TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á

KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN

HỌC PHẦN: TRÍ TUỆ NHÂN TẠO

MÃ ĐỀ THI: 50

XÂY DỰNG HỆ THỐNG NHẬN DIỆN VÀ PHÂN LOẠI BIỂN BÁO GIAO THÔNG VIỆT NAM

LỚP TÍN CHỈ: TTNT.03.K13.10.LH.C04.1_LT

Giảng viên hướng dẫn: Th.S Trần Xuân Thanh

Danh sách sinh viên thực hiện: Nhóm 5

TT	Mã sinh viên	Sinh viên thực hiện	Lớp hành chính
1	20223155	Nguyễn Trí Dũng	DCCNTT 13.10.16
2	20222996	Trần Văn Nam	DCCNTT 13.10.16
3	20223110	Nguyễn Văn Duy	DCCNTT 13.10.16

Bắc Ninh – 2025

MỤC LỤC

DANH MỤC TỪ VIẾT TẮT

STT	Từ viết tắt	Tiếng Anh	Tiếng Việt
1	AI	Artificial Intelligence	Trí tuệ nhân tạo
2	ML	Machine Learning	Học máy
3	DL	Deep Learning	Học sâu
4	CNN	Convolutional Neural Network	Mạng nơ-ron tích chập
5	GPU	Graphics Processing Unit	Bộ xử lý đồ họa
6	CPU	Central Processing Unit	Bộ xử lý trung tâm
7	ANN	Artificial Neural Network	Mạng nơ-ron nhân tạo
8	ReLU	Rectified Linear Unit	Hàm kích hoạt ReLU
9	API	Application Programming Interface	Giao diện lập trình ứng dụng
10	ADAS	Advanced Driver Assistance Systems	Hệ thống hỗ trợ lái xe nâng cao
11	TP	True Positive	Dự đoán đúng dương tính
12	FP	False Positive	Dự đoán sai dương tính
13	FN	False Negative	Dự đoán sai âm tính
14	TN	True Negative	Dự đoán đúng âm tính

DANH MỤC TỪ THUẬT NGỮ

STT	Thuật ngữ	Giải thích ngắn gọn
1	Trí tuệ nhân tạo (AI)	Ngành nghiên cứu các phương pháp giúp máy tính thực hiện các nhiệm vụ cần trí tuệ.
2	Học máy (ML)	Nhánh của AI, tập trung vào thuật toán cho phép máy "học" từ dữ liệu.
3	Học sâu (DL)	Nhánh của ML sử dụng mạng nơ-ron nhiều lớp để tự động trích xuất đặc trưng.
4	Mạng nơ-ron tích chập (CNN)	Kiến trúc mạng nơ-ron chuyên cho dữ liệu dạng ảnh/lưới.
5	Phân loại ảnh	Bài toán gán nhãn (class) cho một ảnh đầu vào.
6	Biển báo giao thông	Hệ thống các ký hiệu, hình vẽ đặt trên đường để điều khiển, hướng dẫn giao thông.
7	Dữ liệu huấn luyện	Tập dữ liệu dùng để học tham số cho mô hình.
8	Dữ liệu kiểm thử (test)	Tập dữ liệu dùng để đánh giá khả năng tổng quát hóa của mô hình.
9	Overfitting	Hiện tượng mô hình học quá "thuộc" dữ liệu huấn luyện, kém trên dữ liệu mới.
10	Data augmentation	Kỹ thuật tăng cường dữ liệu bằng biến đổi ảnh (xoay, lật, thay sáng…).
11	Confusion matrix	Ma trận nhầm lẫn, biểu diễn cấu trúc các dự đoán đúng/sai của mô hình.
12	Accuracy	Độ chính xác, tỷ lệ mẫu dự đoán đúng trên tổng số mẫu.

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

LỜI NÓI ĐẦU

Trong bối cảnh cuộc Cách mạng công nghiệp 4.0 đang diễn ra mạnh mẽ trên toàn cầu, Trí tuệ nhân tạo (Artificial Intelligence – AI) đã trở thành một trong những lĩnh vực cốt lõi, đóng vai trò quan trọng trong việc thúc đẩy đổi mới sáng tạo và tự động hoá. Những ứng dụng của AI ngày càng xuất hiện rộng rãi trong nhiều lĩnh vực như y tế, tài chính, nông nghiệp, công nghiệp sản xuất, giao thông vận tải và đời sống hằng ngày. Trong đó, nhận diện hình ảnh – một nhánh quan trọng của Học sâu (Deep Learning) – đã có những bước tiến vượt bậc và trở thành nền tảng cho nhiều hệ thống thông minh hiện đại.

Giao thông vận tải là lĩnh vực đặc biệt cần đến các công nghệ chính xác và ổn định nhằm đảm bảo an toàn cho người tham gia giao thông. Tại Việt Nam, tình trạng vi phạm luật giao thông, thiếu quan sát biển báo hoặc nhận diện sai quy định giao thông dẫn đến tai nạn vẫn xảy ra thường xuyên. Sự xuất hiện của các hệ thống hỗ trợ lái xe thông minh (ADAS), xe tự hành và các nền tảng giám sát giao thông đã tạo ra nhu cầu cấp thiết về một công nghệ nhận diện và phân loại biển báo giao thông đáng tin cậy và phù hợp với điều kiện thực tế của Việt Nam.

Xuất phát từ nhu cầu đó, nhóm chúng em lựa chọn đề tài "Xây dựng hệ thống nhận diện và phân loại biển báo giao thông Việt Nam" nhằm nghiên cứu, ứng dụng và triển khai các mô hình học sâu trong xử lý ảnh. Đề tài tập trung vào việc tìm hiểu các mô hình hiện đại, tối ưu quy trình tiền xử lý dữ liệu, huấn luyện mô hình và đánh giá hiệu quả nhận diện đối với từng loại biển báo. Hệ thống được kỳ vọng có khả năng hỗ trợ các ứng dụng thực tế như cảnh báo biển báo trong thời gian thực, hỗ trợ người lái xe, và làm nền tảng cho các nghiên cứu phát triển xe tự động trong tương lai.

Báo cáo này được xây dựng nhằm trình bày toàn bộ quá trình thực hiện đề tài, bao gồm cơ sở lý thuyết về Trí tuệ nhân tạo và nhận diện ảnh, mô tả bài toán, phân tích dữ liệu, mô hình áp dụng, quy trình huấn luyện – kiểm thử, cũng như đánh giá kết quả và đề xuất hướng phát triển hệ thống. Thông qua quá trình thực hiện, nhóm sinh viên không chỉ tiếp cận sâu hơn với công nghệ AI hiện đại mà còn rèn luyện kỹ năng làm việc nhóm, giải quyết vấn đề, xử lý dữ liệu, lập trình và tối ưu mô hình.

Nhóm xin gửi lời cảm ơn chân thành tới Thạc sĩ Trần Xuân Thanh, giảng viên hướng dẫn, người đã tận tình hỗ trợ và cung cấp những kiến thức quý báu giúp nhóm hoàn thành bài tập lớn này. Đồng thời, nhóm xin cảm ơn Nhà trường đã tạo điều kiện thuận lợi để chúng em được tiếp cận và triển khai những đề tài mang tính thực tiễn cao.

Mặc dù đã nỗ lực hoàn thiện bài báo cáo một cách đầy đủ và khoa học nhất, nhưng do kiến thức và thời gian có hạn, bài báo cáo khó tránh khỏi những thiếu sót. Nhóm rất mong nhận được sự đóng góp ý kiến của thầy cô và bạn bè để có thể hoàn thiện hơn trong các nghiên cứu tiếp theo.

CHƯƠNG I: CƠ SỞ LÝ THUYẾT

1.1. Tổng quan về Trí tuệ nhân tạo

1.1.1. Khái niệm AI

Trí tuệ nhân tạo (Artificial Intelligence – AI) là một lĩnh vực của khoa học máy tính nghiên cứu các mô hình, phương pháp và hệ thống cho phép máy tính thực hiện những hành vi mà trước đây chỉ con người mới có thể làm được. Theo tài liệu giảng dạy của Học viện Công nghệ Bưu chính Viễn thông, AI có rất nhiều định nghĩa khác nhau và có thể được chia thành bốn nhóm tiếp cận:

Hệ thống hành động như con người,

Hệ thống suy nghĩ như con người,

Hệ thống suy nghĩ hợp lý,

Hệ thống hành động hợp lý.

❖ Các định nghĩa trên phản ánh hai hướng tiếp cận chính:

(1) Dựa trên con người (human-centered) – mô phỏng trí tuệ, nhận thức và hành vi của con người;

(2) Dựa trên lý trí (rationality-centered) – hệ thống tối ưu hoá hành động để đạt mục tiêu.

❖ Mục tiêu tổng quát của AI là xây dựng được thực thể thông minh, có khả năng:

Nhận thức môi trường,

Biểu diễn và lưu trữ tri thức,

Suy luận và lập luận,

Học từ dữ liệu và kinh nghiệm,

Ra quyết định hợp lý trong nhiều tình huống khác nhau.

Như vậy, AI không chỉ tập trung vào mô phỏng trí tuệ con người mà còn hướng tới việc xây dựng các hệ thống tự động – thích nghi – tối ưu, phục vụ nhiều ứng dụng thực tiễn trong đời sống.

1.1.2. Lịch sử phát triển

Theo tài liệu "Giới thiệu chung về Trí tuệ nhân tạo", quá trình hình thành và phát triển của AI có thể chia thành nhiều giai đoạn lớn:

(1) Giai đoạn tiền khởi đầu (1943 – 1955)

❖ Giai đoạn này đặt nền móng cho AI với các công trình nổi bật:

Năm 1943: McCulloch & Pitts mô tả mô hình nơ-ron nhân tạo đầu tiên, chứng minh khả năng biểu diễn các hàm toán học phức tạp.

Năm 1950: Alan Turing đề xuất phép thử Turing, đưa ra câu hỏi nền tảng "Máy móc có thể suy nghĩ được không?" — cùng với các ý tưởng về học máy, thuật toán di truyền và học tăng cường.

Năm 1956: Hội nghị Dartmouth chính thức đặt tên cho lĩnh vực Artificial Intelligence, mở ra kỷ nguyên nghiên cứu AI hiện đại.

(2) Giai đoạn khởi đầu (1956 – 1969)

❖ AI đạt nhiều thành tựu quan trọng:

Chương trình Logic Theorist và General Problem Solver (GPS) có khả năng giải toán theo cách suy luận của con người.

