

BỘ CÔNG THƯƠNG
TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI

=====***=====



BÁO CÁO THÍ NGHIỆM/ THỰC NGHIỆM
HỌC PHẦN: TRÍ TUỆ NHÂN TẠO
ĐỀ TÀI:

**Xây dựng hệ thống nhận dạng biển số xe sử dụng thuật
toán KNN**

Sinh viên thực hiện: Lưu Quang Dũng – 2023605863
 Lê Tiến Đạt – 2023606232
 Chu Thị Sương – 2021602368
 Phan Quốc Bảo – 2023603325

Lớp, khóa: 20241IT6094001_K18

Nhóm: 14

Người hướng dẫn: Ths. Mai Thanh Hồng

Hà Nội, năm 2025

PHIẾU HỌC TẬP CÁ NHÂN/NHÓM

I. Thông tin chung

1. Tên lớp: 20241IT6094001 Khóa: 18
2. Tên nhóm: Nhóm 14
3. Họ và tên thành viên trong nhóm:
 - Lư Quang Dũng – 2023605863
 - Lê Tiến Đạt – 2023606232
 - Chu Thị Sương – 2021602368
 - Phan Quốc Bảo – 2023603325

II. Nội dung học tập

1. Tên chủ đề: Xây dựng hệ thống nhận dạng biển số xe sử dụng thuật toán KNN
2. Hoạt động của sinh viên:
 - Hoạt động/Nội dung 1: Tổ chức nhóm, thảo luận và phân chia nhiệm vụ cho các thành viên trong nhóm. Mục tiêu/chuẩn đầu ra: L4
 - Hoạt động/Nội dung 2: Tìm hiểu khái niệm, định lý, cách giải quyết bài toán bằng thuật toán KNN. Mục tiêu/chuẩn đầu ra: L3
 - Hoạt động/Nội dung 3: Giải bài toán thủ công bằng thuật toán KNN và thực hiện cài đặt thuật toán với ngôn ngữ lập trình Python. Mục tiêu/chuẩn đầu ra: L3
3. Sản phẩm nghiên cứu: Báo cáo thực nghiệm tìm hiểu thuật toán KNN và ứng dụng trong bài toán nhận diện biển số xe..

III. Nhiệm vụ học tập

1. Hoàn thành Tiểu luận, Bài tập lớn, Đồ án/Dự án theo đúng thời gian quy định
2. Báo cáo sản phẩm nghiên cứu theo chủ đề được giao trước giảng viên và những sinh viên khác

IV. Học liệu thực hiện báo cáo Thí nghiệm, Thực nghiệm

- [1] Nguyễn Phương Nga, Trần Hùng Cường(2021), Giáo trình Trí tuệ nhân tạo - Trường Đại học Công Nghiệp Hà Nội, NXB Thống Kê.
- [2] Vũ, H. T. (n.d.). *Machine Learning cơ bản – Các thuật toán phổ biến*. Tài liệu học tập trên trang Machine Learning Cơ Bản.
- [3] Raschka, S. (2015). *Python Machine Learning*. Birmingham, UK: Packt Publishing.

- [4] Bipin Kumar. (n.d.). *Lung Cancer Example Dataset*. Dữ liệu thực hành cho thuật toán KNN.
- [5] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21–27.
- [6] Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

KẾ HOẠCH THỰC HIỆN BÁO CÁO THÍ NGHIỆM THỰC NGHIỆM

Tên lớp: 20241IT6094001 Khóa: K18

Tên nhóm: Nhóm 14

Họ và tên thành viên trong nhóm:

- Lư Quang Dũng – 2023605863
- Lê Tiến Đạt – 2023606232
- Chu Thị Sương – 2021602368
- Phan Quốc Bảo – 2023603325

Tên chủ đề: Xây dựng hệ thống nhận dạng biển số xe sử dụng thuật toán KNN.

Tuần	Người thực hiện	Nội dung công việc	Kết quả đạt được	Phương pháp thực hiện
Tuần 1	Cả nhóm	Thống nhất chủ đề, phân chia công việc, người làm báo cáo bản word (Phan Quốc Bảo, Chu Thị Sương), người làm lập trình (Lư Quang Dũng, Lê Tiến Đạt)	Bầu ra nhóm trưởng và phân chia được nhiệm vụ cho các thành viên.	Trao đổi, thảo luận giữa các thành viên qua Google meet.
	Chu Thị Sương	Phát biểu bài toán, xác định phạm vi nghiên cứu.	Xác định được phạm vi nghiên cứu.	Tìm hiểu trên internet.
	Phan Quốc Bảo	Tìm hiểu thực trạng ứng dụng nhận dạng biển số xe ở Việt Nam.	Xác định được thực trạng ứng dụng công nghệ nhận dạng biển số xe, và nhận thức được tầm quan	Tìm hiểu trên internet.

			trọng của việc phát triển công nghệ này.	
	Lê Tiến Đạt	Tìm hiểu, thu thập dữ liệu về biển số xe.	Tìm được các dữ liệu và các trường hợp lỗi của biển số xe	Tạo file để thu thập dữ liệu từ mọi người.
	Lưu Quang Dũng	Từ dữ liệu, xác định dữ liệu đầu vào và ra	Xác định được dữ liệu đầu vào và ra	Tìm hiểu trên internet và thuật toán.
Tuần 2	Cả nhóm	Thông nhất tổng quan báo cáo, tiến hành thực hiện các đầu mục của báo cáo. Nghiên cứu các môi trường ứng dụng cho bài toán	Nhóm đã xây dựng được khung báo cáo.	Tìm hiểu trên Internet
	Lê Tiến Đạt, Lưu Quang Dũng	Thiết lập môi trường Python, OpenCV, Scikit-learn. Thực hiện các bước xử lý: tiền xử lý ảnh (dùng OpenCV), phân đoạn ký tự, trích xuất đặc trưng,	Thiết lập môi trường và xây dựng bộ dữ liệu thành công	Tham khảo trên Internet

		<p>và phân loại bằng KNN (dùng scikit-learn).</p> <p>- Lập trình và thử nghiệm hệ thống, điều chỉnh tham số để tăng độ chính xác</p>		
	Phan Quốc Bảo	<p>Viết lời mở đầu, trình bày lý do chọn đề tài, mục tiêu nghiên cứu, phạm vi (tập trung vào ảnh biển số xe Việt Nam, ảnh tĩnh, rõ nét), và kết quả mong đợi.</p>	Hoàn thành bản nháp	Tham khảo các cách viết trên Internet
	Chu Thị Sương	<p>Bắt đầu nghiên cứu và viết Chương 1.1: Trí tuệ nhân tạo (khái niệm và vai trò của AI trong giao thông, y học, kinh tế, công nghệ số).</p>		
Tuần 3	Cả nhóm	<p>Tìm hiểu thuật toán KNN (K-Nearest Neighbors) (định nghĩa, phân lớp).</p>	Nhóm đã hiểu được về thuật toán	Xây dựng dựa trên thuật toán tham khảo trên internet, chatGPT, Github,...

	Lê Tiến Đạt, Lưu Quang Dũng	- Thu thập và xây dựng bộ dữ liệu cho bài toán - Kiểm tra và tiền xử lý dữ liệu để chuẩn hóa trước khi sử dụng	Tìm được các dữ liệu, xác định được dữ liệu đầu vào và ra	Các thành viên đưa ra nhận xét và chỉnh sửa
	Phan Quốc Bảo	Viết báo cáo phần 2.1 : Các thuật toán phổ biến	Hoàn thành bản nháp	
	Chu Thị Sương	Viết báo cáo phần 2.2 : Thuật toán KNN và phần 2.3: Kết luận chương		
Tuần 4	Cả nhóm	Kết hợp cả báo cáo và phần code để hoàn thiện báo cáo, demo sản phẩm đã làm được trong các tuần vừa qua	Đã hoàn thành	Tham khảo trên internet
	Lê Tiến Đạt, Lưu Quang Dũng	Tiếp tục hoàn thiện phần code và sửa lỗi	Đóng góp ý kiến và chỉnh sửa nội dung tuần trước	

	Chu Thị Sương	Chỉnh sửa Chương 2, và tiến hành phân chia nhiệm vụ làm Chương 3. Viết phần 3.1 về bài toán và thuật toán áp dụng		
	Phan Quốc Bảo	Viết phần 3.2 và 3.3 kết hợp với thành viên viết code		
Tuần 5	Cả nhóm	Chỉnh sửa nội dung và hoàn thiện báo cáo. Phần code được hoàn thiện tuần trước tiếp tục được demo và đưa ra nhận xét	Dựa vào nội dung từ các tuần trước và hoàn thiện quyền báo cáo	Trao đổi, thảo luận giữa các thành viên qua Google Meet
	Lê Tiến Đạt, Lưu Quang Dũng	Hoàn thiện báo cáo, và bản kế hoạch phân chia theo tuần		
	Chu Thị Sương, Phan Quốc Bảo	Demo sản phẩm, chỉnh sửa lỗi và hoàn thiện bản cuối phần code		

Ngày tháng 06 năm 2025

XÁC NHẬN CỦA GIẢNG VIÊN

(Ký, ghi rõ họ tên)

Mai Thanh Hồng

BỘ CÔNG THƯƠNG TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI -----	CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM Độc lập – Tự do – Hạnh phúc -----
--	--

PHIẾU ĐÁNH GIÁ BÁO CÁO THÍ NGHIỆM/THỰC NGHIỆM **IT 6094-TRÍ TUỆ NHÂN TẠO**

I. THÔNG TIN CHUNG

Người đánh giá: Mai Thanh Hồng Học hàm, học vị: Thạc sỹ

Đơn vị công tác: Khoa CNTT

Tên lớp: 20242IT6094010 Khóa: 18

Họ và tên sinh viên: Lưu Quang Dũng

Tên nhóm: Nhóm 14

Tên sản phẩm: Xây dựng hệ thống nhận dạng biển số xe sử dụng thuật toán KNN.

