**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC VÀ KỸ THUẬT THÔNG TIN**

---------

A blue logo with a black background

AI-generated content may be incorrect.

**BÁO CÁO ĐỒ ÁN**

**Viết trang web để triển khai nhận dạng lửa/khói trong Datacenter**

**Sinh viên thực hiện:**

Hoàng Minh Nhật - MSSV: 24550031

Đoàn Chí Hưng - MSSV: 24550014

**Giảng viên:**

Nguyễn Tấn Cầm

Thành phố Hồ Chí Minh, tháng 6 năm 2025

**BÁO CÁO TÓM TẮT**

**1. Tiêu đề báo cáo: Trang web triển khai nhận dạng lửa/khói trong datacenter**

**2.** **Danh sách thành viên**

|  |  |  |
| --- | --- | --- |
| **MSSV** | **Họ tên** | **Ghi chú** |
| 24550031 | Hoàng Minh Nhật |  |
| 24550014 | Đoàn Chí Hưng |  |

**3.** **Nội dung chính**

***Nội dung 1: Giới thiệu và mục tiêu của đồ án***

***Nội dung 2: Công nghệ sử dụng***

***Nội dung 3: Quy trình thực hiện***

***Nội dung 4: Kết quả và đánh giá***

***Nội dung 5: Tài liệu tham khảo***

Table of Contents

[I. Giới thiệu và mục tiêu của đồ án 1](#_Toc200832586)

[1. Giới thiệu đồ án 1](#_Toc200832587)

[2. Xác định mục tiêu của đồ án 1](#_Toc200832588)

[3. Hướng phát triển và công nghệ được đề xuất 1](#_Toc200832589)

[3.1. Lựa chọn model phù hợp nhất 1](#_Toc200832590)

[3.2. Lựa chọn Object detection model phù hợp nhất 3](#_Toc200832591)

[3.3. Lựa chọn Framework tối ưu nhất cho Web App 4](#_Toc200832592)

[II. Công nghệ sử dụng 5](#_Toc200832593)

[1. Framework và thư viện cho quá trình learning 5](#_Toc200832594)

[2. Nền tảng 6](#_Toc200832595)

[3. Thư viện phát triển web 6](#_Toc200832596)

[- Gradio: Framework dành cho ngôn ngữ lập trình python, với nhiều high level API hỗ trợ việc tích hợp mô hình học máy vào trang web 6](#_Toc200832597)

[4. Thuộc tính của mô hình cuối cùng 6](#_Toc200832598)

[4.1. Kiểu Học (Learning Type) 6](#_Toc200832599)

[4.2. Kiểu Mô hình (Model Type) 6](#_Toc200832600)

[4.3. Mô hình Tiền Huấn luyện (Pre-trained Model) 6](#_Toc200832601)

[4.4. Ultralytics và YOLO 6](#_Toc200832602)

[III. Quy trình thực hiện 9](#_Toc200832603)

[1. Tổng quan cấu trúc dataset 9](#_Toc200832604)

[2. Tiền sử lí và tăng cường dữ liệu 10](#_Toc200832605)

[3. Phân tích dữ liệu khai phá (EDA) 11](#_Toc200832606)

[B.1. Kiểm tra số lượng hình ảnh và class tương ứng: 11](#_Toc200832607)

[B.2. Xác minh trực quan rằng các chú thích là chính xác trước khi tiến hành huấn luyện mô hình. 11](#_Toc200832608)

[B.3 Phân tích phân bổ nhãn trong dataset 14](#_Toc200832609)

[B.4. Kiểm tra nhãn bị thiếu 16](#_Toc200832610)

[B.5. Kết luận 16](#_Toc200832611)

[4. Quy trình huấn luyện 17](#_Toc200832612)

[5. Đánh giá mô hình và quá trình training 19](#_Toc200832613)

[6. Web App Flow và Source Code 21](#_Toc200832614)

[6.1. Tóm tắt source code 22](#_Toc200832615)

[6.2. Sơ hồ hoạt động của hệ thống 23](#_Toc200832616)

[6.3. Giao diện website 26](#_Toc200832617)

[IV. Kết quả và đánh giá 27](#_Toc200832618)

[1. Tổng kết đồ án 27](#_Toc200832619)

[2. Ưu điểm 27](#_Toc200832620)

[3. Nhược điểm 28](#_Toc200832621)

[4. Hướng phát triển trong tương lai 28](#_Toc200832622)

[5. Link các sản phẩm của đồ án 29](#_Toc200832623)

[V. TÀI LIỆU THAM KHẢO 30](#_Toc200832624)

MỤC LỤC BẢNG BIỂU

[Hình 1. So sánh các Vision Model 4](#_Toc200826910)

[Hình 2 So sánh metríc giữa các Object Detection Model trên nền tảng Roboblow 5](#_Toc200826911)

[Hình 3. Tổng quan quy trình thực hiện 10](#_Toc200826912)

[Hình 4. Sơ đồ cấu trúc dataset 10](#_Toc200826913)

[Hình 5. Cấu trúc file data.yaml 11](#_Toc200826914)

[Hình 6. Sơ đồ quy trình tiền sử lý và tăng cường dữ liệu 11](#_Toc200826915)

[Hình 7. Tổng số hình ảnh và class tương ứng trong dataset 12](#_Toc200826916)

[Hình 8. Code định nghiã hàm tạo hộp có viền 12](#_Toc200826917)

[Hình 9. Output ảnh trong dataset đã được vẽ viền 13](#_Toc200826918)

[Hình 10. Code vẽ đồ thị thể hiện tỉ lệ phân bổ các class 14](#_Toc200826919)

[Hình 11. Output đồ thị phân bổ class 15](#_Toc200826920)

[Hình 12. Code kiểm tra nhãn bị thiếu trong dataset 16](#_Toc200826921)

[Hình 13. Output xác nhận không có ảnh bị thiếu nhãn 16](#_Toc200826922)