Các hệ thống chơi cờ, xử lý ngôn ngữ tự nhiên và giải bài toán đơn giản bằng heuristic được phát triển.

(3) Giai đoạn khó khăn – AI Winter (1974 – 1980)

Do hạn chế về dữ liệu, sức mạnh tính toán, và kỳ vọng quá cao, nhiều dự án thất bại, dẫn đến thời kỳ cắt giảm kinh phí nghiên cứu — gọi là "mùa đông AI".

(4) Giai đoạn phục hồi (1980 – 2000)

Sự xuất hiện của hệ chuyên gia, logic mờ, thuật toán tìm kiếm heuristic nâng cao và các mô hình học máy thống kê khởi động lại đà phát triển của AI.

(5) Bùng nổ dữ liệu & Học sâu (2000 – nay)

Sự phát triển của Internet, dữ liệu lớn (Big Data) và GPU đã tạo điều kiện cho Deep Learning phát triển mạnh mẽ. Các mô hình học sâu như CNN, RNN, LSTM, Transformer… đạt độ chính xác vượt trội trong nhận diện ảnh, âm thanh, NLP.

1.1.3. Ứng dụng AI hiện nay

Theo nội dung chương 1 của giáo trình, AI hiện đã len lỏi vào hầu hết các lĩnh vực trong đời sống. Một số nhóm ứng dụng tiêu biểu gồm:

(1) Thị giác máy tính (Computer Vision)

Nhận diện đối tượng (object detection)

Phân loại hình ảnh (image classification)

Nhận diện khuôn mặt, dấu vân tay

Hệ thống giám sát giao thông

Nhận diện & phân loại biển báo giao thông (ứng dụng chính của đề tài)

(2) Xử lý ngôn ngữ tự nhiên (NLP)

Chatbot, trợ lý ảo (Siri, Google Assistant)

Dịch máy (Google Translate)

Trả lời câu hỏi (IBM Watson)

Tách thông tin, phân loại văn bản, lọc spam

(3) Hệ thống gợi ý và phân tích dữ liệu

Gợi ý sản phẩm (Amazon, Shopee)

Gợi ý video (YouTube, TikTok)

Phân tích hành vi khách hàng

Dự đoán thị trường tài chính

(4) Giao thông thông minh & xe tự hành

Nhận diện biển báo

Phát hiện làn đường

Nhận diện vật cản

Lập kế hoạch đường đi

Điều khiển tự động

Google, Tesla, Mercedes… đều đã thử nghiệm hệ thống lái xe tự động đạt mức tương tương hoặc vượt hiệu suất của con người trong một số điều kiện cụ thể.

(5) Y tế & chăm sóc sức khỏe

Chẩn đoán ung thư qua ảnh X-ray, MRI

Phân tích bệnh án

Dự đoán nguy cơ bệnh

Tổng kết: AI đang dần trở thành công nghệ cốt lõi của hầu hết ngành công nghiệp. Đặc biệt trong giao thông thông minh, thị giác máy tính và học sâu đóng vai trò quan trọng trong việc xây dựng hệ thống nhận diện biển báo — nền tảng của đề tài này.

1.2. Giới thiệu các mô hình sử dụng trong đề tài

1.2.1. Mô hình / thuật toán chính

Trong đề tài này, nhóm sử dụng hai mô hình chính thuộc lĩnh vực thị giác máy tính và học sâu:

(1) Mạng nơ-ron tích chập – Convolutional Neural Network (CNN)

CNN là kiến trúc mạng nơ-ron sâu chuyên xử lý dữ liệu dạng lưới, đặc biệt là hình ảnh. CNN được sử dụng để phân loại biển báo giao thông, giúp xác định biển báo thuộc nhóm nào (cấm, nguy hiểm, hiệu lệnh,…).

(2) YOLO – You Only Look Once

YOLO là mô hình phát hiện đối tượng (object detection) thời gian thực. YOLO cho phép nhận diện vị trí + loại biển báo trên hình ảnh hoặc video. Mô hình xử lý toàn bộ ảnh chỉ trong một lần nhìn, đảm bảo tốc độ rất nhanh.

→ Tóm lại:

CNN dùng để phân loại (classification).

YOLO dùng để phát hiện + phân loại trong thời gian thực (real-time detection).

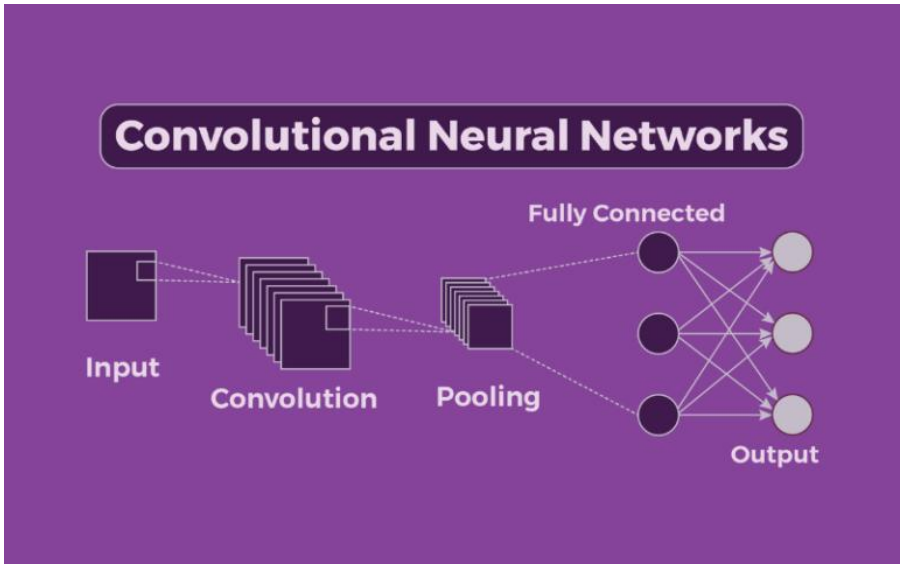
Hai mô hình kết hợp tạo thành hệ thống nhận diện biển báo đầy đủ và chính xác.

1.2.2. Cơ chế hoạt động

(1) Cơ chế hoạt động của CNN

CNN hoạt động dựa trên quá trình trích xuất đặc trưng tự động từ ảnh đầu vào thông qua các tầng tích chập (convolution). Luồng xử lý cơ bản gồm:

1. Input Layer – Nhận ảnh biển báo (thường được resize về 32×32 hoặc 64×64).
2. Convolution Layer – Sử dụng các kernel để trích xuất đặc trưng như cạnh, đường cong, hình dạng biển báo.
3. Activation (ReLU) – Loại bỏ giá trị âm, tăng phi tuyến tính.
4. Pooling Layer – Giảm kích thước dữ liệu, giữ đặc trưng quan trọng.
5. Fully Connected Layer – Kết hợp đặc trưng để đưa ra dự đoán.
6. Output Layer (Softmax) – Cho ra nhãn biển báo.



Hình 1. 1: Mô hình CNN

CNN đặc biệt phù hợp với bài toán biển báo vì hình dạng biển báo có tính đặc trưng mạnh (tròn, tam giác, viền đỏ…).

(2) Cơ chế hoạt động của YOLO

YOLO hoạt động theo tư tưởng phát hiện đối tượng trong một lần xử lý duy nhất. Luồng xử lý cơ bản:

1. Ảnh đầu vào được chia thành S × S ô lưới.
2. Mỗi ô dự đoán:

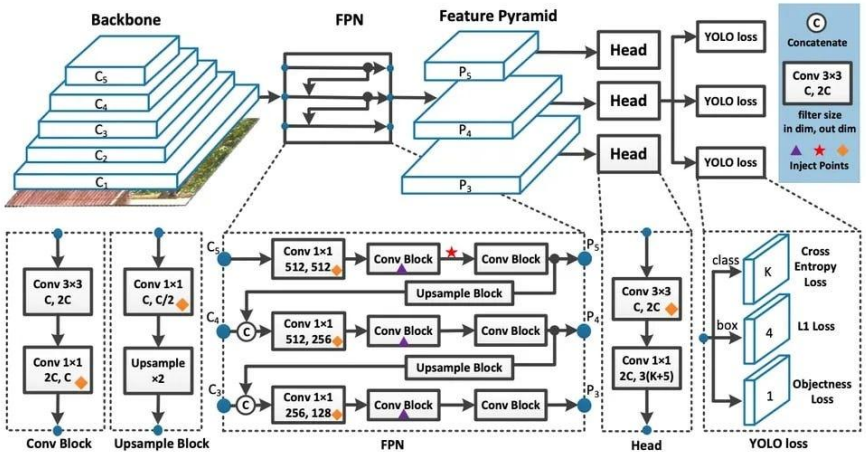
Vị trí bounding box

Kích thước box

Độ tin cậy (confidence)

Xác suất các lớp (class probabilities)

3. Mô hình hợp nhất dự đoán để xác định vị trí biển báo.
4. Áp dụng Non-Max Suppression để loại bỏ các box chồng chéo



Hình 1. 2: Mô hình Yolo

YOLO có ưu điểm chạy thời gian thực ngay cả trên GPU tầm trung, phù hợp cho ứng dụng giao thông.

1.2.3. Ưu – nhược điểm

Bảng 1. 1: Ưu – nhược điểm của CNN

Ưu điểm	**Nhược điểm**
Độ chính xác cao trong các tác vụ phân loại ảnh	Thời gian huấn luyện lâu nếu dữ liệu lớn
Tự động trích xuất đặc trưng, không cần thủ công	Nhạy cảm với thay đổi góc chụp, ánh sáng
Cấu trúc đơn giản, dễ tinh chỉnh	Cần nhiều dữ liệu để đạt hiệu suất tốt

Ổn định cho bài toán phân loại biển báo	Không xác định tọa độ biển báo (chỉ phân loại)

Bảng 1. 2: Ưu – nhược điểm của YOLO

Ưu điểm	Nhược điểm
Tốc độ rất nhanh, hỗ trợ thời gian thực	Độ chính xác thấp hơn các mô hình hai giai đoạn (Faster R-CNN)
Nhận diện vị trí + loại đối tượng	Khó nhận diện đối tượng nhỏ hoặc bị che một phần
Kiến trúc gọn nhẹ, dễ triển khai trên thiết bị	Cần dữ liệu đa dạng để giảm lỗi nhận nhầm
Phù hợp cho hệ thống giao thông thông minh	Huấn luyện phức tạp hơn CNN

1.2.4. Lý do chọn mô hình

(1) Lý do chọn CNN

Bài toán phân loại biển báo là dạng bài toán thị giác máy tính cổ điển, trong đó CNN cho hiệu suất cao và ổn định.

