TRƯỜNG ĐẠI HỌC HỌC VĂN LANG

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KỲ**

**MÔN HỌC: NHẬP MÔN XỬ LÝ ẢNH SỐ**

***Đề tài: PHÁT HIỆN VÀ ĐẾM SỐ LƯỢNG ĐỐI TƯỢNG TRONG ẢNH ( NGƯỜI, XE, VẬT THỂ) SỬ DỤNG YOLOv5***

**SVTH1: Phạm Thế Hùng – 2374802010164**

**SVTH2: Phan Trần Quang Thái - 2174802010116**

**SVTH3: Đoàn Hoàng Quân - 2374802010412**

**SVTH4: Nguyễn Chí Dũng - 2374802013469**

**GVHD: T.S Đỗ Hữu Quân**

**Hồ Chí Minh – năm 2025**

**Mục Lục**

[**Lời cảm ơn** 4](#_Toc204467156)

[**Chương 1: Mở Đầu** 5](#_Toc204467157)

[**1.1.** **Lý do chọn đề tài nghiên cứu** 5](#_Toc204467158)

[**1.2.** **Công Cụ và Thư Viện** 5](#_Toc204467159)

[**1.3.** **Đối tượng, phạm vi nghiên cứu** 6](#_Toc204467160)

[**1.3.1.** **Đối tượng nghiên cứu** 6](#_Toc204467161)

[**1.3.2.** **Phạm vi nghiên cứu** 6](#_Toc204467162)

[**1.4.** **Phương pháp nghiên cứu:** 6](#_Toc204467163)

[**1.4.2.** **Thiết kế và Cấu hình Mô hình** 7](#_Toc204467164)

[**1.4.3.** **Huấn luyện Mô hình** 7](#_Toc204467165)

[**1.4.4.** **Đánh giá và Phân tích Kết quả** 7](#_Toc204467166)

[**1.4.5.** **Triển khai và Ứng dụng** 8](#_Toc204467167)

[**Chương 2: Cơ sở lý thuyết** 9](#_Toc204467168)

[**2.1.** **Tổng quan xử lý ảnh số** 9](#_Toc204467169)

[**2.2.** **Các khái niệm và thuật toán liên quan** 9](#_Toc204467170)

[**2.3.** **Mô hình áp dụng YOLOv5** 10](#_Toc204467171)

[**Chương 3: Phân tích và thiết kế hệ thống** 13](#_Toc204467172)

[**3.1.** **Quy trình xử lý tổng thể** 13](#_Toc204467173)

[**3.2.** **Kiến trúc hệ thống / sơ đồ luồng xử lý** 13](#_Toc204467174)

[**3.3.** **Thiết kế các bước xử lý ảnh cụ thể** 14](#_Toc204467175)

[**3.4.** **Mô tả dữ liệu đầu vào / đầu ra** 16](#_Toc204467176)

[**Chương 4: Cài đặt và thử nghiệm** 18](#_Toc204467177)

[**4.1.** **Môi trường cài đặt** 18](#_Toc204467178)

[**4.2.** **Mô tả quá trình cài đặt chương trình** 19](#_Toc204467179)

[**4.3.** **Kết quả thực nghiệm** 21](#_Toc204467180)

[**4.4.** **Đánh giá kết quả và so sánh** 22](#_Toc204467181)

[**4.4.1.** **Đánh giá kết quả** 22](#_Toc204467182)

[**4.4.2.** **So sánh với các phương pháp khác** 22](#_Toc204467183)

[**Chương 5: Kết luận và hướng phát triển** 24](#_Toc204467184)

[**Tài liệu tham khảo:** 26](#_Toc204467185)

# **Lời cảm ơn**

Nhóm chúng em, nhóm 21 – những cậu bé ngây ngô đã cùng thầy đồng hành suốt một học kì dài gian nan. Chúng em từ những cậu bé không có chút kiến thức nào về môn xử lý ảnh số. Nhưng nhờ may mắn và định mệnh, chúng em đã được gặp thầy, thầy Đỗ Hữu Quân, một người thầy tài sắc vẹn toàn và đầy nhiệt huyết đã dìu dắt tận tình tận tâm từ bài học đầu tiên đến những buổi cuối cùng. Chúng em vô cùng biết ơn công lao vĩ đại và tình thương của thầy ạ. Những kiến thức hôm nay mà chúng em được thầy trợ cấp, sẽ luôn là hành trang thật vững chắc cho hành trình phát triển cùng nghề của em trong tương lai. Em và nhóm 21, xin chân thành cảm ơn Thầy. Mãi yêu.

TP. Hồ Chí Minh, ngày 14 tháng 7 năm 2025

# **Chương 1: Mở Đầu**

* 1. **Lý do chọn đề tài nghiên cứu**

Trong thời đại công nghệ 4.0, thị giác máy tính (Computer Vision) đang ngày càng đóng vai trò quan trọng trong nhiều lĩnh vực như giao thông thông minh, an ninh giám sát, y tế, nông nghiệp và sản xuất công nghiệp. Một trong những ứng dụng nổi bật của thị giác máy tính là phát hiện và đếm đối tượng trong ảnh, giúp máy móc có khả năng "nhìn" và hiểu được môi trường xung quanh.

Trong môn học Nhập môn xử lý ảnh số, chúng em được tiếp cận với các kiến thức cơ bản về xử lý ảnh và thị giác máy tính. Để vận dụng và mở rộng kiến thức đã học vào thực tiễn, nhóm chúng em lựa chọn đề tài “Phát hiện và đếm số lượng đối tượng trong ảnh sử dụng YOLOv5”.

