TRƯỜNG ĐẠI HỌC PHENIKAA

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LỚN
MÔN HỌC: THỊ GIÁC MÁY TÍNH

Nhóm 10

Đề tài: Nhận diện biển báo giao thông bằng mạng CNN và Keras

Thành viên nhóm

Đỗ Văn Nhiên	21012082	K15.CNTT4
Đinh Thái Phúc	21011622	K15.CNTT3
Lê Trung Kiên	21012505	K15.CNTT5

GVHD: Phạm Tiến Lâm

07/2024 – Hà Nội

Mục Lục

Mục lục biểu đồ và ảnh

Mục lục bảng

Phần 1: Mở Đầu

Nhận diện biển báo giao thông là một trong những ứng dụng quan trọng và thực tế của thị giác máy tính, góp phần nâng cao an toàn, hiệu quả và thông minh hóa hệ thống giao thông. Với sự phát triển mạnh mẽ của các kỹ thuật học sâu (deep learning), các phương pháp nhận diện biển báo dựa trên mạng nơ-ron tích chập (Convolutional Neural Networks - CNN) cùng với thư viện Keras đã đạt được nhiều kết quả ấn tượng, vượt trội so với các phương pháp truyền thống.

Báo cáo này sẽ trình bày một nghiên cứu về việc sử dụng mạng CNN và Keras để thực hiện nhiệm vụ nhận diện biển báo giao thông. Đầu tiên, báo cáo sẽ cung cấp tổng quan về bối cảnh và tầm quan trọng của bài toán nhận diện biển báo giao thông trong bối cảnh phát triển của hệ thống giao thông thông minh. Trong những năm gần đây, các hệ thống giao thông thông minh (Intelligent Transportation Systems - ITS) đã trở thành một lĩnh vực được quan tâm và phát triển mạnh mẽ trên toàn thế giới. Các công nghệ ITS như xe tự hành, giao thông quản lý thông minh và hệ thống hỗ trợ người lái đều có sử dụng các module nhận diện biển báo giao thông. Việc nhận diện chính xác các biển báo giao thông giúp các hệ thống ITS có thể tự động điều chỉnh và ra quyết định một cách phù hợp, góp phần nâng cao an toàn và hiệu quả của hệ thống giao thông.

Tiếp theo, báo cáo sẽ trình bày chi tiết về quá trình xây dựng mô hình CNN, bao gồm cả kiến trúc mạng, chuẩn bị tập dữ liệu và quá trình đào tạo mô hình và tìm hiểu thư viện Keras. Đặc biệt, báo cáo sẽ phân tích và so sánh hiệu suất của phương pháp CNN so với các phương pháp truyền thống khác như phân loại dựa trên đặc trưng thủ công (handcrafted features) hoặc các thuật toán học cổ điển khác. Việc so sánh này sẽ giúp làm nổi bật những ưu điểm của phương pháp CNN trong bài toán nhận diện biển báo giao thông.

Từ những kết quả thực nghiệm và phân tích, báo cáo sẽ làm sáng tỏ tiềm năng, ưu điểm và hạn chế của việc sử dụng mạng CNN và Keras trong bài toán nhận diện biển báo giao thông. Đồng thời, cũng sẽ thảo luận về các hướng phát triển tiềm năng của phương pháp đề xuất, hướng tới việc nâng cao độ chính xác và sự ổn định của hệ thống nhận diện biển báo giao thông dựa trên học sâu. Các hướng phát triển có thể bao gồm việc tích hợp thêm các cảm biến bổ sung, sử dụng các kỹ thuật tăng cường dữ liệu hoặc khai thác thêm thông tin ngữ cảnh.

Nội dung của báo cáo sẽ cung cấp một cái nhìn toàn diện về vấn đề nhận diện biển báo giao thông bằng mạng học sâu CNN và Keras, góp phần làm sáng tỏ tiềm năng của phương pháp này trong việc giải quyết bài toán quan trọng này. Các kết quả của nghiên cứu cũng có thể cung cấp những bài học kinh nghiệm và định hướng cho các nghiên cứu và ứng dụng tương lai trong lĩnh vực thị giác máy tính và hệ thống giao thông thông minh.

1.1. Giới thiệu về môn học

Môn học thị giác máy tính là một lĩnh vực đa dạng và đầy thách thức, liên quan đến nhiều lĩnh vực như xử lý tín hiệu, học máy, hình học, thông tin học và nhiều lĩnh vực khác trong khoa học máy tính. Các nhà nghiên cứu trong lĩnh vực này phải giải quyết nhiều vấn đề phức tạp như:

- Xử lý ảnh số: Các kỹ thuật như khử nhiễu, cải thiện chất lượng ảnh, phát hiện và phân đoạn các đối tượng trong ảnh là những thách thức cơ bản cần được giải quyết.
- Nhận dạng và phân loại đối tượng: Phát hiện và phân biệt các đối tượng trong ảnh là một nhiệm vụ khó khăn, đòi hỏi các mô hình học máy tinh vi để có thể phân biệt các đối tượng với độ chính xác cao.
- Nhận dạng khuôn mặt: Đây là một ứng dụng quan trọng của thị giác máy tính, đòi hỏi phải vượt qua nhiều thách thức như khác biệt về khuôn mặt, góc chụp, ánh sáng, che khuất, v.v.
- Theo dõi chuyển động: Theo dõi sự di chuyển của các đối tượng qua các khung hình video là một bài toán phức tạp, đòi hỏi các kỹ thuật như ước lượng chuyển động, phân đoạn, dự đoán.
- Thị giác máy tính 3D: Xây dựng các mô hình 3D chính xác từ các ảnh 2D là một thách thức lớn, yêu cầu kết hợp nhiều kỹ thuật như định vị, tái tạo cảnh 3D, ước lượng độ sâu.
- Nhận dạng cảnh: Mô tả nội dung và ngữ cảnh của các cảnh trong ảnh là một bài toán phức tạp, cần sự hiểu biết về ngữ cảnh, logic và ngôn ngữ tự nhiên.
- Học sâu cho thị giác máy tính: Các mạng nơ-ron sâu đạt được nhiều thành tựu ấn tượng trong các tác vụ thị giác máy tính, nhưng vẫn còn nhiều hạn chế cần được nghiên cứu giải quyết.

Trong những năm gần đây, thị giác máy tính đã thu hút sự quan tâm rộng rãi và có nhiều ứng dụng quan trọng trong cuộc sống hàng ngày, từ điện thoại thông minh

đến xe tự lái. Đây là một lĩnh vực nghiên cứu sôi nổi, mang lại nhiều cơ hội ứng dụng thực tế, nhưng cũng đặt ra nhiều thách thức khoa học và kỹ thuật cần được tiếp tục giải quyết.

1.2. Mô tả đề tài

Nhận diện biển báo giao thông là một ứng dụng quan trọng của thị giác máy tính, đóng vai trò then chốt trong các hệ thống hỗ trợ lái xe thông minh và xe tự lái. Nhiệm vụ này đòi hỏi khả năng phát hiện, phân loại chính xác các loại biển báo giao thông khác nhau, bao gồm biển báo cảnh báo, biển báo hiệu lệnh, biển báo chỉ dẫn, v.v.

Trong đề tài này, trước tiên, chúng tôi khám phá tập dữ liệu biển báo giao thông mẫu, các hình ảnh tiếp theo được sắp xếp và nhãn của chúng được đặt thành danh sách và các danh sách đó được chuyển đổi thành mảng Numpy để cung cấp mô hình. Thứ hai, mô hình CNN được xây dựng để phân loại ảnh thành các loại tương ứng, đây là cách tiếp cận tốt nhất để phân loại ảnh để thực hiện nhiệm vụ nhận diện biển báo giao thông. Sau khi xây dựng mô hình, mô hình được huấn luyện, xác thực và kiểm tra bằng tập dữ liệu thử nghiệm. Cuối cùng, giao diện đồ họa người dùng được xây dựng để nhận dạng biển báo giao thông bằng Tkinter.

Mục tiêu

Mục tiêu chính của đề tài này là phát triển một hệ thống nhận diện biển báo giao thông sử dụng mạng học sâu CNN và Keras, với các mục tiêu cụ thể như sau:

- Xây dựng và huấn luyện một mạng CNN có khả năng phát hiện và phân loại chính xác các loại biển báo giao thông phổ biến.
- Đánh giá hiệu suất của mô hình CNN trên các tập dữ liệu tiêu chuẩn về nhận diện biển báo giao thông.
- Tối ưu hóa kiến trúc và siêu tham số của mô hình CNN để nâng cao độ chính xác và tốc độ nhận diện.
- Nghiên cứu các kỹ thuật tăng cường dữ liệu và chuyển giao học tập nhằm cải thiện hiệu suất của mô hình trên các điều kiện thực tế (ảnh chụp trong điều kiện thời tiết, ánh sáng khác nhau).
- Triển khai và đánh giá hiệu suất của hệ thống nhận diện biển báo giao thông dựa trên mạng CNN trong các ứng dụng thực tế như hỗ trợ lái xe, xe tự lái.

Ý nghĩa và ứng dụng

Hệ thống nhận diện biển báo giao thông sử dụng mạng học sâu CNN và Keras có nhiều ứng dụng quan trọng, bao gồm:

- Hỗ trợ lái xe an toàn: Giúp cảnh báo người lái về các biển báo quan trọng trên đường, tăng cường an toàn giao thông.
- Xe tự lái: Là thành phần then chốt trong các hệ thống xe tự lái, giúp xe nhận biết và tuân thủ các quy định giao thông.
- Quản lý giao thông thông minh: Hệ thống có thể được tích hợp vào các hệ thống giao thông thông minh để giám sát và phân tích tình hình giao thông.
- Bản đồ số hóa: Dữ liệu về vị trí và loại biển báo có thể được sử dụng để cập nhật và hoàn thiện các bản đồ số.

Như vậy, đề tài này không chỉ có ý nghĩa học thuật mà còn có nhiều ứng dụng thực tế quan trọng, góp phần nâng cao an toàn và hiệu quả của hệ thống giao thông.

1.3. Lí do chọn đề tài

Tính an ninh giao thông:

- Nhận biết biển báo giao thông chính xác và kịp thời giúp cải thiện an toàn và hiệu quả của hệ thống giao thông.
- Đây là một trong những ứng dụng quan trọng của trí tuệ nhân tạo và học sâu trong lĩnh vực giao thông vận tải.

Tiến bộ công nghệ:

- Mạng CNN (Convolutional Neural Network) đã chứng minh hiệu quả trong nhiều bài toán nhận dạng hình ảnh.
- Sử dụng thư viện Keras, thư viện này cung cấp giao diện người dùng Python cấp cao với tính linh hoạt của các phần phụ trợ khác nhau để tính toán, thân thiện với người dùng
- Việc áp dụng CNN và thư viện Keras để nhận biết biển báo giao thông là một ứng dụng thú vị của công nghệ học sâu.

Tính thực tiễn và tiềm năng ứng dụng:

- Có thể triển khai ứng dụng này trên các hệ thống xe tự hành, hỗ trợ lái xe, camera giám sát giao thông, v.v.

- Đây là một lĩnh vực đang được quan tâm và đầu tư phát triển, mang lại nhiều cơ hội ứng dụng trong thực tế.