Biển báo giao thông có hình dạng rõ ràng, rất phù hợp với khả năng trích đặc trưng của CNN.

CNN đã được chứng minh là phù hợp trong nhiều nghiên cứu về phân loại biển báo (GTSRB).

Dễ triển khai, tối ưu, và huấn luyện nhanh trên dataset vừa phải.

(2) Lý do chọn YOLO

YOLO là một trong những mô hình nhanh nhất cho bài toán phát hiện biển báo thời gian thực.

Trong giao thông, tốc độ là yếu tố tối quan trọng — YOLO đáp ứng được yêu cầu này.

❖ YOLO có khả năng vừa nhận diện vừa phân loại, phù hợp cho ứng dụng thực tế như:

Camera hành trình,

Cảnh báo biển báo khi lái xe,

Hệ thống giao thông thông minh (ITS).

Kiến trúc YOLO dễ triển khai trên GPU và thiết bị nhúng.

Sự kết hợp giữa CNN (phân loại) và YOLO (nhận diện toàn ảnh) giúp hệ thống vừa chính xác, vừa nhanh, phù hợp với yêu cầu của đề tài về nhận diện biển báo giao thông Việt Nam.

CHƯƠNG II: PHÁT BIỂU VÀ MÔ TẢ BÀI TOÁN

2.1. Bài toán / vấn đề đặt ra

2.1.1. Mô tả bài toán

Nhận diện và phân loại biển báo giao thông là một bài toán thuộc lĩnh vực thị giác máy tính (Computer Vision), trong đó hệ thống cần phân tích hình ảnh đầu vào để xác định:

Biển báo giao thông có xuất hiện trong ảnh hay không.

Vị trí của biển báo trên ảnh (thông qua bounding box).

Biển báo thuộc loại nào (cấm, nguy hiểm, chỉ dẫn, hiệu lệnh…).

Đây là bài toán kết hợp giữa phát hiện đối tượng (object detection) và phân loại hình ảnh (classification).

❖ Để giải quyết bài toán, mô hình cần có khả năng:

Nhận dạng được hình dạng và đặc trưng của biển báo (hình tròn, tam giác, nền xanh, viền đỏ…).

Hoạt động tốt trong điều kiện ánh sáng thay đổi, góc chụp khác nhau, đèn xe, thời tiết hoặc biển báo bị che một phần.

Xử lý nhanh trong thời gian thực (đặc biệt quan trọng trong ứng dụng giao thông).

Tóm lại, bài toán trọng tâm của đề tài là xây dựng hệ thống tự động nhận diện và phân loại các loại biển báo giao thông Việt Nam từ ảnh đầu vào, sử dụng mô hình học sâu như CNN và YOLO.

2.1.2. Bối cảnh triển khai

Trong thực tế, nhu cầu nhận diện tự động biển báo giao thông xuất phát từ nhiều hệ thống hiện đại:

❖ Hệ thống hỗ trợ lái xe thông minh (ADAS)

Cảnh báo người lái khi sắp đến khu vực hạn chế tốc độ, đường cấm, đường nguy hiểm.

Hỗ trợ tài xế quan sát biển báo trong điều kiện trời tối hoặc thời tiết xấu.

❖ Xe tự hành (Autonomous Vehicle): Xe cần hiểu các biển báo để ra quyết định điều khiển như giảm tốc, rẽ, dừng,…

❖ Hệ thống giám sát giao thông

Camera giao thông tự động phát hiện vi phạm biển báo.

Phân tích lưu lượng phương tiện theo khu vực.

❖ Ứng dụng bản đồ số – dẫn đường (GPS)

Hỗ trợ hiển thị biển báo trên bản đồ theo thời gian thực.

❖ Ứng dụng đào tạo lái xe

Hỗ trợ nhận diện biển báo từ video mô phỏng.

Do Việt Nam có đặc thù giao thông phức tạp, nhiều loại biển báo khác nhau, và điều kiện môi trường đa dạng, việc xây dựng hệ thống nhận diện biển báo chính xác và nhanh chóng là vô cùng cần thiết.

2.2. Mục tiêu cần giải quyết

2.2.1. Mục tiêu chức năng

Hệ thống cần đáp ứng được các chức năng chính sau:

❖ Nhận diện biển báo trong ảnh đầu vào

Phát hiện xem ảnh có chứa biển báo hay không.

Xác định vị trí biển báo bằng bounding box.

❖ Phân loại biển báo

Dự đoán biển báo thuộc loại nào theo đúng quy chuẩn giao thông Việt Nam.

Phân loại được các nhóm biển chính (cấm – nguy hiểm – hiệu lệnh – chỉ dẫn…).

❖ Xử lý dữ liệu ảnh

Tiền xử lý dữ liệu (resize, normalize, augmentation).

Tự động trích xuất đặc trưng bằng CNN.

❖ Hiển thị kết quả

Hiển thị bounding box + nhãn biển báo trên ảnh.

Xuất kết quả dưới dạng ảnh hoặc văn bản.

❖ Hỗ trợ thời gian thực (khi chạy YOLO)

Nhận diện liên tục từ camera hoặc video.

2.2.2. Mục tiêu phi chức năng

Bên cạnh chức năng chính, hệ thống cần đảm bảo các yêu cầu phi chức năng quan trọng:

❖ Độ chính xác (Accuracy)

Mô hình phân loại đạt độ chính xác cao trên tập kiểm thử.

Phát hiện biển báo đúng vị trí, ít sai nhãn.

❖ Tốc độ xử lý (Real-time Performance)

YOLO phải xử lý ≥ 20 FPS để đáp ứng yêu cầu thời gian thực.

Tối ưu thời gian suy luận khi xử lý video.

❖ Khả năng mở rộng (Scalability)

Có thể bổ sung thêm các loại biển báo mới trong tương lai.

Dataset dễ mở rộng khi thu thập thêm dữ liệu.

❖ Khả năng chịu nhiễu (Robustness)

Hoạt động tốt khi ánh sáng kém, biển báo mờ, nghiêng, nhỏ hoặc bị che một phần.

❖ Tính ổn định (Reliability): Kết quả nhận diện ổn định qua nhiều khung hình liên tiếp.

❖ Tính dễ triển khai (Deployability): Có thể triển khai trên máy tính, GPU, hoặc thiết bị nhỏ (Jetson Nano, Raspberry Pi).

2.3. Đầu vào – Đầu ra của hệ thống

2.3.1. Dữ liệu đầu vào

Dữ liệu đầu vào của hệ thống là hình ảnh biển báo giao thông Việt Nam, được thu thập từ nhiều nguồn khác nhau và chuẩn hóa trước khi đưa vào mô hình học sâu. Các loại dữ liệu đầu vào bao gồm:

(1) Ảnh tĩnh (Image Input)

Ảnh chụp biển báo giao thông từ camera, điện thoại hoặc dataset công khai.

Định dạng phổ biến: JPEG, PNG.

Ảnh có thể chứa một hoặc nhiều biển báo.

(2) Nhãn dữ liệu (Ground Truth Labels)

Nhãn loại biển báo (P, W, R, I, S…)

Bounding box chuẩn trong trường hợp bài toán detection.

(3) Các thao tác tiền xử lý (Pre-processing)

Nhằm giúp mô hình học tốt hơn, dữ liệu đầu vào được xử lý qua các bước:

Resize về kích thước chuẩn (32×32 cho CNN, 416×416 cho YOLO).

Chuẩn hóa giá trị pixel (Normalization).

Tăng cường dữ liệu (Augmentation): xoay, lật, thay đổi độ sáng, thêm nhiễu,…

Việc tiền xử lý giúp mô hình hoạt động ổn định trong nhiều điều kiện thực tế: ánh sáng yếu, góc chụp nghiêng, biển báo nhỏ hoặc bị che khuất.

2.3.2. Dữ liệu đầu ra

Dữ liệu đầu ra của hệ thống là các thông tin phản ánh kết quả nhận diện và phân loại biển báo giao thông từ ảnh đầu vào. Cụ thể bao gồm:

❖ Nhãn phân loại (Label):

Tên hoặc mã lớp của biển báo giao thông mà hệ thống dự đoán (ví dụ: “Biển cấm rẽ trái”, “Biển hạn chế tốc độ 50 km/h”, “Biển nguy hiểm giao nhau”, …).

Nhãn được sử dụng như kết quả cuối cùng để hiển thị cho người dùng hoặc truyền cho các module khác.

❖ Giá trị dự đoán (Prediction/Score):

Xác suất hoặc điểm tin cậy mà mô hình gán cho từng lớp biển báo.

Có thể được biểu diễn dưới dạng vector phân phối xác suất trên toàn bộ các lớp, trong đó lớp có giá trị cao nhất sẽ được chọn làm kết quả phân loại chính.

❖ Kết quả phân loại (Classification Result):

Thông điệp hoặc cấu trúc dữ liệu tổng hợp, bao gồm: ảnh đầu vào, nhãn dự đoán, xác suất tương ứng và các thông tin mô tả (nếu có).

Có thể được sử dụng để hiển thị trên giao diện (ví dụ: "Biển báo được nhận diện là: Biển cấm dừng, đỗ với độ tin cậy 92%").

❖ Dữ liệu phục vụ đánh giá (nếu ở chế độ kiểm thử):

Cặp (nhãn thực tế, nhãn dự đoán) cho từng mẫu, dùng để tính toán accuracy, ma trận nhầm lẫn và các chỉ số đánh giá khác trong phần kiểm thử & đánh giá.

Như vậy, đầu ra của hệ thống không chỉ dừng lại ở nhãn phân loại mà còn bao gồm thông tin dự đoán định lượng (xác suất, điểm tin cậy) và các cấu trúc dữ liệu hỗ trợ việc đánh giá chất lượng mô hình cũng như hiển thị kết quả cho người dùng.

CHƯƠNG III: CÀI ĐẶT – KIỂM THỬ – ĐÁNH GIÁ

3.1. Mô hình / Thuật toán

3.1.1. Kiến trúc mô hình

Trong đề tài, nhóm sử dụng mô hình mạng nơ-ron tích chập (Convolutional Neural Network – CNN) để thực hiện bài toán nhận diện và phân loại biển báo giao thông. Mô hình được xây dựng bằng thư viện Keras (trên nền tảng TensorFlow) theo kiến trúc tuần tự (Sequential model). Toàn bộ pipeline xử lý của hệ thống có thể mô tả như sau:

Bước 1 – Nạp dữ liệu ảnh:

Chương trình duyệt qua 43 thư mục con tương ứng với 43 lớp biển báo (từ 0 đến 42) trong thư mục train.