II. ĐÁNH GIÁ

TT	Mục tiêu/chuẩn đầu ra học phần	Tiêu chí đánh giá sản phẩm	Điểm tối đa	Điểm đánh giá
1	L3	Vận dụng lý thuyết TTNT và công cụ phần mềm vào triển khai thực nghiệm.	9.5	
Tổng			9.5	

Ngày tháng 06 năm 2025

GIẢNG VIÊN

Mai Thanh Hồng

BỘ CÔNG THƯƠNG TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI -----	CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM Độc lập – Tự do – Hạnh phúc -----
--	--

PHIẾU ĐÁNH GIÁ BÁO CÁO THÍ NGHIỆM/THỰC NGHIỆM **IT 6094-TRÍ TUỆ NHÂN TẠO**

I. THÔNG TIN CHUNG

Người đánh giá: Mai Thanh Hồng

Học hàm, học vị: Thạc sỹ

Đơn vị công tác: Khoa CNTT

Tên lớp: 20242IT6094010 Khóa: 18

Họ và tên sinh viên: Chu Thị Sương

Tên nhóm: Nhóm 14

Tên sản phẩm: Xây dựng hệ thống nhận dạng biển số xe sử dụng thuật toán KNN.

II. ĐÁNH GIÁ

TT	Mục tiêu/chuẩn đầu ra học phần	Tiêu chí đánh giá sản phẩm	Điểm tối đa	Điểm đánh giá
1	L3	Vận dụng lý thuyết TTNT và công cụ phần mềm vào triển khai thực nghiệm.	9.5	
Tổng			9.5	

Ngày tháng 06 năm 2025

GIẢNG VIÊN

Mai Thanh Hồng

BỘ CÔNG THƯƠNG TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI -----	CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM Độc lập – Tự do – Hạnh phúc -----
--	--

PHIẾU ĐÁNH GIÁ BÁO CÁO THÍ NGHIỆM/THỰC NGHIỆM
IT 6094-TRÍ TUỆ NHÂN TẠO

I. THÔNG TIN CHUNG

Người đánh giá: Mai Thanh Hồng

Học hàm, học vị: Thạc sỹ

Đơn vị công tác: Khoa CNTT

Tên lớp: 20242IT6094010 Khóa: 18

Họ và tên sinh viên: Lê Tiến Đạt

Tên nhóm: Nhóm 14

Tên sản phẩm: Xây dựng hệ thống nhận dạng biển số xe sử dụng thuật toán KNN.

II. ĐÁNH GIÁ

TT	Mục tiêu/chuẩn đầu ra học phần	Tiêu chí đánh giá sản phẩm	Điểm tối đa	Điểm đánh giá
1	L3	Vận dụng lý thuyết TTNT và công cụ phần mềm vào triển khai thực nghiệm.	9.5	
Tổng			9.5	

Ngày tháng 06 năm 2025

GIẢNG VIÊN

Mai Thanh Hồng

BỘ CÔNG THƯƠNG TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI -----	CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM Độc lập – Tự do – Hạnh phúc -----
--	--

PHIẾU ĐÁNH GIÁ BÁO CÁO THÍ NGHIỆM/THỰC NGHIỆM

IT 6094-TRÍ TUỆ NHÂN TẠO

I. THÔNG TIN CHUNG

Người đánh giá: Mai Thanh Hồng

Học hàm, học vị: Thạc sỹ

Đơn vị công tác: Khoa CNTT

Tên lớp: 20242IT6094010 Khóa: 18

Họ và tên sinh viên: Phan Quốc Bảo

Tên nhóm: Nhóm 14

Tên sản phẩm: Xây dựng hệ thống nhận dạng biển số xe sử dụng thuật toán KNN.

II. ĐÁNH GIÁ

TT	Mục tiêu/chuẩn đầu ra học phần	Tiêu chí đánh giá sản phẩm	Điểm tối đa	Điểm đánh giá
1	L3	Vận dụng lý thuyết TTNT và công cụ phần mềm vào triển khai thực nghiệm.	9.5	
Tổng			9.5	

Ngày tháng 06 năm 2025

GIẢNG VIÊN

Mai Thanh Hồng

LỜI CẢM ƠN

Trước tiên, nhóm em xin bày tỏ lòng biết ơn sâu sắc đến Trường Đại học Công Nghiệp Hà Nội vì đã đưa môn học Trí tuệ nhân tạo vào chương trình giảng dạy. Đặc biệt, em xin gửi lời cảm ơn chân thành đến cô Mai Thanh Hồng, giảng viên bộ môn, vì sự tận tâm trong việc truyền đạt kiến thức quý giá và những kinh nghiệm thực tế trong suốt quá trình học tập. Những bài giảng của cô không chỉ giúp em mở rộng kiến thức chuyên môn mà còn trang bị cho em các kỹ năng mềm và phương pháp học tập hiệu quả, nghiêm túc.

Môn học Trí tuệ nhân tạo thực sự là một trải nghiệm thú vị và bổ ích, mang lại nền tảng vững chắc cho định hướng nghề nghiệp của chúng em trong tương lai. Chương trình học không chỉ đầy tính thực tiễn mà còn giúp sinh viên nắm bắt được các xu hướng công nghệ đang thay đổi mạnh mẽ trong thời đại hiện nay. Tuy nhiên, do vẫn còn thiếu sót trong kiến thức và kinh nghiệm thực tế, bài báo cáo của em khó tránh khỏi những điểm chưa hoàn thiện hoặc chưa chính xác. Em rất mong cô sẽ xem xét và đóng góp ý kiến để bài báo cáo của em có thể được hoàn thiện hơn.

Em xin chân thành cảm ơn cô một lần nữa vì sự hỗ trợ và hướng dẫn quý báu trong suốt quá trình học.

Nhóm sinh viên thực hiện

Nhóm 14

MỤC LỤC

LỜI CẢM ƠN	13
MỤC LỤC.....	1
DANH MỤC HÌNH ẢNH.....	3
DANH MỤC TỪ VIẾT TẮT.....	5
LỜI MỞ ĐẦU	1
CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI	4
1.1. Trí tuệ nhân tạo.....	4
1.1.1. Khái niệm.....	4
1.1.2. Vai trò.....	5
1.2. Tình hình nhận diện biển số xe.....	6
1.2.1. Tình hình ngoài nước	6
1.2.2. Tình hình trong nước.....	7
1.2.3. Các ứng dụng và tiềm năng phát triển.....	9
1.3 Kết luận chương 1	10
CHƯƠNG 2: CÁC THUẬT TOÁN GIẢI QUYẾT BÀI TOÁN.....	12
2.1. Các thuật toán phổ biến.....	12
2.1.1. Thuật toán ID3 (Decision Tree).....	12
2.1.2. Thuật toán Naive Bayes.....	13
2.2. Thuật toán KNN.....	15
2.2.1. Một số khái niệm.....	15
2.2.2 Ví dụ thuật toán KNN	18
2.3 Kết luận chương 2	20
CHƯƠNG 3: ỨNG DỤNG THUẬT TOÁN KNN VÀO BÀI TOÁN NHẬN	
DIỆN BIỂN SỐ XE.....	22

3.1. Bài toán.....	22
3.1.1. Phát biểu bài toán.....	22
3.1.2. Thuật toán áp dụng.....	22
3.1.3 Hướng giải quyết.....	23
3.2. Cơ sở dữ liệu.....	30
3.3. Cài đặt chương trình.....	30
3.3.1. Thuật toán	30
3.3.2. Cài đặt chương trình chi tiết	31
KẾT LUẬN	45
Tài liệu tham khảo	46

DANH MỤC HÌNH ẢNH

Hình 1.1 Hình ảnh nhận dạng biển số xe trên thế giới	7
Hình 1.2 Hình ảnh nhận diện biển số xe ở Việt Nam	8
Hình 1.3 Hình ảnh các biển số xe được nhận dạng khi bị mờ,...	9
Hình 2.1 Ví dụ mô hình cây quyết định	12
Hình 2.2 Mô hình thuật toán Naive Bayes	14
Hình 2.3 Ví dụ thuật toán KNN	16
Hình 2.4 Ví dụ minh họa thuật toán KNN	18
Hình 2.5 Ví dụ minh họa thuật toán KNN	20
Hình 3.1 : Hình ảnh sơ đồ xác định và tách biển số xe	23
Hình 3.2 Hình ảnh ví dụ giảm nhiễu bằng bộ lọc Gauss	25
Hình 3.3 Tổ chức dữ liệu	31
Hình 3.4 Hàm đọc dữ liệu và chỉnh ảnh	32
Hình 3.5 Dữ liệu của ảnh	33
Hình 3.6 Hàm chia dữ liệu thành tập huấn luyện và tập kiểm tra cho mô hình	33
Hình 3.7 Hàm tạo mô hình nơron	34
Hình 3.8 Hàm huấn luyện mô hình	34
Hình 3.9 Hàm xem độ chính xác trên mô hình	35
Hình 3.10 Hàm lấy đặc trưng	35
Hình 3.11 Hàm đặt lại số chiều	36
Hình 3.12 Dữ liệu của features	36
Hình 3.13 Hàm tạo mô hình KNN và train mô hình	37
Hình 3.14 Hàm xử lý ảnh nghiêng	38
Hình 3.15 Hàm dự đoán kí tự	39

Hình 3.16 Hàm sắp xếp các kí tự	39
Hình 3.17 Hàm tách biến số, kí tự	40
Hình 3.18 Hàm chuyển đổi mã hoá chữ ký tự	41
Hình 3.19 Hình ảnh của label khi chuyển sang số	41
Hình 3.20 Hàm chuyển đổi ngược lại từ số sang ký tự chữ	42
Hình 3.21 Hàm mã hóa từ số sang chữ	42
Hình 3.22 Ảnh ví dụ minh họa	43
Hình 3.23 Ảnh kết quả của ví dụ	44

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Tiếng Anh	Mô tả
ID3	Decision Tree	Cây quyết định
KNN	K-Nearest Neighbors	Tên thuật toán machine learning

LỜI MỞ ĐẦU

1. Lý do chọn đề tài

Trong thời đại công nghệ phát triển mạnh mẽ, việc ứng dụng trí tuệ nhân tạo vào quản lý và giám sát giao thông đang trở thành xu hướng tất yếu. Một trong những giải pháp nổi bật là hệ thống nhận dạng biển số xe (ANPR), hiện đang được triển khai tại nhiều bãi đỗ xe thông minh, trạm thu phí, khu đô thị và khu công nghiệp. Đây là một ứng dụng điển hình của học máy kết hợp với xử lý ảnh – hai lĩnh vực cốt lõi trong trí tuệ nhân tạo hiện đại.