[Hình 14. Code mẫu chạy quy trình huấn luyện 17](#_Toc200826923)

[Hình 15. Sơ đồ tóm tắt quy trình huấn luyện 17](#_Toc200826924)

[Hình 16. Code lưu model vừa được tạo và lưu best weight làm model chính 19](#_Toc200826925)

[Hình 17. Code in biểu đồ training 19](#_Toc200826926)

[Hình 18. Biểu đồ training trên roboflow 20](#_Toc200826927)

[Hình 19. Metrics của model cuôí cùng 20](#_Toc200826928)

[Hình 20. Code tóm tắt Web App 21](#_Toc200826929)

[Hình 21. Sơ đồ hoạt động 22](#_Toc200826930)

[Hình 22.Giao diện Website 24](#_Toc200826931)

# Giới thiệu và mục tiêu của đồ án

## Giới thiệu đồ án

Phát hiện khói và lửa là một nhiệm vụ tối quan trọng để đảm bảo an toàn công cộng và ngăn ngừa các rủi ro tại trung tâm dữ liệu. Với những tiến bộ gần đây trong lĩnh vực Thị giác máy tính (Computer Vision) và Học sâu (Deep Learning), việc xây dựng các hệ thống phát hiện khói và lửa chính xác bằng cách sử dụng tập dữ liệu tùy chỉnh đã trở nên khả thi. Một hệ thống như vậy là YOLO, một mô hình phát hiện đối tượng (object detection model) hiện đại, có thể được huấn luyện (trained) trên một tập dữ liệu tùy chỉnh để phát hiện khói và lửa.

## Xác định mục tiêu của đồ án

* Phát triển một object detection model chuyên dụng trong việc phát hiện khói lửa dựa vào một dataset với dữ liệu có sẵn.
* Tải xuống và đánh giá tập dữ liệu phát hiện khói và lửa.
* Chuẩn bị dữ liệu cho quá trình huấn luyện mô hình phát hiện đối tượng.
* Huấn luyện mô trên tập dữ liệu tùy chỉnh để phát hiện khói và lửa.
* Đánh giá hiệu suất của mô hình đã được huấn luyện.
* Lưu trữ và kiểm thử mô hình đã huấn luyện.
* Áp dụng model này lên một web app mà người dùng có thể tương tác và kiểm thử mô hình nhận diện trên cả video và hình ảnh

## Hướng phát triển và công nghệ được đề xuất

### Lựa chọn model phù hợp nhất

Bài toán đặt ra là phát hiện sự hiện diện của khói và lửa trong hình ảnh hoặc video, đồng thời phải xác định được chính xác vị trí của chúng để đưa ra cảnh báo kịp thời. Để giải quyết yêu cầu này, nhóm đã xem xét các loại mô hình Thị giác máy tính phổ biến:

* **Image Classification (Phân loại ảnh):** Mô hình này chỉ có thể trả lời câu hỏi "có" hoặc "không" có khói/lửa trong một bức ảnh. Nó không thể cho biết đối tượng nằm ở đâu. Điều này không đáp ứng được yêu cầu về việc khoanh vùng đối tượng để cảnh báo chính xác.
* **Image Segmentation (Phân đoạn ảnh):** Mô hình này cung cấp thông tin chi tiết nhất bằng cách phân loại từng pixel thuộc về đối tượng nào. Tuy nhiên, việc xử lý ở cấp độ pixel đòi hỏi tài nguyên tính toán rất lớn và có tốc độ xử lý chậm, không phù hợp cho các ứng dụng cần phản hồi thời gian thực như hệ thống cảnh báo cháy.
* **Object Detection (Phát hiện đối tượng):** Đây là phương pháp dung hòa được ưu điểm của hai loại trên. Mô hình không chỉ xác định sự tồn tại của đối tượng (classification) mà còn khoanh vùng vị trí của chúng bằng các hộp giới hạn (bounding box). Quan trọng hơn, các mô hình Object Detection hiện đại được tối ưu hóa để có thể nhận diện nhiều đối tượng cùng lúc với tốc độ xử lý cao, tiệm cận thời gian thực.

A diagram of a computer model

AI-generated content may be incorrect.

Hình . So sánh các Vision Model

**Kết luận:** Dựa trên các phân tích trên, **Object Detection** là loại mô hình phù hợp nhất với mục tiêu của đồ án: vừa đảm bảo độ chính xác trong việc nhận diện, vừa xác định được vị trí của khói/lửa, lại vừa có tốc độ xử lý đủ nhanh cho các ứng dụng thực tế.

### Lựa chọn Object detection model phù hợp nhất

A graph of a graph with different colored bars

AI-generated content may be incorrect.

Hình So sánh metríc giữa các Object Detection Model trên nền tảng Roboblow

Sau khi xác định hướng đi là Object Detection, bước tiếp theo là lựa chọn một model cụ thể. Thị trường hiện có nhiều kiến trúc nổi bật như series YOLO, SSD, Faster R-CNN, v.v. Để đưa ra quyết định, nhóm đã tiến hành so sánh và đánh giá các mô hình dựa trên các tiêu chí về hiệu năng, tốc độ, sự hỗ trợ từ cộng đồng và tính thực tiễn.