Mỗi thư mục lưu trữ các ảnh thuộc một loại biển báo nhất định; tên thư mục (chỉ số i) được dùng làm nhãn lớp.

Bước 2 – Tiền xử lý ảnh:

Mỗi ảnh được mở bằng thư viện PIL (Pillow).

Ảnh được resize về kích thước 30×30 pixel, đảm bảo tất cả ảnh đầu vào có cùng kích thước.

Ảnh được chuyển sang mảng NumPy và lưu vào danh sách data, nhãn tương ứng được lưu vào labels.

Bước 3 – Chia tập dữ liệu: Dữ liệu được chuyển sang dạng mảng NumPy, sau đó chia thành hai tập: tập huấn luyện (train) và tập kiểm thử (test) với tỷ lệ 80% – 20% bằng hàm train_test_split (tham số test_size=0.2).

Bước 4 – Mã hóa nhãn (one-hot encoding): Các nhãn lớp (0–42) được chuyển sang dạng one-hot vector bằng hàm to_categorical để phù hợp với bài toán phân loại nhiều lớp (multi-class classification).

Bước 5 – Mạng CNN:

Kiến trúc mạng gồm các khối tích chập – gộp – dropout ở phần trích xuất đặc trưng, sau đó là phần phân loại bằng các lớp Fully Connected. Cụ thể:

1. Lớp Conv2D thứ nhất

Số lượng kernel (filters): 32

Kích thước kernel: 5×5

Hàm kích hoạt: ReLU

Input shape: (30, 30, 3) (ảnh màu 30×30, 3 kênh)

2. Lớp Conv2D thứ hai

32 filters, kernel 5×5, activation ReLU.

Tiếp tục trích xuất các đặc trưng phức tạp hơn từ ảnh.

3. Lớp MaxPool2D thứ nhất

Kích thước pool: 2×2.

Giảm kích thước không gian đặc trưng, giúp giảm số tham số và hạn chế overfitting.

4. Lớp Dropout (rate = 0.25)

Ngẫu nhiên "tắt" 25% số neuron trong quá trình huấn luyện, giúp tránh hiện tượng overfitting.

5. Lớp Conv2D thứ ba

64 filters, kernel 3×3, activation ReLU.

6. Lớp Conv2D thứ tư

64 filters, kernel 3×3, activation ReLU.

7. Lớp MaxPool2D thứ hai

Kích thước pool: 2×2.

Lớp Dropout (rate = 0.25)

Tiếp tục giảm overfitting ở tầng trích xuất đặc trưng.

8. Lớp Flatten

Chuyển toàn bộ tensor đặc trưng 2D thành vector 1 chiều để đưa vào các lớp Dense.

9. Lớp Dense (ẩn)

Số neuron: 256

Hàm kích hoạt: ReLU

Đây là lớp Fully Connected chính, học mối quan hệ phi tuyến giữa đặc trưng và lớp biển báo.

10. Lớp Dropout (rate = 0.5)

"Tắt" 50% số neuron ở lớp ẩn nhằm tăng khả năng khái quát hóa.

11. Lớp Dense đầu ra

Số neuron: 43 (tương ứng 43 lớp biển báo).

Hàm kích hoạt: softmax – chuyển đầu ra thành phân phối xác suất trên 43 lớp.

3.1.2. Tham số mô hình

Các tham số chính được sử dụng trong quá trình huấn luyện mô hình (hyperparameters) được thiết lập như sau:

❖ Số lớp đầu ra (số lớp phân loại):

classes = 43

Tương ứng với 43 loại biển báo giao thông trong bộ dữ liệu.

❖ Hàm mất mát (Loss function):

categorical_crossentropy

Phù hợp với bài toán phân loại nhiều lớp với nhãn ở dạng one-hot.

❖ Thuật toán tối ưu (Optimizer):

Adam là thuật toán tối ưu phổ biến, kết hợp ưu điểm của AdaGrad và RMSProp, tự điều chỉnh learning rate trong quá trình huấn luyện.

Trong code không đặt tường minh, nên learning rate sử dụng giá trị mặc định của Adam trong Keras (thường là lr = 0.001).

❖ Batch size:

batch_size = 32

Mỗi lần cập nhật trọng số, mô hình sử dụng 32 mẫu huấn luyện.

Đây là kích thước batch phổ biến, cân bằng giữa tốc độ và độ ổn định của quá trình học.

❖ Số epoch:

epochs = 15

Tập dữ liệu huấn luyện được lặp lại 15 lần qua mô hình.

Số epoch đủ để mô hình hội tụ bước đầu, đồng thời không quá dài để tránh thời gian huấn luyện lớn trong môi trường máy cá nhân.

❖ Tỷ lệ dữ liệu kiểm thử (test_size):

test_size = 0.2

20% dữ liệu dùng để đánh giá mô hình, 80% dữ liệu dùng để huấn luyện.

❖ Dropout:

Hai lớp Dropout với rate = 0.25 trong phần CNN và rate = 0.5 trong phần Fully Connected.

Các giá trị này được lựa chọn nhằm giảm overfitting nhưng vẫn giữ được khả năng học đặc trưng của mạng.

❖ Thước đo đánh giá (Metrics): Độ chính xác được sử dụng làm thước đo chính để đánh giá chất lượng mô hình trong quá trình huấn luyện và kiểm thử.

Các tham số trên được lựa chọn theo hướng cân bằng giữa hiệu năng và chi phí tính toán, phù hợp với khuôn khổ một bài tập lớn, chạy trên máy tính cá nhân nhưng vẫn đạt được độ chính xác phân loại chấp nhận được.

3.1.3. Quy trình huấn luyện

Quy trình huấn luyện mô hình CNN cho bài toán nhận diện và phân loại biển báo giao thông được triển khai theo các bước chính sau:

1. Chuẩn bị và tiền xử lý dữ liệu:

❖ Duyệt và nạp ảnh:

Mỗi lớp biển báo được lưu trong một thư mục tương ứng train/0, train/1, …, train/42.

Chương trình sử dụng os.listdir() để lấy danh sách tất cả các file ảnh trong từng thư mục, sau đó lần lượt mở từng ảnh bằng Image.open().

❖ Chuẩn hóa kích thước ảnh:

Tất cả ảnh được resize về kích thước 30×30 pixel bằng hàm image.resize((30,30)).

Việc chuẩn hóa kích thước đảm bảo đầu vào cho mô hình có kích thước thống nhất.

❖ Chuyển ảnh sang mảng số:

Ảnh sau khi xử lý được chuyển sang mảng NumPy bằng np.array(image) và thêm vào danh sách data.

Nhãn lớp (chỉ số thư mục i) được thêm vào danh sách labels.

❖ Chuyển sang NumPy array:

Sau khi duyệt hết, data và labels được chuyển sang dạng mảng NumPy:

data = np.array(data)

labels = np.array(labels)

2. Chia tập huấn luyện và kiểm thử:

Dữ liệu được chia thành hai tập bằng hàm train_test_split của sklearn:

```
X_train, X_test, y_train, y_test = train_test_split(
    data, labels, test_size=0.2, random_state=43
)
```

Trong đó:

X_train, y_train: dữ liệu và nhãn dùng để huấn luyện.

X_test, y_test: dữ liệu và nhãn dùng để đánh giá.

test_size=0.2 tương ứng với 20% dữ liệu dành cho kiểm thử.

random_state=43 giúp quá trình chia dữ liệu tái lập được.

3. Mã hóa nhãn (One-Hot Encoding):

Các nhãn y_train, y_test ban đầu là số nguyên (0–42).

Sử dụng hàm to_categorical(y_train, 43) và to_categorical(y_test, 43) để chuyển nhãn sang dạng vector one-hot kích thước 43 phần tử.

Bước này giúp nhãn phù hợp với yêu cầu đầu vào của hàm mất mát categorical_crossentropy.

4. Xây dựng và biên dịch mô hình:

Mô hình được khởi tạo dạng Sequential(), sau đó lần lượt thêm các lớp:

Các lớp Conv2D, MaxPool2D, Dropout cho phần trích xuất đặc trưng.

Các lớp Flatten, Dense, Dropout, Dense(43, softmax) cho phần phân loại.

Mô hình được compile với:

```
model.compile(
    loss='categorical_crossentropy',
    optimizer='adam',
    metrics=['accuracy']
)
```

5. Huấn luyện mô hình:

Mô hình được huấn luyện với lệnh:

```
history = model.fit(
    X_train, y_train,
    batch_size=32,
    epochs=15,
    validation_data=(X_test, y_test)
)
```

Trong quá trình huấn luyện:

Ở mỗi epoch, mô hình được tối ưu trên tập train.

Đồng thời, mô hình được đánh giá trên tập validation (ở đây dùng chính X_test, y_test) để theo dõi val_loss và val_accuracy.

6. Lưu mô hình đã huấn luyện:

Sau khi huấn luyện xong, mô hình được lưu lại dưới dạng file my_model.h5:

model.save("my_model.h5")

File này có thể được nạp lại bằng load_model để dùng cho việc dự đoán sau này mà không cần huấn luyện lại.

7. Trực quan hóa quá trình huấn luyện:

❖ Kết quả huấn luyện được trực quan hóa bằng hai biểu đồ:

Biểu đồ Accuracy: so sánh training accuracy và val accuracy theo từng epoch.

Biểu đồ Loss: so sánh training loss và val loss theo từng epoch.

❖ Các biểu đồ này giúp đánh giá:

Mô hình có bị overfitting hay không (training accuracy cao nhưng val accuracy thấp).

Quá trình hội tụ của mô hình (loss có giảm ổn định hay không).

3.2. Dataset

3.2.1. Nguồn dữ liệu

Trong khuôn khổ đề tài, dữ liệu sử dụng cho việc huấn luyện và kiểm thử mô hình nhận diện biển báo giao thông được lấy từ bộ dữ liệu biển báo giao thông dạng mở (open dataset), được phân phối dưới dạng các thư mục con theo từng lớp biển báo.

Cụ thể:

Dữ liệu được tổ chức trong thư mục gốc train, bên trong gồm 43 thư mục con được đánh số từ 0 đến 42.

