**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN**

**HỌC PHẦN: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**

**Đề tài số 14: Xây dựng hệ thống xác nhận đối tượng và đếm đối tượng trong ảnh**

**Giảng viên hướng dẫn: Lương Hồng Lan**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Mã sinh viên** | **Sinh viên thực hiện** | **Lớp hành chính** |
| **1** | 20210272 | Bùi Quốc Toàn | DCCNTT12.10.1 |

**Bắc Ninh, năm 2024**

|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á  **KHOA CÔNG NGHỆ THÔNG TIN** | **KỲ THI KẾT THÚC HỌC PHẦN**  **HỌC KỲ 1, NĂM HỌC 2024** – **2025** |

|  |  |
| --- | --- |
| **PHIẾU CHẤM THI BÀI TẬP LỚN KẾT THÚC HỌC PHẦN**  **Mã đề thi: 14:**  **Xây dựng hệ thống xác nhận đối tượng và đếm đối tượng trong ảnh**  **Tên học phần: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**  **Lớp Tín chỉ: XATGMT.03.K12.01.LH.C04.1\_LT** | |
| **Cán bộ chấm thi 1**  *(Ký và ghi rõ họ tên)*  Lương Thị Hồng Lan | **Cán bộ chấm thi 2**  *(Ký và ghi rõ họ tên)* |

| **TT** | **TIÊU CHÍ** | **THANG ĐIỂM** | **Bùi Quốc Toàn** |
| --- | --- | --- | --- |
| 20210272 |
| **1** | **Nội dung báo cáo trên Word đầy đủ** | **3.5** |  |
| 1.1 | Có bố cục rõ ràng (mục lục, phần mở đầu, nội dung chính, kết luận). | 0,5 |  |
| 1.2 | Nội dung phân tích rõ ràng, logic. | 0,5 |  |
| 1.3 | Có dẫn chứng, số liệu minh họa đầy đủ. | 0,5 |  |
| 1.4 | Ngôn ngữ và trình bày chuẩn, không lỗi chính tả. | 0,5 |  |
| 1.5 | Có trích dẫn tài liệu tham khảo đúng quy cách. | 0,5 |  |
| 1.6 | Được trình bày chuyên nghiệp (canh lề, font chữ, khoảng cách dòng hợp lý). | 0,5 |  |
| 1.7 | Tài liệu đầy đủ, bám sát yêu cầu của đề bài. | 0,5 |  |
| **2** | **Nội dung thuyết trình đầy đủ** | **1.0** |  |
| 2.1 | Trình bày tự tin, phát âm rõ ràng, mạch lạc. | 0,5 |  |
| 2.2 | Nội dung thuyết trình đúng trọng tâm, không lan man. | 0,5 |  |
| **3** | **Slides báo cáo đầy đủ nội dung + Hỏi đáp** | **3.0** |  |
| 3.1 | Slides có bố cục rõ ràng (mở đầu, nội dung, kết luận). | 0,5 |  |
| 3.2 | Thiết kế slides đẹp, chuyên nghiệp (màu sắc, hình ảnh minh họa). | 0,5 |  |
| 3.3 | Nội dung trên slides ngắn gọn, dễ hiểu, súc tích. | 0,5 |  |
| 3.4 | Nội dung slides phù hợp với nội dung báo cáo. | 0,5 |  |
| 3.5 | Trả lời câu hỏi đầy đủ, chính xác. | 0,5 |  |
| 3.6 | Trả lời câu hỏi tự tin, thuyết phục. | 0,5 |  |
| **4** | **Code đầy đủ** | **2.5** |  |
| 1.1 | Code được trình bày rõ ràng, có chú thích đầy đủ. | 0,5 |  |
| 1.2 | Code chạy đúng, không lỗi. | 0,5 |  |
| 1.3 | Code tối ưu, không dư thừa. | 0,5 |  |
| 1.4 | Đáp ứng đầy đủ các yêu cầu chức năng theo đề bài. | 0,5 |  |
| 1.5 | Có tính sáng tạo hoặc cải thiện so với yêu cầu. | 0,5 |  |
| **TỔNG ĐIỂM BẰNG SỐ:** | | **10** |  |
| **TỔNG ĐIỂM BẰNG CHỮ:** | | *Mười tròn* |  |

**MỤC LỤC**

[**LỜI NÓI ĐẦU** 6](#_Toc184454119)

[**CHƯƠNG I: TỔNG QUAN** 7](#_Toc184454120)

[**1.1.** **Tổng quan về thị giác máy tính** 7](#_Toc184454121)

[**1.1.1.** **Thị giác máy tính là gì ?** 7](#_Toc184454122)

[**1.1.2.** **Thị giác máy tính hoạt động như thế nào?** 7](#_Toc184454123)

[**1.1.3.** **Sự phát triển của thị giác máy tính** 9](#_Toc184454124)

[**1.2.** **Tổng quan về nhận dạng đối tượng** 11](#_Toc184454125)

[**1.2.1.** **Nhận dạng đối tượng là gì?** 11](#_Toc184454126)

[**1.2.2.** **Các phương pháp sử dụng để nhận dạng đối tượng** 11](#_Toc184454127)

[**1.3.** **Nhận dạng đối tượng và bài toán đặt ra** 11](#_Toc184454128)

[**1.3.1.** **Phân loại đối tượng** 12](#_Toc184454129)

[**1.3.2.** **Xác định vị trí đối tượng** 12](#_Toc184454130)

[**1.3.3.** **Tính đa dạng và phức tạp của hình ảnh** 13](#_Toc184454131)

[**1.3.4.** **Xử lý chồng lấn:** 14](#_Toc184454132)

[**1.3.5.** **Tốc độ xử lý và hiệu năng:** 14](#_Toc184454133)

[**1.3.6.** **Độ chính xác:** 15](#_Toc184454134)

[**1.3.7.** **Lựa chọn mô hình phù hợp** 15](#_Toc184454135)

[**1.4.** **Tổng quân về học máy** 15](#_Toc184454136)

[**1.4.1.** **Học máy là gì?** 15](#_Toc184454137)

[**1.4.2.** **Các phương pháp học máy** 16](#_Toc184454138)

[**1.5.** **Ngôn ngữ và công cụ lập trình** 17](#_Toc184454139)

[**1.5.1.** **Ngôn ngữ lập trình** 17](#_Toc184454140)

[**1.5.2.** **Công cụ lập trình** 19](#_Toc184454141)

[**CHƯƠNG II: XÂY DỰNG HỆ THỐNG** 20](#_Toc184454142)

[**2.1.** **Yêu cầu bài toán** 20](#_Toc184454143)

[**2.2.** **Mô hình huấn luyện YOLO** 21](#_Toc184454144)

[**2.2.1** **Giới thiệu mô hình YOLOv5s** 22](#_Toc184454145)

[**2.2.2.** **Ứng dụng vào hệ thống** 24](#_Toc184454146)

[**2.3.** **Xây dựng hệ thống** 25](#_Toc184454147)

[**2.3.1.** **Các bước thực hiện** 25](#_Toc184454148)

[**2.3.2.** **Giao diện chương trình** 27](#_Toc184454149)

[**CHƯƠNG III: PHÂN TÍCH QUY TRÌNH HOẠT ĐỘNG CỦA HỆ THỐNG** 28](#_Toc184454150)

[**3.1. Mục tiêu hoạt động** 28](#_Toc184454151)

[**3.2. Quy trình hoạt động** 28](#_Toc184454152)

[**3.3. Mô hình hóa quy trình hoạt động** 29](#_Toc184454153)

[**3.4. Ưu điểm và hạn chế của hệ thống** 29](#_Toc184454154)

[**3.5. Kết luận** 30](#_Toc184454155)

[**TÀI LIỆU THAM KHẢO** 31](#_Toc184454156)

# **LỜI NÓI ĐẦU**

Trong thời đại cách mạng công nghiệp 4.0, công nghệ trí tuệ nhân tạo (AI) và học sâu (Deep Learning) đã và đang mang lại những bước tiến vượt bậc trong lĩnh vực xử lý ảnh và thị giác máy tính (Computer Vision). Một trong những ứng dụng nổi bật là việc nhận diện và đếm đối tượng trong ảnh, một công nghệ có tiềm năng ứng dụng rộng rãi trong nhiều lĩnh vực như an ninh, giám sát, giao thông thông minh, quản lý kho hàng và các hệ thống bán lẻ.

Hệ thống nhận dạng và đếm đối tượng không chỉ giúp tự động hóa các quy trình phức tạp mà còn nâng cao hiệu suất, độ chính xác và khả năng ra quyết định trong thời gian thực. Tuy nhiên, việc xây dựng một hệ thống nhận dạng và đếm đối tượng vẫn đặt ra nhiều thách thức, như xử lý các đối tượng nhỏ, bị che khuất, hoặc trong các môi trường phức tạp. Ngoài ra, sự đa dạng của dữ liệu đầu vào, từ các góc nhìn khác nhau đến điều kiện ánh sáng không ổn định, cũng là một yếu tố cần được xem xét kỹ lưỡng.

Đề tài "Xây dựng hệ thống xác nhận đối tượng và đếm đối tượng trong ảnh" được thực hiện nhằm nghiên cứu và ứng dụng các kỹ thuật tiên tiến trong lĩnh vực thị giác máy tính, đặc biệt là các mô hình học sâu, để xây dựng một hệ thống có khả năng nhận diện và đếm đối tượng hiệu quả. Qua đó, đề tài không chỉ tập trung vào việc phát triển các giải pháp kỹ thuật mà còn hướng đến việc ứng dụng thực tiễn, đáp ứng yêu cầu của các lĩnh vực cụ thể trong đời sống và sản xuất.

Hy vọng rằng, kết quả của đề tài sẽ góp phần vào việc giải quyết các bài toán thực tế, đồng thời mở ra nhiều hướng nghiên cứu và ứng dụng mới trong tương lai.