Tính học thuật và nghiên cứu:

- Đề tài áp dụng các nguyên lý và kiến thức chuyên sâu về CNN, một lĩnh vực nghiên cứu quan trọng trong học máy.
- Tối ưu hóa thuật toán: Nghiên cứu có thể tập trung vào việc cải thiện độ chính xác và hiệu suất của mô hình CNN trong bối cảnh nhận diện biển báo.
- Có thể so sánh hiệu quả của phương pháp đề xuất với các phương pháp truyền thống hoặc các mô hình học sâu khác.

Phần 2. Cơ sở lý thuyết

2.1. Tìm hiểu về ngôn ngữ Python

2.1.1. Khái niệm

Python là ngôn ngữ lập trình máy tính bậc cao thường được sử dụng để xây dựng trang web và phần mềm, tự động hóa các tác vụ và tiến hành phân tích dữ liệu. Python là ngôn ngữ có mục đích chung, nghĩa là nó có thể được sử dụng để tạo nhiều chương trình khác nhau và không chuyên biệt cho bất kỳ vấn đề cụ thể nào.

Tính linh hoạt này, cùng với sự thân thiện với người mới bắt đầu, đã khiến nó trở thành một trong những ngôn ngữ lập trình được sử dụng nhiều nhất hiện nay. Một cuộc khảo sát được thực hiện bởi công ty phân tích ngành RedMonk cho thấy rằng đây là ngôn ngữ lập trình phổ biến thứ hai đối với các nhà phát triển vào năm 2021.

2.1.2. Ứng dụng của Python

Phân tích dữ liệu học máy:

Python đã trở thành một yếu tố chính trong khoa học dữ liệu, cho phép các nhà phân tích dữ liệu và các chuyên gia khác sử dụng ngôn ngữ này để thực hiện các phép tính thống kê phức tạp, tạo trực quan hóa dữ liệu, xây dựng thuật toán học máy, thao tác và phân tích dữ liệu cũng như hoàn thành các nhiệm vụ khác liên quan đến dữ liệu.

Python có thể xây dựng nhiều dạng trực quan hóa dữ liệu khác nhau, chẳng hạn như biểu đồ đường và thanh, biểu đồ hình tròn, biểu đồ 3D. Python cũng có một số thư viện cho phép các lập trình viên viết chương trình để phân tích dữ liệu và học máy nhanh hơn và hiệu quả hơn, như TensorFlow và Keras.

Phát triển Web:

Python thường được sử dụng để phát triển back-end của trang web hoặc ứng dụng những phần mà người dùng không nhìn thấy. Vai trò của Python trong phát triển web có thể bao gồm gửi dữ liệu đến và đi từ máy chủ, xử lý dữ liệu và giao tiếp với cơ sở dữ liệu, định tuyến URL và đảm bảo tính bảo mật. Python cung cấp một số khuôn khổ để phát triển web. Những cái thường được sử dụng bao gồm Django và Flask.

Tự động hóa và phát triển phần mềm:

Trong thế giới mã hóa, tự động hóa có thể được sử dụng để kiểm tra lỗi trên nhiều tệp, chuyển đổi tệp, thực hiện phép toán đơn giản và loại bỏ các bản sao trong dữ liệu.

2.2. Deep Learning

2.2.1. Khái niệm.

Deep Learning là một phương pháp học máy (machine learning) trong lĩnh vực trí tuệ nhân tạo (Artificial Intelligence - AI) mà tập trung vào việc xây dựng và huấn luyện các mạng neural (neural networks) có cấu trúc sâu (deep neural networks) để tự động học và hiểu dữ liệu.

Deep Learning được gọi là "sâu" vì nó bao gồm nhiều tầng (lớp) của các đơn vị tính toán gọi là neurons. Các tầng này kết nối với nhau và truyền dữ liệu qua từ tầng này sang tầng khác, qua đó tạo thành một mạng neural có khả năng học tập phức tạp.

Deep Learning đã gây ra cuộc cách mạng trong nhiều lĩnh vực như thị giác máy tính, xử lý ngôn ngữ tự nhiên và nhận dạng giọng nói. Ví dụ, trong thị giác máy tính, các mạng neural sâu đã đạt được kết quả ấn tượng trong việc phân loại ảnh, nhận dạng khuôn mặt và phát hiện vật thể. Trong xử lý ngôn ngữ tự nhiên, Deep Learning được sử dụng để xây dựng các hệ thống dịch máy, phân loại văn bản và tạo ra các mô hình ngôn ngữ tự động.

2.2.2. Cách thức hoạt động.

Chuẩn bị dữ liệu:

- Thu thập và tiền xử lý dữ liệu
- Gán nhãn cho dữ liệu (đối với học có giám sát)
- Chia dữ liệu thành tập huấn luyện, kiểm tra và xác thực

Thiết kế mạng neural:

- Xác định kiến trúc mạng với nhiều lớp ẩn
- Chọn hàm kích hoạt cho mỗi lớp
- Định nghĩa hàm mất mát (loss function)

Huấn luyện mô hình:

- Lan truyền xuôi (forward propagation) để tính toán đầu ra
- Lan truyền ngược (backpropagation) để tính toán gradient
- Cập nhật trọng số dựa trên gradient

Tối ưu hóa:

- Sử dụng các thuật toán như SGD, Adam, RMSprop
- Điều chỉnh hyperparameters (tốc độ học, kích thước batch, ...)
- Áp dụng các kỹ thuật như regularization, dropout để tránh overfitting

Đánh giá và ứng dụng:

- Kiểm tra hiệu suất trên tập dữ liệu độc lập
- Tinh chỉnh mô hình nếu cần
- Triển khai mô hình cho các ứng dụng thực tế

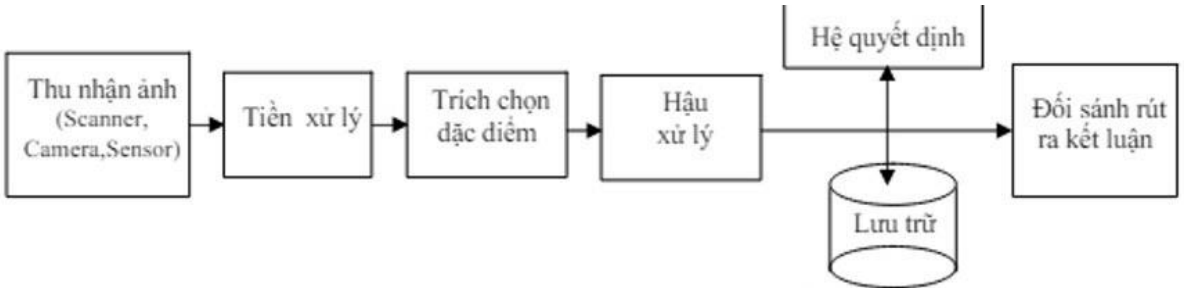
2.3. Tổng quan về xử lý hình ảnh

2.3.1 Xử lý ảnh là gì ?

Con người thu nhận thông tin qua các giác quan, trong đó thị giác đóng vai trò quan trọng nhất. Những năm trở lại đây với sự phát triển của phần cứng máy tính, xử lý ảnh và đồ hoạ đó phát triển một cách mạnh mẽ và có nhiều ứng dụng trong cuộc sống. Xử lý ảnh và đồ hoạ đóng một vai trò quan trọng trong tương tác người máy. Quá trình xử lý ảnh được xem như là quá trình thao tác ảnh đầu vào nhằm cho ra kết quả mong muốn. Kết quả đầu ra của một quá trình xử lý ảnh có thể là một ảnh "tốt hơn" hoặc một kết luận.

Hình 1: Quá trình xử lý ảnh

Ảnh có thể xem là tập hợp các điểm ảnh và mỗi điểm ảnh được xem như là đặc trưng cường độ sáng hay một dấu hiệu nào đó tại một vị trí nào đó của đối tượng trong không gian và nó có thể xem như một hàm n biến P(c1, c2,..., cn). Do đó, ảnh trong xử lý ảnh có thể xem như ảnh n chiều. Sơ đồ tổng quát của một hệ thống xử lý ảnh:

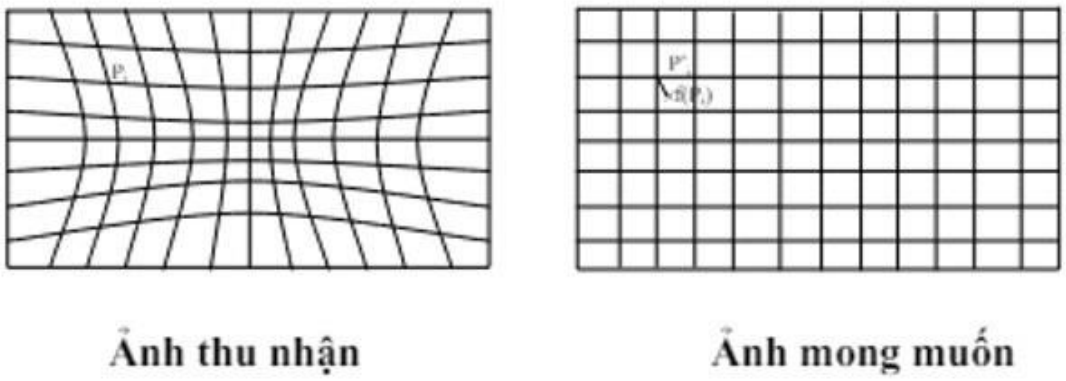


Hình 2. Các bước cơ bản trong một hệ thống xử lý ảnh

2.3.2 Các vấn đề xử lý ảnh

Các khái niệm cơ bản.

- **Ảnh và điểm ảnh**: Điểm ảnh được xem như là dấu hiệu hay cường độ sáng tại một tọa độ trong không gian của đối tượng, và ảnh được xem như là một tập hợp các điểm ảnh. Quá trình xử lý ảnh bao gồm các bước: Thu nhận ảnh (bằng Scanner, Camera, Sensor), Tiền xử lý, Trích chọn đặc điểm, Hệ quyết định, Đối sánh, Rút ra kết luận, và Hậu xử lý. Mục tiêu của xử lý ảnh là biến đổi ảnh đầu vào thành ảnh "tốt hơn" hoặc rút ra kết luận từ ảnh. Quá trình này bao gồm việc lưu trữ, xử lý và phân tích thông tin từ ảnh để đạt được kết quả mong muốn.
- **Mức xám, màu**: Là số các giá trị có thể có của các điểm ảnh của ảnh
- **Nắn chỉnh biến dạng**: Ảnh thu nhận thường bị biến dạng do các thiết bị quang học và điện từ:



Hình 3. Ảnh thu nhận và ảnh mong muốn

- **Khử nhiễu**: Có 2 loại nhiễu cơ bản trong quá trình thu nhận ảnh :

Nhiều hệ thống: là nhiễu có quy luật có thể khử bằng các phép biến đổi

Nhiễu ngẫu nhiên: vết bẩn không rõ nguyên nhân → khắc phục bằng các phép lọc

- **Chỉnh mức xám:**

Nhằm khắc phục tính không đồng đều của hệ thống gây ra. Thông thường có 2 hướng tiếp cận:

• Giảm số mức xám: Thực hiện bằng cách nhóm các mức xám gần nhau thành một bó. Trường hợp chỉ có 2 mức xám thì chính là chuyển về ảnh đen trắng. Ứng dụng: In ảnh màu ra máy in đen trắng.