Mỗi thư mục con tương ứng với một lớp biển báo giao thông (một loại ký hiệu/ý nghĩa cụ thể) và chứa toàn bộ ảnh thuộc lớp đó.

Cấu trúc dữ liệu dạng thư mục giúp chương trình dễ dàng duyệt, nạp ảnh và gán nhãn tự động dựa trên tên thư mục (chỉ số lớp).

3.2.2. Quy mô dữ liệu

Bộ dữ liệu được sử dụng trong đề tài có các đặc điểm chính như sau:

❖ Số lượng lớp (số nhãn):

Tổng cộng 43 lớp biển báo giao thông, đánh số từ 0 đến 42.

Mỗi lớp biểu diễn một loại biển báo khác nhau (ví dụ: biển cấm, biển nguy hiểm, biển chỉ dẫn, giới hạn tốc độ,…).

❖ Số lượng mẫu:

Tổng số lượng ảnh trong toàn bộ dataset là N mẫu (ghi cụ thể theo thống kê thực tế).

Mỗi lớp có số lượng ảnh khác nhau, thể hiện sự phân bố không hoàn toàn cân bằng.

❖ Mức độ cân bằng dữ liệu:

Một số lớp phổ biến (biển cấm, biển giới hạn tốc độ, biển cảnh báo thường gặp) có số lượng mẫu tương đối lớn.

Một số lớp ít xuất hiện trong thực tế có số lượng ảnh ít hơn, dẫn tới hiện tượng mất cân bằng giữa các lớp.

Độ mất cân bằng này có thể ảnh hưởng đến khả năng phân loại chính xác các lớp ít dữ liệu; do đó, trong các hướng phát triển tiếp theo có thể xem xét các biện pháp như tăng cường dữ liệu (data augmentation) hoặc kỹ thuật cân bằng lại dữ liệu.

3.2.3. Tiền xử lý dữ liệu

Trước khi đưa vào mô hình CNN để huấn luyện, dữ liệu ảnh được xử lý qua một số bước tiền xử lý nhằm chuẩn hóa định dạng và hỗ trợ mô hình học đặc trưng tốt hơn:

1. Chuẩn hóa kích thước (Resize):

Các ảnh đầu vào ban đầu có kích thước không đồng nhất.

Trong chương trình, mỗi ảnh được resize về kích thước 30×30 pixel bằng thư viện PIL:

image = image.resize((30,30))

Việc sử dụng kích thước 30×30 giúp giảm số lượng tham số của mô hình, rút ngắn thời gian huấn luyện trong khi vẫn giữ lại đủ thông tin đặc trưng của biển báo.

2. Chuyển đổi sang mảng số (NumPy array):

Ảnh sau khi resize được chuyển sang dạng mảng NumPy bằng:

image = np.array(image)

Mỗi ảnh được biểu diễn dưới dạng tensor 3 chiều (30, 30, 3) tương ứng với chiều cao, chiều rộng và 3 kênh màu (RGB).

Các ảnh được lưu trong mảng data, còn nhãn tương ứng được lưu trong mảng labels.

3. Chia tập huấn luyện và kiểm thử:

Sau khi nạp toàn bộ ảnh, dữ liệu được chia thành hai tập:

Tập huấn luyện (train set)

Tập kiểm thử (test set)

Sử dụng hàm train_test_split với test_size = 0.2, tương ứng 80% mẫu dùng để huấn luyện và 20% mẫu dùng để đánh giá mô hình.

4. Mã hóa nhãn (One-hot encoding):

Nhãn ban đầu là số nguyên (0–42).

Được chuyển sang dạng one-hot vector bằng to_categorical(y_train, 43) và to_categorical(y_test, 43).

Bước này giúp nhãn phù hợp với đầu ra của mô hình softmax và hàm mất mát categorical_crossentropy.

5. Chuẩn hóa giá trị pixel (Normalize):

Thông thường, để mô hình học tốt hơn, các giá trị pixel (0–255) sẽ được chuẩn hóa về khoảng [0,1] bằng cách chia cho 255.

Bước chuẩn hóa có thể được thực hiện bằng:

X_train = X_train / 255.0

X_test = X_test / 255.0

Điều này giúp tăng độ ổn định của quá trình huấn luyện, giảm ảnh hưởng của độ sáng tuyệt đối.

6. Tăng cường dữ liệu (Data Augmentation):

Để cải thiện khả năng tổng quát hóa và giảm overfitting, có thể áp dụng một số phép biến đổi ảnh như:

Xoay nhẹ ảnh (rotation).

Lật ngang (horizontal flip).

Thay đổi nhẹ độ sáng/độ tương phản.

Zoom in/zoom out nhẹ.

Trong bản cài đặt cơ bản, dữ liệu được sử dụng trực tiếp sau bước resize; tuy nhiên, data augmentation là một hướng mở rộng tiềm năng giúp mô hình robust hơn với các điều kiện chụp ảnh khác nhau.

Nhờ các bước tiền xử lý trên, dữ liệu đầu vào trở nên đồng nhất về kích thước, định dạng và cấu trúc, tạo điều kiện thuận lợi cho quá trình huấn luyện và đánh giá mô hình CNN trong các mục tiếp theo.

3.3. Cấu hình hệ thống

3.3.1. Phần cứng

Trong phạm vi đề tài, hệ thống được cài đặt và thử nghiệm trên máy tính cá nhân với cấu hình tham khảo như sau:

Bộ vi xử lý (CPU): Intel® Core™ i5 / i7 thế hệ … (hoặc tương đương)

Bộ nhớ trong (RAM): 8 GB / 16 GB

Ổ cứng: 256 GB / 512 GB SSD (khuyến khích dùng SSD để rút ngắn thời gian đọc/ghi dữ liệu)

Card đồ họa (GPU) (nếu có): NVIDIA GeForce GTX 1050 / 1650 / RTX … với bộ nhớ đồ họa từ 4 GB trở lên

Trong trường hợp không có GPU chuyên dụng, mô hình vẫn có thể huấn luyện được trên CPU, tuy nhiên thời gian huấn luyện sẽ lâu hơn. Cấu hình trên đáp ứng đủ yêu cầu cho việc xử lý ảnh kích thước 30×30 và huấn luyện mô hình CNN với số epoch vừa phải (15 epoch như trong code).

3.3.2. Phần mềm

Hệ thống được triển khai trên nền tảng phần mềm với các thành phần chính:

❖ Hệ điều hành: Windows 10 64-bit / Windows 11 64-bit

❖ Ngôn ngữ lập trình: Python 3.x (ví dụ: Python 3.8 / 3.9)

❖ Thư viện và framework chính:

TensorFlow / Keras: dùng để xây dựng và huấn luyện mô hình CNN

keras.models.Sequential, Conv2D, MaxPool2D, Dense, Flatten, Dropout, load_model, …

scikit-learn: dùng cho việc chia dữ liệu train/test (train_test_split)

NumPy: xử lý mảng số, thao tác dữ liệu ảnh dạng mảng

Pillow (PIL): đọc và resize ảnh (Image.open, image.resize)

Matplotlib: vẽ biểu đồ Accuracy và Loss theo số epoch

pip cho việc cài đặt thư viện Python.

Các thư viện trên đóng vai trò nền tảng cho quá trình nạp dữ liệu, tiền xử lý, xây dựng mô hình, huấn luyện, đánh giá và trực quan hóa kết quả trong đề tài.

3.3.3. Môi trường chạy

Trong khuôn khổ đề tài, quá trình cài đặt và huấn luyện mô hình được thực hiện chủ yếu trên nền tảng Google Colab. Đây là dịch vụ notebook trực tuyến do Google cung cấp, hỗ trợ sẵn môi trường Python và có thể sử dụng GPU miễn phí để tăng tốc quá trình huấn luyện mô hình học sâu. Các đặc điểm chính của môi trường chạy:

Nền tảng: Google Colab (dựa trên Jupyter Notebook, chạy trong môi trường cloud).

❖ Cấu hình runtime:

Runtime type: Python 3

Hardware accelerator: GPU (NVIDIA Tesla K80/T4/P100, tùy thời điểm Colab cấp phát).

Bộ nhớ RAM và dung lượng lưu trữ tạm thời do Colab cung cấp (khoảng 12–25 GB RAM, dung lượng đĩa ~70–100 GB, có thể thay đổi theo phiên).

❖ Quản lý mã nguồn và dữ liệu: Mã nguồn huấn luyện mô hình (code Python) được lưu trữ dưới dạng file .ipynb trên Google Drive hoặc tải trực tiếp lên Colab.

Bộ dữ liệu biển báo giao thông được:

Hoặc tải lên Google Drive và mount vào Colab (sử dụng from google.colab import drive).

Hoặc tải trực tiếp lên thư mục làm việc của Colab (nếu kích thước không quá lớn).

❖ Quy trình thực thi:

Bước 1: Kết nối runtime Colab, bật GPU trong phần Runtime → Change runtime type → Hardware accelerator: GPU.

Bước 2: Chuẩn bị dữ liệu (tải/mount từ Google Drive, giải nén, đặt đúng cấu trúc thư mục train/0 … train/42).

Bước 3: Chạy lần lượt các cell mã nguồn: nạp thư viện, đọc dữ liệu, tiền xử lý, xây dựng và huấn luyện mô hình CNN.

Bước 4: Sau khi huấn luyện xong, mô hình my_model.h5 được lưu xuống Google Drive hoặc tải về máy cục bộ.

Bước 5: Vẽ và hiển thị các biểu đồ accuracy, loss trực tiếp trong notebook để phục vụ phân tích, đánh giá.