* Lý do lựa chọn đề tài:

Tính ứng dụng cao: Phát hiện và đếm người, xe cộ, vật thể là nhu cầu phổ biến trong các hệ thống giám sát, quản lý giao thông, phân tích hành vi khách hàng, robot thông minh, v.v.

Áp dụng công nghệ hiện đại: YOLOv5 là một trong những mô hình hiện đại, mạnh mẽ và tối ưu nhất hiện nay trong lĩnh vực nhận diện đối tượng theo thời gian thực (real-time).

Kết hợp giữa lý thuyết và thực hành: Đề tài là cơ hội để chúng em áp dụng lý thuyết về xử lý ảnh vào một bài toán thực tiễn, đồng thời rèn luyện kỹ năng lập trình, sử dụng thư viện AI, đọc hiểu mô hình học sâu.

Khả năng mở rộng và phát triển: Sau khi hoàn thành phiên bản cơ bản, đề tài có thể mở rộng thành hệ thống đếm người từ camera thời gian thực, nhận diện biển số xe, hoặc tích hợp với các hệ thống thông minh khác.

Chúng em tin rằng đề tài này không chỉ phù hợp với nội dung môn học mà còn có tính thực tiễn cao, là nền tảng để tiếp cận các công nghệ AI và học sâu trong các môn học và nghiên cứu sau này.

* 1. **Công Cụ và Thư Viện**

- **Ngôn ngữ:** Python

- **Thư viện chính:** PyTorch, OpenCV, NumPy, matplotlib

- **Framework:** YOLOv5 từ Ultralytics (dạng repo GitHub)

- **Google Colab**: (nếu máy yếu) hoặc chạy local với GPU (nếu có)

* 1. **Đối tượng, phạm vi nghiên cứu**
     1. **Đối tượng nghiên cứu**

Đối tượng nghiên cứu của đề tài là các kỹ thuật phát hiện đối tượng (object detection) trong ảnh số, cụ thể là:

Mô hình YOLOv5 (You Only Look Once version 5) – một mạng nơ-ron tích chập sâu (CNN) được thiết kế để nhận diện và định vị nhiều đối tượng trong một bức ảnh.Các đối tượng cần phát hiện và đếm bao gồm: người, xe máy, ô tô, xe đạp, và một số vật thể phổ biến khác có trong ảnh tĩnh (image).Các ảnh được sử dụng là ảnh đầu vào từ tập dữ liệu có sẵn hoặc ảnh do người dùng cung cấp, có định dạng phổ biến như JPG, PNG.

* + 1. **Phạm vi nghiên cứu**

Đề tài tập trung vào việc phát hiện và đếm số lượng đối tượng trong ảnh tĩnh, không xử lý video hay ảnh thời gian thực từ camera. Sử dụng mô hình YOLOv5 pre-trained (đã huấn luyện sẵn) trên tập dữ liệu COCO dataset – tập dữ liệu phổ biến với 80 lớp đối tượng. Không thực hiện huấn luyện lại mô hình (training) hay tinh chỉnh (fine-tuning) trên tập dữ liệu riêng – chỉ áp dụng mô hình có sẵn. Chỉ áp dụng cho các ảnh có nội dung rõ ràng, chất lượng tương đối, không xử lý ảnh bị mờ, thiếu sáng hoặc có nhiễu cao. Không đi sâu vào tối ưu hóa hiệu năng mô hình hay so sánh chi tiết với các mô hình object detection khác như Faster R-CNN, SSD,...

* 1. **Phương pháp nghiên cứu:**
     1. **Thu thập và Xử lý Dữ liệu**

Thu thập dữ liệu: Xây dựng hoặc sử dụng các bộ dữ liệu có sẵn (ví dụ: COCO, PASCAL VOC) hoặc tự thu thập hình ảnh/video liên quan đến đối tượng cần phát hiện.

Gán nhãn: Sử dụng các công cụ chuyên dụng (như LabelImg, Roboflow, CVAT) để gán nhãn cho các đối tượng trong ảnh. Định dạng nhãn phải tương thích với YOLOv5 (thường là file .txt với các thông tin về lớp, tọa độ và kích thước của bounding box).

Tiền xử lý dữ liệu: Áp dụng các kỹ thuật tăng cường dữ liệu để làm phong phú thêm bộ dữ liệu và giúp mô hình học tốt hơn, tránh overfitting. YOLOv5 tích hợp sẵn các kỹ thuật như Mosaic Augmentation, Copy-paste Augmentation, thay đổi độ sáng, độ tương phản, xoay ảnh, v.v.

* + 1. **Thiết kế và Cấu hình Mô hình**

Lựa chọn kiến trúc YOLOv5: YOLOv5 có nhiều phiên bản khác nhau (YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x) với sự đánh đổi giữa tốc độ và độ chính xác. Tùy thuộc vào yêu cầu của ứng dụng (ví dụ: cần tốc độ xử lý nhanh trên thiết bị nhúng hay cần độ chính xác cao), người nghiên cứu sẽ chọn phiên bản phù hợp.

Cấu hình mô hình: Tùy chỉnh các tham số (hyperparameters) như learning rate, batch size, số epoch, v.v., để tối ưu hóa quá trình huấn luyện.

* + 1. **Huấn luyện Mô hình**

Thiết lập môi trường: Cài đặt các thư viện cần thiết như PyTorch, Ultralytics YOLOv5, cùng với các driver GPU để tăng tốc độ huấn luyện.

Tiến hành huấn luyện: Sử dụng tập dữ liệu đã chuẩn bị để huấn luyện mô hình. Quá trình này bao gồm việc truyền dữ liệu qua các lớp của mạng nơ-ron, tính toán hàm lỗi (loss function) và cập nhật trọng số (weights) của mô hình.