• Tăng số mức xám: Thực hiện nội suy ra các mức xám trung gian bằng kỹ thuật nội suy. Kỹ thuật này nhằm tăng cường độ mịn cho ảnh

- **Trích chọn đặc điểm**

Các đặc điểm của đối tượng được trích chọn tuỳ theo mục đích nhận dạng trong quá trình xử lý ảnh. Có thể nêu ra một số đặc điểm của ảnh sau đây:

Đặc điểm không gian: Phân bố mức xám, phân bố xác suất, biên độ, điểm uốn v.v..

Đặc điểm biến đổi: Các đặc điểm loại này được trích chọn bằng việc thực hiện lọc vùng (zonal filtering). Các bộ vùng được gọi là "mặt nạ đặc 10 điểm" (feature mask) thường là các khe hẹp với hình dạng khác nhau (chữ nhật, tam giác, cung tròn v.v..)

Đặc điểm biên và đường biên: Đặc trưng cho đường biên của đối tượng và do vậy rất hữu ích trong việc trích trọn các thuộc tính bất biến được dùng khi nhận dạng đối tượng. Các đặc điểm này có thể được trích chọn nhờ toán tử gradient, toán tử la bàn, toán tử Laplace, toán tử "chéo không" (zero crossing) v.v..

Việc trích chọn hiệu quả các đặc điểm giúp cho việc nhận dạng các đối tượng ảnh chính xác, với tốc độ tính toán cao và dung lượng nhớ lưu trữ giảm xuống.

- **Nhận dạng**

Nhận dạng tự động (automatic recognition), mô tả đối tượng, phân loại và phân nhóm các mẫu là những vấn đề quan trọng trong thị giác máy, được ứng dụng trong nhiều ngành khoa học khác nhau. Tuy nhiên, một câu hỏi đặt ra là: mẫu (pattern) là gì? Watanabe, một trong những người đi đầu trong lĩnh vực này đã định nghĩa: "Ngược lại với hỗn loạn (chaos), mẫu là một thực thể (entity), được xác định một cách ang áng (vaguely defined) và có thể gán cho nó một tên gọi nào đó". Ví dụ mẫu có thể là ảnh của vân tay, ảnh của một vật nào đó được chụp, một chữ viết, khuôn mặt người hoặc một ký đồ tín hiệu tiếng nói. Khi biết một mẫu nào đó, để nhận dạng hoặc phân loại mẫu đó có thể:

Hoặc phân loại có mẫu (supervised classification), chẳng hạn phân tích phân biệt (discriminant analyis), trong đó mẫu đầu vào được định danh như một thành phần của một lớp đã xác định.

Hoặc phân loại không có mẫu (unsupervised classification hay clustering) trong đó các mẫu được gán vào các lớp khác nhau dựa trên một tiêu chuẩn đồng dạng nào đó. Các lớp này cho đến thời điểm phân loại vẫn chưa biết hay chưa được định danh. Hệ thống nhận dạng tự động bao gồm ba khâu tương ứng với ba giai đoạn chủ yếu sau đây:

- Thu nhận dữ liệu và tiền xử lý.
- Biểu diễn dữ liệu.
- Nhận dạng, ra quyết định.

Bốn cách tiếp cận khác nhau trong lý thuyết nhận dạng là:

Đối sánh mẫu dựa trên các đặc trưng được trích chọn.

- Phân loại thống kê.
- Đối sánh cấu trúc.
- Phân loại dựa trên mạng nơ-ron nhân tạo.

Trong các ứng dụng rõ ràng là không thể chỉ dùng có một cách tiếp cận đơn lẻ để phân loại "tối ưu" do vậy cần sử dụng cùng một lúc nhiều phương pháp và cách tiếp cận khác nhau. Do vậy, các phương thức phân loại tổ hợp hay được sử dụng khi nhận dạng và nay đã có những kết quả có triển vọng dựa trên thiết kế các hệ thống lai (hybrid system) bao gồm nhiều mô hình kết hợp.

Việc giải quyết bài toán nhận dạng trong những ứng dụng mới, nảy sinh trong cuộc sống không chỉ tạo ra những thách thức về thuật giải, mà còn đặt ra những yêu cầu về tốc độ tính toán. Đặc điểm chung của tất cả những ứng dụng đó là những đặc điểm đặc trưng cần thiết thường là nhiều, không thể do chuyên gia đề xuất, mà phải được trích chọn dựa trên các thủ tục phân tích dữ liệu.

- **Nén ảnh**

Nhằm giảm thiểu không gian lưu trữ. Thường được tiến hành theo cả hai cách khuynh hướng là nén có bảo toàn và không bảo toàn thông tin. Nén không bảo toàn thì thường có khả năng nén cao hơn nhưng khả năng phục hồi thì kém hơn. Trên cơ sở hai khuynh hướng, có 4 cách tiếp cận cơ bản trong nén ảnh:

• Nén ảnh thống kê: Kỹ thuật nén này dựa vào việc thống kê tần xuất xuất hiện của giá trị các điểm ảnh, trên cơ sở đó mà có chiến lược mã hóa thích hợp. Một ví dụ điển hình cho kỹ thuật mã hóa này là *.TIF

• Nén ảnh không gian: Kỹ thuật này dựa vào vị trí không gian của các điểm ảnh để tiến hành mã hóa. Kỹ thuật lợi dụng sự giống nhau của các điểm ảnh trong các vùng gần nhau. Ví dụ cho kỹ thuật này là mã nén *.PCX

• Nén ảnh sử dụng phép biến đổi: Đây là kỹ thuật tiếp cận theo hướng nén không bảo toàn và do vậy, kỹ thuật thướng nến hiệu quả hơn. *.JPG chính là tiếp cận theo kỹ thuật nén này.

• Nén ảnh Fractal: Sử dụng tính chất Fractal của các đối tượng ảnh, thể hiện sự lặp lại của các chi tiết. Kỹ thuật nén sẽ tính toán để chỉ cần lưu trữ phần gốc ảnh và quy luật sinh ra ảnh theo nguyên lý Fractal

2.3.3 Thu nhận và biểu diễn ảnh

Ảnh trên máy tính là kết quả thu nhận theo các phương pháp số hoá được nhúng trong các thiết bị kỹ thuật khác nhau. Quá trình lưu trữ ảnh nhằm 2 mục đích:

• Tiết kiệm bộ nhớ

• Giảm thời gian xử lý Việc lưu trữ thông tin trong bộ nhớ có ảnh hưởng rất lớn đến việc hiển thị, in ấn và xử lý ảnh được xem như là 1 tập hợp các điểm với cùng kích thước nếu sử dụng càng nhiều điểm ảnh thì bức ảnh càng đẹp, càng mịn và càng thể hiện rõ hơn chi tiết của ảnh người ta gọi đặc điểm này là độ phân giải.

Việc lựa chọn độ phân giải thích hợp tuỳ thuộc vào nhu cầu sử dụng và đặc trưng của mỗi ảnh cụ thể, trên cơ sở đó các ảnh thường được biểu diễn theo 2 mô hình cơ bản

- **Mô hình Raster**

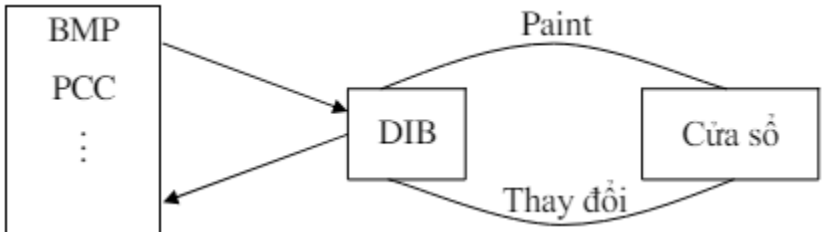
Đây là cách biểu diễn ảnh thông dụng nhất hiện nay, ảnh được biểu diễn dưới dạng ma trận các điểm (điểm ảnh). Thường thu nhận qua các thiết bị như camera, scanner. Tuỳ theo yêu cầu thực thế mà mỗi điểm ảnh được biểu diễn qua 1 hay nhiều bít

Mô hình Raster thuận lợi cho hiển thị và in ấn. Ngày nay công nghệ phần cứng cung cấp những thiết bị thu nhận ảnh Raster phù hợp với tốc độ nhanh và chất lượng cao cho cả đầu vào và đầu ra. Một thuận lợi cho việc hiển thị trong môi trường Windows là Microsoft đưa ra khuôn dạng ảnh DIB (Device Independent Bitmap) làm trung gian.

Một trong những hướng nghiên cứu cơ bản trên mô hình biểu diễn này là kỹ thuật nén ảnh các kỹ thuật nén ảnh lại chia ra theo 2 khuynh hướng là nén bảo toàn

và không bảo toàn thông tin nén bảo toàn có khả năng phục hồi hoàn toàn dữ liệu ban đầu còn nếu không bảo toàn chỉ có khả năng phục hồi độ sai số cho phép nào đó. Theo cách tiếp cận này người ta đã đề ra nhiều quy cách khác nhau như BMP, TIF, GIF, PCX…

Hiện nay trên thế giới có trên 50 khuôn dạng ảnh thông dụng bao gồm cả trong đó các kỹ thuật nén có khả năng phục hồi dữ liệu 100% và nén có khả năng phục hồi với độ sai số nhận được



Hình 4. Mô hình Raster

- **Mô hình vector**

Biểu diễn ảnh ngoài mục đích tiết kiệm không gian lưu trữ dễ dàng cho hiển thị và in ấn còn đảm bảo dễ dàng trong lựa chọn sao chép di chuyển tìm kiếm… Theo những yêu cầu này kỹ thuật biểu diễn vector tỏ ra ưu việt hơn.

Trong mô hình vector người ta sử dụng hướng giữa các vector của điểm ảnh lân cận để mã hoá và tái tạo hình ảnh ban đầu ảnh vector được thu nhận trực tiếp từ các thiết bị số hoá như Digital hoặc được chuyển đổi từ ảnh Raster thông qua các chương trình số hoá

Công nghệ phần cứng cung cấp những thiết bị xử lý với tốc độ nhanh và chất lượng cho cả đầu vào và ra nhưng lại chỉ hỗ trợ cho ảnh Raster.

Do vậy, những nghiên cứu về biểu diễn vector đều tập trung từ chuyển đổi từ ảnh Raster.

Hình 5. Mô hình Vector

2.3.4 Nhận diện đối tượng

Nhận diện đối tượng thường được sử dụng làm gì và ứng dụng như thế nào?

Công nghệ phát hiện đối tượng tồn tại xung quanh chúng ta. Ứng dụng có thể nhìn thấy rõ ràng nhất chính là phần mềm mở khóa bằng nhận diện khuôn mặt trên điện thoại, hoặc các hệ thống camera giám sát an ninh tại các hệ thống cửa hàng, kho bãi.

Một số ứng dụng chính của công nghệ phát hiện đối tượng:

Nhận dạng biển số – sử dụng cả công nghệ phát hiện đối tượng và nhận dạng ký tự quang học (OCR) để nhận dạng các ký tự chữ và số trên biển số xe. Tính năng phát hiện đối tượng được sử dụng thông qua việc lưu giữ lại hình ảnh và phát hiện các đối tượng cụ thể như xe cộ, phương tiện đi lại trên bức ảnh đó. Sau khi đã xác định được tọa độ của phương tiện trên ảnh, chúng ta có thể sử dụng các mô hình Object Detection khác để phát hiện biển số xe và các ký tự trên biển số. Từ đó, áp dụng các mô hình OCR hoặc image classification để nhận dạng ký tự trên hình, ánh xạ từ dạng dữ liệu hình ảnh sang dạng chữ (text).