Việc sử dụng Google Colab mang lại một số lợi ích:

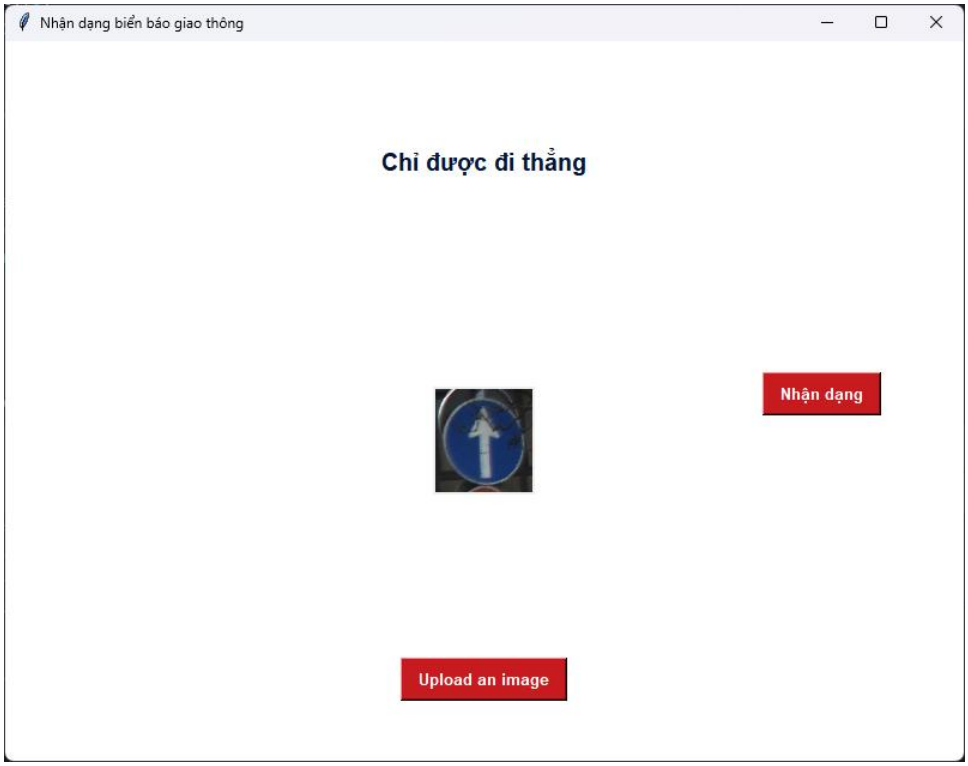
Không yêu cầu máy cá nhân phải có GPU mạnh;

Môi trường cài đặt sẵn các thư viện như TensorFlow/Keras, NumPy, Matplotlib, giúp giảm thời gian cấu hình;

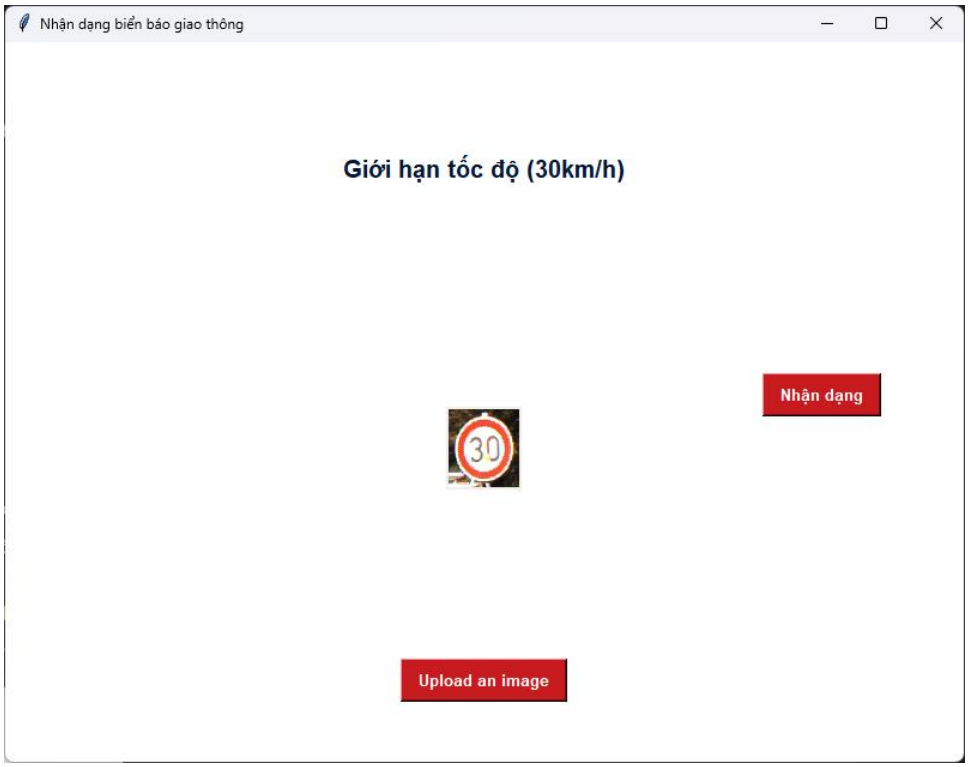
Dễ dàng chia sẻ notebook, tái hiện quá trình huấn luyện và kiểm thử mô hình.