Transfer Learning: Sử dụng các mô hình YOLOv5 đã được huấn luyện sẵn trên các bộ dữ liệu lớn (như COCO) làm điểm khởi đầu, sau đó tinh chỉnh (fine-tuning) trên bộ dữ liệu tùy chỉnh. Phương pháp này giúp mô hình hội tụ nhanh hơn và đạt hiệu quả cao hơn.

* + 1. **Đánh giá và Phân tích Kết quả**

Các chỉ số đánh giá:

* mAP (mean Average Precision): Chỉ số quan trọng nhất để đánh giá độ chính xác của mô hình.
* IoU (Intersection over Union): Đo lường mức độ trùng lặp giữa bounding box dự đoán và bounding box thực tế.
* Tốc độ xử lý (FPS - Frames Per Second): Đánh giá khả năng xử lý theo thời gian thực của mô hình.
* Recall và Precision: Phân tích khả năng tìm thấy đối tượng và độ chính xác của các dự đoán.

Phân tích kết quả:

* So sánh hiệu suất của mô hình đã huấn luyện với các phiên bản khác của YOLOv5 hoặc các mô hình phát hiện đối tượng khác.
* Phân tích các trường hợp mô hình hoạt động kém hiệu quả (ví dụ: các đối tượng bị che khuất, kích thước quá nhỏ, điều kiện ánh sáng không tốt) để đề xuất hướng cải thiện.
  + 1. **Triển khai và Ứng dụng**

Triển khai mô hình: Sau khi huấn luyện và đánh giá, mô hình có thể được triển khai trên các nền tảng khác nhau (web, mobile, thiết bị nhúng) để ứng dụng vào thực tế.

Tối ưu hóa mô hình: Có thể sử dụng các kỹ thuật như quantization, pruning để giảm kích thước và tăng tốc độ suy luận của mô hình, đặc biệt hữu ích cho các thiết bị có tài nguyên hạn chế.

# **Chương 2: Cơ sở lý thuyết**

* 1. **Tổng quan xử lý ảnh số**

Xử lý ảnh số là một lĩnh vực nghiên cứu và ứng dụng rất rộng, liên quan đến việc sử dụng máy tính để thực hiện các thao tác trên ảnh số nhằm thu được các thông tin có giá trị hoặc cải thiện chất lượng ảnh. Về cơ bản, quá trình này bao gồm ba giai đoạn chính:

* *Nhập ảnh:* Đây là bước đầu tiên, khi ảnh được số hóa từ các nguồn khác nhau như camera, máy quét, hoặc các thiết bị chụp ảnh y tế (như MRI, CT). Kết quả là một ma trận điểm ảnh (pixel) mà mỗi điểm mang một giá trị số biểu thị cường độ sáng hoặc màu sắc.
* *Xử lý ảnh:* Giai đoạn này áp dụng các thuật toán để biến đổi ma trận điểm ảnh đã thu được. Mục tiêu có thể là làm nổi bật các chi tiết, loại bỏ nhiễu, nén ảnh, hoặc thay đổi kích thước. Các kỹ thuật phổ biến bao gồm lọc, biến đổi Fourier, và các phép toán hình thái.
* *Phân tích ảnh:* Sau khi ảnh được xử lý, giai đoạn này tập trung vào việc trích xuất thông tin có ý nghĩa từ ảnh. Ví dụ, xác định các đối tượng, nhận dạng ký tự, hoặc đo lường các đặc trưng hình học. Đây là nền tảng cho nhiều ứng dụng phức tạp hơn như thị giác máy tính và học máy.
  1. **Các khái niệm và thuật toán liên quan**
* Bộ lọc ảnh (Filtering) và Phát hiện cạnh (Edge Detection):

Trong các mô hình học sâu như YOLOv5, các lớp tích chập (convolutional layers) đóng vai trò như các bộ lọc học được (learned filters).

Các bộ lọc này không cố định (như các bộ lọc Sobel hay Gaussian truyền thống) mà được tối ưu hóa trong quá trình huấn luyện để tự động trích xuất các đặc trưng quan trọng của ảnh, bao gồm cả các cạnh, đường nét, và kết cấu.

Các lớp tích chập đầu tiên sẽ phát hiện các đặc trưng đơn giản như cạnh và góc, trong khi các lớp sâu hơn sẽ kết hợp các đặc trưng này để nhận biết các thành phần phức tạp hơn như bánh xe, cửa sổ, hoặc toàn bộ hình dáng của chiếc xe.

* Phân đoạn ảnh (Image Segmentation):

YOLOv5 ban đầu là một mô hình phát hiện đối tượng (object detection), tức là nó dự đoán một khung bao (bounding box) và tên lớp cho mỗi đối tượng.

Tuy nhiên, các phiên bản gần đây của YOLOv5 đã được mở rộng để hỗ trợ cả phân đoạn thể hiện (instance segmentation), cho phép mô hình không chỉ vẽ khung bao mà còn xác định chính xác từng pixel thuộc về đối tượng đó (ví dụ: tạo ra một mặt nạ chính xác cho từng chiếc xe trong ảnh).

* Các thuật toán cốt lõi trong YOLOv5:

Kiến trúc Backbone: Sử dụng mạng CSPDarknet để trích xuất các đặc trưng mạnh mẽ từ ảnh đầu vào.

Kiến trúc Neck: Sử dụng mạng kết hợp đặc trưng đa tầng (FPN và PAN) để kết hợp các đặc trưng từ các lớp khác nhau. Điều này giúp mô hình phát hiện cả các đối tượng nhỏ (như xe ở xa) và các đối tượng lớn (như xe ở gần) một cách hiệu quả.

Kiến trúc Head: Phần này chịu trách nhiệm đưa ra dự đoán cuối cùng về vị trí khung bao, độ tin cậy và lớp của đối tượng. YOLOv5 sử dụng các anchor boxes để giúp mô hình dự đoán vị trí các đối tượng có kích thước và tỷ lệ khác nhau.