Phát hiện và nhận dạng khuôn mặt – một trong những ứng dụng chính của phát hiện đối tượng là nhận diện và phát hiện khuôn mặt. Với sự trợ giúp của các thuật toán hiện đại, chúng ta có thể phát hiện khuôn mặt người trong một hình ảnh hoặc video.

Theo dõi đối tượng – Công nghệ này có thể ứng dụng để theo dõi chuyển động của một đối tượng hay đồ vật cụ thể. Chẳng hạn, trong khi xem một trận bóng chày hoặc cricket, quả bóng có thể bị đập ra xa. Trong những tình huống này, thuật toán giúp chúng ta phát hiện, nhận dạng và theo dõi (tracking) vị trí và đường bay của quả bóng hiện tại.

Ô tô tự lái – đối với ô tô tự lái, điều quan trọng là phải nghiên cứu các yếu tố môi trường xung quanh ô tô khi lái xe. Một mô hình phát hiện đối tượng được đào tạo trên

nhiều đối tượng để phát hiện các thực thể khác nhau cản trở việc lái xe nhằm đảo bảo an toàn cho người lái.

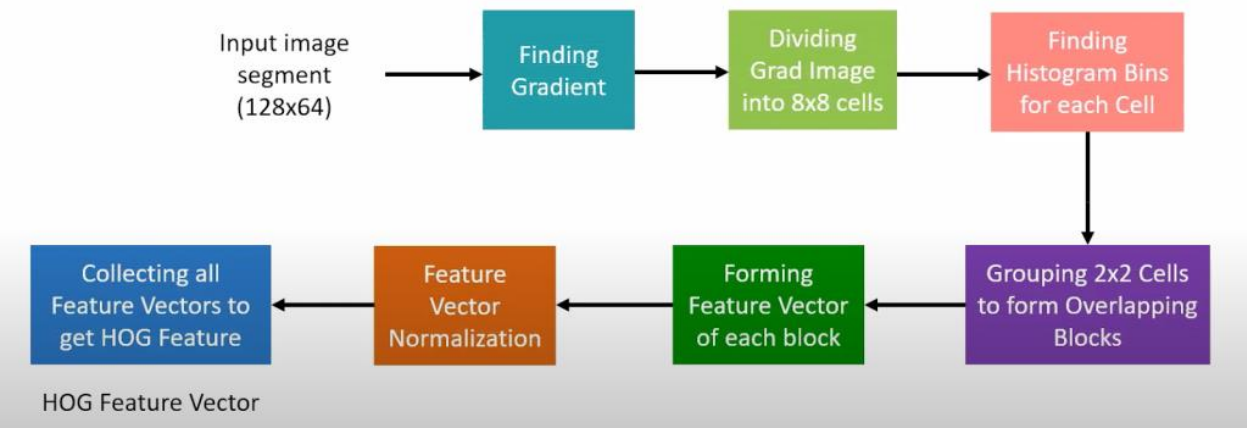
Người máy – Hiện nay thuật toán phát hiện đối tượng thường được ứng dụng trong robot để nhận diện và di chuyển các vật nặng, giảm thiểu những công việc về thể chất cho con người và tự động hóa công việc.

Các thuật toán nhận dạng đối tượng phổ biến:

- **Phương pháp mô tả đặc trưng (Histogram of Oriented Gradients – HOG)**

Phương pháp mô tả đặc trưng (Histogram of Oriented Gradient – HOG) là một trong những phương pháp phát hiện đối tượng lâu đời nhất. Nó được giới thiệu lần đầu tiên vào năm 1986. HOG được sử dụng như 1 thuật toán trích chọn đặc trưng của đối tượng trong ảnh.

Mục đích của HOG là trừu tượng hóa đối tượng bằng cách trích xuất ra những đặc trưng của đối tượng đó và bỏ đi những thông tin không hữu ích. Vì vậy, HOG được sử dụng chủ yếu để mô tả hình dạng và sự xuất hiện của một đối tượng trong ảnh. HOG dựa trên việc chia ảnh đầu vào thành các ảnh con, tính toán histogram của ảnh để tổng hợp và trích rút ra các vector gọi là vector đặc trưng ứng với từng đối tượng.



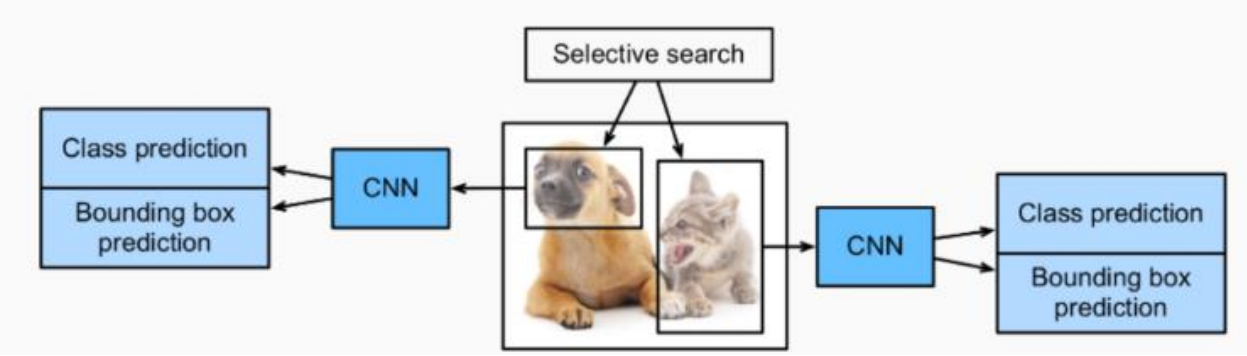
Hình 6. Kiến trúc hệ thống HOG để phát hiện đối tượng

Bản chất của phương pháp HOG là sử dụng thông tin về sự phân bố của các cường độ gradient (intensity gradient) hoặc của hướng biên (edge directions) để mô tả các đối tượng cục bộ trong ảnh. Các toán tử HOG được cài đặt bằng cách chia nhỏ một bức ảnh thành các vùng con, được gọi là cell và với mỗi cell, ta sẽ tính toán một histogram về

các hướng của gradients cho các điểm nằm trong cell. Ghép các histogram lại với nhau ta sẽ có một biểu diễn cho bức ảnh ban đầu. Để tăng cường hiệu năng nhận dạng, các histogram cục bộ có thể được chuẩn hóa về độ tương phản bằng cách tính một ngưỡng cường độ trong một vùng lớn hơn cell, gọi là các khối (blocks) và sử dụng giá trị ngưỡng đó để chuẩn hóa tất cả các cell trong khối. Kết quả sau bước chuẩn hóa sẽ là một vector đặc trưng có tính bất biến đối với các thay đổi về điều kiện ánh sáng.

- **Mạng nơ-ron tích chập theo vùng (R-CNN)**

Mạng nơ-ron tích chập theo vùng (R-CNN) là một cải tiến mới trong kỹ thuật phát hiện đối tượng từ các phương pháp trước đây của HOG và SIFT. Trong các mô hình R-CNN thường trích xuất các đặc trưng cần thiết nhất của đối tượng (thường là khoảng 2000 đặc trưng) bằng cách sử dụng 1 giải thuật chọn lọc (gọi là selective search). Quá trình lựa chọn các đặc trưng quan trọng nhất có thể được tính toán với sự trợ giúp của thuật toán tìm kiếm chọn lọc.



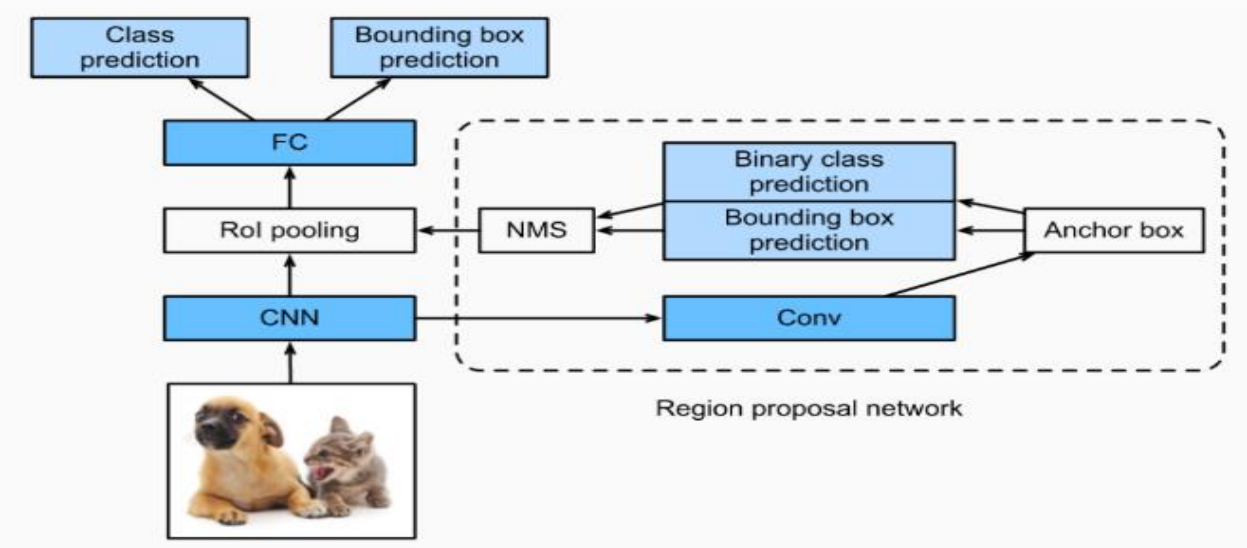
Hình 7. Chu trình phát hiện đối tượng với Mạng nơ-ron tích chập dựa trên khu vực (R-CNN)

Với R-CNN, việc trích xuất các vùng region proposal được thực hiện thông qua thuật toán Selective Search để trích chọn ra các vùng có khả năng chứa đối tượng (khoảng 2000 vùng). Sau đó, các vùng (ảnh) này được resize về 1 kích thước cố định và đưa qua 1 pretrained CNN model (imagenet), rồi từ đó tiến hành xác định offset và nhãn đối tượng. Tuy nhiên, việc đưa các vùng region proposal qua mạng CNN 2000 lần khiến tốc độ thực thi của model cực kì chậm!

- **Faster R-CNN**

Mặc dù mô hình R-CNN có thể thực hiện tính toán phát hiện đối tượng và đạt được hiệu quả nhưng vẫn tồn đọng nhiều điểm bất cập, điển hình là tốc độ mô hình. Vì vậy, một số phương pháp đã được phát triển để giải quyết vấn đề này và xử lý các nhược điểm của R-CNN. Trong số đó nổi bật nhất chính là mô hình Fast R-CNN và Faster R-CNN.

Với Faster-RCNN, thay vì việc sử dụng Selective Search, mô hình được thiết kế thêm 1 mạng con gọi là RPN (Region Proposal Network) để trích rút các vùng có khả năng chứa đối tượng của ảnh. Nhìn chung, sau khi thực hiện RPN, các bước xử lý sau tương tự như Fast-RCNN nhưng nhanh hơn nhiều (vì không sử dụng Selective Search) và được thiết kế như 1 mạng end-to-end trainable network!