# **CHƯƠNG I: TỔNG QUAN**

* 1. **Tổng quan về thị giác máy tính**
     1. **Thị giác máy tính là gì ?**

Thị giác máy tính là một lĩnh vực của Trí tuệ nhân tạo (AI - Artificial Intelligence), cho phép máy tính và hệ thống lấy thông tin hữu ích từ hình ảnh kỹ thuật số, video và các đầu vào trực quan khác.

Sau đó, công nghệ này thực hiện phân tích, đưa ra đề xuất dựa trên thông tin đó. Nếu AI cho phép máy tính suy nghĩ, thì thị giác máy tính cho phép chúng nhìn, quan sát và hiểu.

Thị giác máy tính hoạt động giống như thị giác của con người. Tuy nhiên, ở thị giác con người, khi nhìn thấy hình ảnh có thể gợi nhớ hoặc tạo nên ký ức, suy nghĩ. Với máy tính, tất cả các hình ảnh đều là một mảng các pixel, các giá trị số đại diện cho các sắc độ của màu đỏ, xanh lá cây và xanh dương.

Thị giác con người có lợi thế trong việc lưu trữ, phân tích bối cảnh. Thị giác máy tính cần được huấn luyện để thực hiện những chức năng này trong thời gian rất ngắn.

Một hệ thống ứng dụng thị giác máy tính được đào tạo để kiểm tra sản phẩm có thể phân tích hàng nghìn sản phẩm hoặc quy trình cùng lúc, phát hiện lỗi hoặc những vấn đề mà con người chưa kịp nhận thấy.

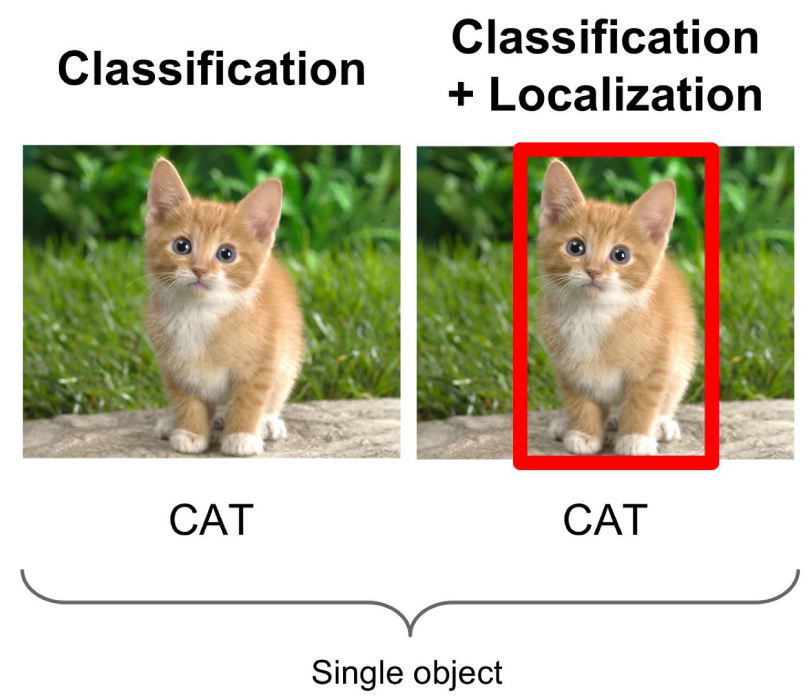
* + 1. **Thị giác máy tính hoạt động như thế nào?**

Một trong những câu hỏi thường được đặt ra trong cả Khoa học thần kinh và Học máy là: Bộ não của chúng ta hoạt động chính xác như thế nào và làm thế nào chúng ta có thể ước tính điều đó bằng các thuật toán của riêng mình? Thực tế là có rất ít lý thuyết toàn diện về tính toán của não; vì vậy, mặc dù thực tế là Mạng thần kinh nhân tạo được cho là “bắt chước cách thức hoạt động của não”, nhưng không ai dám chắc điều đó có thực sự đúng hay không.

Nghịch lý tương tự cũng đúng với thị giác máy tính - vì chúng ta không quyết định cách não và mắt xử lý hình ảnh, nên rất khó để nói các thuật toán được sử dụng trong sản xuất gần đúng với các quá trình tinh thần bên trong của chúng ta như thế nào.

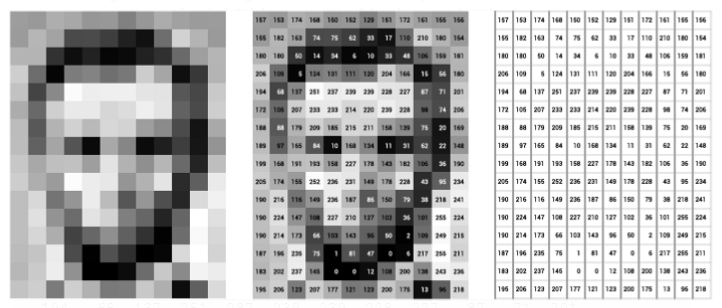
Ở một mức độ nhất định, Thị giác máy tính là thuộc về nhận dạng mẫu. Vì vậy, một cách để “huấn luyện” máy tính hiểu dữ liệu trực quan là cung cấp cho nó hàng nghìn, hàng triệu hình ảnh nếu có thể đã được “đánh dấu” và sau đó áp dụng các kỹ thuật phần mềm hoặc thuật toán khác nhau cho phép máy tính tìm kiếm trong tất cả các phần tử có liên quan đến các mẫu được đánh dấu đó. Đây là lúc các kĩ thuật AI, cụ thể là machine learning hay deep learning được áp dụng để phân loại đối tượng.  Để làm được điều này, trước đó mô hình (model) AI đã được cho học ảnh mẫu, thế nào là người, thế nào là xe hơi, thế nào là con mèo… Khi gặp một ảnh mới cần dự đoán, nó sẽ lấy những gì từng được học ra để đoán.

Ví dụ: Nếu bạn cho máy tính nhận biết một triệu hình ảnh con mèo, tất cả chúng sẽ phải tuân theo các thuật toán cho phép chúng phân tích màu sắc trong ảnh, hình dạng, khoảng cách giữa các hình... để nó xác định một hồ sơ về “mèo” có nghĩa là gì. Khi quá trình hoàn tất, máy tính (về lý thuyết) sẽ có thể sử dụng trải nghiệm của mình nếu được cung cấp các hình ảnh không được gắn nhãn khác để tìm những hình ảnh của mèo.



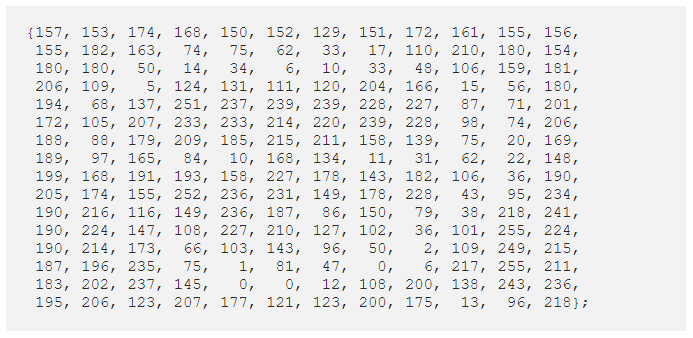
*Hình 1.1: Ảnh minh họa cho máy tính nhận biết một triệu hình ảnh con mèo*

Dưới đây là một minh họa đơn giản về bộ đệm hình ảnh thang độ xám lưu trữ hình ảnh. Độ sáng của mỗi pixel được biểu thị bằng một số 8 bit, có phạm vi từ 0 (đen) đến 255 (trắng):



Hình 1.2: Bộ đệm hình ảnh thang độ xám lưu trữ hình ảnh

Trên thực tế, các giá trị pixel hầu như được lưu trữ phổ biến, ở cấp độ phần cứng, trong một mảng một chiều. Ví dụ: dữ liệu từ hình ảnh trên được lưu trữ theo cách tương tự như danh sách dài các ký tự không dấu này:



Bây giờ bạn hãy tưởng tượng bức tranh trên có màu. Mọi thứ bắt đầu trở nên phức tạp hơn. Máy tính thường đọc màu dưới dạng một chuỗi 3 giá trị - đỏ, lục và lam (RGB) - trên cùng một thang 0–255. Bây giờ, mỗi pixel thực sự có 3 giá trị để máy tính lưu trữ ngoài vị trí của nó. Nếu chúng ta tô màu cho bức tranh đó sẽ dẫn đến giá trị 12 x 16 x 3 hoặc 576 số.

Đó là yêu cầu rất nhiều bộ nhớ cho một hình ảnh và rất nhiều pixel để một thuật toán lặp lại. Tuy nhiên, để đào tạo một mô hình với độ chính xác có ý nghĩa, đặc biệt là khi bạn đang nói về Học sâu, bạn thường cần hàng chục nghìn hình ảnh và càng nhiều hình ảnh càng tốt.

* + 1. **Sự phát triển của thị giác máy tính**

Trước khi Học sâu ra đời, các tác vụ mà thị giác máy tính có thể thực hiện rất hạn chế và đòi hỏi nhiều nỗ lực và mã hóa thủ công của các nhà phát triển và người vận hành. Ví dụ: nếu bạn muốn thực hiện nhận dạng khuôn mặt, bạn sẽ phải thực hiện các bước sau:

* Tạo cơ sở dữ liệu: Bạn phải chụp từng ảnh của tất cả các đối tượng bạn muốn theo dõi ở một định dạng cụ thể.
* Chú thích hình ảnh: Sau đó, đối với mỗi hình ảnh riêng lẻ, bạn sẽ phải nhập một số điểm dữ liệu chính, chẳng hạn như khoảng cách giữa hai mắt, chiều rộng của sống mũi, khoảng cách giữa môi trên và mũi và hàng chục phép đo khác xác định các đặc điểm riêng biệt của mỗi người.
* Chụp ảnh mới: Tiếp theo, bạn sẽ phải chụp ảnh mới, kể cả ảnh hay nội dung video. Và sau đó bạn phải thực hiện lại quá trình đo lường, đánh dấu các điểm chính trên hình ảnh. Bạn cũng phải tính đến góc ảnh được chụp.