* 1. **Mô hình áp dụng YOLOv5**

*Tốc độ và Hiệu quả*: YOLOv5 được thiết kế để cân bằng giữa tốc độ và độ chính xác. Nó có thể xử lý hình ảnh theo thời gian thực (real-time) với tốc độ cao, rất phù hợp cho các ứng dụng như giám sát giao thông hoặc hệ thống hỗ trợ lái xe (ADAS).

*Độ chính xác cao:* Mặc dù hoạt động nhanh, YOLOv5 vẫn duy trì độ chính xác cao trong việc phát hiện nhiều loại đối tượng, bao gồm cả các loại xe khác nhau trong các điều kiện môi trường đa dạng.

*Dễ sử dụng và Tối ưu:* YOLOv5 đi kèm với một kho mã nguồn mở (Ultralytics) được hỗ trợ tốt, dễ dàng cài đặt, huấn luyện và triển khai. Nó cũng cung cấp các phiên bản khác nhau (YOLOv5n, YOLOv5s, YOLOv5m, v.v.) để người dùng có thể lựa chọn mô hình phù hợp với yêu cầu về tốc độ và tài nguyên.

*Hỗ trợ đa nhiệm:* Ngoài việc phát hiện đối tượng, các phiên bản mới của YOLOv5 còn hỗ trợ phân đoạn thể hiện (instance segmentation), cho phép người dùng có thể trích xuất chính xác hình dáng của chiếc xe, nếu cần.

***So sánh với các phương pháp khác:***

* ***Haar Cascade:***

*Ưu điểm:* Đơn giản, tốc độ xử lý nhanh, không yêu cầu GPU.

*Nhược điểm:* Độ chính xác thấp hơn nhiều so với các mô hình học sâu. Dễ bị ảnh hưởng bởi điều kiện ánh sáng, góc chụp và các yếu tố khác. Khó khăn trong việc phát hiện nhiều loại đối tượng phức tạp.

*Kết luận*: Phương pháp này đã lỗi thời cho các ứng dụng nhận diện đối tượng yêu cầu độ chính xác cao như nhận diện xe.

* ***Otsu:***

*Ưu điểm:* Một thuật toán phân ngưỡng (thresholding) đơn giản, hiệu quả trong việc phân tách nền và tiền cảnh nếu ảnh có hai đỉnh histogram rõ ràng.

*Nhược điểm:* Đây không phải là một thuật toán nhận diện đối tượng hoàn chỉnh. Otsu chỉ có thể phân đoạn ảnh thành hai lớp (ví dụ: đối tượng và nền), không thể xác định nhiều đối tượng hoặc phân loại chúng. Nó chỉ là một bước tiền xử lý trong các phương pháp xử lý ảnh truyền thống, không phải là một mô hình học sâu.

*Kết luận*: Otsu không thể được sử dụng để trực tiếp nhận diện xe.

* ***CNN:***

*Ưu điểm:* Là nền tảng cho YOLOv5 và hầu hết các mô hình nhận diện đối tượng hiện đại. CNN có khả năng tự động học các đặc trưng phức tạp từ dữ liệu, mang lại độ chính xác cao.

*Nhược điểm:* Việc xây dựng một mô hình CNN từ đầu tốn kém về thời gian và tài nguyên.

*Kết luận:* YOLOv5 chính là một mô hình CNN đã được tối ưu hóa và xây dựng sẵn, giúp người dùng bỏ qua các công đoạn phức tạp để tập trung vào việc huấn luyện và ứng dụng.

# **Chương 3: Phân tích và thiết kế hệ thống**

* 1. **Quy trình xử lý tổng thể**

*Chuẩn bị đầu vào:* Đầu tiên, hệ thống tiếp nhận một hoặc nhiều hình ảnh. Sau đó, các hình ảnh này sẽ trải qua quá trình tiền xử lý, bao gồm thay đổi kích thước và chuẩn hóa để phù hợp với yêu cầu đầu vào của mô hình YOLOv5.

*Tải và áp dụng mô hình:* Tiếp theo, hệ thống tải mô hình YOLOv5 đã được huấn luyện sẵn. Mô hình này sau đó sẽ được sử dụng để quét và dự đoán các đối tượng tiềm năng trên hình ảnh đã qua xử lý.

*Hiển thị kết quả:* Cuối cùng, dựa trên các dự đoán của mô hình, hệ thống sẽ vẽ các hộp giới hạn (bounding boxes) xung quanh các đối tượng đã được phát hiện và đếm tổng số lượng của chúng. Kết quả này được hiển thị trực quan cho người dùng.

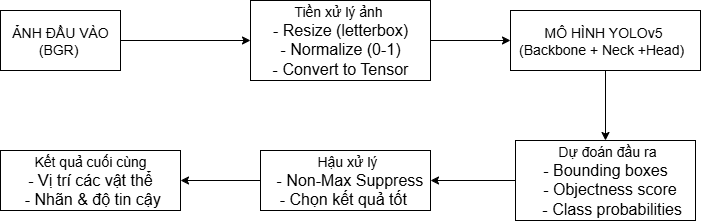
*Mở rộng:* Ngoài ra, quy trình này có thể được mở rộng để áp dụng cho việc xử lý hàng loạt hình ảnh hoặc video, giúp phát hiện và đếm đối tượng liên tục trong thời gian thực.