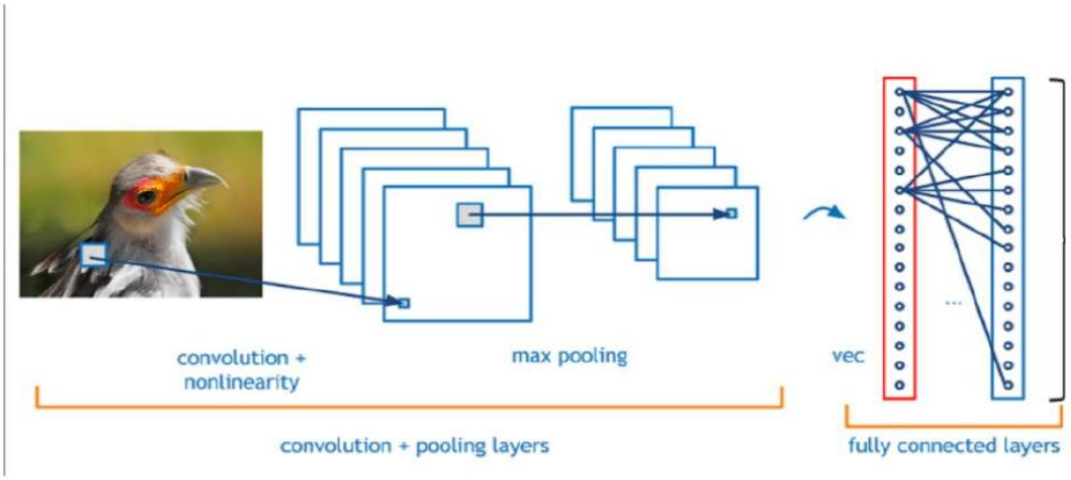
Hình 8. Minh họa mô hình Faster R-CNN

Mạng RPN giảm thời gian tính toán để trích chọn đặc trưng, thường là 10ms cho mỗi hình ảnh. Mạng này bao gồm các lớp tích hợp mà từ đó chúng ta có thể thu được các đặc trưng cần thiết thông qua từng lớp tích chập liên tiếp nhau. Để đưa ra các vùng đặc trưng, chúng ta sử dụng các hộp neo (anchor box) với các tỉ lệ, kích thước và độ lớn khác nhau. Đối với mỗi anchor box tại RPN, chúng ta thực hiện 1 binary classifier để phân loại vùng trích chọn đó có khả năng chứa đối tượng hay không, và dự đoán ra các hộp giới hạn (bounding box) tương ứng.

Sau đó, các vùng trích chọn sẽ được đưa qua 1 bộ lọc gọi là Non maximum suppression (NMS) để loại bỏ các bounding box dư thừa. Đầu ra của NMS được cho qua 1 lớp gọi là RoI Align (Region of Interest) để cố định kích thước đầu ra của các vùng đặc trưng đã trích chọn được. Sau đó, phần xử lý tiếp theo của mô hình sẽ tương tự như mô hình Fast-RCNN

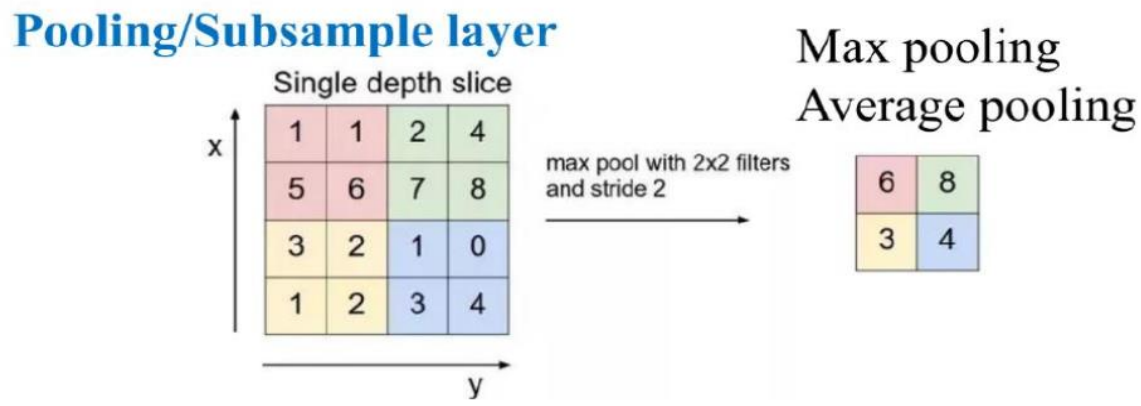
2.4. Convolutional Neural Networks (CNN)

- Chi tiết các thành phần và thuật toán
  CNN**:** Là một loại mạng nơ-ron nhân tạo chuyên dùng cho xử lý ảnh, được thiết kế để tự động và thích ứng trích xuất các đặc trưng quan trọng từ dữ liệu hình ảnh.



Hình 9. Thuật toán CNN

- Hoạt động dựa trên các lớp **Conv2D** (lớp tích chập) và **MaxPooling2D** (lớp tổng hợp).



Hình 10. Hình ảnh thực hiện MaxPooling

- **Flatten** (Lớp tích chập): Đây là lớp cơ bản nhất trong CNN, nó thực hiện phép tích chập (convolution) giữa một filter (bộ lọc) và input data (dữ liệu đầu vào) để tạo ra một feature map (bản đồ đặc trưng). Quá trình này giúp mô hình học được các đặc trưng không gian của hình ảnh.
- **Dense** (Lớp kết nối đầy đủ): Sau khi qua các lớp convolutional và pooling, các đặc trưng đã được trích xuất sẽ được chuyển đổi thành một vector và đưa vào các lớp fully connected để thực hiện phân loại.
- **Softmax** (Hàm kích hoạt): Hàm Softmax là một hàm kích hoạt thường được sử dụng trong các mô hình phân loại, đặc biệt ở lớp đầu ra của các mạng nơ-ron. Nó biến đổi một vector các giá trị thực thành một vector xác suất, trong đó tổng các xác suất bằng 1.
- **Dropout** : Kỹ thuật regularization giúp ngăn chặn overfitting (hiện tượng mô hình học thuộc dữ liệu huấn luyện), bằng cách ngẫu nhiên loại bỏ một số nơ-ron trong quá trình huấn luyện.

2.5. Thư viện Keras

Keras là một open source cho Neural Network được viết bởi ngôn ngữ Python. Nó là một library được phát triển vào năm 2005 bởi Francois Chollet, là một kỹ sư nghiên cứu Deep Learning. Keras có thể sử dụng chung với các thư viện nổi tiếng như Tensorflow, CNTK, Theano. Một số ưu điểm của Keras như:

- Dễ sử dụng, dùng đơn giản hơn Tensor, xây dựng model nhanh.
- Run được trên cả CPU và GPU.
- Hỗ trợ xây dựng CNN , RNN hoặc cả hai. Với những người mới tiếp cận đến Deep như mình thì mình chọn sử dụng Keras để build model vì nó đơn giản,dễ nắm bắt hơn các thư viện khác

TensorFlow là framework duy nhất trong số các framework này chấp nhận Keras làm API cấp cao chính thức của nó. Keras là một framework học sâu được xây dựng trên nền tảng TensorFlow và có các module tích hợp cho tất cả các hoạt động mạng nơ-ron.

Đồng thời, các phép tính tensor, đồ thị tính toán, phiên làm việc và các tính toán tùy chỉnh khác có thể được thực hiện bằng TensorFlow Core API, cho bạn sự linh hoạt và kiểm soát hoàn toàn đối với ứng dụng của mình và cho phép bạn nhanh chóng triển khai ý tưởng. Keras là API cấp cao của TensorFlow, một giao diện thân thiện với người dùng, năng suất cao để giải quyết các vấn đề học máy với trọng tâm là học sâu hiện đại. Nó cung cấp các trừu tượng và khối xây dựng cần thiết để thiết kế và triển khai nhanh chóng các giải pháp học máy.

2.5. Thư viện GUI Tkinter

Tkinter là thư viện GUI tiêu chuẩn cho Python. Tkinter trong Python cung cấp một cách nhanh chóng và dễ dàng để tạo các ứng dụng GUI. Tkinter cung cấp giao diện hướng đối tượng cho bộ công cụ Tk GUI.

Sau đây là các bước để tạo một ứng dụng Tkinter:

1. import Tkinter module.
2. Tạo cửa sổ chính của ứng dụng GUI.
3. Thêm một hoặc nhiều widget nói trên vào ứng dụng GUI.
4. Gọi vòng lặp sự kiện chính để các hành động có thể diễn ra trên màn hình máy tính của người dùng.

2.5.1. Các Widget của Tkinter trong Python

Có nhiều widget khác nhau như button, canvas, checkbutton, entry, … chúng dùng để xây dựng các ứng dụng GUI trong Python:

- Button : Button được sử dụng để thêm nhiều nút khác nhau vào ứng dụng python.
- Canvas: Canvas được sử dụng để vẽ các hình trên cửa sổ.
- Checkbutton: Checkbutton được sử dụng để hiển thị CheckButton trên cửa sổ.
- Entry: Entry được sử dụng để hiển thị trường văn bản một dòng cho người dùng. Nó thường được sử dụng để nhập các giá trị của người dùng.
- Frame : Frame có thể được định nghĩa là một vùng chứa mà có thể chứa một hoặc nhiều widget khác.

- Label: Label là một văn bản được sử dụng để hiển thị một số thông báo hoặc thông tin cho các widget khác.

2.5.2. Bố cục trong Python Tkinter

Python Tkinter cung cấp các phương thức để bố cục các widget sau:

- Phương thức pack(): được sử dụng để tổ chức widget theo khối. Vị trí các widget được thêm vào ứng dụng python bằng phương thức pack() có thể được kiểm soát bằng cách sử dụng các tùy chọn khác nhau được chỉ định trong lệnh gọi phương thức.

- Phương thức grid(): Trình quản lý layout grid() sắp xếp các widget ở dạng bảng. Chúng ta có thể chỉ định các hàng và cột. Chúng ta cũng có thể chỉ định khoảng cột (chiều rộng) hoặc chiều dài hàng (chiều cao) của widget con.

- Phương thức place(): Trình quản lý layout place() sắp xếp các widget theo các tọa độ x và y.

Phần 3. Triển khai đề tài

Giải pháp

Để phát hiện và nhận dạng biển báo giao thông dựa trên quá trình xử lý hình ảnh được đề xuất, được kết hợp với mạng thần kinh tích chập (CNN) và Keras để sắp xếp các biển giao thông. CNN có thể được sử dụng để thực hiện nhiều nhiệm vụ thị giác máy tính do tỷ lệ nhận dạng cao. Cùng với đó là TensorFlow được sử dụng để triển khai CNN và Keras. Bằng cách đó, ta có thể xác định biểu tượng giao thông với tỷ lệ chính xác cao và tốn ít thời gian hơn.

