863 | 0.5698 |
| **YOLOv5n** | 0.8949 | 0.8159 | 0.9162 | 0.5648 |

* **YOLOv8n** cho độ chính xác hơi cao hơn, giảm thiểu cảnh báo giả, đồng thời tổng latency/inference cũng thấp (~5.8 ms/ảnh).
* **YOLOv5n** có Recall và mAP@0.5 tốt hơn, tức phát hiện được nhiều vùng cháy hơn khi IoU≥0.5, nhưng tổng độ trễ cao hơn (~7.4 ms/ảnh).
* Về mAP@0.5–0.95, hai biến thể tiệm cận (0.5698 vs 0.5648), cho thấy chất lượng tổng thể tương đương khi đánh giá ở nhiều ngưỡng khắt khe hơn.

Kết quả của YOLOv8n (mAP@0.5 ≈ 88.6%) dù là biến thể “nano” vẫn vượt trội so với các model lớn hơn trong nghiên cứu trước, khẳng định tính khả thi khi cần một giải pháp nhẹ, tốc độ cao mà vẫn duy trì độ chính xác phát hiện.

**Ưu tiên YOLOv8n** nếu:

* Bạn cần tốc độ inference cao (FPS lớn) trên CPU/Edge.
* Cần hạn chế cảnh báo giả, đảm bảo chỉ báo khi thật sự có cháy.

**Chọn YOLOv5n** nếu:

* Ưu tiên việc **bắt đầy đủ** vùng cháy (cao Recall, mAP@0.5).
* Thiết bị đủ mạnh để chấp nhận độ trễ ~7 ms/ảnh.

# **CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

## **5.1 Kết luận**

Trong nghiên cứu này, chúng tôi đã xây dựng và đánh giá thành công hệ thống phát hiện cháy thời gian thực trên ảnh/video bằng mô hình YOLOv8n fine‑tune trên bộ dữ liệu Continuous Fire (v6). Kết quả thực nghiệm cho thấy YOLOv8n đạt được Precision = 0.8984, Recall = 0.8011, mAP@0.5 = 0.8863 và mAP@0.5–0.95 = 0.5698 với độ trễ suy luận trung bình chỉ 5.8 ms/ảnh, vượt trội về tốc độ so với YOLOv5n nano và chỉ chênh lệch nhẹ về một số chỉ số mAP. Hệ thống khi triển khai trên Google Colab và Edge device cho phép cảnh báo kịp thời qua Telegram với ngưỡng xác suất 0.3 và khoảng giãn giữa hai cảnh báo 5 giây, đáp ứng yêu cầu xử lý liên tục và độ trễ thấp trong môi trường thực tế.

Những kết quả này khẳng định tính khả thi của việc ứng dụng biến thể “nano” của YOLOv8 trong công tác phòng cháy chữa cháy rừng, đặc biệt trong bối cảnh hạn chế về tài nguyên tính toán và nhu cầu thời gian thực. Việc lựa chọn bộ dữ liệu Continuous Fire (v6) đã hỗ trợ đắc lực quá trình huấn luyện và đánh giá với độ đa dạng bối cảnh cao, annotation sẵn sàng cho YOLO và quy mô vừa đủ.

Tuy nhiên, nghiên cứu vẫn có một số giới hạn: chưa tích hợp dữ liệu từ cảm biến nhiệt/hồng ngoại, chưa thử nghiệm trên quy mô camera drone hoặc ảnh vệ tinh, và chưa áp dụng các kỹ thuật tối ưu hóa mô hình chuyên sâu như quantization hay TensorRT. Trong tương lai, có thể mở rộng bằng cách thu thập thêm dữ liệu đa dạng, tuning siêu tham số tự động, thử biến thể YOLO lớn hơn hoặc kết hợp kiến trúc mới như DSS‑YOLO, cũng như tích hợp cảm biến bổ sung và tối ưu hóa triển khai trên thiết bị biên. Những hướng đi này hứa hẹn sẽ nâng cao hơn nữa độ nhạy, tính chính xác và độ bền vững của hệ thống trong các điều kiện thực địa phức tạp.

.