3.4. Kiểm thử & đánh giá

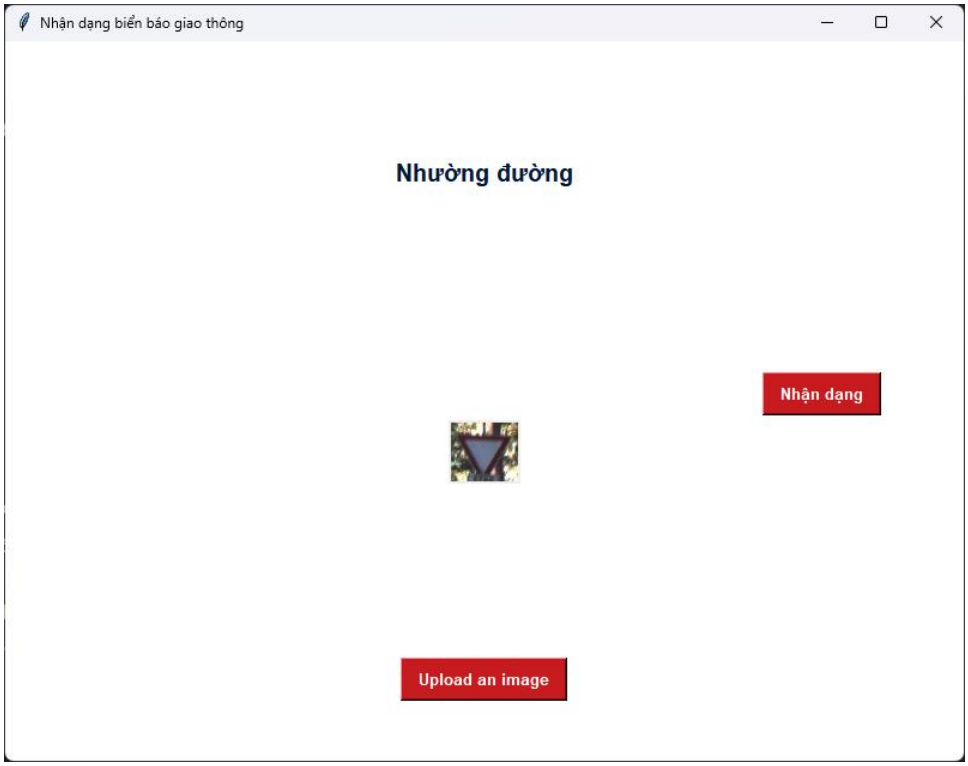
3.4.1. Hình ảnh minh họa



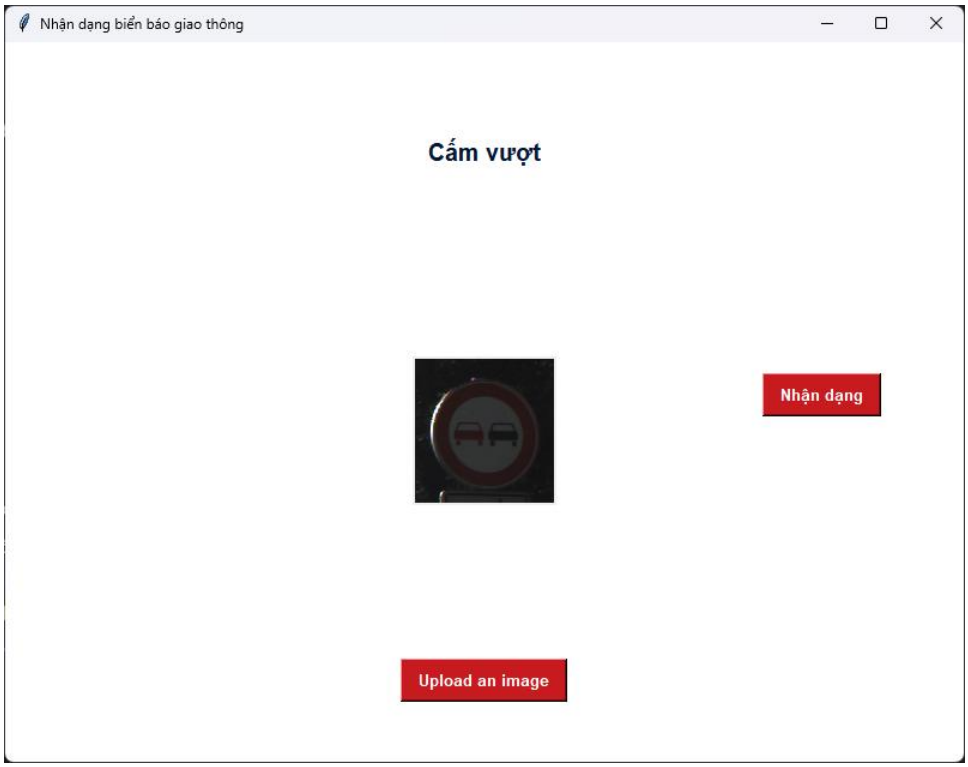
Hình 3. 1: Nhận diện lần 1



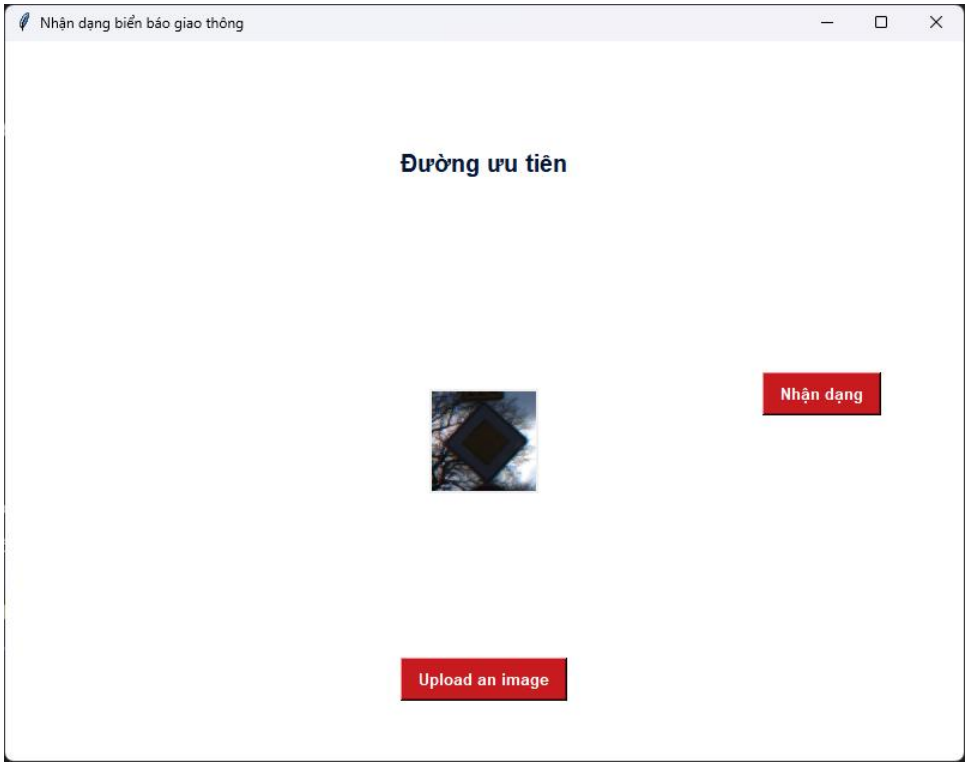
Hình 3. 2: Nhận diện lần 2



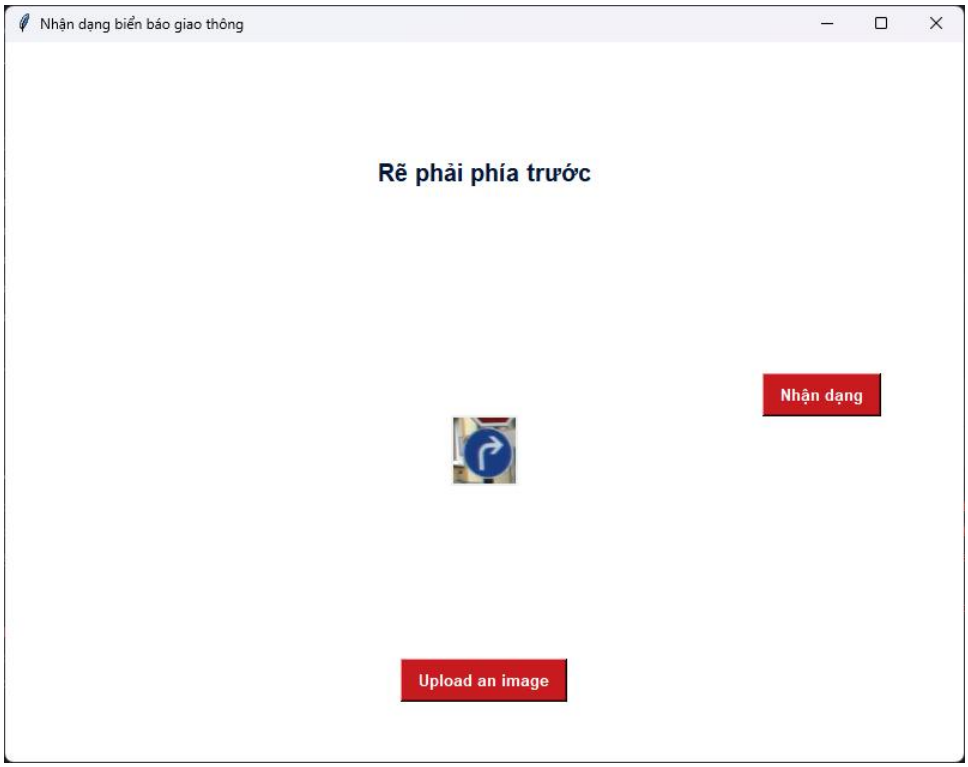
Hình 3. 3: Nhận diện lần 3



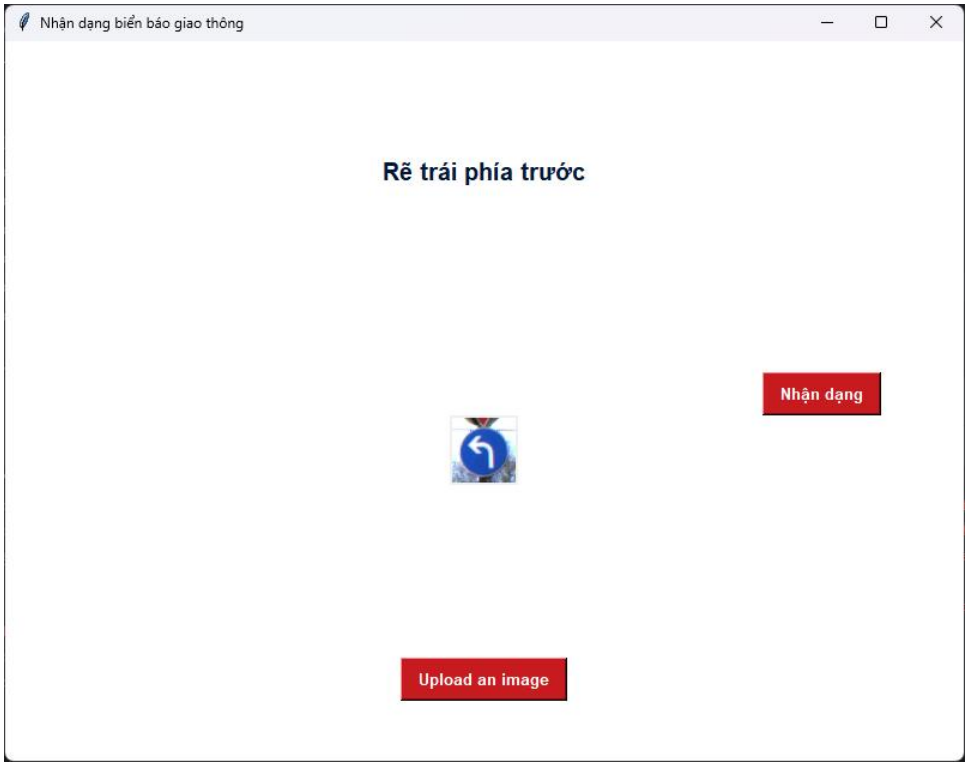
Hình 3. 4: Nhận diện lần 4



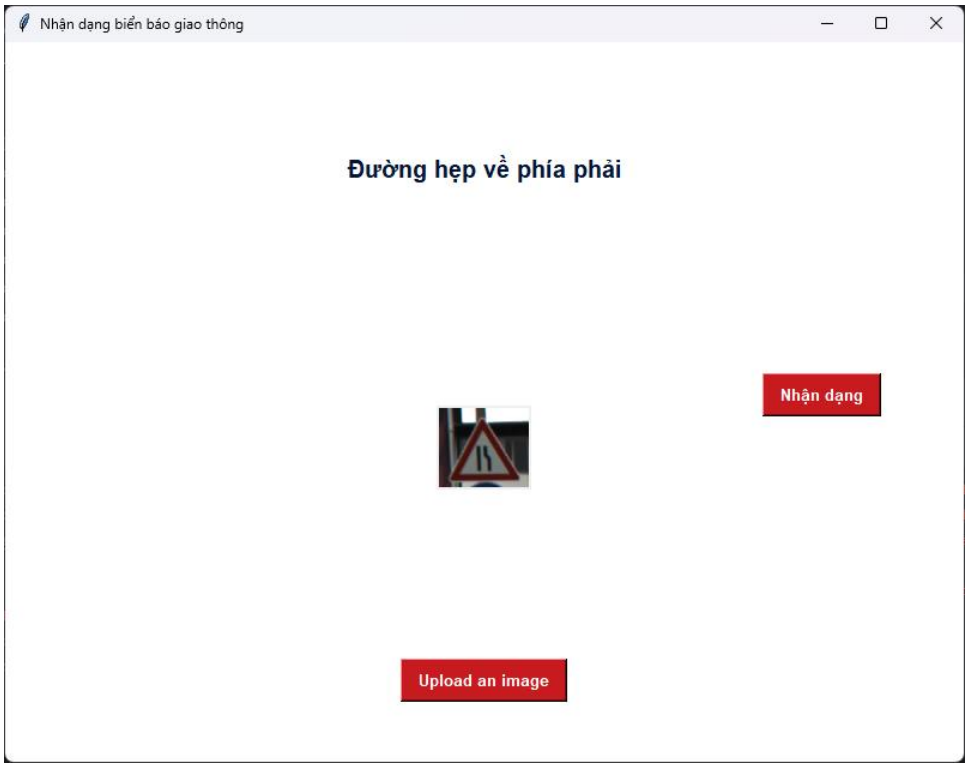
Hình 3. 5: Nhận diện lần 5



Hình 3. 6: Nhận diện lần 6



Hình 3. 7: Nhận diện lần 7



Hình 3. 8: Nhận diện lần 8

3.4.2. Phân tích kết quả

❖ Về khả năng dự đoán đúng:

Mô hình cho kết quả độ chính xác tương đối cao trên tập kiểm thử.

Các lớp biển báo có hình dạng và màu sắc đặc trưng rõ ràng, xuất hiện với số lượng mẫu lớn trong dữ liệu (ví dụ: "Giới hạn tốc độ", "Chỉ được đi thẳng", "Cấm vượt", "Đường ưu tiên", "Nhường đường", …) thường được mô hình nhận diện rất tốt.

Các ví dụ giao diện minh họa cho thấy mô hình dự đoán đúng trong nhiều tình huống ảnh có độ sáng khác nhau, nền cảnh phức tạp, nhưng biển báo vẫn rõ ràng.

❖ Về các trường hợp dự đoán sai:

Mô hình dễ nhầm lẫn giữa các biển báo có hình dạng tương tự nhau, đặc biệt là các biển tròn viền đỏ với các số tốc độ khác nhau, hoặc các biển cảnh báo tam giác có biểu tượng gần giống.

Một số ảnh có chất lượng kém (nhòe, thiếu sáng, bị chói, bị che khuất một phần) làm cho mô hình khó trích xuất đặc trưng chính xác, dẫn đến dự đoán sai.

Ở những lớp có số lượng mẫu ít, mô hình thường có xu hướng dự đoán nhầm sang các lớp "gần giống" về hình dạng nhưng nhiều dữ liệu hơn. Điều này phản ánh tác động của mất cân bằng dữ liệu.