Từ mong muốn tìm hiểu các ứng dụng thực tiễn của trí tuệ nhân tạo (AI), đặc biệt trong xử lý ảnh và học máy, nhóm em lựa chọn đề tài “**Xây dựng hệ thống nhận dạng biển số xe sử dụng thuật toán KNN**”. Thông qua đề tài này, nhóm mong muốn nâng cao kiến thức về học máy, kỹ năng lập trình, xử lý ảnh cơ bản, cũng như khả năng tư duy hệ thống và giải quyết vấn đề.

2. Tình hình nghiên cứu

Trong những năm gần đây, công nghệ nhận dạng biển số xe đã phát triển mạnh mẽ nhờ vào sự tiến bộ của các phương pháp học máy và xử lý ảnh. Các hệ thống này được ứng dụng rộng rãi trong nhiều lĩnh vực như kiểm soát giao thông, giám sát an ninh, thu phí tự động và nhiều dịch vụ công cộng khác. Tuy nhiên, việc xây dựng một hệ thống nhận dạng biển số xe chính xác, nhanh chóng và hiệu quả vẫn là một thách thức lớn, đặc biệt trong điều kiện môi trường phức tạp như ánh sáng thay đổi, góc chụp không đồng đều và tốc độ di chuyển của phương tiện.

3. Tính cấp thiết của đề tài

Tính cấp thiết của đề tài này xuất phát từ nhu cầu thực tế trong việc quản lý giao thông thông minh, đặc biệt trong các thành phố lớn. Hệ thống nhận dạng biển số xe giúp các cơ quan chức năng giám sát và xử lý các vi phạm giao thông một cách nhanh chóng và chính xác, đồng thời hỗ trợ các hoạt động như thu phí tự động và phát hiện xe vi phạm. Bằng cách sử dụng thuật toán KNN (K-Nearest Neighbors) – một phương

pháp học máy đơn giản nhưng hiệu quả – nhóm chúng em hy vọng sẽ xây dựng được một hệ thống có thể nhận diện biển số xe trong thời gian ngắn và đạt độ chính xác cao.

4. Mục tiêu của đề tài

Đề tài hướng đến việc xây dựng một mô hình đơn giản có khả năng nhận diện các ký tự trên biển số xe thông qua ảnh đã được xử lý trước. Cụ thể, các mục tiêu nhóm hướng đến bao gồm:

- Tìm hiểu nguyên lý hoạt động của thuật toán K-Nearest Neighbors (KNN) và cách áp dụng vào bài toán phân loại ký tự.
- Nắm được quy trình cơ bản của một hệ thống nhận dạng biển số xe, bao gồm các bước xử lý ảnh và triển khai mô hình học máy.
- Sử dụng ngôn ngữ lập trình Python và các thư viện như OpenCV, scikit-learn trong quá trình thực nghiệm.

5. Nội dung của đề tài

Đề tài sẽ tập trung vào nghiên cứu và phát triển hệ thống nhận dạng biển số xe, bao gồm các công đoạn từ thu thập dữ liệu ảnh biển số xe, tiền xử lý ảnh, áp dụng thuật toán KNN để phân loại biển số, đến việc đánh giá hiệu quả của hệ thống. Ngoài ra, nhóm nghiên cứu cũng sẽ tìm hiểu các phương pháp học máy khác để so sánh và cải tiến kết quả nhận dạng.

6. Cách tiếp cận và phương pháp nghiên cứu

Phương pháp nghiên cứu của đề tài bao gồm các bước chính: nghiên cứu lý thuyết về hệ thống nhận dạng biển số xe, tìm hiểu về thuật toán KNN và các thuật toán học máy khác, thu thập và tiền xử lý dữ liệu ảnh biển số xe, xây dựng và huấn luyện mô hình nhận dạng, và đánh giá hiệu quả của hệ thống. Các công cụ lập trình phổ biến như Python và các thư viện hỗ trợ học máy (scikit-learn, OpenCV, NumPy) sẽ được sử dụng để phát triển hệ thống.

7. Đối tượng phạm vi nghiên cứu

Đề tài tập trung vào việc phát triển hệ thống nhận dạng biển số xe tự động dành cho các phương tiện giao thông trong điều kiện thực tế, nhằm giải quyết các vấn đề thực tiễn liên quan đến quản lý giao thông và kiểm soát phương tiện. Hệ thống sẽ được thiết kế để nhận dạng biển số xe trên các loại phương tiện giao thông như xe con, xe tải, xe buýt và xe máy, đảm bảo hoạt động hiệu quả với các biển số có kích thước, vị trí và đặc điểm khác nhau. Nghiên cứu sẽ bao gồm việc thu thập và xử lý dữ liệu hình ảnh thực tế trong các điều kiện như thay đổi ánh sáng, góc chụp và chất lượng hình ảnh. Các bước tiền xử lý hình ảnh như lọc nhiễu, chuyển đổi thang xám, và phân đoạn biển số sẽ được tối ưu hóa để làm nổi bật ký tự cần nhận dạng. Đề tài cũng tập trung vào việc áp dụng thuật toán K-Nearest Neighbors (KNN) để nhận dạng ký tự trên biển số, đồng thời tìm hiểu cách tối ưu các tham số quan trọng của thuật toán như số lượng láng giềng (k) và phương pháp đo khoảng cách nhằm đạt được độ chính xác cao nhất. Mục tiêu là xây dựng một hệ thống có khả năng hoạt động hiệu quả trong thực tế, góp phần tự động hóa các quy trình liên quan đến giao thông và kiểm soát phương tiện.

8. Dự kiến kết quả đạt được

Sau khi hoàn thành đề tài, nhóm mong muốn đạt được một số kết quả cụ thể như sau:

- Về kiến thức: Hiểu rõ nguyên lý hoạt động của thuật toán KNN và nắm được quy trình tổng quát của một hệ thống nhận dạng ký tự từ ảnh.
- Về kỹ năng thực hành: Thành thạo các thao tác xử lý ảnh cơ bản và triển khai mô hình học máy đơn giản với Python, OpenCV và Scikit-learn.
- Về sản phẩm mô phỏng: Xây dựng được một mô hình có khả năng phân loại ký tự từ ảnh biển số đã tiền xử lý và đánh giá được độ chính xác ở mức cơ bản.
- Về kỹ năng mềm: Nâng cao khả năng làm việc nhóm, phối hợp thực hiện đề tài và trình bày báo cáo kỹ thuật một cách khoa học, logic.

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

1.1. Trí tuệ nhân tạo

1.1.1. Khái niệm

Trí tuệ nhân tạo (AI) là một lĩnh vực nghiên cứu và ứng dụng trong khoa học máy tính, với mục tiêu phát triển những hệ thống có khả năng thực hiện các nhiệm vụ mà trước đây chỉ có con người mới có thể làm được, chẳng hạn như nhận thức, học hỏi, suy luận và giải quyết vấn đề. AI không chỉ đơn thuần là việc tạo ra các chương trình máy tính thông minh mà còn là việc mô phỏng và ứng dụng các quá trình tư duy, hành động của con người vào trong máy móc và hệ thống tự động.

Một trong những khái niệm nền tảng trong AI là khả năng học máy (machine learning), nơi hệ thống được "dạy" từ dữ liệu và có thể cải thiện khả năng thực hiện nhiệm vụ mà không cần sự can thiệp của con người. Máy tính có thể nhận diện mẫu, đưa ra dự đoán và tự cải tiến qua thời gian, từ đó thực hiện các công việc như nhận diện hình ảnh, phân tích ngữ nghĩa của văn bản, dự đoán hành vi của người dùng, hay tự động hóa các quy trình công việc.

AI không chỉ dừng lại ở việc xử lý các công việc đơn giản mà còn có thể thực hiện các nhiệm vụ phức tạp như ra quyết định dựa trên các yếu tố biến động trong môi trường xung quanh. Ví dụ, trong y học, AI có thể phân tích hàng triệu dữ liệu y tế để đưa ra các phương án điều trị hoặc chẩn đoán bệnh chính xác hơn so với con người. Trong giao thông, các hệ thống AI giúp điều khiển các phương tiện tự lái một cách an toàn và hiệu quả.

Tuy nhiên, cùng với những ứng dụng tuyệt vời, AI cũng đặt ra một số vấn đề và lo ngại. Một trong những mối quan tâm lớn nhất là việc các hệ thống AI có thể phát triển đến mức tự quyết định và tự hành động mà không có sự kiểm soát của con người, từ đó dẫn đến những hệ quả không mong muốn. Do đó, việc nghiên cứu về đạo đức trong AI và phát triển các quy định, chính sách quản lý các công nghệ này trở thành yếu tố quan trọng trong việc đảm bảo rằng AI sẽ mang lại lợi ích lâu dài cho nhân loại mà không gây hại.