Sau tất cả công việc thủ công này, ứng dụng cuối cùng sẽ có thể so sánh các phép đo trong hình ảnh mới với các phép đo được lưu trữ trong cơ sở dữ liệu của nó và cho bạn biết liệu nó có tương ứng với bất kỳ cấu hình nào mà nó đang theo dõi hay không. Trên thực tế, hầu hết công việc được thực hiện thủ công và biên độ lỗi vẫn còn lớn.

Học máy cung cấp một cách tiếp cận khác để giải quyết các vấn đề về thị giác máy tính. Với Học máy, các nhà phát triển không còn cần phải viết mã thủ công từng quy tắc vào các ứng dụng thị giác của họ. Thay vào đó, họ lập trình “các tính năng”, các ứng dụng nhỏ hơn có thể phát hiện các mẫu cụ thể trong hình ảnh. Sau đó, họ sử dụng một thuật toán học thống kê như hồi quy tuyến tính, hồi quy logistic, hoặc máy vectơ hỗ trợ (SVM) để phát hiện các mẫu, phân loại hình ảnh và phát hiện các đối tượng trong đó.

Học sâu cung cấp một cách tiếp cận cơ bản khác để thực hiện học máy. Học sâu dựa trên mạng nơ-ron, một chức năng có mục đích chung có thể giải quyết bất kỳ vấn đề nào có thể biểu diễn thông qua các ví dụ. Khi bạn cung cấp một mạng nơ-ron có nhiều ví dụ được gắn nhãn về một loại dữ liệu cụ thể, nó sẽ có thể trích xuất các mẫu chung giữa các ví dụ đó và biến đổi nó thành một phương trình toán học giúp phân loại các phần thông tin trong tương lai.

Ví dụ: việc tạo một ứng dụng nhận dạng khuôn mặt với học sâu chỉ yêu cầu bạn phát triển hoặc chọn một thuật toán được cấu trúc sẵn và đào tạo nó với các ví dụ về khuôn mặt của những người mà nó phải phát hiện.

Học sâu là một phương pháp rất hiệu quả để thực hiện thị giác máy tính. Trong hầu hết các trường hợp, việc tạo ra một thuật toán học sâu giúp thu thập một lượng lớn dữ liệu đào tạo được gắn nhãn và điều chỉnh các tham số như loại và số lớp của mạng nơ-ron... So với các loại học máy trước đây, học sâu dễ phát triển và triển khai hơn.

Hiện nay, thị giác máy tính được ứng dụng triển khai trong phát hiện ung thư, ô tô tự lái và nhận dạng khuôn mặt đều sử dụng học sâu. Học sâu và mạng nơ-ron sâu đã chuyển từ lĩnh vực khái niệm sang các ứng dụng thực tế nhờ tính khả dụng và những tiến bộ trong tài nguyên phần cứng và điện toán đám mây.

* 1. **Tổng quan về nhận dạng đối tượng**
     1. **Nhận dạng đối tượng là gì?**

**Nhận dạng đối tượng** (Object Recognition) là một lĩnh vực cốt lõi của thị giác máy tính, tập trung vào việc xác định, phân loại và đôi khi xác định vị trí của các đối tượng trong hình ảnh hoặc video. Đây là công nghệ nền tảng giúp máy móc "hiểu" và tương tác với thế giới thực thông qua dữ liệu trực quan, từ đó hỗ trợ nhiều ứng dụng trong đời sống và công nghiệp.

Nhận dạng đối tượng là một phần quan trọng trong thị giác máy tính, giúp máy móc "hiểu" và tương tác với thế giới thực. Với sự phát triển nhanh chóng của trí tuệ nhân tạo và học sâu, công nghệ này đang ngày càng trở nên chính xác và mạnh mẽ hơn, mở ra nhiều cơ hội trong việc tự động hóa và cải tiến hiệu suất trong các lĩnh vực từ y tế, giao thông đến công nghiệp và giải trí.

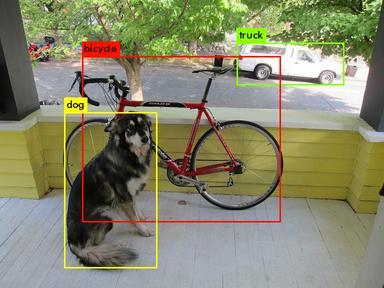
* + 1. **Các phương pháp sử dụng để nhận dạng đối tượng**

Nhận dạng đối tượng là một lĩnh vực quan trọng trong thị giác máy tính, với nhiều phương pháp được phát triển từ kỹ thuật truyền thống đến học sâu. Phương pháp truyền thống dựa trên đặc trưng thủ công như HOG, SIFT hoặc SURF, kết hợp với các thuật toán học máy như SVM hoặc KNN, phù hợp với dữ liệu nhỏ và các bài toán đơn giản. Tuy nhiên, học sâu đã mang lại bước đột phá nhờ khả năng tự động học đặc trưng từ dữ liệu. Các mạng nơ-ron tích chập (CNN) như AlexNet, VGG, ResNet và EfficientNet được sử dụng rộng rãi, trong khi các mô hình phát hiện đối tượng hiện đại như YOLO, SSD, và Faster R-CNN cho phép nhận dạng nhanh và chính xác.

Ngoài ra, học chuyển giao (Transfer Learning) giúp tận dụng các mô hình huấn luyện sẵn để áp dụng vào bài toán cụ thể, tiết kiệm thời gian và tài nguyên. Các phương pháp mới như DETR (Transformer-based) và GANs cũng đang dần thay đổi cách tiếp cận nhận dạng đối tượng nhờ khả năng tạo dữ liệu và chú ý vào chi tiết. Cuối cùng, việc kết hợp học sâu và học tăng cường đã mở ra tiềm năng mới cho các ứng dụng trong môi trường động. Việc lựa chọn phương pháp phụ thuộc vào yêu cầu bài toán, độ phức tạp của dữ liệu, và khả năng xử lý của hệ thống.

* 1. **Nhận dạng đối tượng và bài toán đặt ra**

Nhờ khả năng nhận diện và phân loại đối tượng một cách chính xác, công nghệ phát hiện đối tượng được ứng dụng rộng rãi trong nhiều lĩnh vực. Trong an ninh, nó được sử dụng để phát hiện kẻ xâm nhập hoặc nhận diện khuôn mặt. Trong hệ thống xe tự lái, công nghệ này giúp nhận biết người đi bộ, xe cộ và biển báo giao thông để đưa ra quyết định lái xe an toàn. Bên cạnh đó, trong y tế, phát hiện đối tượng được dùng để xác định các vùng bất thường trong ảnh chụp X-quang hay MRI. Các ứng dụng khác bao gồm phân tích hành vi khách hàng trong bán lẻ, theo dõi động vật hoang dã trong nghiên cứu sinh thái, và thậm chí là hỗ trợ sản xuất trong nhà máy.



Ví dụ trên là một bài toán của một mô hình nhận diện đối tượng cơ bản. Trong hình, mô hình đã phát hiện và phân loại các đối tượng như một xe Oto (gắn nhãn "truck"), một con chó (gắn nhãn "dog"), và xe đạp (gắn nhãn "bicyde"). Các đối tượng này được bao quanh bởi các khung chữ nhật (bounding boxes), đi kèm nhãn để chỉ rõ loại đối tượng.

* + 1. **Phân loại đối tượng**

Các đối tượng đã được phân loại và gắn nhãn rõ ràng, bao gồm: dog (chó) ở phía trước, bicycle (xe đạp) ở giữa, và truck (xe tải) ở nền xa. Mục tiêu của bài toán phân loại đối tượng là xác định nhãn chính xác cho từng đối tượng trong ảnh.

Hệ thống không chỉ phân loại mà còn xác định vị trí chính xác của các đối tượng thông qua các khung giới hạn (bounding boxes). Trong khi phân loại đối tượng thường áp dụng để nhận diện một lớp đối tượng chính, phát hiện đối tượng lại xử lý nhiều đối tượng và vị trí của chúng trong cùng một hình ảnh.

* + 1. **Xác định vị trí đối tượng**

Xác định vị trí đối tượng (Object Localization) là một bài toán quan trọng trong lĩnh vực Thị giác máy tính (Computer Vision), với mục tiêu chính là xác định vị trí của một hoặc nhiều đối tượng trong hình ảnh bằng cách sử dụng các khung giới hạn (bounding boxes). Trong hình minh họa, các đối tượng đã được xác định vị trí rõ ràng: dog (chó) được khoanh vùng bằng khung màu vàng, bicycle (xe đạp) bằng khung màu đỏ, và truck (xe tải) bằng khung màu xanh lá. Bài toán này không chỉ dừng ở việc xác định tọa độ khung mà còn có thể kết hợp với phân loại để gắn nhãn đúng cho từng đối tượng trong khung.

Xác định vị trí đối tượng là nền tảng của các bài toán phức tạp hơn, như phát hiện đối tượng (Object Detection) hoặc phân đoạn ảnh (Image Segmentation). Ứng dụng của bài toán rất đa dạng, từ hệ thống xe tự lái (xác định vị trí người đi bộ, xe cộ, biển báo giao thông) đến giám sát an ninh (phát hiện người hoặc vật thể đáng ngờ) và y tế (xác định khu vực bất thường trong ảnh chụp X-quang, CT, hoặc MRI). Với vai trò quan trọng trong việc hiểu và phân tích hình ảnh, bài toán này là bước đệm cho nhiều ứng dụng công nghệ cao trong đời sống và công nghiệp.