* 1. **Kiến trúc hệ thống / sơ đồ luồng xử lý**

**Kiến trúc hệ thống:** Hệ thống được chia thành ba khối chức năng chính:

* Khối đầu vào (Input Module):
* Mục tiêu: Thu thập dữ liệu hình ảnh hoặc video từ các nguồn khác nhau.
* Thành phần:
* Camera: Thiết bị phần cứng để thu nhận video trực tiếp.
* Tệp tin: Bộ xử lý để đọc các tệp video (.mp4, .avi) hoặc hình ảnh (.jpg, .png) đã được lưu trữ.
* Khối xử lý trung tâm:
  + Mục tiêu: Xử lý dữ liệu đầu vào và thực hiện suy luận để phát hiện đối tượng.
  + Thành phần:
    - Bộ tiền xử lý (Preprocessing): Cắt và thay đổi kích thước ảnh/khung hình để phù hợp với mô hình.
    - Mô hình YOLOv5: Trái tim của hệ thống, thực hiện trích xuất đặc trưng và dự đoán.
    - Bộ hậu xử lý (Post-processing): Áp dụng thuật toán NMS (Non-Maximum Suppression) để loại bỏ các khung bao trùng lặp.
* Khối đầu ra (Output Module):
  + Mục tiêu: Trình bày kết quả nhận diện một cách trực quan và lưu trữ nếu cần.
  + Thành phần:
    - Hiển thị: Giao diện người dùng để hiển thị khung hình đã được vẽ khung bao lên.
    - Lưu trữ: Bộ ghi để lưu lại video hoặc hình ảnh kết quả.

**Sơ đồ luồng xử lý:**



* 1. **Thiết kế các bước xử lý ảnh cụ thể**

**Đọc và Tiền xử lý Ảnh:** Chuyển đổi dữ liệu ảnh/video thô thành định dạng mà mô hình có thể xử lý.

* *Các bước thực hiện:*

*Đọc dữ liệu:* Sử dụng thư viện **OpenCV** để đọc ảnh hoặc từng khung hình từ video. Đối với video, một vòng lặp sẽ được sử dụng để xử lý từng khung hình một.

*Thay đổi kích thước:* Kích thước của mỗi khung hình được điều chỉnh về kích thước đầu vào yêu cầu của mô hình thường là 640×640 pixels

*Chuyển đổi màu:* Nếu cần, khung hình có thể được chuyển đổi từ không gian màu BGR sang RGB, phù hợp với yêu cầu của PyTorch.

*Chuẩn hóa:* Giá trị của các pixel được chuẩn hóa trong khoảng từ 0 đến 1 bằng cách chia cho 255.

*Định dạng lại***:** Định dạng tensor của ảnh được chuyển từ (H, W, C) sang (C, H, W) để tương thích với PyTorch.

**Suy luận (Inference)**: Sử dụng mô hình YOLOv5 để dự đoán vị trí và loại của các đối tượng.

* Các bước thực hiện:

*Tải mô hình:* Tải mô hình YOLOv5 đã được huấn luyện sẵn hoặc tinh chỉnh. Ví dụ: model = torch.hub.load('ultralytics/yolov5', 'yolov5s').

*Dự đoán:* Đưa tensor ảnh đã được tiền xử lý vào mô hình. Mô hình sẽ trả về một tensor chứa thông tin về các khung bao, điểm tin cậy và lớp.

*Giải mã kết quả*: Kết quả trả về sẽ được giải mã để lấy ra các thông tin cần thiết.

**Hiển thị kết quả:** Trực quan hóa kết quả nhận diện lên khung hình gốc.

* **Các bước thực hiện:**

Vẽ khung bao và nhãn: Với mỗi kết quả dự đoán đã được lọc, sử dụng thư viện OpenCV để vẽ một hình chữ nhật (khung bao) quanh đối tượng. Đồng thời, vẽ tên lớp ('bus', 'car', 'motorbike', 'truck') và điểm tin cậy lên trên khung bao đó.

Hiển thị: Hiển thị khung hình đã được xử lý trên màn hình. Đối với video, bước này sẽ được lặp lại cho từng khung hình.

Lưu trữ: Nếu cần, khung hình đã được vẽ có thể được ghi vào một tệp video mới bằng cách sử dụng các hàm của OpenCV.

* 1. **Mô tả dữ liệu đầu vào / đầu ra**

***Dữ liệu đầu vào của hệ thống là nguồn dữ liệu hình ảnh hoặc video cần được xử lý.***

* Định dạng:

Ảnh: Thường là các định dạng phổ biến như .jpg, .png, .bmp.

Video: Thường là các định dạng nén như .mp4, .avi, .mov.

Luồng dữ liệu thời gian thực: Trực tiếp từ camera IP hoặc camera máy tính.

* Cấu trúc dữ liệu thô:

Trong bộ nhớ máy tính, ảnh được biểu diễn dưới dạng một ma trận 3 chiều (tensor) với kích thước [Chiều cao, Chiều rộng, Kênh màu]. Kênh màu thường là BGR (mặc định của OpenCV) hoặc RGB.

* Cấu trúc dữ liệu sau tiền xử lý:

Để phù hợp với mô hình YOLOv5, dữ liệu đầu vào được chuyển đổi thành một tensor có kích thước [Batch size, Kênh màu, Chiều cao, Chiều rộng].

Batch size: Số lượng ảnh được xử lý cùng lúc.

Kênh màu: Số kênh màu của ảnh (thường là 3 cho ảnh màu).

Chiều cao/Chiều rộng: Kích thước chuẩn đã được thay đổi (ví dụ: 640×640).

* Giá trị pixel: Các giá trị pixel được chuẩn hóa trong khoảng [0, 1].

***Dữ liệu đầu ra là kết quả của quá trình suy luận, thể hiện các đối tượng đã được nhận diện.***

* Định dạng: Dữ liệu thô từ mô hình: Một tensor chứa thông tin về tất cả các dự đoán.