Trí tuệ nhân tạo hiện nay đã và đang phát triển mạnh mẽ, tạo ra những ứng dụng đầy tiềm năng trong nhiều lĩnh vực như chăm sóc sức khỏe, giao thông, tài chính, sản xuất, và nhiều ngành khác. Sự kết hợp của AI với các công nghệ khác như Internet of Things (IoT), Blockchain, và Big Data sẽ mở ra những cơ hội mới cho thế giới, giúp con người giải quyết các vấn đề phức tạp mà trước đây tưởng chừng không thể.

Tuy nhiên, cùng với những ứng dụng tuyệt vời, AI cũng đặt ra một số vấn đề và lo ngại. Một trong những mối quan tâm lớn nhất là việc các hệ thống AI có thể phát triển đến mức tự quyết định và tự hành động mà không có sự kiểm soát của con người, từ đó dẫn đến những hệ quả không mong muốn. Do đó, việc nghiên cứu về đạo đức trong AI và phát triển các quy định, chính sách quản lý các công nghệ này trở thành yếu tố quan trọng trong việc đảm bảo rằng AI sẽ mang lại lợi ích lâu dài cho nhân loại mà không gây hại.

Trí tuệ nhân tạo hiện nay đã và đang phát triển mạnh mẽ, tạo ra những ứng dụng đầy tiềm năng trong nhiều lĩnh vực như chăm sóc sức khỏe, giao thông, tài chính, sản xuất, và nhiều ngành khác. Sự kết hợp của AI với các công nghệ khác như Internet of Things (IoT), Blockchain, và Big Data sẽ mở ra những cơ hội mới cho thế giới, giúp con người giải quyết các vấn đề phức tạp mà trước đây tưởng chừng không thể.

1.1.2. Vai trò

Trong giao thông: Công nghệ AI ứng dụng trong nhận dạng biển số xe, phát hiện vượt đèn đỏ, cảnh báo nguy hiểm khi tham gia giao thông. Công nghệ AI ứng dụng vào hệ thống cảnh báo va chạm nhận diện vật cản đang là xu hướng phát triển mạnh.

Trong y học: Công nghệ AI đã mở ra một trang mới cho nền y học thế giới, đặc biệt là nền y học nước nhà. Nó mang đến cho con người những giá trị đáng kinh ngạc trong việc bảo vệ sức khỏe và điều trị bệnh tật. Tại lĩnh vực này, trí tuệ nhân tạo có vai trò quan trọng trong việc hỗ trợ điều trị y tế như định lượng thuốc, các phương pháp điều trị khác nhau cho bệnh nhân và quy trình phẫu thuật trong phòng mổ. Chúng sử dụng những thuật toán phân tích để hỗ trợ bệnh nhân theo dõi kết quả điều trị 24/7.

Trong kinh tế: Ngoài việc hỗ trợ con người chăm sóc sức khỏe, AI còn có vai trò quan trọng trong ngành tài chính ngân hàng. AI là công cụ giúp con người xử lý các hoạt động trong ngân hàng như xử lý giao dịch, theo dõi số dư, quản lý tài sản và các tài khoản tiền gửi lớn một cách nhanh chóng và chính xác nhất. Trí tuệ nhân tạo không những giúp các ngân hàng hợp lý hóa giao dịch mà còn có thể ước tính cung, cầu và định giá chứng khoán một cách dễ dàng hơn.

Trong công nghệ số hiện nay: Hiện nay, những tập đoàn lớn đang ngày càng thúc đẩy việc sử dụng máy móc thông minh vào dây chuyền sản xuất. AI được sử dụng như các robot có thể thay thế một phần công việc của con người. Khối lượng công việc và thời gian hoàn thành sẽ nhanh chóng và nhẹ nhàng hơn dưới sự hoạt động của máy móc tích hợp trí tuệ nhân tạo. Tiêu biểu là với các sản phẩm như ô tô tự lái và trò chơi điện tử. Trong trò chơi điện tử, trí tuệ nhân tạo AI sẽ tự phân tích các hành vi và đưa ra những đáp án không kém cạnh với trí tuệ con người. Với ô tô tự lái, hệ thống AI tính toán tất cả các dữ liệu bên trong động cơ, tìm hiểu cách đi và ngăn chặn va chạm bởi chướng ngại vật.

1.2. Tình hình nhận diện biển số xe

Trong bối cảnh công nghệ phát triển vượt bậc, việc ứng dụng các giải pháp thông minh vào lĩnh vực giao thông. Nhận diện biển số xe tự động là một trong những công nghệ tiên tiến, đóng vai trò then chốt trong việc nâng cao hiệu quả quản lý và đảm bảo an ninh giao thông. Qua đó, hệ thống nhận diện biển số xe **ANPR** (Automatic Number Plate Recognition) không chỉ góp phần tự động hóa việc giám sát giao thông mà còn hỗ trợ công tác an ninh, quản lý vé và thu phí, đáp ứng nhu cầu phát triển của thành phố thông minh.

1.2.1. Tình hình ngoài nước

Nhận diện biển số xe (ANPR) là một công nghệ đang được áp dụng rộng rãi trên toàn cầu để giải quyết các vấn đề giao thông, an ninh, và quản lý phương tiện. Công nghệ này giúp tự động nhận diện biển số xe từ hình ảnh hoặc video, nhờ vào việc sử dụng các thuật toán và trí tuệ nhân tạo (AI) để phân tích các đặc điểm của biển số, bất kể là ban ngày hay ban đêm. Trong nhiều quốc gia, nhận diện biển số xe

đã được ứng dụng thành công trong các hệ thống giám sát giao thông, quản lý bãi đỗ xe, thu phí đường bộ, và kiểm soát truy cập vào các khu vực cấm.



Hình 1.1 Hình ảnh nhận dạng biển số xe trên thế giới

Các nước phát triển như Mỹ, Anh, Nhật Bản, và các quốc gia châu Âu đã triển khai hệ thống nhận diện biển số xe tại nhiều thành phố lớn, giúp tối ưu hóa việc quản lý giao thông và giảm thiểu vi phạm. Công nghệ này cũng được sử dụng trong việc xác định và theo dõi các phương tiện liên quan đến tội phạm, cung cấp các dữ liệu hỗ trợ cho lực lượng cảnh sát và các cơ quan an ninh.

Tuy nhiên, các hệ thống nhận diện biển số xe vẫn gặp phải một số thách thức như độ chính xác của việc nhận diện trong các điều kiện khắc nghiệt (mưa, sương mù, ánh sáng yếu), hay việc phát hiện biển số giả mạo. Đặc biệt, chất lượng hình ảnh và vị trí của camera cũng ảnh hưởng lớn đến khả năng nhận diện.

1.2.2. Tình hình trong nước

Trong vài năm qua, công nghệ nhận diện biển số xe đã khẳng định vai trò then chốt trong hệ thống quản lý giao thông và đảm bảo an ninh tại Việt Nam. Nhiều thành phố lớn như Hà Nội và TP. Hồ Chí Minh đã triển khai các giải pháp ANPR/LPR nhằm theo dõi giao thông, phát hiện nhanh chóng các hành vi vi phạm như vượt tốc

độ, đỗ xe trái phép, hay không đóng phí trên các tuyến đường cao tốc. Các hệ thống này không chỉ được áp dụng trong việc thu phí tự động mà còn hỗ trợ quản lý bãi đỗ xe và giám sát an ninh tại các khu vực công cộng.

Tuy nhiên, hiệu quả của các hệ thống này cũng bị ảnh hưởng bởi một số yếu tố. Thứ nhất: chất lượng hình ảnh không ổn định, vì trong bối cảnh giao thông phức tạp và đông đúc, hiện tượng ùn tắc, ảnh chất lượng kém hoặc điều kiện ánh sáng yếu—đặc biệt vào giờ cao điểm—làm giảm độ chính xác của hệ thống. Thứ hai: tình trạng biển số xe, do các điều kiện môi trường và mức độ bảo dưỡng, biển số xe thường gặp tình trạng bị mờ, bẩn hoặc bị che khuất, điều này trực tiếp cản trở khả năng nhận dạng của công nghệ. Thứ ba: hệ thống dữ liệu không đồng bộ, việc thiếu một cơ sở dữ liệu thống nhất và toàn diện về biển số xe trên toàn quốc gây khó khăn trong việc tích hợp thông tin, từ đó ảnh hưởng đến hiệu suất của các ứng dụng giám sát và thu phí. Các cơ quan chức năng hiện đang tích cực nghiên cứu và cải tiến công nghệ, hướng đến việc tích hợp các thuật toán xử lý ảnh tiên tiến và trí tuệ nhân tạo nhằm nâng cao khả năng nhận dạng, giảm thiểu các hạn chế về chất lượng hình ảnh và tạo ra một hệ thống dữ liệu đồng bộ hơn.



Hình 1.2 Hình ảnh nhận diện biển số xe ở Việt Nam



Hình 1.3 Hình ảnh các biển số xe được nhận dạng khi bị mờ,...

1.2.3. Các ứng dụng và tiềm năng phát triển

Công nghệ nhận diện biển số xe không chỉ giúp giải quyết các vấn đề liên quan đến giao thông mà còn đóng góp vào công tác bảo vệ an ninh, giám sát hành vi tội phạm và hỗ trợ quản lý các phương tiện trong các khu vực công cộng. Những ứng dụng của công nghệ nhận diện biển số xe có thể được mở rộng trong các lĩnh vực như:

- **Quản lý giao thông:** Giúp giám sát và điều chỉnh lưu lượng giao thông, giảm ùn tắc và vi phạm.
- **Thu phí tự động:** Hệ thống nhận diện biển số được sử dụng để thu phí đường bộ, đặc biệt là ở các tuyến cao tốc hoặc khu vực thu phí.