* + 1. **Tính đa dạng và phức tạp của hình ảnh**

 **Độ sáng và ánh sáng**:

* Hình ảnh có thể có độ sáng khác nhau hoặc có thể được chụp dưới ánh sáng yếu, khiến các đối tượng trở nên khó nhận diện.
* Tình trạng ánh sáng thay đổi có thể làm cho các đối tượng trở nên mờ hoặc bị che khuất.

 **Góc nhìn và vị trí**:

* Các đối tượng có thể được nhìn thấy từ nhiều góc độ khác nhau. Ví dụ, một chiếc xe có thể được nhìn từ phía trước, phía sau, hoặc một góc nghiêng.
* Sự thay đổi trong góc nhìn có thể làm cho một đối tượng trở nên khó nhận diện hoặc nhận diện sai.

 **Chất lượng hình ảnh**:

* Hình ảnh có thể bị nhiễu, mờ hoặc có độ phân giải thấp, điều này có thể ảnh hưởng đến độ chính xác của các mô hình nhận dạng đối tượng.
* Các yếu tố như mờ, sự khúc xạ ánh sáng hoặc nhiễu từ cảm biến có thể tạo ra những khó khăn trong việc xác định chính xác đối tượng.

 **Sự che khuất và chồng lắp**:

* Đôi khi, các đối tượng có thể bị che khuất một phần hoặc bị chồng lên nhau trong hình ảnh, khiến cho việc nhận diện hoặc xác định vị trí trở nên khó khăn.
* Ví dụ, một người đứng sau chiếc xe có thể bị che khuất hoàn toàn hoặc chỉ một phần.

 **Đối tượng tương tự và sự phân biệt**:

* Các đối tượng có thể rất giống nhau về hình dạng, màu sắc hoặc kết cấu, ví dụ như các loài động vật giống nhau hoặc các loại phương tiện giao thông tương tự nhau.
* Việc phân biệt giữa các đối tượng tương tự có thể gây khó khăn cho các mô hình nhận diện, đặc biệt là khi chúng có ít đặc điểm phân biệt rõ rệt.

 **Phông nền và môi trường**:

* Hình ảnh có thể có phông nền rất đa dạng, từ các cảnh vật thiên nhiên đến các bối cảnh đô thị phức tạp. Sự thay đổi này làm cho việc phân tách đối tượng khỏi nền trở nên khó khăn hơn.
* Hình ảnh có thể chứa những yếu tố làm nhiễu (noise), chẳng hạn như các đối tượng không liên quan hoặc có sự thay đổi về môi trường trong hình.
  + 1. **Xử lý chồng lấn:**

Chồng lấn xảy ra khi các đối tượng trong hình ảnh bị che khuất một phần hoặc hoàn toàn bởi các đối tượng khác, làm giảm khả năng nhận diện và xử lý hình ảnh của các mô hình. Xử lý chồng lấn là một yếu tố quan trọng để cải thiện hiệu suất của các hệ thống nhận diện đối tượng trong các tình huống thực tế phức tạp.

Ví dụ trong hình, chiếc xe đạp có thể bị che lấp bới con chó

* + 1. **Tốc độ xử lý và hiệu năng:**

**Tốc độ xử lý:** Một hệ thống nhận diện đối tượng phải có khả năng xử lý từng khung hình trong video hoặc ảnh đầu vào trong thời gian ngắn, thường là dưới 100ms mỗi khung hình để có thể phản ứng trong thời gian thực.

**Yêu cầu về hiệu năng**: Hệ thống cần phải cân bằng giữa độ chính xác (accuracy) và tốc độ xử lý. Mô hình học sâu (deep learning models), chẳng hạn như Convolutional Neural Networks (CNNs), yêu cầu tài nguyên tính toán mạnh mẽ để đạt được kết quả chính xác, tuy nhiên cũng cần phải tối ưu hóa để tránh việc quá tải phần cứng.

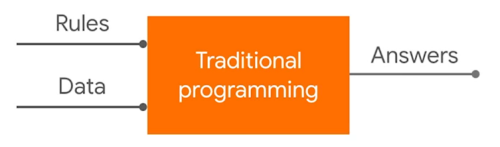
* + 1. **Độ chính xác:**

Hệ thống nhận diện được ba đối tượng chính: **dog**, **bicycle**, và **truck**, với các khung giới hạn sát thực tế. Dựa trên quan sát, độ chính xác tổng thể rất cao, với **Precision** và **Recall** đều gần 100% vì tất cả các đối tượng đều được phát hiện đúng và không có nhãn nào sai. Tuy nhiên, để đánh giá chi tiết hơn, cần so sánh với **ground truth** và tính toán các chỉ số như **IoU** cho từng đối tượng.

* + 1. **Lựa chọn mô hình phù hợp**
* YOLO (You Only Look Once) là mô hình phù hợp nhất cho bài toán này nhờ khả năng phát hiện nhanh và chính xác nhiều đối tượng trong một hình ảnh. Với tốc độ xử lý cao, YOLO có thể hoạt động tốt trong các ứng dụng thời gian thực, chẳng hạn như giám sát giao thông hoặc phân tích video. Phiên bản mới nhất (YOLOv5, YOLOv8) còn cải thiện đáng kể cả về tốc độ lẫn độ chính xác, giúp nhận diện hiệu quả các đối tượng như dog, bicycle, và truck trong ví dụ trên. Đây là lựa chọn tối ưu nếu yêu cầu chính của bài toán là xử lý nhanh và hiệu suất cao.
* Nếu ưu tiên độ chính xác cao hơn tốc độ, Faster R-CNN hoặc EfficientDet sẽ là lựa chọn tốt.
  1. **Tổng quan về học máy**
     1. **Học máy là gì?**

Học máy (machine learning) là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kĩ thuật cho phép các hệ thống “học” tự động từ dữ liệu để giải quyết những vấn đề cụ thể. Một cách tổng quát, trong cuốn sách Machine Learning của tác giả Tom Mitchell xuất bản năm 1997, học máy được định nghĩa như sau: “A computer program is said to learn to perform a task T from experience E, if its performance at task T, as measured by a performance metric P, improves with experience E over time” (một chương trình máy tính được cho là học để thực hiện một nhiệm vụ T từ kinh nghiệm E, nếu hiệu suất thực hiện công việc T của nó được đo bởi chỉ số hiệu suất P và được cải thiện bởi kinh nghiệm E theo thời gian).

Như vậy, học máy nói đến một chương trình giúp cho máy tính có thể giải một bài toán cụ thể bằng những “kinh nghiệm” mà nó đã được “học” từ dữ liệu. Chương trình này có sự khác biệt lớn so với các chương trình lập trình truyền thống như. Sự khác biệt này có thể được mô tả trực quan như trong hình 1. Ở các chương trình lập trình truyền thống, con người sẽ phải viết ra các qui tắc, điều kiện để máy tính thực hiện nhiệm vụ và đưa ra câu trả lời. Ngược lại, đối với Học máy, nhiệm vụ của máy tính là phải dựa vào dữ liệu quan sát để tìm ra được các qui tắc này.

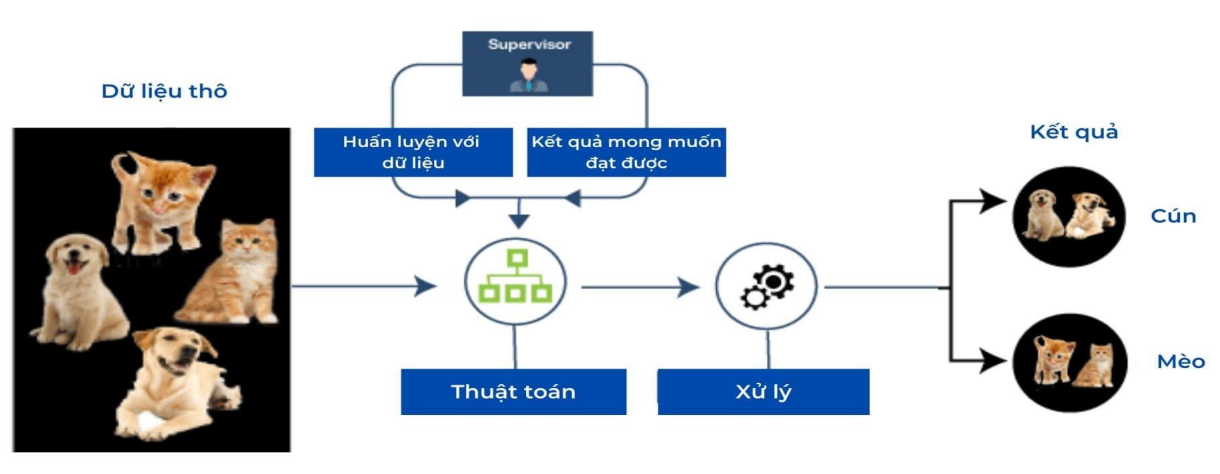




Tương tự như con người, máy tính cũng tự cải thiện khả năng giải quyết vấn đề thông qua việc học. Điểm khác biệt là ở chỗ, con người học được từ những quan sát thực tế, những điều mà con người được nhìn, được nghe, còn máy tính học từ dữ liệu.