Qua bảng so sánh metrics của một số model được huấn luyện trên cùng một bộ dữ liệu, có thể thấy các model mới như YOLOv11 hay Roboflow 3.0 OD cho kết quả về độ chính xác (mAP, Precision, Recall) nhỉnh hơn một chút. Tuy nhiên, khi xem xét một cách toàn diện, **YOLOv8** được lựa chọn là model cuối cùng cho đồ án này vì những lý do chiến lược sau:

* **Sự cân bằng giữa hiệu năng và tính thực tiễn:** Mặc dù metrics của YOLOv8 thấp hơn một chút trong phép so sánh cụ thể này, nhưng mức chênh lệch không quá lớn và hiệu suất của nó vẫn ở mức rất cao (mAP@50 đạt 97.0%). Quan trọng hơn, YOLOv8 mang lại sự cân bằng tuyệt vời giữa độ chính xác và tốc độ xử lý, một yếu tố sống còn cho các ứng dụng thời gian thực.
* **Sự trưởng thành và cộng đồng hỗ trợ lớn:** YOLOv8 được phát triển bởi Ultralytics, một tổ chức uy tín với lịch sử phát triển các phiên bản YOLO thành công. Nhờ vậy, YOLOv8 có một hệ sinh thái hoàn chỉnh: tài liệu hướng dẫn chi tiết, cộng đồng người dùng đông đảo, và rất nhiều bộ dữ liệu (dataset) đã được tối ưu hóa sẵn. Điều này giúp quá trình triển khai, gỡ lỗi và phát triển sau này trở nên thuận lợi hơn rất nhiều.
* **Dễ dàng sử dụng và tích hợp:** Kiến trúc của YOLOv8 được thiết kế để đơn giản hóa quá trình huấn luyện, kiểm thử và xuất model. Việc này giúp giảm đáng kể thời gian phát triển, cho phép nhóm tập trung nhiều hơn vào việc tinh chỉnh dữ liệu và ứng dụng cuối.
* **Tính linh hoạt và khả năng mở rộng:** YOLOv8 không chỉ là một model mà là một framework hoàn chỉnh, hỗ trợ nhiều tác vụ khác nhau ngoài Object Detection như Segmentation, Classification. Điều này mở ra tiềm năng phát triển và mở rộng dự án trong tương lai.

Vì những lý do trên, YOLOv8 là lựa chọn tối ưu, kết hợp giữa hiệu năng mạnh mẽ đã được kiểm chứng và các yếu tố thực tiễn quan trọng cho sự thành công của đồ án.

### Lựa chọn Framework tối ưu nhất cho Web App

Để người dùng có thể tương tác và kiểm thử mô hình đã huấn luyện, việc xây dựng một giao diện Web App là cần thiết. Thay vì sử dụng các framework web full-stack phức tạp như Django hay Flask, vốn đòi hỏi nhiều thời gian và công sức cho việc thiết kế front-end và back-end, đồ án lựa chọn **Gradio** làm công nghệ chính.

Lý do lựa chọn Gradio:

* **Đơn giản và nhanh chóng:** Gradio là một thư viện Python được thiết kế đặc biệt để tạo giao diện người dùng (UI) cho các mô hình Machine Learning. Chỉ với vài dòng code Python, ta có thể xây dựng một ứng dụng web hoàn chỉnh với chức năng tải lên hình ảnh/video, hiển thị kết quả đầu ra trực quan mà không cần kiến thức về HTML, CSS hay JavaScript.
* **Tập trung vào mục tiêu chính:** Việc sử dụng Gradio giúp nhóm tập trung hoàn toàn vào việc trình diễn khả năng của model YOLOv8, thay vì sa đà vào các vấn đề kỹ thuật của phát triển web. Chức năng và độ phức tạp của Gradio hoàn toàn phù hợp với phạm vi (scope) của đồ án.
* **Tích hợp hoàn hảo với hệ sinh thái Hugging Face:** Gradio được hỗ trợ mạnh mẽ bởi Hugging Face. Điều này mang lại một lợi thế cực lớn: khả năng triển khai ứng dụng lên **Hugging Face Spaces** một cách miễn phí và liền mạch. Người dùng trên toàn thế giới có thể truy cập và sử dụng sản phẩm chỉ qua một đường link mà không cần nhóm phải lo lắng về việc quản lý máy chủ hay hạ tầng.
* **Cộng đồng lớn và tài liệu đầy đủ:** Là một công cụ phổ biến trong cộng đồng AI/ML, Gradio có nguồn tài liệu phong phú và được nhiều người sử dụng.

# Công nghệ sử dụng

## Framework và thư viện cho quá trình learning

* **Kagglehub**: Để tải tập dữ liệu từ Kaggle.
* **OS**: Để tương tác với hệ điều hành và quản lý đường dẫn file.
* **Random**: Để tạo số ngẫu nhiên, được sử dụng khi chọn mẫu ảnh.
* **CV2 (OpenCV)**: Thư viện xử lý ảnh và thị giác máy tính, được sử dụng để đọc, xử lý và vẽ lên ảnh (ví dụ: vẽ bounding box).
* **Glob**: Để tìm kiếm các file theo pattern trong hệ thống file.
* **Pandas**: Để phân tích và xử lý dữ liệu, đặc biệt là làm việc với các file CSV (ví dụ: kết quả huấn luyện).
* **Matplotlib**: Để tạo biểu đồ và đồ thị hóa dữ liệu (ví dụ: phân bố class, kết quả huấn luyện).
* **Collections.Counter**: Để đếm số lượng các đối tượng (ví dụ: số lượng class trong dataset).
* **Albumentations**: Thư viện tăng cường dữ liệu cho hình ảnh.
* **Albumentations.pytorch.ToTensorV2**: Biến đổi ảnh từ Albumentations sang tensor PyTorch.
* **Ultralytics**: Framework chứa mô hình YOLOv8 và các công cụ liên quan để huấn luyện và dự đoán.
* **IPython.display**: Để hiển thị hình ảnh và output trong notebook.
* **Shutil**: Để thực hiện các thao tác cấp cao trên file và thư mục (ví dụ: sao chép file).

## Nền tảng

* **Google Colab:** Phân tích dataset vàModel training
* **Roboflow:** Production training nhiều model khác nhau trên cùng một dataset và workflow. Đây là nền tảng train model cuối cùng.
* **HuggingFace:** Lưu trữ và công khai final model.
* **HuggingFace Space:** Lưu mã nguồn cho Web App và host trang web.
* **Github:** Lưu Source code cho mã nguồn kiểm thử trên máy local.