Qua đó cần một số giải pháp như sau:

3.1. Thu thập và chuẩn bị dữ liệu:

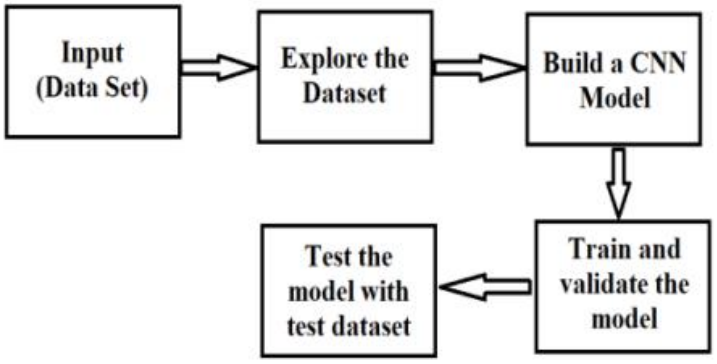
Ta sử dụng mạng thần kinh tích chập (CNN) và thư viện Keras để tạo mô hình phân loại biển báo giao thông trong hình ảnh thành nhiều danh mục trong dự án Deep Learning này. Hơn 50.000 hình ảnh về các biển báo giao thông khác nhau tạo nên tập dữ liệu hình ảnh (giới hạn tốc độ, lối băng qua, tín hiệu giao thông, v.v.). Tập dữ liệu chứa khoảng 43 phân loại phân loại hình ảnh khác nhau. Các lớp tập dữ liệu có kích thước khác nhau, một số có tương đối ít ảnh và một số khác có số lượng lớn. Vì tập dữ liệu chỉ có kích thước 314,36 MB nên mất rất ít thời gian và dung lượng để tải xuống. Nó có hai thư mục: train và test, với thư mục train bao gồm các lớp và mỗi danh mục chứa đồ họa khác nhau.

Mỗi thư mục trong số 43 thư mục trong thư mục dữ liệu của đại diện cho một lớp khác nhau. Kích thước của thư mục nằm trong khoảng từ 0 đến 42. Chúng ta lặp qua tất cả các lớp bằng mô-đun OS, thêm hình ảnh và nhãn của chúng vào danh sách data và labels. Để mở nội dung hình ảnh thành một mảng, thư viện PIL được sử dụng.

Cuối cùng, sắp xếp tất cả hình ảnh và nhãn vào các danh sách (data và labels). Để cung cấp cho mô hình, ta phải chuyển danh sách thành các mảng NumPy. Dữ liệu có hình dạng (39209, 30, 30, 3), cho biết có 39.209 hình ảnh kích thước 30x30 pixel và ba số cuối cùng chỉ ra rằng dữ liệu bao gồm hình ảnh màu (giá trị RGB).

```
(39209, 30, 30, 3) (39209,)
(31367, 30, 30, 3) (7842, 30, 30, 3) (31367,) (7842,)
```

Phương thức train_test_split() trong gói sklearn được sử dụng để chia dữ liệu huấn luyện và kiểm tra. Để chuyển đổi các nhãn trong y_train và y_test thành mã hóa one-hot, chúng ta sử dụng phương thức to_categorical từ gói Keras.utils.



Hình 11: Sơ đồ khối tập dữ liệu

Tệp dữ liệu được sử dụng ở định dạng CSV

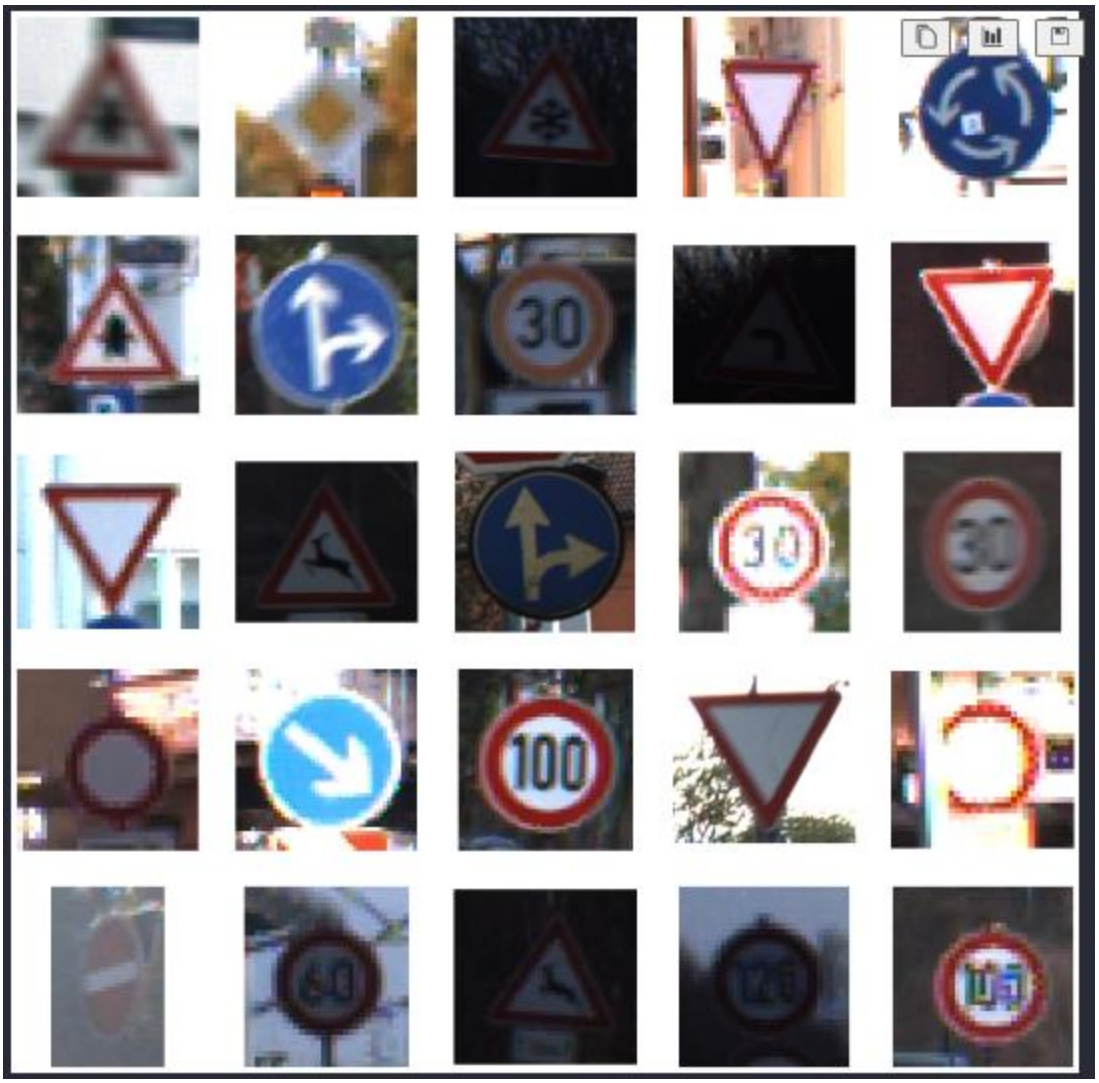
	Width	Height	label	path
0	27	26	20	Train/20/00020_00000_00000.png
1	28	27	20	Train/20/00020_00000_00001.png
2	29	26	20	Train/20/00020_00000_00002.png
3	28	27	20	Train/20/00020_00000_00003.png
4	28	26	20	Train/20/00020_00000_00004.png

Hình 12: Tệp Train.csv

	Width	Height	label	path
0	53	54	16	Test/00000.png
1	42	45	1	Test/00001.png
2	48	52	38	Test/00002.png
3	27	29	33	Test/00003.png
4	60	57	11	Test/00004.png

Hình 13: Tệp Test.csv

Ta có thể trích xuất ngẫu nhiên một tập hợp các hình ảnh từ dữ liệu kiểm tra (test data) về biển báo giao thông.



Hình 14: Hình ảnh ngẫu nhiên test data

Mục đích chính của việc này là để kiểm tra trực quan và xác nhận tính đúng đắn của dữ liệu trước khi tiến hành huấn luyện và đánh giá mô hình CNN (Convolutional Neural Network) được xây dựng bằng Keras.

3.2. Thiết kế và huấn luyện mô hình CNN:

Cấu trúc mô hình học máy:

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 26, 26, 32)	2,432
conv2d_1 (Conv2D)	(None, 22, 22, 32)	25,632
max_pooling2d (MaxPooling2D)	(None, 11, 11, 32)	0
dropout (Dropout)	(None, 11, 11, 32)	0
conv2d_2 (Conv2D)	(None, 9, 9, 64)	18,496
conv2d_3 (Conv2D)	(None, 7, 7, 64)	36,928
max_pooling2d_1 (MaxPooling2D)	(None, 3, 3, 64)	0
dropout_1 (Dropout)	(None, 3, 3, 64)	0
flatten (Flatten)	(None, 576)	0
dense (Dense)	(None, 256)	147,712
dropout_2 (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 43)	11,051

Total params: 242,251 (946.29 KB)

Trainable params: 242,251 (946.29 KB)

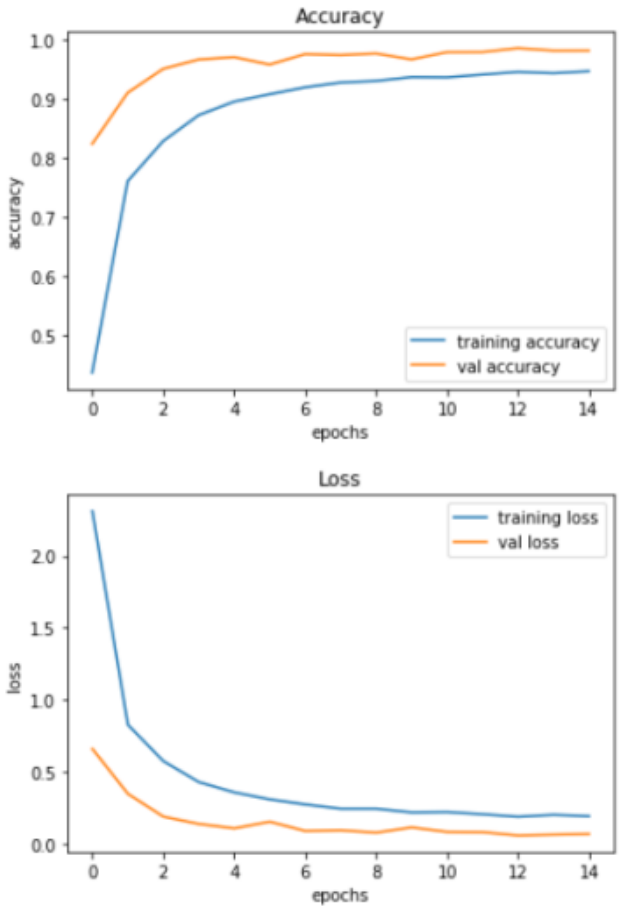
Non-trainable params: 0 (0.00 B)

Hình 15: Cấu trúc mô hình học máy

Với:

- Layer (type): Tên của lớp và loại lớp (Conv2D, MaxPooling2D, Dropout, Flatten, Dense).
- Output Shape: Kích thước đầu ra của lớp sau khi dữ liệu đã qua lớp đó. Điều này giúp bạn hiểu được sự thay đổi kích thước của dữ liệu qua từng lớp.
- Param #: Số lượng tham số cần huấn luyện trong lớp đó. Điều này bao gồm các trọng số và độ chệch (bias).
- Total params: Tổng số tham số trong mô hình.
- Trainable params: Số lượng tham số có thể được cập nhật trong quá trình huấn luyện.
- Non-trainable params: Số lượng tham số không được cập nhật trong quá trình huấn luyện (thường là các tham số được cố định trước).

Sử dụng mô hình CNN để phân loại ảnh thành các nhóm thích hợp (Convolutional Neural Network). Để phân loại hình ảnh thì ta dùng CNN là lựa chọn tốt nhất.