- **Giám sát an ninh:** Công nghệ này có thể giúp xác định và theo dõi các phương tiện liên quan đến tội phạm hoặc vi phạm pháp luật.
- **Bãi đỗ xe thông minh:** Các hệ thống nhận diện biển số giúp tự động hóa quá trình xác thực xe ra vào các bãi đỗ xe.

Tuy nhiên, để phát triển và ứng dụng công nghệ nhận diện biển số xe hiệu quả hơn, cần cải thiện chất lượng hình ảnh, tăng cường cơ sở dữ liệu, và nâng cấp các thuật toán AI để nâng cao độ chính xác và độ tin cậy. Cùng với đó, việc bảo mật thông tin và quyền riêng tư của người dân cũng cần được đặc biệt chú trọng khi triển khai các hệ thống này.

1.3 Kết luận chương 1

Trong chương này, chúng ta đã tìm hiểu về trí tuệ nhân tạo (AI), vai trò quan trọng của nó trong nhiều lĩnh vực như giao thông, y học, kinh tế, và công nghệ số. AI không chỉ giúp tự động hóa các quy trình và cải thiện hiệu quả công việc mà còn góp phần vào việc nâng cao chất lượng cuộc sống con người. Tuy nhiên, với sự phát triển mạnh mẽ của AI, vấn đề đạo đức và quản lý công nghệ là những yếu tố cần được quan tâm để đảm bảo rằng AI phục vụ lợi ích chung của nhân loại mà không gây ra những hệ lụy không mong muốn.

Chương này cũng đã đề cập đến tình hình ứng dụng công nghệ nhận diện biển số xe (ANPR), một trong những ứng dụng AI nổi bật trong giao thông và quản lý phương tiện. Nhận diện biển số xe không chỉ giúp nâng cao hiệu quả trong việc giám sát giao thông mà còn hỗ trợ an ninh và quản lý các khu vực công cộng. Dù công nghệ này đã được triển khai thành công ở nhiều quốc gia, việc ứng dụng trong nước, đặc biệt tại Việt Nam, vẫn còn một số thách thức như độ chính xác của hệ thống trong điều kiện môi trường khắc nghiệt và chất lượng hình ảnh. Tuy nhiên, với sự phát triển của các thuật toán AI và cơ sở hạ tầng công nghệ, tiềm năng phát triển của công nghệ nhận diện biển số xe là rất lớn, và việc cải thiện chất lượng hình ảnh, dữ liệu và bảo mật sẽ giúp tối ưu hóa hiệu quả ứng dụng này trong tương lai.

Trong tổng thể, sự kết hợp giữa trí tuệ nhân tạo và các công nghệ tiên tiến sẽ mở ra nhiều cơ hội mới, giúp giải quyết các vấn đề phức tạp và tạo ra các giải pháp thông minh trong nhiều lĩnh vực khác nhau.

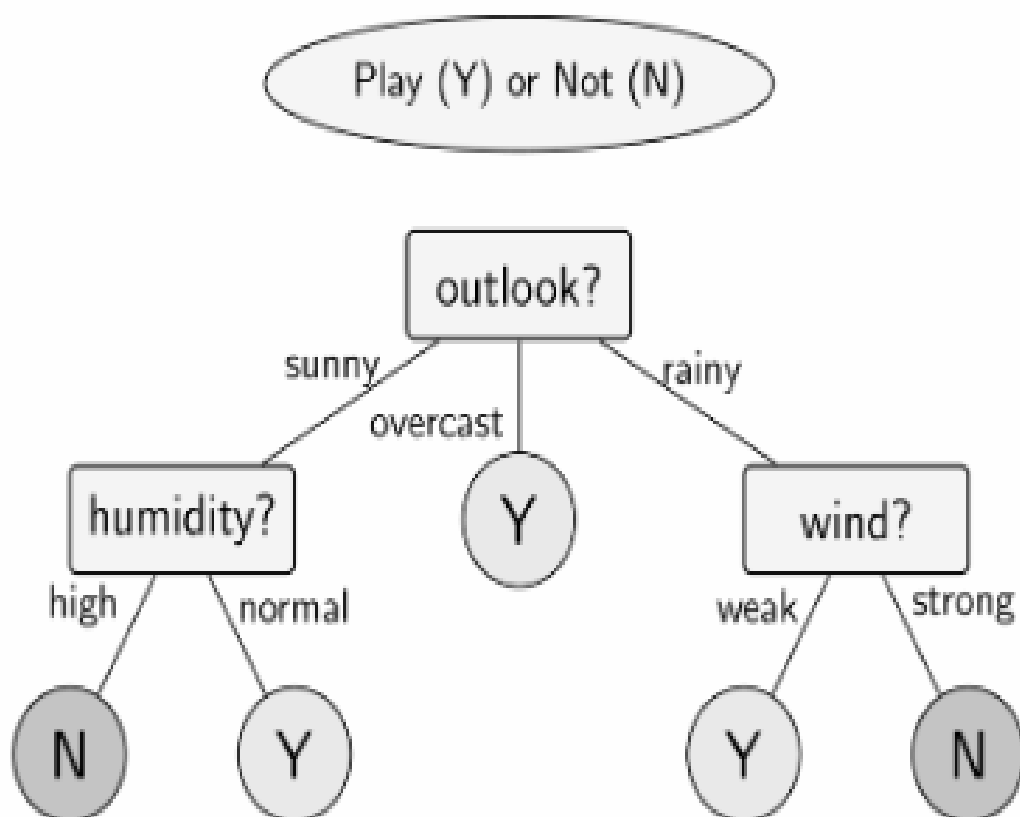
CHƯƠNG 2: CÁC THUẬT TOÁN GIẢI QUYẾT BÀI TOÁN

2.1. Các thuật toán phổ biến

2.1.1. Thuật toán ID3 (Decision Tree)

Thuật toán ID3 (Iterative Dichotomiser 3) là một phương pháp xây dựng cây quyết định dựa trên tiêu chí lựa chọn thuộc tính tốt nhất tại mỗi bước dựa trên độ lợi thông tin (Information Gain). Với mỗi thuộc tính được chọn, tập dữ liệu sẽ được chia thành các nhánh con (child node) tương ứng với từng giá trị của thuộc tính đó, và tiếp tục áp dụng quy trình này cho từng nhánh con.

Việc lựa chọn thuộc tính tốt nhất tại mỗi bước theo cách tham lam (greedy) giúp đơn giản hóa quá trình xây dựng cây, dù không đảm bảo tối ưu toàn cục. Một phép phân chia được coi là tốt nếu các nhánh con thu được có độ tinh khiết cao, tức là dữ liệu thuộc cùng một lớp. Để đánh giá chất lượng của một phép phân chia, các thước đo như entropy hoặc gain ratio thường được sử dụng.



Hình 2.1 Ví dụ mô hình cây quyết định

- *Ưu điểm của ID3*

- Mô hình dễ hiểu và dễ giải thích.
- Cần ít dữ liệu để huấn luyện.
- Có thể xử lý tốt với dữ liệu dạng số (rời rạc và liên tục) và dữ liệu hạng mục.
- Mô hình dạng white box rõ ràng.
- Xây dựng nhanh. Phân lớp nhanh.

- *Nhược điểm của ID3*

- Không đảm bảo xây dựng được cây tối ưu.
- Có thể overfitting (tạo ra những cây quá khớp với dữ liệu huấn luyện hay quá phức tạp).
- Thường ưu tiên thuộc tính có nhiều giá trị (khắc phục bằng cách sử dụng Gain Ratio).

2.1.2. Thuật toán Naive Bayes

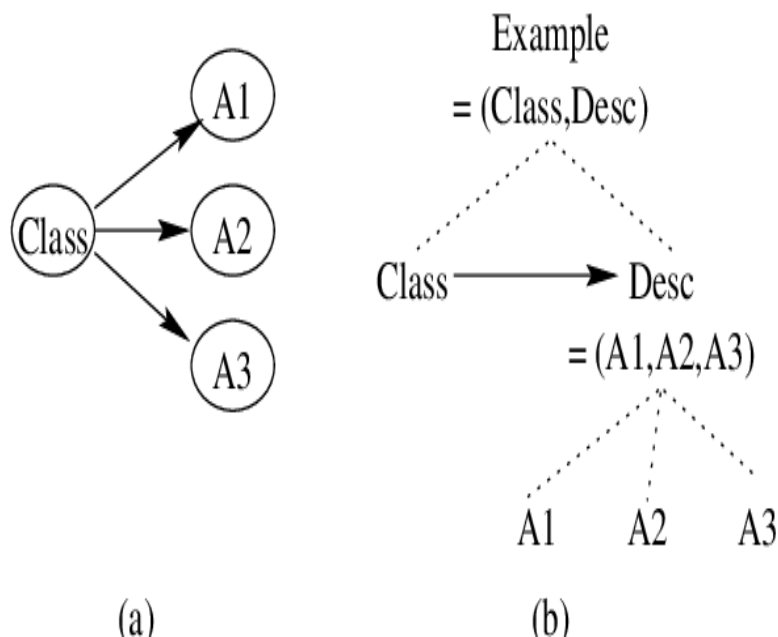
Naive Bayes là một trong nhóm các thuật toán áp dụng định lý Bayes với một giả định khá ngây thơ - đúng nghĩa đen của từ Naive, rằng mọi features đầu vào đều độc lập với nhau. Bạn có thể hiểu features ở đây là danh sách các biến đầu vào: độ tuổi, giới tính, mức lương, tình trạng hôn nhân, ... Ví dụ 2 biến độc lập là: size giày và giới tính của bạn. Ví dụ 2 biến phụ thuộc là: số tiền quảng cáo bỏ ra và doanh số thu được.