## Thư viện phát triển web

* **Gradio:** Framework dành cho ngôn ngữ lập trình python, với nhiều high level API hỗ trợ việc tích hợp mô hình học máy vào trang web

## Thuộc tính của mô hình cuối cùng

### Kiểu Học (Learning Type)

Mô hình được sử dụng trong notebook này áp dụng Học có giám sát (Supervised Learning). Điều này là do mô hình được huấn luyện trên một tập dữ liệu đã được gán nhãn (labeled dataset), trong đó mỗi hình ảnh đi kèm với các hộp giới hạn (bounding box) và nhãn lớp (class label) cho các đối tượng khói và lửa.

### Kiểu Mô hình (Model Type)

Đây là một mô hình Phát hiện đối tượng thời gian thực (Real-time Object Detection). YOLO (You Only Look Once) là một họ các mô hình nổi tiếng về khả năng phát hiện đối tượng hiệu quả và nhanh chóng, phù hợp cho các ứng dụng yêu cầu xử lý video hoặc hình ảnh trực tiếp.

### Mô hình Tiền Huấn luyện (Pre-trained Model)

Mô hình cơ sở được sử dụng là YOLOv8n.pt. Đây là một phiên bản nhỏ (nano) của kiến trúc YOLOv8, đã được tiền huấn luyện trên một tập dữ liệu lớn ([COCO dataset](https://www.google.com/url?q=https%3A%2F%2Fcocodataset.org%2F%23home)).

### Ultralytics và YOLO

YOLO (You Only Look Once): Là tên của một kiến trúc mạng nơ-ron (neural network architecture) được thiết kế cho tác vụ phát hiện đối tượng. Các mô hình YOLO xử lý toàn bộ hình ảnh trong một lần duy nhất để dự đoán các hộp giới hạn và xác suất lớp, giúp đạt được tốc độ xử lý cao. Ultralytics: Là công ty phát triển và duy trì các phiên bản mới nhất của YOLO, bao gồm YOLOv8. Ultralytics cung cấp một framework và các công cụ đi kèm để dễ dàng sử dụng, huấn luyện và triển khai các mô hình YOLO.

Framework ultralytics được sử dụng trong notebook này để làm việc với mô hình YOLOv8. Chuyển giao Học tập (Transfer Learning) Quá trình huấn luyện trong notebook này sử dụng kỹ thuật Chuyển giao học tập (Transfer Learning).

Mô hình cuối cùng sau khi được huấn luyện sẽ có khả năng Phát hiện và định vị các đối tượng khói và lửa trong hình ảnh hoặc video đầu vào. Đối với mỗi đối tượng được phát hiện, mô hình sẽ cung cấp:

Lớp của đối tượng (Smoke hoặc Fire). Hộp giới hạn (Bounding Box) chỉ ra vị trí của đối tượng. Điểm tin cậy (Confidence Score) cho biết mức độ chắc chắn của mô hình về dự đoán đó.

Vận dụng vision model cho web app nhận diện khói lửa

Với mô hình YOLOv8 đã được huấn luyện trên tập dữ liệu tùy chỉnh về khói và lửa, đồ án này hướng tới việc xây dựng một ứng dụng thực tế để trình diễn khả năng phát hiện này.

Các bước tiếp theo bao gồm:

* Sử dụng mô hình YOLOv8 đã huấn luyện để xây dựng một ứng dụng web.
* Ứng dụng web sẽ cho phép người dùng tải lên hình ảnh hoặc video để mô hình dự đoán vị trí khói và lửa.
* Sử dụng framework Gradio để phát triển giao diện người dùng cho ứng dụng web một cách nhanh chóng.
* Triển khai (host) mô hình và ứng dụng web trên nền tảng Hugging Face Spaces để mọi người có thể dễ dàng truy cập và kiểm thử.
* Lưu trữ mô hình đã huấn luyện trên Hugging Face Hub để có thể tái sử dụng hoặc chia sẻ.

# Quy trình thực hiện

A close-up of a sign

AI-generated content may be incorrect.

Hình . Tổng quan quy trình thực hiện

## Tổng quan cấu trúc dataset

**A diagram of a fire detector

AI-generated content may be incorrect.**

Hình . Sơ đồ cấu trúc dataset

Data set này chứa ảnh sample có khói và lửa, tất cả đều được lưu dưới định dạng YOLO, nhằm hỗ trợ cho supervised, transfer learning cho object detection vision model.

Có 3 folder:

* **Train**: Chứa ảnh được label, dùng trực tiếp cho quá trình train
* **Valid**: Chứa ảnh được label, dùng cho quá trình training để đối chiếu kết quả thực tế với kết quả mong đợi
* **Test**: Dùng sau quá trình training, dùng để đánh giá hiệu suất của model khi xử lý data chưa từng gặp

Mỗi ảnh ở folder images sẽ có một file txt ở folder labels gán tọa độ cho biết khói và lửa nằm ở đâu (vd: **1 0.69921875 0.39921875 0.6015625 0.6546875**)

File **data.yaml** trong dataset giúp định vị các folder và liệt kê các label class, ở đây là khói (smoke) và lửa (fire)

**A screenshot of a computer program

AI-generated content may be incorrect.**

Hình . Cấu trúc file data.yaml

## Tiền sử lí và tăng cường dữ liệu

**A diagram of a structure

AI-generated content may be incorrect.**

Hình . Sơ đồ quy trình tiền sử lý và tăng cường dữ liệu

* Thay đổi kích thước: Resize ảnh về 608x608 pixel (có thể tùy chỉnh về 640x640 để phù hợp YOLOv8).
* Chuẩn hóa: Theo tiêu chuẩn ImageNet (mean = [0.485, 0.456, 0.406], std = [0.229, 0.224, 0.225]).
* Tăng cường dữ liệu:
* Lật ngang (50% xác suất).
* Điều chỉnh độ sáng/tương phản ngẫu nhiên.
* Biến đổi màu sắc.
* Xoay 90 độ ngẫu nhiên.
* Xử lý hộp giới hạn: Căn chỉnh bounding box sau biến đổi.
* Chia tập dữ liệu: Đã chia sẵn thành train, valid, test.