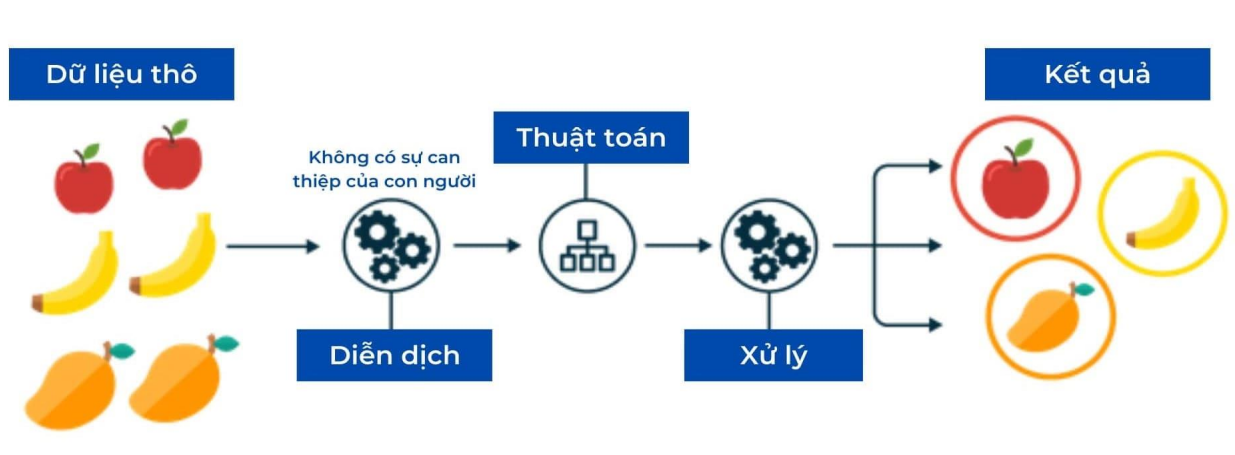
* + 1. **Các phương pháp học máy**

Học có giám sát (Supervised Learning) là phương pháp huấn luyện mô hình bằng tập dữ liệu đã được gán nhãn, trong đó đầu vào và đầu ra đều rõ ràng. Mục tiêu là xây dựng mối quan hệ giữa dữ liệu đầu vào và kết quả để dự đoán cho các trường hợp mới. Phương pháp này thường được áp dụng trong các bài toán phân loại và hồi quy với các thuật toán phổ biến như hồi quy tuyến tính, cây quyết định, và mạng nơ-ron nhân tạo.



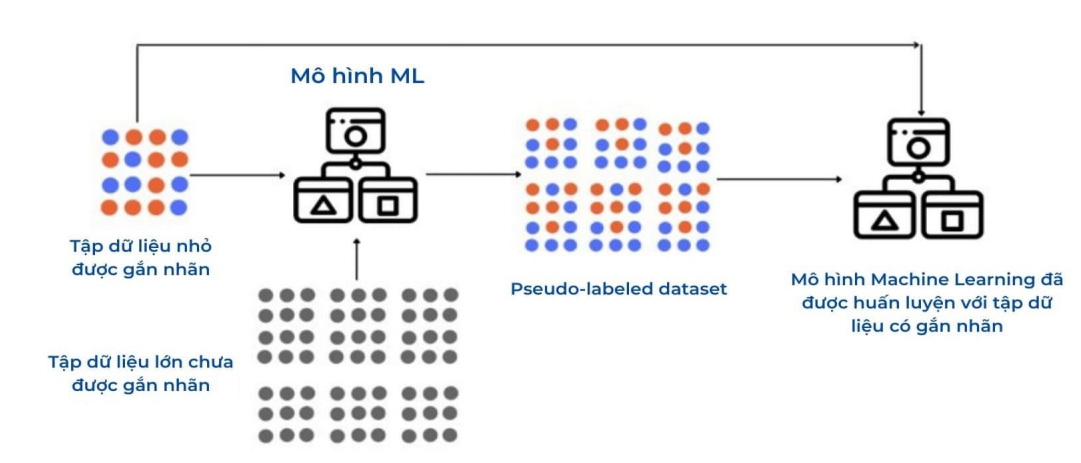
*Hình 1.1: Mô tả học máy có giám sát*

Học không giám sát (Unsupervised Learning) không yêu cầu dữ liệu gán nhãn mà tập trung vào việc tìm ra cấu trúc hoặc mẫu ẩn trong dữ liệu. Nó thường được áp dụng trong các bài toán phân cụm và giảm chiều dữ liệu, sử dụng các thuật toán như K-Means Clustering hoặc phân tích thành phần chính (PCA).



*Hình 1.2: Mô tả học máy không giám sát*

Học bán giám sát (Semi-Supervised Learning) kết hợp dữ liệu có nhãn và không nhãn, phù hợp trong các trường hợp khó thu thập dữ liệu gán nhãn đầy đủ. Trong khi đó, học tăng cường (Reinforcement Learning) cho phép mô hình học cách đưa ra quyết định thông qua tương tác với môi trường, tối đa hóa phần thưởng thông qua các thuật toán như Q-Learning hay Deep Q-Networks.



*Hình 1.3: Mô tả học máy bán giám sát*

Một phương pháp đặc biệt khác là học truyền tải (Transfer Learning), nơi mô hình tận dụng kiến thức từ các bài toán trước để áp dụng cho các bài toán mới, thường được sử dụng trong thị giác máy tính và xử lý ngôn ngữ tự nhiên. Bên cạnh đó, học sâu (Deep Learning), sử dụng các mạng nơ-ron sâu, đã trở thành một phương pháp mạnh mẽ cho các bài toán phức tạp như nhận diện hình ảnh, xử lý ngôn ngữ và phân tích dữ liệu lớn.

* 1. **Ngôn ngữ và công cụ lập trình**
     1. **Công cụ lập trình**

Visual Studio Code (VS Code) là một công cụ chỉnh sửa mã nguồn miễn phí và mã nguồn mở, phát triển bởi Microsoft, được sử dụng rộng rãi trong cộng đồng lập trình. Với hỗ trợ nhiều ngôn ngữ lập trình như Python, JavaScript, C++, Java, và nhiều ngôn ngữ khác, VS Code cung cấp tính năng IntelliSense tự động hoàn thành mã, công cụ gỡ lỗi tích hợp, và khả năng quản lý mã nguồn qua Git. Giao diện người dùng thân thiện và có thể tùy chỉnh, cùng với kho tiện ích mở rộng phong phú, giúp người dùng linh hoạt trong phát triển ứng dụng. VS Code hoạt động mượt mà trên các hệ điều hành Windows, macOS và Linux, mang lại hiệu suất cao và dễ dàng mở rộng, trở thành lựa chọn lý tưởng cho lập trình viên ở mọi cấp độ.



* + 1. **Ngôn ngữ lập trình**

1. **Python**

Python là một ngôn ngữ lập trình bậc cao, thông dịch và dễ học, được phát triển bởi Guido van Rossum vào cuối những năm 1980 và ra mắt lần đầu vào năm 1991. Python nổi bật với cú pháp đơn giản, dễ đọc và dễ hiểu, giúp lập trình viên dễ dàng tiếp cận và viết mã hiệu quả. Khác với nhiều ngôn ngữ khác, Python không yêu cầu dấu chấm phẩy hay dấu ngoặc nhọn, điều này giúp mã nguồn trở nên gọn gàng và dễ bảo trì. Chính sự đơn giản này đã khiến Python trở thành một lựa chọn phổ biến trong cộng đồng lập trình viên trên toàn thế giới.



Python rất linh hoạt và có thể được ứng dụng trong nhiều lĩnh vực khác nhau, bao gồm phát triển phần mềm, ứng dụng web, trí tuệ nhân tạo (AI), học máy (machine learning), phân tích dữ liệu (data science), xử lý ảnh (image processing), phát triển trò chơi (game development), và nhiều ứng dụng khác. Ngôn ngữ này hỗ trợ cả lập trình hướng đối tượng (OOP), lập trình hàm, và lập trình thủ tục, mang lại sự linh hoạt trong cách tổ chức mã nguồn và phát triển ứng dụng.

Một trong những điểm mạnh của Python là hệ sinh thái thư viện phong phú và mạnh mẽ. Các thư viện như NumPy, Pandas, Matplotlib, TensorFlow, PyTorch, và OpenCV giúp lập trình viên giải quyết các vấn đề phức tạp trong các lĩnh vực như phân tích dữ liệu, học máy, xử lý ảnh, và vẽ đồ thị. Ngoài ra, Python cũng hỗ trợ nhiều công cụ quản lý gói như pip và môi trường ảo qua virtualenv, giúp người dùng dễ dàng cài đặt, quản lý và cô lập các thư viện cho từng dự án riêng biệt.

Python có một cộng đồng lập trình viên lớn và rất năng động, cung cấp vô số tài nguyên học tập, diễn đàn hỗ trợ và mã nguồn mở. Điều này giúp lập trình viên dễ dàng tìm kiếm sự trợ giúp và học hỏi từ cộng đồng khi gặp khó khăn trong quá trình phát triển phần mềm. Với tính đơn giản, tính linh hoạt và hệ sinh thái mạnh mẽ, Python đã trở thành một trong những ngôn ngữ lập trình phổ biến nhất, được ưa chuộng trong nhiều ngành công nghiệp và lĩnh vực nghiên cứu.

1. **Thư viện OpenCV**

OpenCV (Open Source Computer Vision Library) là một thư viện mã nguồn mở mạnh mẽ, được phát triển với mục đích hỗ trợ xử lý ảnh, video và các ứng dụng thị giác máy tính. Được ra mắt lần đầu vào năm 1999 bởi Intel, OpenCV nhanh chóng trở thành một công cụ phổ biến trong lĩnh vực này nhờ vào sự linh hoạt và khả năng xử lý các tác vụ phức tạp liên quan đến hình ảnh và video. Thư viện này cung cấp hàng loạt các thuật toán và công cụ cho nhận diện và phân tích đối tượng trong ảnh, phát hiện chuyển động, phân đoạn ảnh, theo dõi đối tượng trong video, và nhiều ứng dụng khác như nhận diện khuôn mặt và phân tích dữ liệu hình ảnh trong y học.

OpenCV hỗ trợ rất nhiều kỹ thuật xử lý ảnh cơ bản và nâng cao, bao gồm làm mịn, làm nét, chuyển đổi màu sắc, phát hiện đường biên, phân tích hình học và các phép toán ảnh phức tạp. Thư viện này cũng tích hợp các công cụ học máy cơ bản, giúp xây dựng và huấn luyện các mô hình nhận diện đối tượng từ dữ liệu hình ảnh. OpenCV có thể chạy trên nhiều nền tảng khác nhau, bao gồm Windows, Linux, macOS, và các hệ điều hành di động như Android và iOS, làm cho nó trở thành một công cụ đa năng và dễ tiếp cận cho nhiều loại ứng dụng.