Hình 16: Biểu đồ Accuracy (chính xác) và Loss(tổn thất)

Vì có nhiều lớp cần phân loại, ta biên dịch mô hình với trình tối ưu hóa Adam, một phương pháp hoạt động tốt, và sử dụng hàm mất mát "categorical_crossentropy". Sau khi xây dựng kiến trúc mô hình, chúng ta sử dụng model.fit để huấn luyện mô hình. Ta thử nghiệm với kích thước batch là 32 và 64. Với batch size 64, mô hình của chúng ta hoạt động tốt hơn. Độ chính xác ổn định sau 15 epoch. Trên tập dữ liệu huấn luyện, mô hình của chúng ta đạt tỷ lệ chính xác 95%. Chúng ta trực quan hóa biểu đồ độ accuracy(chính xác) và loss(tổn thất) bằng matplotlib.

```
Epoch 1/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 81s 78ms/step - accuracy: 0.3668 - loss: 3.0234 - val_accuracy: 0.9227 - val_loss: 0.3143
Epoch 2/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 84s 85ms/step - accuracy: 0.8551 - loss: 0.5370 - val_accuracy: 0.9621 - val_loss: 0.1446
Epoch 3/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 86s 88ms/step - accuracy: 0.9043 - loss: 0.3451 - val_accuracy: 0.9709 - val_loss: 0.1147
Epoch 4/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 135s 81ms/step - accuracy: 0.9237 - loss: 0.2810 - val_accuracy: 0.9677 - val_loss: 0.1201
Epoch 5/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 67s 68ms/step - accuracy: 0.9318 - loss: 0.2524 - val_accuracy: 0.9756 - val_loss: 0.0896
Epoch 6/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 65s 66ms/step - accuracy: 0.9342 - loss: 0.2517 - val_accuracy: 0.9842 - val_loss: 0.0612
Epoch 7/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 67s 69ms/step - accuracy: 0.9370 - loss: 0.2446 - val_accuracy: 0.9841 - val_loss: 0.0633
Epoch 8/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 65s 66ms/step - accuracy: 0.9515 - loss: 0.1935 - val_accuracy: 0.9768 - val_loss: 0.0850
Epoch 9/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 67s 68ms/step - accuracy: 0.9490 - loss: 0.1918 - val_accuracy: 0.9881 - val_loss: 0.0482
Epoch 10/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 60s 61ms/step - accuracy: 0.9554 - loss: 0.1755 - val_accuracy: 0.9885 - val_loss: 0.0498
Epoch 11/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 56s 57ms/step - accuracy: 0.9513 - loss: 0.2054 - val_accuracy: 0.9890 - val_loss: 0.0443
Epoch 12/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 57s 58ms/step - accuracy: 0.9544 - loss: 0.1928 - val_accuracy: 0.9890 - val_loss: 0.0433
Epoch 13/15
...
Epoch 14/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 58s 59ms/step - accuracy: 0.9575 - loss: 0.1811 - val_accuracy: 0.9879 - val_loss: 0.0506
Epoch 15/15
981/981 ━━━━━━━━━━━━━━━━━━━━ 58s 59ms/step - accuracy: 0.9554 - loss: 0.1898 - val_accuracy: 0.9814 - val_loss: 0.0658
```

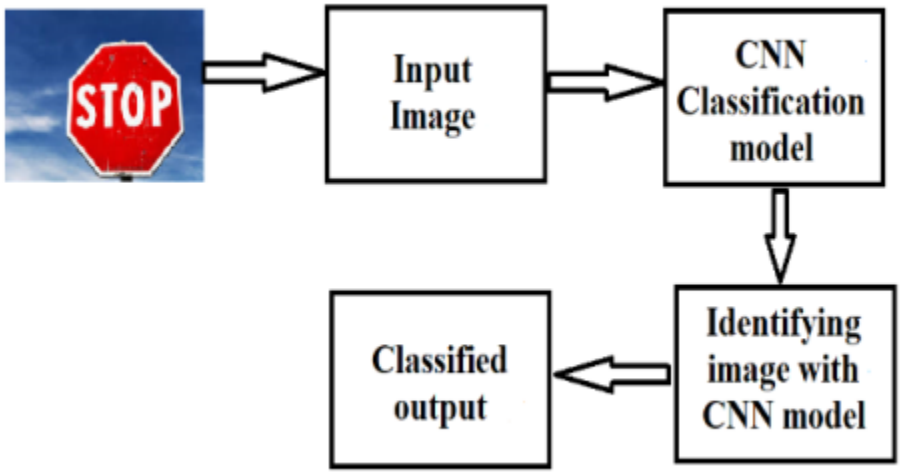
Hình 17: Độ chính xác sau 15 epoch

Các chi tiết liên quan đến đường dẫn hình ảnh và nhãn lớp tương ứng của chúng được chứa trong một tệp test.csv trong bộ dữ liệu. Sử dụng pandas, trích xuất đường dẫn hình ảnh (image path) và nhãn (label). Sau đó, để dự đoán mô hình, ta phải chia tỷ lệ ảnh của mình thành 30x30 pixel và tạo một mảng NumPy chứa tất cả dữ liệu hình ảnh. Sử dụng điểm chính xác (accuracy) từ sklearn.metrics để xem mô hình dự đoán nhãn(label) thực tế như thế nào. Trong mô hình này,ta đã đạt được tỷ lệ chính xác 95%.

Bây giờ chúng ta sẽ xây dựng một giao diện người dùng đồ họa cho bộ phân loại biển báo giao thông của chúng ta bằng Tkinter. Tkinter là một bộ công cụ GUI trong thư viện chuẩn của Python.

Chúng ta bắt đầu bằng cách tải mô hình đã được huấn luyện 'my_model.h5' bằng Keras. Sau đó, chúng ta tạo giao diện người dùng để tải lên hình ảnh, với một nút phân loại khởi chạy hàm mã classify(). Hàm classify() chuyển đổi một hình ảnh thành kích thước (1, 30, 30, 3). Điều này là vì chúng ta phải cung cấp cùng một kích thước mà chúng ta đã sử dụng để phát triển mô hình để dự đoán biển báo giao thông. Sau đó, chúng ta dự đoán lớp, và model.predict_classes(image) trả về cho chúng ta một số từ (0-42) đại diện cho lớp mà nó thuộc về. Chúng ta tra cứu thông tin về lớp trong từ điển.

Đây là mã cho tệp gui.py. Nhập các gói cần thiết, truy xuất hình ảnh và bảng của chúng, chuyển đổi danh sách thành mảng NumPy, chia tập dữ liệu thành tập huấn luyện và kiểm tra, chuyển đổi nhãn, xây dựng mô hình, biên dịch mô hình, vẽ đồ thị cho độ chính xác, kiểm tra độ chính xác trên tập dữ liệu kiểm tra, độ chính xác với dữ liệu kiểm tra.



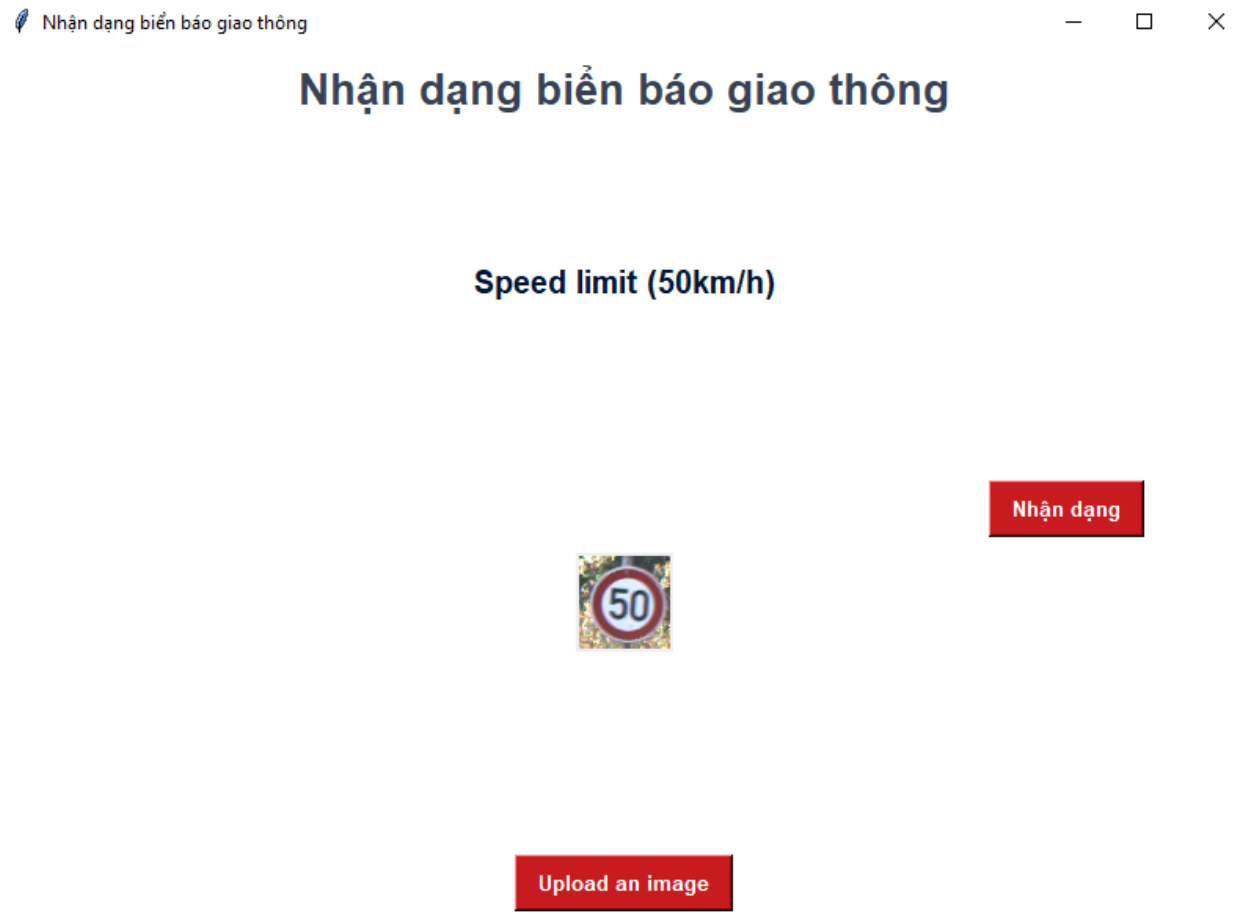
Hình 18: Sơ đồ khối đề xuất

Tất cả các chi tiết cần thiết để Python chạy chương trình đã được import. Các hình ảnh có sẵn trong bộ dữ liệu được truy xuất cùng với bảng dữ liệu của chúng. Danh sách các giá trị số của hình ảnh được chuyển đổi thành mảng NumPy. Bộ dữ liệu được chia thành tập huấn luyện và tập kiểm tra. Các nhãn được chuyển đổi thành dạng mã hóa one-hot. Việc xây dựng mô hình được thực hiện bằng cách sử dụng bộ dữ liệu đã có sẵn và chúng ta đã có được trong quá trình chia tách dữ liệu. Quá trình biên dịch các mô hình được hoàn tất. Để kiểm tra độ chính xác, các đồ thị được vẽ dựa trên kết quả nhận được từ giải pháp. Độ chính xác của kết quả được kiểm tra bằng cách sử dụng bộ dữ liệu đã huấn luyện mà chúng ta đã chia thành tập kiểm tra và tập huấn luyện. Độ chính xác cuối cùng của kết quả được đưa ra với sự hỗ trợ của các metric từ thư viện Sklearn.