## Phân tích dữ liệu khai phá (EDA)

### B.1. Kiểm tra số lượng hình ảnh và class tương ứng:

A black screen with white text

AI-generated content may be incorrect.

Hình . Tổng số hình ảnh và class tương ứng trong dataset

### B.2. Xác minh trực quan rằng các chú thích là chính xác trước khi tiến hành huấn luyện mô hình.

A screenshot of a computer program

AI-generated content may be incorrect.

Hình . Code định nghiã hàm tạo hộp có viền

A collage of images of fire and smoke

AI-generated content may be incorrect.

Hình . Output ảnh trong dataset đã được vẽ viền

### B.3 Phân tích phân bổ nhãn trong dataset

**A screen shot of a computer program

AI-generated content may be incorrect.**

Hình . Code vẽ đồ thị thể hiện tỉ lệ phân bổ các class

A blue bar graph with white text

AI-generated content may be incorrect.

Hình . Output đồ thị phân bổ class

### B.4. Kiểm tra nhãn bị thiếu

A computer screen shot of text

AI-generated content may be incorrect.

Hình . Code kiểm tra nhãn bị thiếu trong dataset

A screenshot of a computer

AI-generated content may be incorrect.

Hình . Output xác nhận không có ảnh bị thiếu nhãn

### B.5. Kết luận

* Các hình ảnh rất đa dạng về hậu cảnh, ánh sáng và khả năng nhìn thấy khói/lửa.
* Các hộp giới hạn bao quanh chính xác các vùng khói hoặc lửa.
* Phân bổ class trong tập huấn luyện đã được vẽ biểu đồ.
* Class 0 và Class 1 (lửa và khói) đều có mặt đầy đủ
* Đã quan sát thấy một chút mất cân bằng class, nhưng không quá nghiêm trọng.
* Đã kiểm tra rằng mỗi hình ảnh đều có tệp nhãn tương ứng.
* Không tìm thấy nhãn bị thiếu trên các tập huấn luyện, kiểm định và kiểm thử.
* Sự mất cân bằng class nhỏ có thể được xử lý trong quá trình huấn luyện mô hình bằng các kỹ thuật như class weights nếu cần thiết.
  + **Chất lượng bộ dữ liệu tốt và sẵn sàng cho việc huấn luyện mô hình học sâu.**

## Quy trình huấn luyện

A screenshot of a computer program

AI-generated content may be incorrect.

Hình . Code mẫu chạy quy trình huấn luyện

**A diagram of a diagram

AI-generated content may be incorrect.**

Hình . Sơ đồ tóm tắt quy trình huấn luyện

**B.1 Bắt đầu Training**

* Khởi động quá trình huấn luyện mô hình.

**B.2 Scanning Dataset – Kiểm tra & Thống kê**

* Duyệt và kiểm tra cấu trúc tập dữ liệu (các nhãn, ảnh có hợp lệ không).
* Nếu phát hiện lỗi (ví dụ: nhãn sai định dạng, ảnh hỏng), hệ thống sẽ báo lỗi & dừng để người dùng xử lý.

**B.3 Chuẩn bị cho Epoch**

* Nếu dữ liệu hợp lệ, hệ thống sẽ chuẩn bị cho quá trình huấn luyện từng epoch.

**B.4 Training trên Tập Train**

* Mô hình bắt đầu học từ dữ liệu huấn luyện.

**B.5 Validation trên Tập Validation**

* Sau mỗi epoch, mô hình được đánh giá trên tập validation.

**B.6 Tính toán & Lưu Metrics + Mô hình**

* Lưu lại các chỉ số đánh giá (như mAP, loss) và mô hình tạm thời (checkpoint).

**B.7 Đã đủ số Epoch hoặc Dừng sớm?**

* Kiểm tra nếu đã đạt số epoch mong muốn hoặc đạt điều kiện hội tụ sớm (early stopping).
* Nếu chưa: Quay lại bước chuẩn bị epoch và tiếp tục training.
* Nếu đã đủ: Chuyển sang bước kiểm tra với tập test.

**B.8 Có Tập Test không?**

* Nếu có tập test, tiến hành đánh giá hiệu suất cuối cùng.

**B.9 Testing trên Tập Test**

* Mô hình sẽ chạy trên tập test để đo lường khả năng tổng quát hóa.

**B.10 Kết thúc Training**

* Quá trình huấn luyện dừng lại, mô hình cuối cùng được xác nhận.

**B.11 Lưu mô hình cuối cùng**

* Mô hình đã huấn luyện xong sẵn sàng để triển khai hoặc tích hợp.

**A screenshot of a computer program

AI-generated content may be incorrect.**

Hình . Code lưu model vừa được tạo và lưu best weight làm model chính

## Đánh giá mô hình và quá trình training

A screenshot of a computer program

AI-generated content may be incorrect.

Hình . Code in biểu đồ training

A group of graphs showing the size of a column

AI-generated content may be incorrect.

Hình . Biểu đồ training trên roboflow

**Thời gian huấn luyện:** Mô hình được huấn luyện trong tổng cộng 200 epoch.

* **Xu hướng hội tụ:**
  + Các biểu đồ **loss** (box\_loss, cls\_loss, dfl\_loss) trên cả tập train và tập validation đều giảm mạnh trong khoảng 50 epoch đầu và hội tụ ổn định về giá trị rất thấp.
  + Các biểu đồ **metrics** (Precision, Recall, mAP) tăng nhanh và đạt mức gần như tối đa, sau đó duy trì sự ổn định trong suốt phần còn lại của quá trình huấn luyện.
* **Đánh giá:** Xu hướng của đồ thị validation bám sát đồ thị training cho thấy mô hình không có dấu hiệu **overfitting** (học vẹt), chứng tỏ khả năng tổng quát hóa tốt trên dữ liệu mới. Quá trình huấn luyện diễn ra thành công và mô hình đã hội tụ tốt.