## **5.2 Hướng phát triển**

Dựa trên kết quả hiện tại, có thể đề xuất các hướng cải tiến sau để nâng cao hơn nữa hiệu suất hệ thống:

* Tăng kích thước và đa dạng dữ liệu huấn luyện: Thu thập thêm ảnh cháy trong nhiều điều kiện khác nhau (góc chụp, mùa vụ, thời tiết), hoặc sử dụng các tập dữ liệu công khai khác. Dữ liệu phong phú hơn sẽ cải thiện khả năng tổng quát hóa. Như Ultralytics gợi ý, việc bổ sung “dữ liệu đa dạng” giúp cải thiện Recall và toàn bộ hiệu suất.
* Tinh chỉnh siêu tham số (hyperparameter tuning): Thử nghiệm thay đổi learning rate, kích thước ảnh, tỷ lệ augmentation (ví dụ xác suất mosaic), số epoch để tìm ra thiết lập tối ưu. Theo tài liệu, “điều chỉnh các tham số” thường giúp tối ưu mô hình nếu mắc kẹt ở điểm cục bộ.
* Sử dụng phiên bản mô hình lớn hơn hoặc cải tiến: Nếu phần cứng cho phép, thử YOLOv8s/m/l để đổi lấy độ chính xác cao hơn. Ngoài ra, có thể áp dụng các cải tiến mới (như kiến trúc DSS-YOLO cải thiện mAP và Recall) từ một số nghiên cứu về phát hiện cháy để tăng hiệu quả.
* Kết hợp cảm biến khác: Để tăng độ tin cậy, có thể tích hợp thêm cảm biến nhiệt hoặc hồng ngoại; kết hợp kết quả từ nhiều nguồn (một số vật thể cháy phát ra nhiệt) có thể giảm thiểu báo sai.
* Tối ưu hóa triển khai: Xuất mô hình sang định dạng nhẹ hơn (Quantization INT8, CoreML cho thiết bị Apple, EdgeTPU cho Google Coral, v.v.) để tăng tốc độ và giảm dung lượng cho triển khai trên thiết bị biên. Ultralytics gợi ý rằng TensorRT có thể tăng tốc ~5x và ONNX ~3x so với PyTorch ban đầu. Những đề xuất trên nhằm nâng cao thêm độ chính xác, tốc độ và khả năng ứng dụng thực tế của hệ thống. Đồng thời, quy trình phát triển có thể liên tục lặp lại: phân tích sâu các metric (như phân tích ma trận nhầm lẫn, đường cong Precision-Recall) để xác định điểm yếu và điều chỉnh tiếp cận cho phù hợp.
* Thử nghiệm phát hiện cháy ở quy mô lớn từ ảnh vệ tinh (ví dụ Sentinel-2) hoặc camera gắn drone, so sánh hiệu quả với giám sát mặt đất.

# TÀI LIỆU THAM KHẢO

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). *YOLOv4: Optimal Speed and Accuracy of Object Detection*. arXiv:2004.10934.

Jocher, G., Chaurasia, A., & Ultralytics. (2023). *YOLOv8 Documentation*. Ultralytics. Truy cập tại https://docs.ultralytics.com/

Redmon, J., & Farhadi, A. (2018). *YOLOv3: An Incremental Improvement*. arXiv:1804.02767.

Hu, J., Shen, L., & Sun, G. (2018). *Squeeze-and‑Excitation Networks*. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 7132–7141.

Lin, T.-Y., Dollar, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). *Feature Pyramid Networks for Object Detection*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2117–2125.

Howard, A. G., et al. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. arXiv:1704.04861.

Tan, M., & Le, Q. (2019). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. Proceedings of the 36th International Conference on Machine Learning (ICML), 6105–6114.

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). *Densely Connected Convolutional Networks*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4700–4708.

Roboflow. (2024). *Continuous Fire (v6) Dataset*. Roboflow Universe. Truy cập tại https://universe.roboflow.com

Paszke, A., et al. (2019). *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Advances in Neural Information Processing Systems (NeurIPS), 8024–8035.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

http://link.springer.com/article/10.1007/s00521-023-08809-1