❖ Về overfitting/underfitting:

So sánh đường training/validation accuracy và loss cho thấy mô hình không bị overfitting quá nặng: đường validation không tụt mạnh sau một số epoch, và không có khoảng cách quá xa so với training.

Tuy nhiên, vẫn có thể xuất hiện hiện tượng overfitting nhẹ khi số epoch tăng cao hơn (nếu tiếp tục huấn luyện), do đó việc sử dụng Dropout và dừng ở 15 epoch là hợp lý trong khuôn khổ đề tài.

Tổng hợp lại, mô hình CNN đã nắm bắt tốt đặc trưng của đa số lớp biển báo, hoạt động ổn định trên dữ liệu kiểm thử và các ảnh thực nghiệm bên ngoài, tuy vẫn còn một số lỗi dự đoán ở các lớp khó và ít dữ liệu.

3.4.3. Nhận xét về độ phức tạp

❖ Độ phức tạp tính toán (time complexity):

Mỗi epoch huấn luyện phải duyệt qua toàn bộ tập dữ liệu huấn luyện với một kiến trúc CNN gồm nhiều lớp tích chập và fully-connected, nên thời gian huấn luyện tỷ lệ với:

Nhờ sử dụng kích thước ảnh nhỏ (30×30) và kiến trúc mạng ở mức vừa phải, thời gian huấn luyện trên GPU của Google Colab là chấp nhận được cho 15 epoch.

❖ Thời gian suy luận (inference time):Khi chạy trên ứng dụng giao diện desktop,

việc dự đoán cho một ảnh đơn lẻ diễn ra rất nhanh (chỉ trong khoảng từ vài chục đến vài trăm mili–giây, tùy cấu hình CPU/GPU), nên người dùng cảm nhận được kết quả gần như tức thời sau khi nhấn nút "Nhận dạng".

❖ Mức sử dụng bộ nhớ (memory usage):

Mô hình CNN sử dụng nhiều lớp tích chập và dense, tuy nhiên kích thước ảnh nhỏ và số lớp không quá lớn nên dung lượng mô hình (my_model.h5) vẫn ở mức vừa phải, có thể nạp và chạy được trên nhiều máy tính cá nhân.

Việc lưu trữ dữ liệu ảnh để huấn luyện chiếm phần lớn dung lượng bộ nhớ, nhưng vẫn nằm trong khả năng của cả môi trường Google Colab và máy tính cài ứng dụng chạy mô hình.

Nhìn chung, độ phức tạp thời gian và bộ nhớ của mô hình là hoàn toàn phù hợp với một hệ thống nhận diện biển báo giao thông ở quy mô bài tập lớn, có thể triển khai trên máy tính cá nhân hoặc tích hợp vào các ứng dụng thử nghiệm.

3.4.4. Hạn chế

❖ Về dữ liệu:

Quy mô bộ dữ liệu còn hạn chế so với toàn bộ hệ thống biển báo giao thông Việt Nam ngoài thực tế.

Phân bố dữ liệu giữa các lớp chưa thật sự cân bằng; một số lớp có rất ít mẫu, dẫn tới khả năng nhận diện còn chưa ổn định.

Dữ liệu chủ yếu là ảnh biển báo rõ ràng; chưa bao quát đầy đủ các điều kiện khó như: trời mưa, sương mù, ban đêm, biển bị bẩn, bị che khuất nhiều, v.v.

❖ Về mô hình:

Mô hình CNN sử dụng kiến trúc tương đối đơn giản, số lớp và số tham số vừa phải, chưa áp dụng các kiến trúc hiện đại hơn như ResNet, MobileNet, EfficientNet…

Chưa thực hiện tối ưu siêu tham số (hyperparameter tuning) một cách hệ thống (learning rate, số epoch, số lớp, số filter, v.v.), nên hiệu năng đạt được mới ở mức tốt nhưng chưa chắc đã tối ưu.

Chưa sử dụng các kỹ thuật nâng cao như data augmentation mạnh, regularization bổ sung, hoặc transfer learning.

❖ Về phạm vi bài toán:

Hệ thống hiện tại chỉ xử lý ảnh đã cắt sẵn chứa biển báo, chưa giải quyết bài toán phát hiện vị trí biển báo trong ảnh toàn cảnh (object detection).

Ứng dụng demo mới dừng ở mức nhận dạng một biển báo mỗi lần, chưa xử lý tình huống nhiều biển báo xuất hiện cùng lúc trong một khung hình.

Những hạn chế trên đồng thời mở ra nhiều hướng phát triển tiếp theo cho đề tài, như mở rộng và cân bằng dữ liệu, áp dụng mô hình sâu hơn, kết hợp nhánh phát hiện đối tượng (YOLO, SSD, Faster R-CNN…), và tối ưu mô hình để có thể triển khai trên thiết bị di động hoặc hệ thống nhúng trên xe.

KẾT LUẬN

Trong khuôn khổ bài tập lớn học phần Trí tuệ nhân tạo, đề tài "Xây dựng hệ thống nhận diện và phân loại biển báo giao thông Việt Nam" đã tiến hành nghiên cứu từ cơ sở lý thuyết đến triển khai thực nghiệm. Trên phương diện lý thuyết, báo cáo đã hệ thống hóa các khái niệm về trí tuệ nhân tạo, học máy, học sâu và mạng nơ-ron tích chập (CNN), đồng thời trình bày quy trình tổng quát của một bài toán phân loại ảnh gồm các bước: thu thập và tổ chức dữ liệu, tiền xử lý, xây dựng mô hình, huấn luyện và đánh giá.

Về mặt triển khai, đề tài đã sử dụng bộ dữ liệu biển báo giao thông gồm 43 lớp, tổ chức theo cấu trúc thư mục, xây dựng mô hình CNN trên nền Keras/TensorFlow, huấn luyện trên môi trường Google Colab và tích hợp mô hình vào một ứng dụng có giao diện trực quan cho phép người dùng tải ảnh và nhận kết quả phân loại biển báo.

Kết quả thực nghiệm cho thấy mô hình CNN đề xuất đạt độ chính xác tương đối cao trên tập kiểm thử, các đường cong accuracy và loss trong quá trình huấn luyện thể hiện sự hội tụ ổn định, không xuất hiện hiện tượng quá khớp nghiêm trọng trong số epoch đã lựa chọn. Mô hình nhận diện tốt các biển báo có hình dạng và màu sắc đặc trưng, xuất hiện nhiều trong dữ liệu, và hoạt động hiệu quả trên các ảnh thử nghiệm ngoài tập huấn luyện thông qua ứng dụng giao diện, nơi kết quả được hiển thị bằng tiếng Việt tương ứng với ý nghĩa của từng biển báo.

Bên cạnh đó, quá trình đánh giá cũng chỉ ra rằng mô hình còn gặp khó khăn với các lớp ít dữ liệu hoặc có hình dạng gần giống nhau, phản ánh những hạn chế về mặt dữ liệu và kiến trúc chưa được tối ưu toàn diện.

Trong tương lai, đề tài có thể được mở rộng theo nhiều hướng nhằm nâng cao chất lượng và tính ứng dụng của hệ thống. Về dữ liệu, cần thu thập thêm ảnh biển báo trong nhiều điều kiện môi trường và ánh sáng khác nhau, đồng thời cân bằng số lượng mẫu giữa các lớp kết hợp với các kỹ thuật tăng cường dữ liệu để tăng khả năng khái quát hóa. Về mô hình, có thể thử nghiệm các kiến trúc CNN hiện đại hơn, áp dụng transfer learning và tối ưu siêu tham số một cách hệ thống.

Về phạm vi bài toán, một hướng phát triển quan trọng là mở rộng từ phân loại biển báo đã cắt sẵn sang bài toán phát hiện và nhận dạng biển báo trong ảnh toàn cảnh,

cũng như tối ưu mô hình để triển khai trên thiết bị di động hoặc hệ thống nhúng trên ô tô.

Với những định hướng này, hệ thống nhận diện biển báo giao thông được xây dựng trong đề tài không chỉ dừng ở mức bài tập lớn mà có thể trở thành nền tảng cho các ứng dụng hỗ trợ học luật và hỗ trợ lái xe an toàn trong các hệ thống giao thông thông minh.

TÀI LIỆU THAM KHẢO

[1] Từ Minh Phương. (n.d.). Nhập môn trí tuệ nhân tạo [Giáo trình]. Học viện Công nghệ Bưu chính Viễn thông, Hà Nội, Việt Nam.

[2] Russell, S. J., & Norvig, P. (2020). Artificial intelligence: A modern approach (4th ed.). Pearson.

[3] Trường Đại học Sư phạm Hà Nội. (n.d.). Giáo trình trí tuệ nhân tạo [Giáo trình]. Hà Nội, Việt Nam.

[4] Đinh Mạnh Tường. (n.d.). Trí tuệ nhân tạo: Tri thức và lập luận [Giáo trình]. Hà Nội, Việt Nam.

[5] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.

[6] Bishop, C. M. (2006). Pattern recognition and machine learning. Springer.

[7] Russell, S. J., & Norvig, P. (2021). Artificial intelligence: A modern approach (4th ed.). Pearson.

[8] Nguyễn, D. H. (2019). Trí tuệ nhân tạo – Cơ sở và ứng dụng (Tái bản lần 2). NXB Khoa học và Kỹ thuật. (Ví dụ sách tiếng Việt, bạn thay bằng sách thật mà bạn dùng)

[9] Cireşan, D. C., Meier, U., Masci, J., & Schmidhuber, J. (2012). Multi-column deep neural networks for traffic sign classification. Neural Networks, 32, 333–338. https://doi.org/10.1016/j.neunet.2012.02.023

[10] Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. Neural Networks, 32, 323–332. https://doi.org/10.1016/j.neunet.2012.02.016

[11] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 770–778). https://doi.org/10.1109/CVPR.2016.90

[12] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In Advances in neural information processing systems (Vol. 25).

[13] TensorFlow. (n.d.). TensorFlow documentation. Retrieved May 10, 2025, from https://www.tensorflow.org/

[14] Kaggle. (n.d.). German traffic sign recognition benchmark. Retrieved May 10, 2025, from https://www.kaggle.com/

[15] Bộ Giao thông Vận tải. (2019). Quy chuẩn kỹ thuật quốc gia về báo hiệu đường bộ QCVN 41:2019/BGTVT. Truy cập từ https://www.mt.gov.vn/