A blue line with black text

AI-generated content may be incorrect.

Hình . Metrics của model cuôí cùng

* **Precision (99.3%):** Tỉ lệ dự đoán đúng trên tổng số các dự đoán mà mô hình tạo ra là cực kỳ cao. Điều này có nghĩa là mô hình rất đáng tin cậy.
* **Recall (99.6%):** Mô hình có khả năng phát hiện gần như toàn bộ các đối tượng khói/lửa có trong thực tế.
* **mAP@50 (99.4%):** Mức 99.4% là một con số xuất sắc, khẳng định mô hình không chỉ nhận diện đúng mà còn khoanh vùng đối tượng với độ chính xác cao.

## Web App Flow và Source Code

Mô hình được đẩy lên Hugging Face Hub: [kappH/NhanDienKhoiLua · Hugging Face](https://huggingface.co/kappH/NhanDienKhoiLua)

WebApp Gradio công khai: [Gradio](https://kapph-nhandienkhoilua.hf.space/)

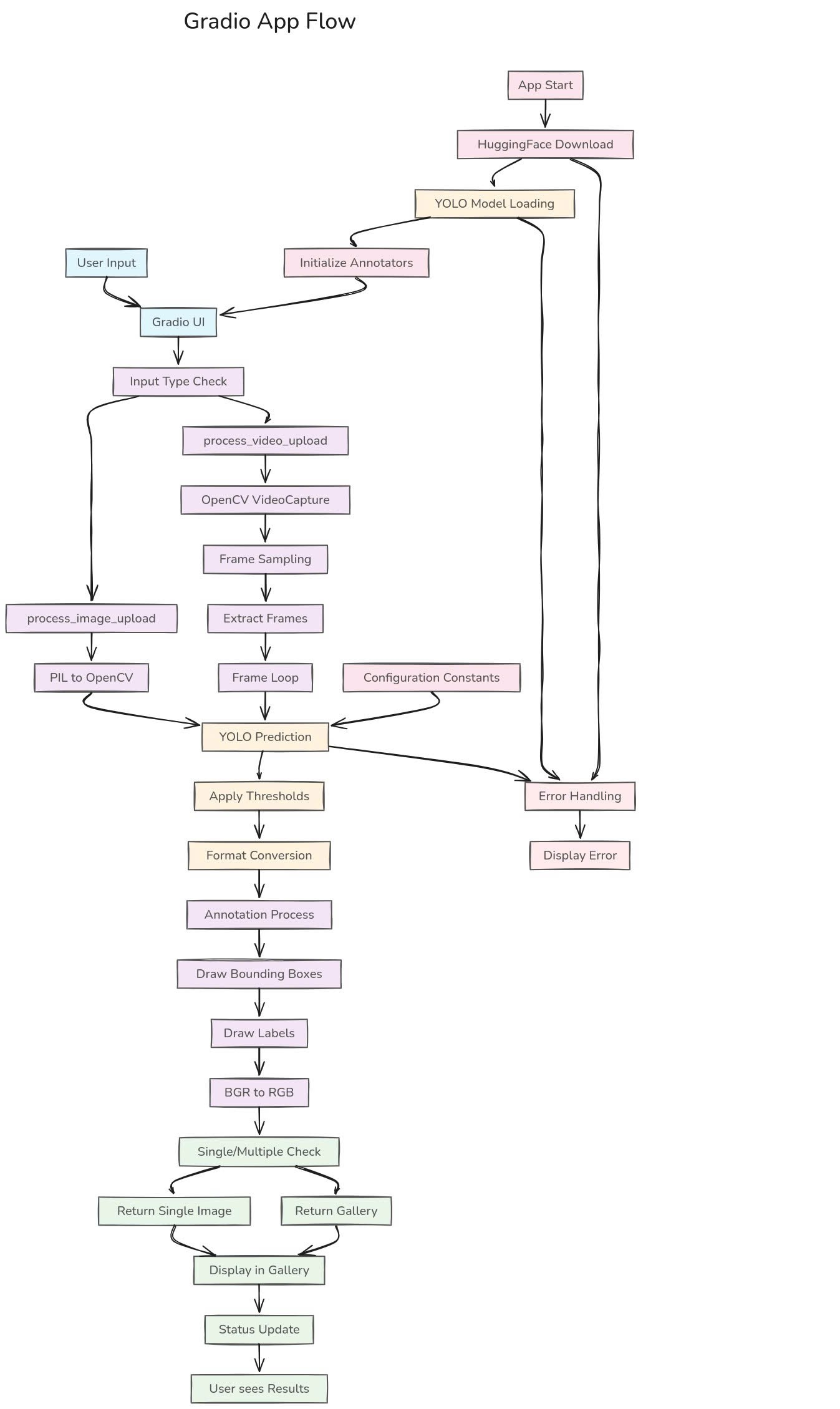
### 6.1. Tóm tắt source code

A screenshot of a computer program

AI-generated content may be incorrect.

Hình . Code tóm tắt Web App

### 6.2. Sơ hồ hoạt động của hệ thống



Hình . Sơ đồ hoạt động

**B.1. App Start → Tải mô hình từ Hugging Face**

* + Ứng dụng bắt đầu và thực hiện tải mô hình đã huấn luyện từ Hugging Face.

**B.2. YOLO Model Loading**

* + Mô hình được load vào bộ nhớ, sẵn sàng dự đoán.

**B.3. User Input → Gradio UI**

* + Người dùng tương tác thông qua giao diện Gradio và upload ảnh/video.

**B.4. Kiểm tra kiểu đầu vào (ảnh hoặc video)**

* + Nếu là ảnh, hệ thống chuyển đổi từ PIL sang OpenCV.
  + Nếu là video, hệ thống:
  + Dùng OpenCV để trích xuất các khung hình,
  + Lặp qua từng khung hình (frame loop) để xử lý.