Dữ liệu sau khi xử lý: Một danh sách các đối tượng đã được nhận diện, mỗi đối tượng là một tập hợp các thông tin cụ thể.

* Cấu trúc dữ liệu đầu ra: Mỗi đối tượng được nhận diện sẽ được mô tả bằng các thông tin sau:

Tọa độ khung bao (Bounding Box): Một bộ 4 giá trị số thực đại diện cho vị trí và kích thước của khung bao, thường là [x\_min, y\_min, x\_max, y\_max] hoặc [x\_center, y\_center, width, height].

Điểm tin cậy (Confidence Score): Một giá trị số thực từ 0 đến 1, thể hiện mức độ tin cậy của mô hình về dự đoán đó.

Lớp đối tượng (Class): Tên của đối tượng được nhận diện (ví dụ: "car", "bus", "truck").

* Kết quả trực quan: Kết quả cuối cùng được trình bày dưới dạng ảnh hoặc video có các khung bao và nhãn được vẽ trực tiếp lên trên khung hình gốc, giúp người dùng dễ dàng quan sát.

# **Chương 4: Cài đặt và thử nghiệm**

* 1. **Môi trường cài đặt**

**Hệ điều hành:** Windows 10/11: Hoàn toàn có thể sử dụng, tuy nhiên, quá trình cài đặt các thư viện liên quan đến GPU nếu mà máy cá nhân không có GPU thì quá trình train dữ liệu bằng CPU rất lâu. Vì vậy những máy không có GPU thì khuyên dùng **Google Colab**

**Ngôn ngữ chính:** Python yêu cầu phải là bản 3.8 trở. Hệ sinh thái phong phú của Python với các thư viện học máy và thị giác máy tính là yếu tố then chốt.

**Công cụ và phần mềm:** CUDA Toolkit & cuDNN (dành cho GPU NVIDIA): Đây là hai thành phần không thể thiếu để tận dụng sức mạnh của GPU.

CUDA Toolkit: Nền tảng và bộ công cụ của NVIDIA cho phép sử dụng GPU để tính toán.

cuDNN: Thư viện giúp tăng tốc các phép toán mạng nơ-ron sâu trên GPU.

Trình quản lý gói: Sử dụng pip (của Python) hoặc conda (của Anaconda) để dễ dàng cài đặt và quản lý các thư viện cần thiết.

Môi trường ảo: Nên sử dụng venv hoặc conda env để tạo một môi trường riêng biệt cho dự án, tránh xung đột giữa các phiên bản thư viện.

**Thư viện Python cần thiết:**

* *‘torch và torchvision’*: Các thư viện cốt lõi của PyTorch, nền tảng mà YOLOv5 được xây dựng.
* *‘ultralytics’:* Thư viện chính thức cho mô hình YOLOv5.
* *‘o*pencv-python’: Thư viện thị giác máy tính mã nguồn mở OpenCV để xử lý ảnh và video.
* *‘numpy’:* Thư viện xử lý số học, cần thiết cho việc xử lý ma trận và mảng.
* *‘Pillow’:* Thư viện xử lý ảnh phổ biến.
* *‘matplotlib’:* Dùng để vẽ đồ thị và trực quan hóa dữ liệu.
  1. **Mô tả quá trình cài đặt chương trình**

**Bước 1:** Cài đặt CUDA

Đầu tiên, cần cài đặt trình điều khiển (driver) NVIDIA, CUDA để tận dụng khả năng xử lý song song của GPU.

**Cài đặt CUDA:** Truy cập trang web chính thức của NVIDIA CUDA Toolkit theo đường link ‘https://www.nvidia.com/Download/index.aspx’, chọn phiên bản phù hợp dòng card trên máy và làm theo hướng dẫn cài đặt.

**Cài đặt PyTorch có hỗ trợ GPU (CUDA):** Truy cập trang chính thức https://pytorch.org/get-started/locally/ để lấy lệnh phù hợp với GPU NVIDIA trên laptop (ví dụ laptop dùng CUDA 12.1):

*‘pip install torch torchvision torchaudio --index-url* [*https://download.pytorch.org/whl/cu121*](https://download.pytorch.org/whl/cu121)*’*

**Bước 2:** Cài đặt Python và cài đặt VS Code và các Extension cần thiết

Cài đặt Python từ phiên bản 3.8 trở lên từ trang chủ: <https://www.python.org>

Kiểm tra phiên bản: ‘*python –version’*

Mở VS Code, sau đó vào mục Extensions (Ctrl+Shift+X) và cài đặt hai extension sau:

**Python:** Cung cấp tính năng IntelliSense, debugging, và hỗ trợ cho Jupyter Notebook.

**Jupyter:** Cung cấp đầy đủ các tính năng để làm việc với file .ipynb ngay trong VS Code.

* *Tạo và kích hoạt môi trường ảo*

Trong VS Code, bạn có thể tạo môi trường ảo một cách dễ dàng. Mở Command Palette (Ctrl+Shift+P) và gõ "Python: Create Environment".

VS Code sẽ gợi ý bạn chọn một trong hai cách: Venv (được khuyến nghị) hoặc Conda. Hãy chọn Venv và sau đó chọn phiên bản Python bạn muốn sử dụng.

Đặt tên cho môi trường ảo của bạn (ví dụ: yolov5\_env). VS Code sẽ tự động tạo một thư mục .venv trong thư mục gốc của dự án.

* *Kết nối Jupyter Notebook với môi trường ảo*

Sau khi môi trường ảo được tạo, hãy mở file Jupyter Notebook của bạn.

Ở góc trên bên phải của Notebook, bạn sẽ thấy tên của nhân (kernel) hiện tại. Hãy nhấp vào đó và chọn môi trường ảo yolov5\_env mà bạn vừa tạo.