Naive Bayes là bộ phân loại theo xác suất (probability classifier) nên chúng ta sẽ đi tính toán xác suất bằng cách sử dụng định lý Bayes. Ví dụ khi bạn nhận được một tấm ảnh chứa một chữ số bất kì từ 0-9, bạn sẽ đi tính toán xác suất ảnh này với 10 con số từ 0-9 $P(8 | \text{Ảnh}) = x$, $P(7 | \text{Ảnh}) = y$, ..., sau đó chọn ra cặp nào có xác suất cao nhất thì đấy chính là kết quả. Naive Bayes Classifier đã được áp dụng thành công trong nhiều lĩnh vực, một trong số đó là Xử lý ngôn ngữ tự nhiên (Natural Language Processing).

- Ta hãy xét một ví dụ sau:

- + Hãy tưởng tượng chúng ta có hai người bạn là Alice và Bob. Và ta có những thông tin sau. Alice là người thường xuyên sử dụng những từ như: “love, great, wonderful” và Bob là người thường xuyên sử dụng những từ gồm: “dog, ball wonderful”.
- + Vào một ngày đẹp trời nào đó, bạn đột nhiên nhận được một email ẩn danh với nội dung: “I love beach sand. Additionally the sunset at beach offers wonderful view”. Khả năng là một trong hai người là Alice và Bob gửi email. Bạn có thể đoán được là ai với thông tin trên không? Chính xác là Alice. Ta dựa theo những từ như “Love”, “wonderful” được sử dụng. Một trường hợp khác, ta nhận được mail với nội dung là: “Wonderful Love”. Bạn sẽ đoán là ai?

Đó chính là Bob. Nhưng nếu bạn không đoán ra được là ai, thì đừng quá lo lắng, trong trường hợp này, ta sẽ sử dụng Định lý Bayes.



Hình 2.2 Mô hình thuật toán Naive Bayes

- *Ưu điểm của Naive Bayes*
 - Giả định độc lập: hoạt động tốt cho nhiều bài toán/miền sử liệu và ứng dụng. Đơn giản nhưng đủ tốt để giải quyết nhiều bài toán như phân lớp văn bản, lọc spam,..

- Cho phép kết hợp tri thức tiên nghiệm và dữ liệu quan sát được. Tốt khi có sự chênh lệch số lượng giữa các lớp phân loại.
- Huấn luyện mô hình (ước lượng tham số) dễ và nhanh.
- *Nhược điểm của Naive Bayes*
 - Giả định độc lập (ưu điểm cũng chính là nhược điểm) hầu hết các trường hợp thực tế trong đó có các thuộc tính trong các đối tượng thường phụ thuộc lẫn nhau.
 - Vấn đề zero (đã nêu cách giải quyết ở phía trên)
 - Mô hình không được huấn luyện bằng phương pháp tối ưu mạnh và chặt chẽ. Tham số của mô hình là các ước lượng xác suất điều kiện đơn lẻ. Không tính đến sự tương tác giữa các ước lượng này.

2.2. Thuật toán KNN

2.2.1. Một số khái niệm

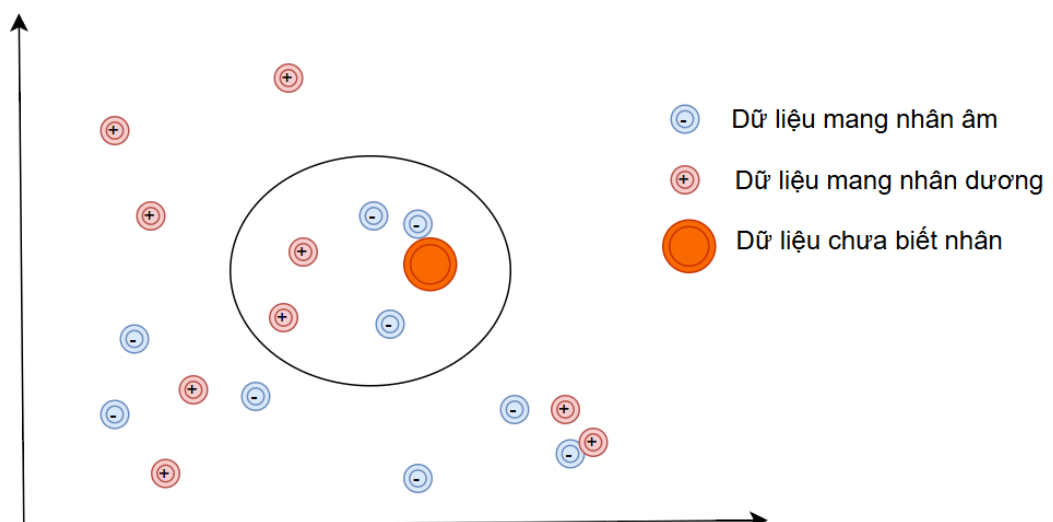
KNN (K-Nearest Neighbors) là một thuật toán học có giám sát đơn giản nhưng rất hiệu quả trong nhiều ứng dụng khai phá dữ liệu và học máy. Đây là một thuật toán được sử dụng rộng rãi trong nhiều lĩnh vực, từ phân loại văn bản, nhận dạng hình ảnh, đến dự đoán trong các hệ thống khuyến nghị. Một trong những lý do khiến KNN trở thành một trong những thuật toán phổ biến là vì tính đơn giản và khả năng thực thi dễ dàng của nó.

- Nguyên lý hoạt động của KNN :
 - + Ý tưởng cơ bản của thuật toán KNN là không thực sự "học" một mô hình hay hàm phân loại từ tập dữ liệu huấn luyện như các thuật toán học có giám sát khác. Thay vào đó, thuật toán này được xếp vào loại lazy learning (học lười) vì nó chỉ thực hiện các tính toán khi cần dự đoán nhãn cho một đối tượng dữ liệu mới. KNN dựa vào một nguyên tắc rất đơn giản: Lớp (nhãn) của một đối tượng dữ liệu mới có thể được dự đoán từ các lớp (nhãn) của k hàng xóm gần nhất.
 - + Cụ thể hơn, khi nhận được một điểm dữ liệu mới (A), KNN sẽ tính toán khoảng cách giữa điểm đó và tất cả các điểm trong tập huấn luyện (D) đã có nhãn, sau đó xác định k điểm dữ liệu gần nhất. Lớp của điểm A sẽ được xác

định dựa vào đa số các nhãn của các điểm gần nhất này. Thuật toán này có thể được áp dụng cho các bài toán phân loại (classification) hoặc hồi quy (regression) tùy thuộc vào yêu cầu của bài toán.

- Quy trình phân loại trong KNN :

+ Giả sử ta có D là tập các dữ liệu đã được phân loại thành 2 nhãn (+) và (-) được biểu diễn trên trục tọa độ như hình vẽ và một điểm dữ liệu mới A chưa biết nhãn. Vậy làm cách nào để chúng ta có thể xác định được nhãn của A là (+) hay (-)? Có thể thấy cách đơn giản nhất là so sánh tất cả các đặc điểm của dữ liệu A với tất cả tập dữ liệu học đã được gán nhãn và xem nó giống cái nào nhất, nếu dữ liệu (đặc điểm) của A giống với dữ liệu của điểm mang nhãn (+) thì điểm A mang nhãn (+), nếu dữ liệu A giống với dữ liệu nhãn (-) hơn thì nó mang nhãn (-), trông có vẻ rất đơn giản nhưng đó là những gì mà KNN làm. Trong trường hợp của KNN, thực tế nó không so sánh dữ liệu mới (không được phân lớp) với tất cả các dữ liệu khác, thực tế nó thực hiện một phép tính toán học để đo khoảng cách giữa dữ liệu mới với tất cả các điểm trong tập dữ liệu học D để thực hiện phân lớp. Phép tính khoảng cách giữa 2 điểm có thể là Euclidean, Manhattan, trọng số, Minkowski, ...



Hình 2.3 Ví dụ thuật toán KNN

Bước 1: Ta có D là tập các điểm dữ liệu đã được gán nhãn và A là dữ liệu chưa được phân loại.

Bước 2: Đo khoảng cách (Euclidean, Manhattan, Minkowski, Minkowski hoặc Trọng số) từ dữ liệu mới A đến tất cả các dữ liệu khác đã được phân loại trong D.

Bước 3: Chọn K (K là tham số mà bạn định nghĩa) khoảng cách nhỏ nhất.

Bước 4: Kiểm tra danh sách các lớp có khoảng cách ngắn nhất và đếm số lượng của mỗi lớp xuất hiện.

Bước 5: Lấy đúng lớp (lớp xuất hiện nhiều lần nhất).

Bước 6: Lớp của dữ liệu mới là lớp mà bạn đã nhận được ở bước 5.

- *Ưu điểm của KNN*

- Thuật toán đơn giản, dễ dàng triển khai.
- Độ phức tạp tính toán nhỏ.
- Xử lý tốt với tập dữ liệu nhiễu

- *Nhược điểm của KNN*

- Với K nhỏ dễ gặp nhiễu dẫn tới kết quả đưa ra không chính xác
- Cần nhiều thời gian để thực hiện do phải tính toán khoảng cách với tất cả các đối tượng trong tập dữ liệu.
- Cần chuyển đổi kiểu dữ liệu thành các yếu tố định tính.