**B.5. YOLO Prediction**

* + Mỗi khung hình (hoặc ảnh) được đưa vào mô hình để dự đoán vùng chứa khói/lửa.

**B.6. Áp dụng Ngưỡng (Apply Thresholds)**

* + Loại bỏ những kết quả không đủ tin cậy.

**B.7. Chuyển đổi định dạng → Vẽ kết quả**

* + Kết quả được chuyển về định dạng phù hợp,
  + Vẽ bounding boxes và nhãn (labels) tương ứng lên ảnh/video,
  + Chuyển màu từ BGR sang RGB.

**B.8. Kiểm tra đầu ra là đơn ảnh hay nhiều ảnh**

* + Nếu là một ảnh duy nhất → trả ảnh về trực tiếp.
  + Nếu là nhiều ảnh → hiển thị dưới dạng thư viện (gallery).

**B.9. Hiển thị kết quả & Cập nhật trạng thái**

* + Ảnh đã xử lý được hiển thị cho người dùng.
  + Giao diện cập nhật trạng thái xử lý và trả kết quả.

**B.10. Xử lý lỗi**

* Nếu có lỗi ở bất kỳ bước nào, hệ thống chuyển hướng tới phần hiển thị lỗi.

### 6.3. Giao diện website

A screenshot of a computer

AI-generated content may be incorrect.

Hình .Giao diện Website

Trên trang web, người dung có thể:

* Upload ảnh hoặc video
* Nhấn ‘Bắt đầu phát hiện tử Hình ảnh/Video’
* Nhận lại một khung hình hoặc 5 khung hình (trích từ video) được nhận dịên khói lửa
* Ngoài ra, trang web còn có mục examples để người dùng có thể trải nghiệm trang web mà không cần chuẩn bị trước hình ảnh hoặc video

# Kết quả và đánh giá

## Tổng kết đồ án

* Đồ án đã xây dựng thành công hệ thống phát hiện khói/lửa bằng YOLOv8.
* Ứng dụng web được triển khai trên Hugging Face Spaces, dễ dàng tiếp cận và kiểm thử.

## Ưu điểm

* **Ứng dụng model lên trang web**: Thay vì chạy trên máy local, dự án này sử dụng nền tảng online hosting để khiến việc tiếp cận và thử nghiệm model dễ dàng hơn.
* **Phát hiện sớm**: Hệ thống có khả năng phát hiện khói và lửa từ hình ảnh hoặc video, giúp cảnh báo sớm hơn so với các phương pháp truyền thống dựa vào cảm biến nhiệt hoặc khói tại một điểm cố định.
* **Không xâm lấn**: Quá trình phát hiện dựa trên phân tích hình ảnh, không yêu cầu tiếp xúc vật lý với môi trường, phù hợp cho các khu vực rộng lớn hoặc khó tiếp cận.
* **Chi phí triển khai linh hoạt**: Có thể tận dụng hạ tầng camera giám sát sẵn có, giảm chi phí so với việc lắp đặt hệ thống cảm biến dày đặc.
* **Khả năng mở rộng**: Mô hình có thể được tinh chỉnh (fine-tune) hoặc huấn luyện lại với dữ liệu mới để cải thiện hiệu suất hoặc thích ứng với các môi trường khác nhau.
* **Thông tin chi tiết**: Cung cấp thông tin về vị trí (thông qua bounding box) và loại đối tượng (khói hoặc lửa), giúp xác định rõ ràng mối đe dọa.

## Nhược điểm

* **Chưa cạnh tranh được so với cảm biến vật lý**: Trong một số trường hợp, cảm biến vật lý vẫn có thể phản ứng nhanh hơn hoặc chính xác hơn trong điều kiện lý tưởng (ví dụ: cảm biến khói trong không gian kín).
* **Yêu cầu GPU hiệu năng cao**: Quá trình huấn luyện và suy luận (inference) của các mô hình học sâu như YOLOv8 yêu cầu phần cứng có GPU hiệu năng cao để đạt được tốc độ và hiệu quả mong muốn, đặc biệt là khi xử lý video thời gian thực.
* **Chưa thể áp dụng nhận diện trên live webcam cho Gradio web app**: Do giới hạn về tài nguyên tính toán và băng thông trên các nền tảng triển khai miễn phí như Hugging Face Spaces, việc xử lý video từ webcam trực tiếp trên ứng dụng web còn gặp khó khăn.

## Hướng phát triển trong tương lai

* **Tích hợp vào Homeserver/Homelab**: Triển khai mô hình đã huấn luyện lên một homeserver hoặc homelab với GPU cục bộ để xử lý video từ các camera an ninh trong nhà hoặc văn phòng.
* **Hệ thống cảnh báo tự động**: Khi hệ thống nhận diện khói hoặc lửa, tự động kích hoạt các hành động cảnh báo như gửi thông báo đến điện thoại, email, bật còi báo động, hoặc thậm chí là kích hoạt hệ thống chữa cháy tự động (trong môi trường kiểm soát và an toàn).
* **Cải thiện hiệu suất trên thiết bị biên (Edge Devices)**: Nghiên cứu và tối ưu hóa mô hình để chạy hiệu quả hơn trên các thiết bị biên có tài nguyên hạn chế (ví dụ: Raspberry Pi, Jetson Nano) để giảm chi phí triển khai và tăng tính di động.
* **Tăng cường dữ liệu và huấn luyện**: Mở rộng tập dữ liệu với nhiều kịch bản khói và lửa đa dạng hơn, đồng thời áp dụng các kỹ thuật huấn luyện nâng cao để cải thiện độ chính xác và khả năng chống nhiễu của mô hình.
* **Phát hiện các dấu hiệu khác**: Mở rộng khả năng của hệ thống để phát hiện các dấu hiệu khác liên quan đến hỏa hoạn như thay đổi nhiệt độ bất thường (sử dụng camera nhiệt) hoặc chuyển động bất thường của đám đông trong trường hợp sơ tán.
* **Phát triển ứng dụng web thời gian thực**: Tìm kiếm các giải pháp triển khai ứng dụng web cho phép xử lý video thời gian thực từ webcam, có thể thông qua các dịch vụ đám mây chuyên dụng hoặc tối ưu hóa luồng xử lý dữ liệu.