Một điểm mạnh của OpenCV là khả năng tích hợp với nhiều ngôn ngữ lập trình như C++, Python, Java, và MATLAB, giúp các lập trình viên dễ dàng sử dụng thư viện này trong các dự án của mình mà không phải chuyển đổi mã nguồn phức tạp. Ngoài ra, OpenCV cũng có hiệu suất cao nhờ được tối ưu hóa cho việc xử lý ảnh và video với khả năng chạy đa lõi và hỗ trợ GPU, giúp cải thiện tốc độ xử lý trong các ứng dụng thời gian thực.

Với sự hỗ trợ mạnh mẽ từ cộng đồng toàn cầu, OpenCV không ngừng phát triển và cập nhật, cung cấp cho người dùng các tài liệu học tập, diễn đàn hỗ trợ và mã nguồn mở để giúp giải quyết các vấn đề trong quá trình phát triển. Nhờ vào tính linh hoạt, hiệu suất cao và khả năng mở rộng, OpenCV là một công cụ lý tưởng cho các nhà phát triển và nhà nghiên cứu trong việc xây dựng các ứng dụng nhận diện đối tượng, phân tích video và xử lý ảnh, ứng dụng trong các lĩnh vực như an ninh, robot học, y học và nhiều ngành công nghiệp khác.

# **CHƯƠNG II: XÂY DỰNG HỆ THỐNG**

## **2.1. Yêu cầu bài toán**

* **Mục tiêu của bài toán:**

Hệ thống nhận diện và đếm đối tượng trong ảnh hoặc video được xây dựng với mục tiêu tự động hóa việc phát hiện, định danh và thống kê số lượng các đối tượng trong môi trường thực tế. Cụ thể, hệ thống cần xác định chính xác vị trí của các đối tượng (bounding box), gắn nhãn cho từng loại đối tượng (như người, xe, động vật), đồng thời đếm số lượng của mỗi loại. Bên cạnh đó, hệ thống phải hoạt động hiệu quả trong các điều kiện phức tạp như đối tượng bị che khuất, chồng lấn hoặc hình ảnh có nhiều loại đối tượng.

* **Yêu cầu của hệ thống**

Hệ thống được xây dựng với hai nhóm yêu cầu chính:

* **Yêu cầu chức năng**: Hệ thống phải có khả năng nhận diện chính xác các đối tượng trong ảnh và hiển thị kết quả bằng cách vẽ khung giới hạn (bounding box) và gắn nhãn đối tượng. Ngoài ra, cần tổng hợp và hiển thị số lượng từng loại đối tượng để phục vụ các mục đích thống kê hoặc phân tích.
* **Yêu cầu phi chức năng**: Hệ thống cần đạt độ chính xác cao, với **Precision** và **Recall** trên 90%. Tốc độ xử lý phải đảm bảo đạt ít nhất 20 FPS để hỗ trợ ứng dụng thời gian thực. Bên cạnh đó, hệ thống cần dễ dàng mở rộng để thêm mới các loại đối tượng hoặc tích hợp vào nhiều môi trường khác nhau.
* **Công nghệ đề xuất**

Để đáp ứng yêu cầu bài toán, các công nghệ học sâu và công cụ hỗ trợ sau được đề xuất:

* **Mô hình học sâu**: YOLO (You Only Look Once) là lựa chọn ưu tiên nhờ tốc độ xử lý nhanh và khả năng nhận diện đa đối tượng trong thời gian thực. Các phiên bản như YOLOv5 hoặc YOLOv8 đạt sự cân bằng tốt giữa tốc độ và độ chính xác.
* **Công cụ hỗ trợ**: Các công cụ như **OpenCV** để xử lý hình ảnh/video, **PyTorch/TensorFlow** để huấn luyện mô hình và **LabelImg** để gán nhãn dữ liệu được khuyến nghị nhằm tối ưu hóa quy trình phát triển hệ thống.
* **Quy trình xây dựng hệ thống**

Hệ thống được phát triển qua các bước chính như sau:

* **Thu thập dữ liệu**: Tập hợp tập dữ liệu chứa nhiều loại đối tượng với các điều kiện phức tạp như thay đổi góc nhìn, ánh sáng và chồng lấn đối tượng. Gán nhãn thủ công hoặc tự động để chuẩn bị cho giai đoạn huấn luyện.
* **Tiền xử lý dữ liệu**: Sử dụng các kỹ thuật tăng cường dữ liệu (data augmentation) như xoay, cắt, hoặc phóng to để cải thiện khả năng tổng quát hóa của mô hình.
* **Huấn luyện mô hình**: Huấn luyện mô hình YOLO hoặc các mô hình khác trên tập dữ liệu đã chuẩn bị. Tinh chỉnh siêu tham số như learning rate hoặc batch size để tối ưu hóa hiệu suất.
* **Đánh giá mô hình**: Sử dụng các chỉ số như IoU (Intersection over Union), Precision, Recall và F1-Score để đo lường độ chính xác và hiệu năng của hệ thống.
* **Triển khai hệ thống**: Xây dựng giao diện nhận ảnh hoặc video đầu vào và hiển thị kết quả đầu ra, bao gồm khung giới hạn đối tượng và tổng số lượng từng loại.

## **2.2. Xây dựng hệ thống**

### **2.2.1. Mô hình học sâu YOLO**

YOLO (viết tắt của You Only Look Once) là một kiến trúc mạng học sâu hiện đại, được thiết kế để giải quyết bài toán nhận diện đối tượng (object detection). Mô hình YOLO được biết đến với khả năng phân tích toàn bộ ảnh chỉ trong một lần tính toán duy nhất, từ đó xác định vị trí (bounding boxes) và loại đối tượng (class labels) trong ảnh với tốc độ nhanh và độ chính xác đáng kể.

**Các đặc điểm nổi bật của YOLO:**

* Xử lý toàn cục (Global Context):
* YOLO xem toàn bộ ảnh đầu vào cùng một lúc để nhận diện đối tượng, thay vì chia nhỏ hoặc quét từng phần ảnh như các phương pháp truyền thống (như R-CNN).
* Dự đoán trong thời gian thực:
* Tốc độ xử lý rất nhanh, có thể đạt đến hàng chục khung hình mỗi giây (frames per second - FPS), thích hợp cho các ứng dụng thời gian thực như giám sát giao thông hoặc robot.
* Kết hợp giữa nhận diện và phân loại:
* YOLO không chỉ xác định vị trí của đối tượng (bounding boxes) mà còn phân loại các đối tượng trong ảnh.
* Độ chính xác và khả năng tổng quát hóa:
* Mô hình học cách nhận diện đối tượng trong các bối cảnh đa dạng, ngay cả với những đối tượng có kích thước hoặc hình dạng khác nhau.

### **2.2.2. Mô hình YOLOv5s**

YOLOv5s là một biến thể nhẹ của dòng mô hình YOLO, được thiết kế để thực hiện nhiệm vụ phát hiện đối tượng trong ảnh với độ chính xác cao nhưng vẫn đảm bảo tốc độ nhanh và khả năng xử lý trên các thiết bị phần cứng hạn chế. Mô hình YOLOv5s sử dụng kiến trúc mạng nơ-ron sâu để phát hiện nhiều đối tượng cùng lúc, với khả năng chia ảnh thành các ô nhỏ và dự đoán đồng thời các bounding boxes (vùng bao quanh đối tượng), độ tin cậy của dự đoán, và các nhãn lớp của đối tượng.

* **Khả năng:** Mô hình YOLOv5s có thể phát hiện các đối tượng trong ảnh với độ chính xác cao, nhờ vào việc học các đặc trưng từ dữ liệu huấn luyện.
* **Ưu điểm:** Tiết kiệm tài nguyên tính toán, tốc độ phát hiện nhanh chóng, có thể áp dụng cho nhiều ứng dụng thời gian thực.

### **2.2.3. Ứng dụng vào hệ thống**

Trong hệ thống nhận diện và đếm đối tượng trong ảnh, **YOLOv5s** được sử dụng để nhận diện các đối tượng có trong ảnh và xác định vị trí của chúng thông qua các **bounding boxes** (hình chữ nhật bao quanh đối tượng). Mô hình YOLOv5s có thể phân tích ảnh và nhận diện các đối tượng theo từng loại (chó, người, xe, v.v.), sau đó đếm số lượng các đối tượng này.

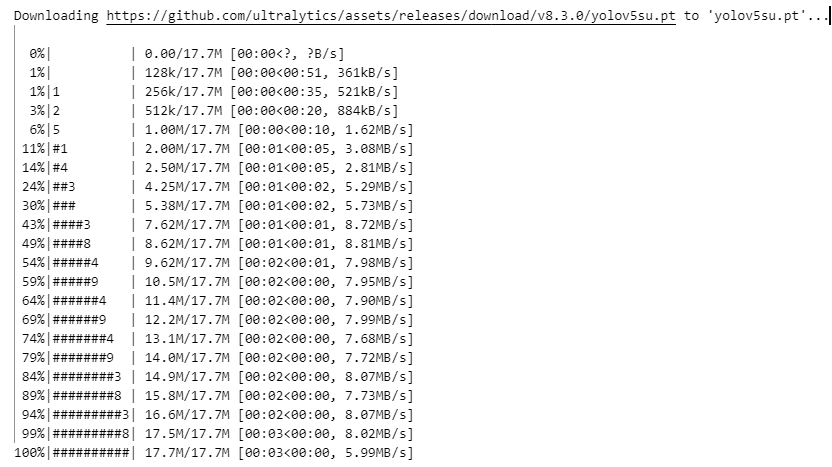
**Cách YOLOv5s hoạt động trong hệ thống:**

* **Tiền xử lý ảnh**: Đầu tiên, ảnh được đưa vào hệ thống, và có thể được tiền xử lý như thay đổi kích thước, chuẩn hóa màu sắc để phù hợp với yêu cầu của mô hình YOLOv5s.
* **Nhận diện đối tượng**: YOLOv5s nhận diện các đối tượng trong ảnh thông qua các **bounding boxes**. Mỗi bounding box đại diện cho một đối tượng được nhận diện và mô hình xác định loại đối tượng này (ví dụ: người, xe, thú cưng, v.v.) và xác định độ tin cậy của dự đoán.
* **Đếm đối tượng**: Sau khi các đối tượng được nhận diện, hệ thống có thể đếm số lượng đối tượng theo từng loại (ví dụ: bao nhiêu chiếc xe, bao nhiêu con chó, v.v.). Quá trình đếm này có thể được thực hiện thông qua việc phân tích các nhãn đối tượng và các giá trị bounding boxes.
* **Hiển thị kết quả**: Kết quả nhận diện và đếm đối tượng có thể được hiển thị lên ảnh dưới dạng bounding boxes và nhãn, đồng thời hệ thống có thể đưa ra kết quả đếm các loại đối tượng.