Công thức của Euclidien: Đây là công thức phổ biến nhất và được sử dụng khi không gian dữ liệu có dạng không gian Euclidean (ví dụ như trong không gian 2D, 3D). Công thức tính khoảng cách Euclidean giữa hai điểm A và B có tọa độ (x_1, x_2, \dots, x_k) và (y_1, y_2, \dots, y_k) là:

$$d = \sqrt{\sum_{i=1}^k (x_i - y_i)^2}$$

Công thức Manhattan: Khoảng cách này tính tổng các sự khác biệt tuyệt đối giữa các tọa độ của hai điểm. Công thức là:

$$d(x_i, y_i) = \sum_{i=1}^k |x_i - y_i|$$

Khoảng cách Manhattan thường được sử dụng khi các dữ liệu có đơn vị không đồng nhất hoặc khi dữ liệu được tổ chức trong không gian dạng lưới.

Công thức Minkowski: Khoảng cách Minkowski là sự tổng quát của các loại khoảng cách trên. Nó được tính theo công thức:

$$d(x_i, y_i) = \left(\sum_{i=1}^k |x_i - y_i|^p \right)^{\frac{1}{p}}$$

(Khi $p = 2$ ta có đây là khoảng cách Euclidien)

2.2.2 Ví dụ thuật toán KNN

Ví dụ:

Chúng ta có dữ liệu là tuổi, khoản vay và khả năng vỡ nợ như hình:

Age	Loan	Default	Distance
25	40000	N	
35	60000	N	
45	80000	N	
20	20000	N	
35	120000	N	
52	18000	N	
23	95000	Y	
40	62000	Y	
60	100000	Y	
48	220000	Y	
33	150000	Y	
48	142000	?	

Hình 2.4 Ví dụ minh họa thuật toán KNN

$$D1=\sqrt{(48-25)^2+(142000-40000)^2}$$

$$D1 = 102000$$

$$D2=\sqrt{(48-35)^2+(142000-60000)^2}$$

$$D2 = 82000$$

$$D3=\sqrt{(48-45)^2+(142000-80000)^2}$$

$$D3 = 62000$$

$$D4=\sqrt{(48-20)^2+(142000-20000)^2}$$

$$D4 = 122000$$

$$D5=\sqrt{(48-35)^2+(142000-120000)^2}$$

$$D5 = 22000$$

$$D6=\sqrt{(48-52)^2+(142000-18000)^2}$$

$$D6 = 124000$$

$$D7=\sqrt{(48-23)^2+(142000-95000)^2}$$

$$D7 = 47000$$

$$D8=\sqrt{(48-40)^2+(142000-62000)^2}$$

$$D8 = 80000$$

$$D9=\sqrt{(48-60)^2+(142000-100000)^2}$$

$$D9 = 42000$$

$$D_{10} = \sqrt{(48 - 48)^2 + (142000 - 220000)^2}$$

$$D_{10} = 78000$$

$$D_{11} = \sqrt{(48 - 33)^2 + (142000 - 150000)^2}$$

$$D_{11} = 8000$$

Age	Loan	Default	Distance
25	40000	N	102000
35	60000	N	82000
45	80000	N	62000
20	20000	N	122000
35	120000	N	22000
52	18000	N	124000
23	95000	Y	47000
40	62000	Y	80000
60	100000	Y	42000
48	220000	Y	78000
33	150000	Y	8000
48	142000	?	

Hình 2.5 Ví dụ minh họa thuật toán KNN

- Sau khi tính các giá trị Distance ta sẽ lấy ra k Distance, trong bài trên ta sẽ lấy ra 3 Distance nhỏ nhất. So sánh các giá trị default của 3 Distance vừa chọn, chọn ra default có tỉ lệ cao nhất và đó là giá trị default của dữ liệu mới.
- Do trong bài trên lấy k=3 và trong đó có 2 giá trị cho kết quả là Y.
- > với số tuổi 48, tiền: 142000 thì theo thuật toán sẽ là vỡ nợ với kết quả là Y.

2.3 Kết luận chương 2

KNN là một thuật toán học máy rất dễ sử dụng và có thể ứng dụng trong nhiều lĩnh vực khác nhau. Tuy nhiên, để có được kết quả tối ưu, cần phải chú ý đến việc chọn lựa giá trị k phù hợp, cũng như sử dụng các phép tính khoảng cách chính xác

để đảm bảo độ chính xác cao nhất. KNN là một ví dụ điển hình của các thuật toán đơn giản nhưng mạnh mẽ, có thể được áp dụng vào các bài toán phân loại và hồi quy với hiệu quả cao khi dữ liệu không quá lớn.

CHƯƠNG 3: ỨNG DỤNG THUẬT TOÁN KNN VÀO BÀI TOÁN NHẬN DIỆN BIỂN SỐ XE

3.1. Bài toán

3.1.1. Phát biểu bài toán

Bài toán nhận diện biển số xe sử dụng thuật toán KNN (K-Nearest Neighbors) nhằm mục tiêu tự động nhận diện và phân loại biển số của các phương tiện giao thông từ các hình ảnh hoặc video. Mỗi biển số xe sẽ được biểu diễn dưới dạng các đặc trưng hình ảnh, chẳng hạn như các ký tự và số trên biển số, màu sắc, hình dạng và các yếu tố khác. Thuật toán KNN sẽ được áp dụng để phân loại biển số xe bằng cách so sánh các đặc trưng của biển số xe cần nhận diện với các biển số đã được phân loại trước đó trong cơ sở dữ liệu.

Mục tiêu của bài toán là xác định chính xác biển số xe từ hình ảnh hoặc video, bao gồm việc nhận diện từng ký tự hoặc số trên biển số, sau đó phân loại biển số vào đúng danh mục tương ứng. Thuật toán KNN sẽ dựa vào khoảng cách giữa các đặc trưng của biển số xe mới và các biển số đã có trong tập huấn luyện để dự đoán biển số của phương tiện đó.

3.1.2. Thuật toán áp dụng

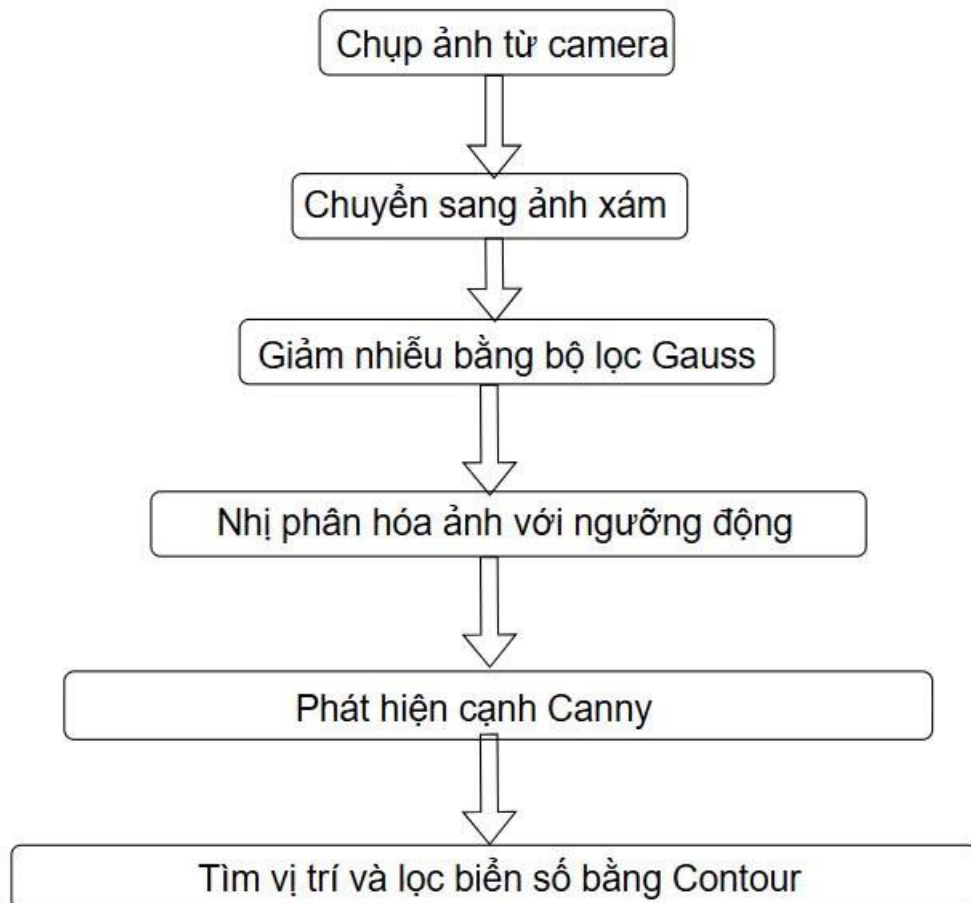
Bài toán nhận diện biển số xe áp dụng thuật toán KNN (K-Nearest Neighbors) để phân loại các ký tự (chữ và số) sau khi biển số đã được trích xuất từ hình ảnh. Thuật toán KNN là một phương pháp học máy đơn giản nhưng hiệu quả, trong đó mỗi ký tự trên biển số xe được phân loại dựa trên sự tương đồng với các ký tự đã có trong tập huấn luyện. Cụ thể, khi một ký tự mới được trích xuất từ hình ảnh, thuật toán KNN sẽ tính toán khoảng cách giữa ký tự đó và các ký tự trong tập huấn luyện (ví dụ: bằng cách sử dụng khoảng cách Euclidean). Sau đó, thuật toán sẽ chọn k ký tự gần nhất (hàng xóm) và dự đoán ký tự dựa trên đa số của các hàng xóm này.