## Link các sản phẩm của đồ án

Kết quả đo lường metrics cho các model đề xuất được lưu tại roboflow project:

<https://app.roboflow.com/nhatml/fire-smoke-detection-eozii-yqmn9/models>

Google colab notebook cho quá trình phân tích dữ liệu và model training

<https://colab.research.google.com/drive/1J-5NthnIKvsM4a6-RZzOcqjEdOUhlF1T?usp=sharing>

Model hoàn chỉnh được lưu tại huggingface:

[https://huggingface.co/kappH/NhanDienKhoiLua/](https://www.google.com/url?q=https%3A%2F%2Fhuggingface.co%2FkappH%2FNhanDienKhoiLua%2F)

Ứng dụng trên gradio tại:

[https://kapph-nhandienkhoilua.hf.space/](https://www.google.com/url?q=https%3A%2F%2Fkapph-nhandienkhoilua.hf.space%2F)

Source Code cho ứng dụng gradio

<https://huggingface.co/spaces/kappH/NhanDienKhoiLua/tree/main>

Github repo để test mô hình trên máy local:

<https://github.com/potatomat0/yolo-smoke-and-fire-detector>

Graphical Abstract

<https://excalidraw.com/#json=iQJ3YostLShr1IqHZ3910,DsvsoERoApkttNnDclezpg>

Các video demo

<https://drive.google.com/drive/folders/1Q8ITgK58ZRnN91aLsYXEPUHmxqCm2BOY?usp=sharing>

# TÀI LIỆU THAM KHẢO

Dataset: [https://www.kaggle.com/datasets/roscoekerby/firesmoke-detection-yolo-v9](https://www.google.com/url?q=https%3A%2F%2Fwww.kaggle.com%2Fdatasets%2Froscoekerby%2Ffiresmoke-detection-yolo-v9)

**So sánh các mô hình và tác vụ học máy**

* [A Comprehensive Survey on Deep Learning for Computer Vision](https://www.google.com/search?q=https://www.researchgate.net/publication/335809538_A_Comprehensive_Survey_on_Deep_Learning_for_Computer_Vision): Một bài khảo sát học thuật chi tiết về các lĩnh vực trong thị giác máy tính.
* [Computer Vision: The Evolution of AGI](https://www.ibm.com/topics/computer-vision): Bài viết từ IBM cung cấp cái nhìn tổng quan về thị giác máy tính và vai trò của nó hướng tới AGI.
* [Image Classification vs. Object Detection vs. Image Segmentation](https://www.google.com/search?q=https://www.v7labs.com/blog/image-classification-vs-object-detection-vs-image-segmentation): So sánh trực quan và dễ hiểu giữa ba tác vụ phổ biến nhất trong thị giác máy tính.
* [A Gentle Introduction to Computer Vision](https://machinelearningmastery.com/what-is-computer-vision/): Bài viết nhập môn về các khái niệm và ứng dụng của thị giác máy tính.

**YOLO và Ultralytics**

* [Ultralytics YOLOv8 Documentation](https://docs.ultralytics.com/): Trang tài liệu chính thức và đầy đủ nhất cho mọi thứ liên quan đến YOLO của Ultralytics.
* [How to Train a Custom YOLOv8 Model on Google Colab](https://blog.roboflow.com/how-to-train-yolov8-on-a-custom-dataset/): Hướng dẫn chi tiết từ Roboflow về cách huấn luyện mô hình YOLOv8 trên Colab.
* [Getting Started with YOLOv8 and Ultralytics](https://github.com/ultralytics/ultralytics): Kho mã nguồn chính thức trên GitHub, chứa các ví dụ và hướng dẫn cốt lõi.

**Gradio, Hugging Face, và Roboflow**

* [Gradio Documentation](https://www.google.com/search?q=https://www.gradio.app/docs/main): Trang tài liệu chính thức của Gradio, hướng dẫn cách xây dựng giao diện cho mô hình học máy.
* [Hugging Face Documentation](https://huggingface.co/docs): Cổng tài liệu chính của Hugging Face, bao gồm thư viện transformers, datasets, và nhiều hơn nữa.
* [Hugging Face Spaces Documentation](https://huggingface.co/docs/hub/spaces): Hướng dẫn về cách triển khai ứng dụng Gradio và Streamlit trên nền tảng Hugging Face Spaces.
* [Roboflow Documentation](https://docs.roboflow.com/): Tài liệu chính thức của Roboflow, bao gồm toàn bộ quy trình từ gán nhãn, xử lý dữ liệu đến huấn luyện mô hình.
* [Roboflow Quick Start Guide](https://www.google.com/search?q=https://docs.roboflow.com/quick-start): Hướng dẫn nhanh cho người mới bắt đầu sử dụng Roboflow.

**Phân tích dữ liệu khám phá (EDA)**

* [Exploratory Data Analysis (EDA) - A Practical Guide](https://www.google.com/search?q=https://www.ibm.com/cloud/learn/exploratory-data-analysis): Hướng dẫn thực tế về EDA từ IBM, giải thích mục đích và các kỹ thuật chính.
* [How to Structure an Exploratory Data Analysis (EDA) Report](https://www.google.com/search?q=https://towardsdatascience.com/how-to-structure-an-exploratory-data-analysis-eda-report-3b7075bfe20): Bài viết chuyên sâu về cách cấu trúc và trình bày một báo cáo EDA hiệu quả.
* [A comprehensive guide to Exploratory Data Analysis](https://www.google.com/search?q=https://www.analyticsvidhya.com/blog/2021/08/a-comprehensive-guide-to-exploratory-data-analysis/): Một hướng dẫn toàn diện khác về các bước và công cụ để thực hiện EDA.