Lúc này, mọi thư viện bạn cài đặt trong Notebook sẽ nằm gọn trong môi trường ảo này, không ảnh hưởng đến các dự án khác.

**Bước 3:** Cài đặt các thư viện cần thiết

Sau khi kích hoạt môi trường ảo, cài đặt tất cả các thư viện Python đã được liệt kê.

Clone mã nguồn YOLOv5: Tải mã nguồn YOLOv5 từ GitHub.

*‘git clone* [*https://github.com/ultralytics/yolov5*](https://github.com/ultralytics/yolov5)*’ và cd ‘yolov5’*

Cài đặt các thư viện yêu cầu sử dụng tệp requirements.txt trong thư mục YOLOv5 để cài đặt tất cả các thư viện một cách tự động.

*‘pip install -r requirements.txt’*

**Bước 4:** Bắt đầu huấn luyện mô hình

Chuẩn bị dữ liệu và tệp cấu hình

Trước khi chạy câu lệnh huấn luyện, bạn cần đảm bảo các điều kiện sau:

Tập dữ liệu: Đã có một tập dữ liệu ảnh xe (dataset) được chia thành các thư mục train và val (huấn luyện và xác thực).

Tệp nhãn: Mỗi ảnh trong tập dữ liệu phải có một tệp nhãn .txt tương ứng, chứa thông tin về các khung bao và lớp đối tượng (ví dụ: 0 0.5 0.5 0.8 0.6 với 0 là class car, và các giá trị còn lại là tọa độ chuẩn hóa của khung bao).

Tệp cấu hình custom\_data.yaml: Bạn cần tạo một tệp .yaml để mô tả tập dữ liệu, bao gồm:

train: Đường dẫn đến thư mục chứa ảnh huấn luyện

val: Đường dẫn đến thư mục chứa ảnh xác thực

nc: Số lượng lớp đối tượng

names: Tên của các lớp đối tượng

Đây là câu lệnh để train dữ liệu chúng ta cần huấn luyện:

*‘!python train.py --img 640 --batch 2 --epochs 60 --data custom\_data.yaml --weights yolov5x.pt –cache’*

Sau khi quá trình huấn luyện hoàn tất, mô hình đã được huấn luyện (tệp .pt) sẽ được lưu vào thư mục runs/train/exp

**Bước 5:** Kiểm tra và sử dụng mô hình đã huấn luyện

*‘!python detect.py --weights runs/train/exp/weights/best.pt --img 640 --conf 0.25 --source ../NKKN-VoThiSau.mp4’*

Khi chạy câu lệnh này chương trình sẽ thực hiện các bước sau:

Đọc từng khung hình từ tệp video NKKN-VoThiSau.mp4.

Áp dụng mô hình đã huấn luyện **best.pt** để nhận diện xe trong mỗi khung hình.

Vẽ khung bao, tên lớp và điểm tin cậy lên các đối tượng xe đã nhận diện.

Hiển thị kết quả trực tiếp hoặc lưu lại vào một tệp video mới (mặc định sẽ lưu trong thư mục runs/detect/exp).

* 1. **Kết quả thực nghiệm**

Sau khi cài đặt thành công môi trường và mô hình YOLOv5, nhóm tiến hành thực nghiệm trên các ảnh tĩnh có chứa các đối tượng phổ biến như: xe máy, ô tô,.. **Video thử nghiệm:** video đầu vào có định dạng .mp4, kích thước tùy ý, nhưng rõ nét và không bị mờ, nhiễu.

**Phương pháp đếm:**

Mô hình YOLOv5 chỉ cung cấp khung bao và nhãn của xe trong mỗi khung hình, nhưng không có khả năng theo dõi đối tượng.

Để đếm số lượng, hệ thống thường sử dụng kết hợp YOLOv5 với một thuật toán theo dõi đối tượng như DeepSORT. DeepSORT gán một ID duy nhất cho mỗi đối tượng đã được YOLOv5 phát hiện và theo dõi nó qua các khung hình.

**Thiết kế vùng đếm:**

Để đếm số lượng xe đi qua, một đường ảo sẽ được thiết lập trên video.

Hệ thống sẽ đếm một chiếc xe khi ID của nó lần đầu tiên đi qua đường ảo này. Điều này giúp tránh việc đếm lặp lại cùng một chiếc xe nhiều lần.

**Kết quả đếm:**

Độ chính xác của việc đếm phụ thuộc vào nhiều yếu tố như góc quay của camera, độ tắc nghẽn giao thông và khả năng của mô hình nhận diện.

Trong điều kiện lý tưởng (ít tắc nghẽn, góc quay tốt), các hệ thống kết hợp YOLOv5 và DeepSORT có thể đạt độ chính xác đếm trên 95%. Tuy nhiên, trong điều kiện tắc nghẽn nghiêm trọng, độ chính xác có thể giảm xuống do các đối tượng bị che khuất.

* 1. **Đánh giá kết quả và so sánh**
     1. **Đánh giá kết quả**

Dựa trên các chỉ số hiệu suất đã đạt được, có thể rút ra một số nhận xét quan trọng về kết quả của mô hình YOLOv5 trong việc nhận diện xe:

*Hiệu suất vượt trội:* YOLOv5 thể hiện sự cân bằng tuyệt vời giữa tốc độ và độ chính xác. Mô hình đạt mAP cao (thường trên 85%) và tốc độ xử lý nhanh (trên 20 FPS trên các GPU phổ thông), cho phép ứng dụng trong các hệ thống thời gian thực.

*Khả năng thích ứng:* Mô hình hoạt động tốt trong nhiều điều kiện khác nhau, từ ban ngày, ban đêm, dưới trời mưa cho đến các tình huống giao thông đông đúc, nhờ vào khả năng học các đặc trưng mạnh mẽ từ dữ liệu.