### **2.2.4. Quy trình hoạt động của hệ thống**

 **Tải Mô hình YOLOv5s**:

* Mô hình được tải từ file đã được huấn luyện trước ("yolov5s.pt").



 **Chọn và Xử lý Ảnh**:

* Hệ thống sử dụng giao diện người dùng Tkinter để cho phép người dùng chọn ảnh từ máy tính. Sau đó, ảnh được đọc bằng OpenCV và chuyển sang định dạng RGB phù hợp với đầu vào của mô hình YOLO.

 **Nhận diện và Phát hiện Đối tượng**:

* Mô hình YOLOv5s được sử dụng để dự đoán các đối tượng trong ảnh. Kết quả trả về bao gồm:
  + **Bounding Boxes**: Vùng bao quanh đối tượng trong ảnh.
  + **Confidence Scores**: Mức độ tin cậy của các đối tượng được phát hiện.
  + **Class Labels**: Các nhãn xác định loại đối tượng (ví dụ: người, ô tô, động vật...).

 **Đếm Các Đối tượng**:

* Sau khi phát hiện các đối tượng, hệ thống sẽ đếm số lượng đối tượng của mỗi loại. Các đối tượng được phân loại dựa trên các nhãn lớp mà mô hình YOLOv5s dự đoán. Các đối tượng sẽ được lưu trữ trong một dictionary để đếm số lượng và hiển thị lên giao diện.

 **Hiển thị và Lưu Kết quả**:

* Ảnh đã xử lý (với các bounding box) được hiển thị trên giao diện. Đồng thời, kết quả đếm các đối tượng sẽ được hiển thị trên giao diện người dùng.
* Người dùng có thể lưu lại kết quả dưới dạng ảnh đã được vẽ bounding box hoặc dưới dạng một file chứa thông tin đối tượng đã phát hiện.

# **CHƯƠNG III: KẾT QUẢ THỰC NGHIỆM**

## **3.1. Dữ liệu**

Mô hình đã được huấn luyện trên các bộ dữ liệu như COCO, với khả năng nhận diện 80 lớp đối tượng khác nhau.

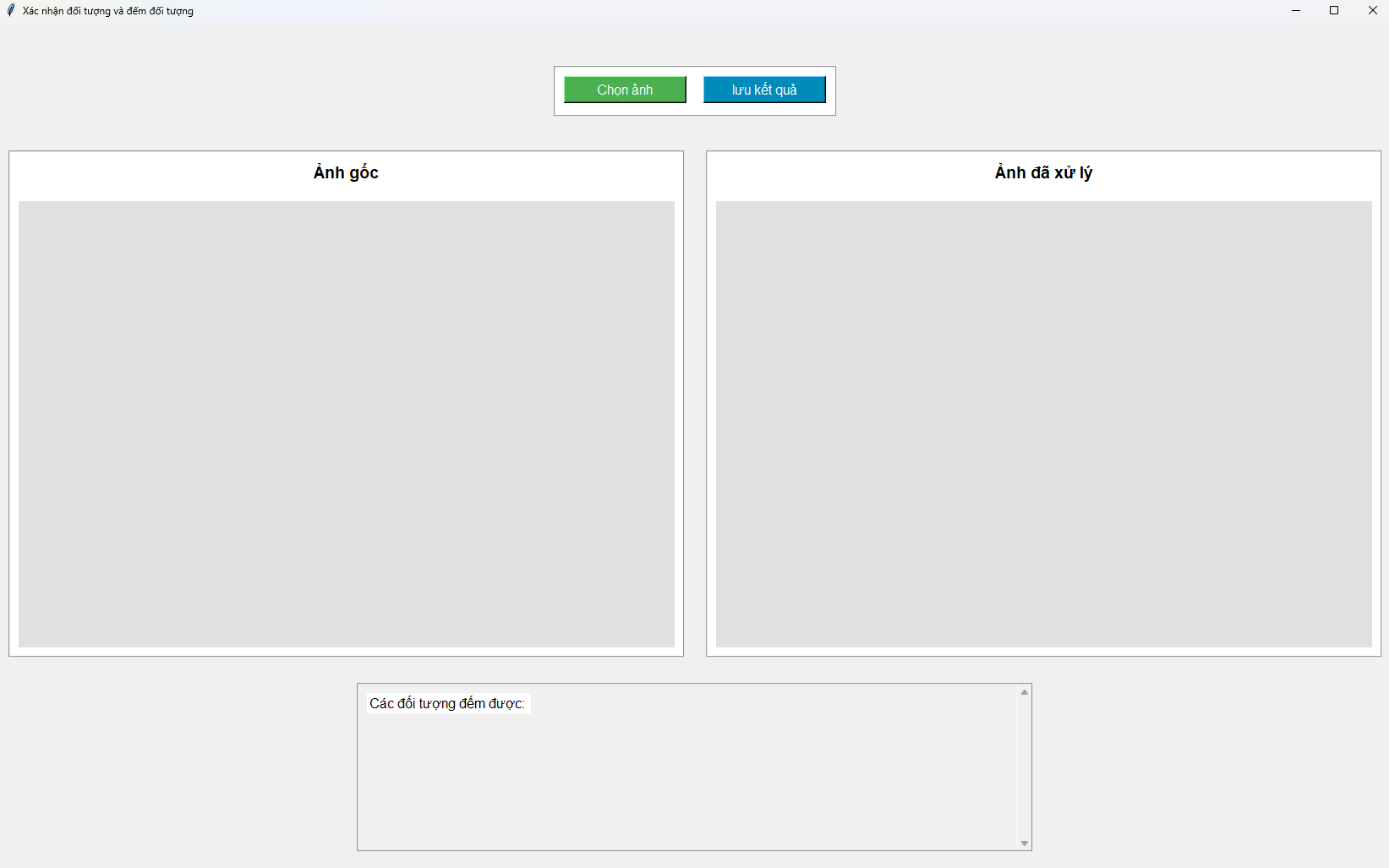
* COCO (Common Objects in Context): Đây là một tập dữ liệu lớn và đa dạng, chứa hơn 330K hình ảnh và 80 lớp đối tượng. COCO bao gồm các loại đối tượng như người, xe cộ, động vật, và nhiều đối tượng khác. Đây là một tập dữ liệu phổ biến cho các bài toán nhận diện đối tượng.