Phần 4. Kết quả và thảo luận

Với sự hỗ trợ của các thuật toán đã cho và các thư viện Python khác, chúng tôi đã thành công trong việc nhận diện biển báo giao thông trong các môi trường khác nhau. Thông qua bộ dữ liệu từ Kaggle, ta đã tạo ra các tập huấn luyện dựa trên bộ dữ liệu đó. Bằng cách sử dụng bộ dữ liệu này, một tệp phân loại .h5 đã được tạo ra. Với tệp phân loại đó, tacó thể nhanh chóng thu được kết quả. Hệ thống có khả năng phân loại các hình ảnh đầu vào với tỷ lệ chính xác 95%.

Hình 19: Thông tin biển báo



Hình 20: Biển báo tốc độ

Các hình ảnh được chụp từ các cảm biến trong xe. Hình ảnh đó được đưa vào hệ thống làm dữ liệu đầu vào. Với sự hỗ trợ của mạng nơ-ron tích chập (CNN) và Keras, các hình ảnh được phân loại. Bộ dữ liệu được sử dụng để huấn luyện thuật toán của chúng tôi, trong đó có các tập dữ liệu khác nhau. Có dữ liệu về các điều kiện thời tiết khác nhau và các biển báo bị can thiệp khác nhau. Trong những tình huống khó khăn đó, thuật toán của chúng tôi với sự hỗ trợ của CNN và Keras có thể đạt được tỷ lệ chính xác cao trong việc phân loại các biển báo giao thông.

Phần 5. Tổng kết

5.1. Kế hoạch

5.1.1. Kế hoạch dự kiến

Giai đoạn 1:

Thu thập và chuẩn bị bộ dữ liệu:

- Tập hợp hình ảnh biển báo giao thông từ nhiều nguồn khác nhau, bao gồm cả dữ liệu từ Kaggle.
- Đảm bảo dữ liệu đa dạng về điều kiện ánh sáng, thời tiết, và góc chụp.
- Tiến hành gán nhãn và phân loại dữ liệu một cách chính xác.
- Chia dữ liệu thành các tập huấn luyện, kiểm tra và xác thực.

Thiết kế và xây dựng mô hình CNN:

- Sử dụng thư viện Keras để xây dựng kiến trúc mạng nơ-ron tích chập.
- Thử nghiệm với các kiến trúc CNN khác nhau để tìm ra mô hình tối ưu.

Huấn luyện mô hình:

- Áp dụng kỹ thuật augmentation dữ liệu để tăng cường bộ dữ liệu huấn luyện.
- Tinh chỉnh các siêu tham số như learning rate, batch size, và số epoch.

Đánh giá và tối ưu hóa:

Tích hợp vào hệ thống thực tế:

Giai đoạn 2:

Đỗ Văn Nhiên	Thu thập và chuẩn bị bộ dữ liệu. Thiết kế và xây dựng mô hình CNN Huấn luyện mô hình. Viết báo cáo
Đinh Thái Phúc	Thu thập và chuẩn bị bộ dữ liệu. Thiết kế và xây dựng mô hình CNN Huấn luyện mô hình. Viết báo cáo, làm slide
Lê Trung Kiên	Thiết kế và xây dựng mô hình CNN. Đánh giá và tối ưu hóa. Tích hợp vào hệ thống thực tế. Làm slide

Bảng 1: Phân chia công việc

5.1.2. Phân chia công việc

Vai trò		Lê Trung Kiên	Đinh Thái Phúc	Đô Văn Nhiên
		Thành viên	Trưởng nhóm	Thành viên
STT	Nội dung công việc	Phần % công việc thực hiện		
1	Báo cáo BTL phần 1, 2	50%	50%	50%
2	Báo cáo BTL phần 3, 4, 5	25%	50%	25%
3	Thu thập và chuẩn bị bộ dữ liệu.	50%	50%	50%
4	Thiết kế và xây dựng mô hình CNN	50%	50%	50%
5	Huấn luyện mô hình.	25%	50%	25%
6	Đánh giá và tối ưu hóa.	50%	50%	50%
7	Tích hợp vào hệ thống thực tế.	50%	50%	0%
8	Làm slide	0%	50%	50%
Tổng kết		100%		

Bảng 2: Nội dung công việc

5.2. Kết quả đạt được

Trong bài báo này, một phương pháp nhận dạng biển báo giao thông dựa trên học sâu với sự trợ giúp của mạng nơ ron tích chập (CNN) và Keras được đề xuất, chủ yếu nhắm đến các biển báo giao thông khác nhau. Bằng cách sử dụng tính năng xử lý trước hình ảnh, tập dữ liệu từ Kaggle, phát hiện, nhận dạng và phân loại biển báo giao thông, phương pháp này có thể phát hiện và xác định biển báo giao thông

một cách hiệu quả. Với sự trợ giúp của những kết quả này, ta có thể xác định được biển báo giao thông. Nó giúp người dùng theo hai cách, khi người dùng ở chế độ thủ công, nó sẽ hiển thị kết quả trên màn hình bảng điều khiển và khi người lái xe được đặt ở chế độ tự động, nó sẽ giúp xe lái xe an toàn bằng cách xác định các biển báo giao thông. Kết quả thử nghiệm cho thấy độ chính xác của phương pháp này rất cao.

5.3. Ưu điểm và hạn chế

5.3.1. Ưu điểm

Độ chính xác cao: Mô hình đạt được tỷ lệ chính xác 95% trong việc nhận diện biển báo, giúp tăng độ tin cậy và an toàn cho người lái xe.

Khả năng thích ứng: Có thể nhận diện biển báo trong nhiều điều kiện môi trường khác nhau, bao gồm cả điều kiện ánh sáng kém và thời tiết xấu.

Tốc độ xử lý nhanh: Nhờ sử dụng CNN và tối ưu hóa mô hình, hệ thống có thể xử lý và nhận diện biển báo trong thời gian thực, đáp ứng yêu cầu của ứng dụng trên xe.

Khả năng mở rộng: Mô hình có thể dễ dàng cập nhật và mở rộng để nhận diện các loại biển báo mới hoặc thích ứng với các quy định giao thông mới.

Tính linh hoạt: Có thể tích hợp vào nhiều loại phương tiện khác nhau, từ xe cá nhân đến xe tự lái.

Giảm thiểu sai sót của con người: Hệ thống có thể phát hiện và nhắc nhở người lái về các biển báo quan trọng mà họ có thể bỏ qua.

5.3.2. Hạn chế

Phụ thuộc vào dữ liệu: Hiệu suất của mô hình phụ thuộc nhiều vào chất lượng và đa dạng của bộ dữ liệu huấn luyện. Nếu dữ liệu không đại diện đầy đủ cho các tình huống thực tế, mô hình có thể gặp khó khăn trong một số trường hợp.

Khó khăn với biển báo bị che khuất: Mô hình có thể gặp trở ngại khi nhận diện các biển báo bị che khuất một phần hoặc bị hư hỏng nặng.

Yêu cầu về phần cứng: Để chạy mô hình CNN trong thời gian thực trên xe, cần có phần cứng đủ mạnh, điều này có thể làm tăng chi phí triển khai.

Ảnh hưởng của điều kiện cực đoan: Hiệu suất nhận diện có thể giảm trong các điều kiện thời tiết cực đoan như mưa to, tuyết dày, hoặc sương mù dày đặc.

Cần cập nhật thường xuyên: Để đảm bảo tính chính xác và phù hợp, mô hình cần được cập nhật định kỳ để nhận diện các loại biển báo mới hoặc thích ứng với sự thay đổi trong quy định giao thông.

Vấn đề đạo đức và pháp lý: Việc sử dụng hệ thống tự động trong giao thông có thể gặp phải các vấn đề về trách nhiệm pháp lý trong trường hợp xảy ra tai nạn do nhận diện sai.

Khả năng bị đánh lừa: Mô hình có thể bị đánh lừa bởi các biển báo giả mạo hoặc bị sửa đổi cố ý, gây ra nguy cơ về an toàn.

5.4. Phương hướng phát triển

- Mở rộng bộ dữ liệu:

- Thu thập thêm dữ liệu từ nhiều quốc gia khác nhau để tăng tính đa dạng và khả năng áp dụng toàn cầu.
- Bổ sung dữ liệu về biển báo trong điều kiện thời tiết khắc nghiệt (mưa lớn, tuyết rơi, sương mù) và điều kiện ánh sáng khác nhau (ban đêm, chạng vạng).
- Tạo dữ liệu tổng hợp bằng cách sử dụng các kỹ thuật tạo ảnh 3D để mô phỏng các tình huống hiếm gặp.

- Cải thiện kiến trúc mô hình: Thử nghiệm với các kiến trúc CNN tiên tiến hơn như EfficientNet, Vision Transformer, hoặc MobileNetV3 để cải thiện độ chính xác và hiệu suất.

- Tối ưu hóa cho thiết bị di động và hệ thống nhúng:

- Áp dụng pruning để loại bỏ các tham số không cần thiết, giảm độ phức tạp của mô hình.
- Chuyển đổi mô hình sang các định dạng tối ưu như TensorFlow Lite hoặc ONNX để triển khai trên các thiết bị có tài nguyên hạn chế.

- Tích hợp học liên tục (Continuous Learning):

- Phát triển cơ chế cho phép mô hình học từ dữ liệu mới mà không cần huấn luyện lại toàn bộ.
- Xây dựng hệ thống phản hồi từ người dùng để cải thiện và cập nhật mô hình theo thời gian thực.

- Kết hợp với các kỹ thuật xử lý ảnh nâng cao:

- Tích hợp thuật toán phát hiện đối tượng như YOLO hoặc SSD để xác định vị trí chính xác của biển báo trong khung hình.
- Sử dụng các thuật toán khôi phục ảnh để xử lý biển báo bị mờ hoặc bị che khuất một phần.

- Phát triển khả năng đa nhiệm:

- Mở rộng mô hình để không chỉ nhận diện biển báo mà còn phát hiện vạch kẻ đường, người đi bộ, và các đối tượng giao thông khác.
- Tích hợp khả năng đọc và hiểu nội dung văn bản trên biển báo.

- Tăng cường tính bảo mật và độ tin cậy:

- Phát triển các phương pháp phát hiện và ngăn chặn các cuộc tấn công đối nghịch (adversarial attacks) nhằm vào mô hình.
- Tích hợp cơ chế tự kiểm tra và báo cáo độ tin cậy của các dự đoán.

Phần 6. Tài liệu tham khảo

[1]: Tài liệu bài giảng từ giảng viên

[2] : Convolutional Neural Network (CNN) Tutorial

https://www.kaggle.com/code/kanncaa1/convolutional-neural-network-cnn-tutorial

[3]: Deep Learning, thư viện Keras

https://viblo.asia/p/gioi-thieu-ve-deep-learning-thu-vien-keras-63vKjDGAl2R

[4]: Tkinter – Python Interface

https://docs.python.org/3/library/tkinter.html

[5]: Traffic Sign Recognition Data

https://www.kaggle.com/datasets/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign

[6]: Traffic Sign Classification with Keras and Deep Learning

https://pyimagesearch.com/2019/11/04/traffic-sign-classification-with-keras-and-deep-learning/

[7]: Traffic Signs Recognition using CNN and Keras in Python

https://www.analyticsvidhya.com/blog/2021/12/traffic-signs-recognition-using-cnn-and-keras-in-python/