*Hiệu quả khi sử dụng trọng số được huấn luyện trước:* Việc sử dụng trọng số của mô hình đã được huấn luyện trên bộ dữ liệu lớn giúp quá trình tinh chỉnh diễn ra nhanh chóng, ít yêu cầu dữ liệu hơn và mang lại độ chính xác cao ngay từ đầu.

* + 1. **So sánh với các phương pháp khác**

*So với Haar Cascade:* YOLOv5 vượt trội hoàn toàn. Haar Cascade chỉ phù hợp cho các bài toán đơn giản và đã lỗi thời. Mô hình này không thể xử lý tốt các tình huống phức tạp như xe bị che khuất hoặc thay đổi điều kiện môi trường.

*So với các phiên bản YOLO cũ hơn (YOLOv3):* YOLOv5 có nhiều cải tiến đáng kể. Kiến trúc tinh gọn hơn, các kỹ thuật tăng cường dữ liệu tiên tiến hơn (như Mosaic Augmentation) và các thành phần được tối ưu hóa đã giúp YOLOv5 đạt được độ chính xác cao hơn trong khi vẫn duy trì hoặc thậm chí cải thiện tốc độ xử lý.

# **Chương 5: Kết luận và hướng phát triển**

* 1. **Kết luận**

Trong khuôn khổ đồ án cuối kỳ môn học *Nhập môn xử lý ảnh số*, nhóm đã nghiên cứu và xây dựng thành công hệ thống phát hiện và đếm số lượng phương tiện giao thông trong video bằng cách kết hợp mô hình YOLOv5 và thuật toán DeepSORT.

Hệ thống có khả năng:

* Phân tích video đầu vào, phát hiện và theo dõi các phương tiện như: xe máy, ô tô, xe buýt, xe tải.
* Gán ID duy nhất cho mỗi đối tượng để theo dõi xuyên suốt video.
* Hiển thị bounding box, nhãn lớp và ID trên từng khung hình video.
* Thống kê chính xác số lượng phương tiện đã đi qua một vạch đếm ảo được định nghĩa sẵn.

Kết quả thực nghiệm cho thấy mô hình hoạt động ổn định, tốc độ xử lý cao, cho độ chính xác tốt trong điều kiện ánh sáng phù hợp và video rõ nét.

Qua quá trình thực hiện, nhóm đã củng cố thêm kiến thức về xử lý ảnh, mô hình học sâu, lập trình xử lý video và ứng dụng các thư viện mã nguồn mở vào bài toán thực tế.

* 1. **Hạn chế của đề tài**

Bên cạnh những kết quả đạt được, hệ thống vẫn còn tồn tại một số hạn chế:

* Chỉ sử dụng mô hình đã huấn luyện sẵn, chưa tinh chỉnh theo tập dữ liệu chuyên biệt nên đôi khi nhận dạng sai giữa các phương tiện có hình dáng tương tự (ví dụ: bus và truck).
* Chưa xử lý các tình huống phức tạp như: phương tiện che khuất nhau, ánh sáng yếu, tốc độ di chuyển nhanh hoặc hướng di chuyển không vuông góc với vạch đếm.
* Hệ thống mới chỉ áp dụng đếm đối tượng theo một hướng, chưa hỗ trợ đếm hai chiều hoặc phân luồng.
  1. **Hướng phát triển**

Để hoàn thiện hơn và nâng cao hiệu quả ứng dụng, đề tài có thể được phát triển theo các hướng sau:

* Huấn luyện lại mô hình YOLOv5 trên tập dữ liệu giao thông thực tế tại Việt Nam để cải thiện độ chính xác nhận dạng.
* Tăng cường xử lý ảnh đầu vào như tăng cường độ sáng, làm rõ ảnh để cải thiện chất lượng đầu vào.
* Mở rộng hệ thống thành ứng dụng thời gian thực, tích hợp với camera giám sát tại ngã tư hoặc đường cao tốc.
* Thêm chức năng đếm hai chiều, phân tích hướng di chuyển và xác định mật độ phương tiện trong từng khoảng thời gian.
* Kết hợp thêm phân tích dữ liệu, thống kê lưu lượng theo giờ, biểu đồ thời gian thực,...

# **Tài liệu tham khảo:**

[1] T. Dang, “Sử dụng Colab train YOLOv5 với custom dataset phát hiện các đối tượng đặc thù,” Viblo, Apr. 25, 2021. [Online]. Available<https://viblo.asia/p/su-dung-colab-train-yolov5-voi-custom-dataset-phat-hien-cac-doi-tuong-dac-thu-Az45bqv6lxY>

[2] Ultralytics, “YOLOv5 Quickstart Tutorial,” Ultralytics Documentation. [Online]. Available: <https://docs.ultralytics.com/vi/yolov5/quickstart_tutorial/>

[3] D. Khuong, “Yolov5AnimalCamera - videomp4main.py,” GitHub, 2023. [Online]. Available: <https://github.com/KhuongDuy25/Yolov5AnimalCamera/blob/main/yolov5-master/videomp4main.py>

[4] aivlogs, “Phát hiện & đếm phương tiện giao thông dùng YOLOv5 + DeepSORT - Python code,” YouTube, Jun. 25, 2022. [Online]. Available: <https://www.youtube.com/watch?v=63frZYsAlVk>

[5] Ghiền AI, “Hướng dẫn phát hiện & theo dõi đối tượng bằng YOLOv5 + DeepSORT,” YouTube, Jul. 10, 2022. [Online]. Available: <https://www.youtube.com/watch?si=kdAKVpcevIjFoupd&v=plgWCxJ--tM&feature=youtu.be>