## **3.2. Kết quả nghiệm thực**

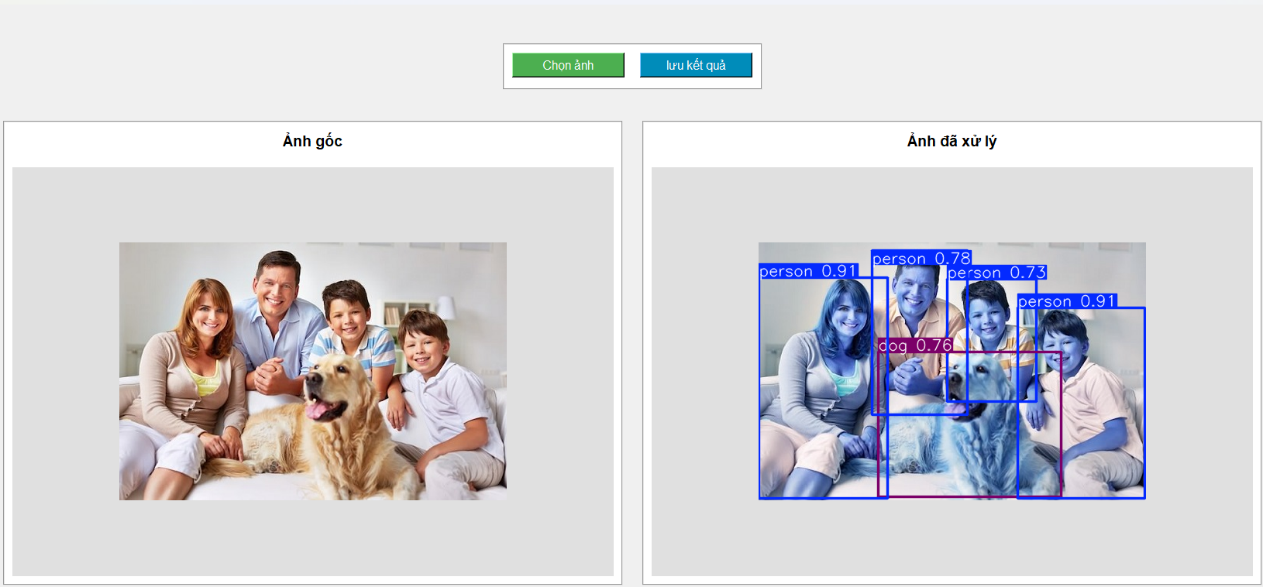
### **3.2.1. Các đoạn code quan trọng**

|  |  |
| --- | --- |
| Khởi tạo mô hình YOLO | import tkinter as tk  from tkinter import filedialog, messagebox  from PIL import Image, ImageTk  import cv2  from ultralytics import YOLO  # Khởi tạo mô hình YOLO  model = YOLO("yolov5s.pt") |
| Lấy ảnh và hiển thị ảnh gốc ra hệ thống | def select\_image():      global panel\_original, panel\_processed, img\_path      # Chọn ảnh từ máy tính      img\_path = filedialog.askopenfilename(filetypes=[("Image files", "\*.jpg \*.jpeg \*.png")])      if not img\_path:          return      # Hiển thị ảnh gốc      image = cv2.imread(img\_path)      image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)      image\_pil = Image.fromarray(image)      img\_display = ImageTk.PhotoImage(image\_pil)      panel\_original.config(image=img\_display)      panel\_original.image = img\_display      process\_image(image) |
| Xuất kết quả ảnh đã xử lý và đếm đối tượng | def process\_image(image):      global panel\_processed, result\_label      # Nhận diện đối tượng      results = model(image)      result\_image = results[0].plot()  # Vẽ bounding box lên ảnh      # Đếm số lượng đối tượng      objects = results[0].names      counts = {}      for obj in results[0].boxes.cls:          obj\_name = objects[int(obj)]          counts[obj\_name] = counts.get(obj\_name, 0) + 1      # Hiển thị kết quả số lượng đối tượng      result\_text = "\n".join([f"{obj}: {count}" for obj, count in counts.items()])      result\_label.config(text=f"Các đối tượng đếm được:\n{result\_text}")      # Hiển thị ảnh đã xử lý      result\_image = cv2.cvtColor(result\_image, cv2.COLOR\_BGR2RGB)      result\_pil = Image.fromarray(result\_image)      img\_display = ImageTk.PhotoImage(result\_pil)      panel\_processed.config(image=img\_display)      panel\_processed.image = img\_display |
| Xuất ảnh đã xử lý về máy | def save\_results():      if img\_path:          # Đọc ảnh gốc          image = cv2.imread(img\_path)            # Nhận diện đối tượng và vẽ bounding boxes lên ảnh          results = model(image)          result\_image = results[0].plot()  # Vẽ bounding box lên ảnh          # Lưu ảnh đã xử lý          save\_path = filedialog.asksaveasfilename(defaultextension=".jpg", filetypes=[("Image files", "\*.jpg \*.jpeg \*.png")])          if save\_path:              cv2.imwrite(save\_path, result\_image)  # Lưu ảnh với bounding boxes              messagebox.showinfo("Save Image", "Image saved successfully!") |
| Xây dựng giao diện cho hệ thống | # Giao diện chính  root = tk.Tk()  root.title("Xác nhận đối tượng và đếm đối tượng")  root.geometry("1650x1000")  # Tăng kích thước cửa sổ  root.configure(bg="#f0f0f0")  # Màu nền giao diện  # Frame cho phần điều khiển  control\_frame = tk.Frame(root, bg="#ffffff", bd=2, relief="groove")  control\_frame.place(relx=0.5, rely=0.05, anchor="n", width=335, height=60)  # Nút chọn ảnh  btn\_select = tk.Button(control\_frame, text="Chọn ảnh", command=select\_image, bg="#4CAF50", fg="white", font=("Arial", 12), width=15)  btn\_select.grid(row=0, column=0, padx=10, pady=10)  # Nút lưu kết quả  btn\_save = tk.Button(control\_frame, text="lưu kết quả", command=save\_results, bg="#008CBA", fg="white", font=("Arial", 12), width=15)  btn\_save.grid(row=0, column=1, padx=10, pady=10)  # Frame cho ảnh gốc  original\_frame = tk.Frame(root, bg="#ffffff", bd=2, relief="groove")  original\_frame.place(relx=0.25, rely=0.15, anchor="n", width=800, height=600)  original\_label = tk.Label(original\_frame, text="Ảnh gốc", bg="#ffffff", font=("Arial", 14, "bold"))  original\_label.pack(pady=10)  panel\_original = tk.Label(original\_frame, bg="#e0e0e0")  panel\_original.pack(fill="both", expand=True, padx=10, pady=10)  # Frame cho ảnh đã xử lý  processed\_frame = tk.Frame(root, bg="#ffffff", bd=2, relief="groove")  processed\_frame.place(relx=0.75, rely=0.15, anchor="n", width=800, height=600)  processed\_label = tk.Label(processed\_frame, text="Ảnh đã xử lý", bg="#ffffff", font=("Arial", 14, "bold"))  processed\_label.pack(pady=10)  panel\_processed = tk.Label(processed\_frame, bg="#e0e0e0")  panel\_processed.pack(fill="both", expand=True, padx=10, pady=10)  # Frame cho kết quả với Canvas và thanh cuộn  result\_frame = tk.Frame(root, bg="#ffffff", bd=2, relief="groove")  result\_frame.place(relx=0.5, rely=0.78, anchor="n", width=800, height=200)  # Tăng chiều cao khung kết quả  # Canvas cho kết quả và thanh cuộn  result\_canvas = tk.Canvas(result\_frame)  scrollbar = tk.Scrollbar(result\_frame, orient="vertical", command=result\_canvas.yview)  result\_canvas.configure(yscrollcommand=scrollbar.set)  # Gắn các widget vào Canvas  result\_canvas.grid(row=0, column=0, sticky="nsew")  scrollbar.grid(row=0, column=1, sticky="ns")  # Thiết lập cấu trúc của grid  result\_frame.grid\_rowconfigure(0, weight=1)  result\_frame.grid\_columnconfigure(0, weight=1)  result\_label = tk.Label(result\_canvas, text="Các đối tượng đếm được: ", bg="#ffffff", font=("Arial", 12), justify="left", anchor="w")  result\_canvas.create\_window(10, 10, anchor="nw", window=result\_label)  root.mainloop() |

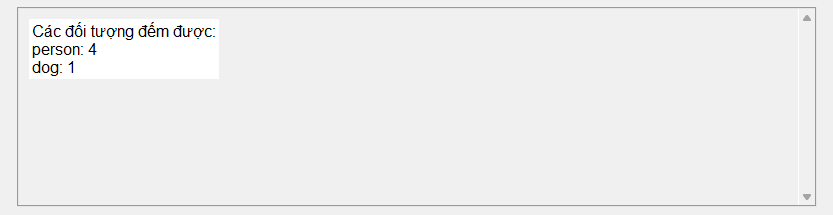
### **3.2.2. Kết quả thực nghiệm**

****

*Hình 3.1: Giao diện ban đầu của hệ thống*

****

*Hình 3.2: Ảnh sau khi được xử lý*

****

*Hình 3.3: Đưa ra kết quả đếm đối tượng rút ra từ ảnh đã xử lý*

# **KẾT LUẬN**

Mô hình YOLOv5s là một công cụ mạnh mẽ và hiệu quả trong việc nhận diện và phân loại đối tượng trong ảnh. Được thiết kế để cung cấp tốc độ xử lý nhanh và độ chính xác cao, YOLOv5s rất phù hợp với các ứng dụng yêu cầu nhận diện đối tượng thời gian thực, như giám sát an ninh, phân tích hình ảnh và kiểm tra chất lượng sản phẩm.

**Ưu điểm:**

* **Tốc độ xử lý nhanh:** YOLOv5s có khả năng nhận diện đối tượng trong thời gian thực, điều này rất quan trọng trong các ứng dụng yêu cầu phản hồi ngay lập tức.
* **Độ chính xác cao:** Mô hình có thể phát hiện nhiều loại đối tượng khác nhau trong ảnh với độ chính xác cao, giúp nâng cao hiệu quả của các hệ thống nhận diện.
* **Giao diện người dùng thân thiện:** Giao diện sử dụng Tkinter dễ hiểu và dễ sử dụng, giúp người dùng dễ dàng tương tác với phần mềm để chọn ảnh, xử lý và lưu kết quả.
* **Ứng dụng đa dạng:** YOLOv5s có thể được áp dụng vào nhiều lĩnh vực, từ giám sát an ninh đến các nghiên cứu y khoa hay phân tích hình ảnh.

**Hạn chế:**

* **Phụ thuộc vào bộ dữ liệu huấn luyện:** Mô hình chỉ hoạt động tốt trên các bộ dữ liệu đã được huấn luyện trước đó, nếu dữ liệu đặc biệt hoặc hiếm, độ chính xác có thể giảm.
* **Giới hạn với ảnh có độ phân giải cao:** Việc xử lý ảnh có độ phân giải quá lớn có thể gây chậm trễ hoặc tiêu tốn nhiều tài nguyên tính toán, đặc biệt trên các thiết bị có cấu hình yếu.
* **Khó phát hiện đối tượng nhỏ:** Mô hình gặp khó khăn trong việc phát hiện các đối tượng nhỏ hoặc các đối tượng bị che khuất.
* **Hạn chế với đối tượng chưa có trong dữ liệu huấn luyện:** Mô hình khó nhận diện các đối tượng chưa được huấn luyện, điều này có thể yêu cầu huấn luyện lại mô hình với bộ dữ liệu mới.
* **Chưa tối ưu cho thiết bị di động:** Mặc dù YOLOv5s là phiên bản nhẹ của YOLO, nhưng vẫn cần tối ưu thêm khi triển khai trên các thiết bị di động hoặc hệ thống có tài nguyên hạn chế.

# **TÀI LIỆU THAM KHẢO**

[1] Slide bài giảng Xử lý ảnh và thị giác máy tính, Trường đại học Công nghệ Đông Á, Lương Hồng Lan.

[2] url.td/MohinhYOLO, Phát Hiện Phương Tiện Giao Thông Trong Không Ảnh Bằng Mô Hình YoloV5 Nhỏ Gọn, 3/12/2024.