Bài toán này được xây dựng bằng ngôn ngữ lập trình Python, một ngôn ngữ lập trình bậc cao nổi bật trong các ứng dụng học máy và xử lý hình ảnh. Python có bộ thư viện phong phú, đặc biệt là các thư viện như OpenCV, scikit-learn và TensorFlow, giúp dễ dàng triển khai các thuật toán học máy, bao gồm KNN. Thêm

vào đó, Python có cú pháp dễ hiểu, khả năng mở rộng và cộng đồng phát triển mạnh mẽ, chính vì vậy nó là lựa chọn lý tưởng để giải quyết bài toán nhận diện biển số xe.

3.1.3 Hướng giải quyết

Sơ đồ dưới đây sẽ tóm gọn các bước để xác định và tách biển số xe từ ảnh:



Hình 3.1 : Hình ảnh sơ đồ xác định và tách biển số xe

Đầu tiên từ ảnh đầu vào ta sẽ cắt biển số xử lý. Ở phạm vi đề án này, ý tưởng chủ yếu là nhận diện được biển số từ sự thay đổi đột ngột về cường độ ánh sáng giữa biển số và môi trường xung quanh nên ta sẽ loại bỏ các dữ liệu màu sắc RGB bằng cách chuyển sang ảnh xám. Ta sẽ tiến hành giảm nhiễu bằng bộ lọc Gauss để loại bỏ những chi tiết nhiễu có thể gây ảnh hưởng đến quá trình nhận diện, đồng thời làm tăng tốc độ xử lý.

Tiếp đó ta sử dụng thuật toán phát hiện cạnh Canny để trích xuất những chi tiết cạnh của biên số. Trong quá trình xử lý máy tính có thể nhầm lẫn biên số với những chi tiết nhiễu, việc lọc lần cuối bằng các tỉ lệ cao/rộng hay diện tích của biên số sẽ giúp xác định được đúng biên số. Cuối cùng, ta sẽ xác định vị trí của biên số trong ảnh bằng cách vẽ Contour bao quanh.

1. Chuyển ảnh xám

Ảnh xám (Gray Scale) đơn giản là một hình ảnh trong đó các màu là các sắc thái của màu xám với 256 cấp độ xám biến thiên từ màu đen đến màu trắng, nằm trong dải giá trị từ 0 đến 255, nghĩa là cần 8 bits hay 1 byte để biểu diễn mỗi điểm ảnh này. Lý do cần phải phân biệt giữa ảnh xám và các ảnh khác nằm ở việc ảnh xám cung cấp ít thông tin hơn cho mỗi pixel. Với ảnh thông thường thì mỗi pixel thường được cung cấp 3 trường thông tin trong khi với ảnh xám chỉ có 1 trường thông tin, việc giảm khối lượng thông tin giúp tăng tốc độ xử lý, đơn giản hóa giải thuật nhưng vẫn đảm bảo các tác vụ cần thiết

2. Giảm nhiễu bằng bộ lọc Gauss

1. Bộ lọc Gauss (Gauss filter)

Bộ lọc Gauss được cho là bộ lọc hữu ích nhất, được thực hiện bằng cách nhân chập ảnh đầu vào với một ma trận lọc Gauss sau đó cộng chúng lại để tạo thành ảnh đầu ra.

Ý tưởng chung là giá trị mỗi điểm ảnh sẽ phụ thuộc nhiều vào các điểm ảnh ở gần hơn là các điểm ảnh ở xa. Trọng số của sự phụ thuộc được lấy theo hàm Gauss (cũng được sử dụng trong quy luật phân phối chuẩn).

Giả sử ảnh là một chiều. Điểm ảnh ở trung tâm sẽ có trọng số lớn nhất. Các điểm ảnh ở càng xa trung tâm sẽ có trọng số giảm dần khi khoảng cách từ chúng tới điểm trung tâm tăng lên. Như vậy điểm càng gần trung tâm sẽ càng đóng góp nhiều hơn.

	
<p>Ảnh gốc</p>	<p>Ảnh sau khi làm mờ, giảm nhiễu</p>

Hình 3.2 Hình ảnh ví dụ giảm nhiễu bằng bộ lọc Gauss

3. Nhị phân hóa

1. Ảnh nhị phân

Là ảnh mà giá trị của các điểm ảnh chỉ được biểu diễn bằng hai giá trị là 0 (Đen) và 255 (Trắng).

2. Nhị phân hóa

Là quá trình biến đổi một ảnh xám thành ảnh nhị phân.

- Gọi giá trị cường độ sáng tại một điểm ảnh là $I(x,y)$.
- $INP(x,y)$ là cường độ sáng của điểm ảnh trên ảnh nhị phân .
- (Với $0 < x < \text{image.width}$) và $(0 < y < \text{image.height})$.

Để biến đổi ảnh xám thành ảnh nhị phân. Ta so sánh giá trị cường độ sáng của điểm ảnh với một ngưỡng nhị phân T .

- Nếu $I(x,y) > T$ thì $INP(x, y) = 0$.
- Nếu $I(x,y) > T$ thì $INP(x, y) = 255$.

- Giá trị của T trong bài là 127.

4. Phát hiện cạnh Canny (Canny Edge Detection)

Trong hình ảnh, thường tồn tại các thành phần như: vùng trơn, góc/cạnh và nhiễu. Cạnh trong ảnh mang đặc trưng quan trọng, thường là thuộc đối tượng trong ảnh. Do đó, để phát hiện cạnh trong ảnh, có nhiều giải thuật khác nhau như toán tử Sobel, toán tử Prewitt, Zero crossing nhưng ở đây em chọn giải thuật Canny vì hương pháp này hơn hẳn các phương pháp khác do ít bị tác động của nhiễu và cho khả năng phát hiện các biên yếu. Phương pháp này đi theo 4 bước chính:

1. Giảm nhiễu (Noise reduction)
2. Tính toán Gradient (Gradient calculation)
3. Loại bỏ những điểm không phải là cực đại (Non-maximum suppression)
4. Lọc ngưỡng (Double threshold)

a. Giảm nhiễu

Làm mờ ảnh, giảm nhiễu dùng bộ lọc Gauss kích thước 5x5. Kích thước 5x5 thường hoạt động tốt cho giải thuật Canny.

b. Tính toán Gradient

Ta dùng 2 bộ lọc Sobel X và Sobel Y (3x3) để tính đạo hàm Gx và Gy

- Bộ lọc Sobel theo trục X (Gx):

$$Kx = -1 \ 0 \ 1 \ -2 \ 0 \ 2 \ -1 \ 0 \ 1$$

- Bộ lọc Sobel theo trục Y (Gy):

$$Ky = 1 \ 2 \ 1 \ 0 \ 0 \ 0 \ -1 \ -2 \ -1$$

Sau khi thu được giá trị Gx và Gy tại mỗi điểm ảnh, ta tính:

- Độ lớn của gradient: $G = \sqrt{G_x^2 + G_y^2}$
- Góc phương của gradient: $\theta = \frac{G_x}{G_y}$

Góc gradient sau đó được lượng tử hoá (làm tròn) về một trong bốn hướng chính: hướng ngang (0 độ), hướng chéo bên phải (45 độ), hướng dọc (90 độ) và hướng chéo trái (135 độ).

c. Loại bỏ những điểm không phải là cực đại

Ở bước này, dùng một filter 3x3 lần lượt chạy qua các pixel trên ảnh gradient. Trong quá trình lọc, xem xét xem độ lớn gradient của pixel trung tâm có phải là cực đại so với các gradient ở các pixel xung quanh. Nếu là cực đại, ta sẽ ghi nhận sẽ giữ pixel đó lại. Còn nếu pixel tại đó không phải là cực đại lân cận, ta sẽ set độ lớn gradient của nó về zero. Ta chỉ so sánh pixel trung tâm với 2 pixel lân cận theo hướng gradient. Ví dụ: nếu hướng gradient đang là 0 độ, ta sẽ so pixel trung tâm với pixel liền trái và liền phải nó. Trường hợp khác nếu hướng gradient là 45 độ, ta sẽ so sánh với 2 pixel hàng xóm là góc trên bên phải và góc dưới bên trái của pixel trung tâm

d. Lọc ngưỡng

Lọc ngưỡng: ta sẽ xét các pixel dương trên mặt nạ nhị phân kết quả của bước trước. Nếu giá trị gradient vượt ngưỡng max_val thì pixel đó chắc chắn là cạnh. Các pixel có độ lớn gradient nhỏ hơn ngưỡng min_val sẽ bị loại bỏ. Còn các pixel nằm trong khoảng 2 ngưỡng trên sẽ được xem xét rằng nó có nằm liền kề với những pixel được cho là "chắc chắn là cạnh" hay không. Nếu liền kề thì ta giữ, còn không liền kề bất cứ pixel cạnh nào thì ta loại. Sau bước này ta có thể áp dụng thêm bước hậu xử lý loại bỏ nhiễu (tức những pixel cạnh rời rạc hay cạnh ngắn) nếu muốn

e. Kết quả

Sau khi sử dụng phát hiện biên canny, dù đã trích xuất được những chi tiết cạnh của ảnh, tuy nhiên vẫn còn quá nhiều chi tiết thừa trong hình ảnh, từ đây chúng ta sẽ vẽ contour, áp dụng những đặc điểm của ảnh để lọc lấy ra ảnh chính xác.

5. Loc biến số với contour

1. Một số phương pháp tìm contour

Có thể hiểu Contour là tập hợp các điểm tạo thành đường cong kín bao quanh một đối tượng nào đó. Thường dùng để xác định vị trí, đặc điểm của đối tượng. Có 4 thuật toán Contour Tracing chung nhất. Hai trong số đó có tên là: Square Tracing algorithm và Moore – Neighbor Tracing là dễ để thực hiện và thường xuyên được dùng để dò tìm contour của một mẫu. Với thư viện OpenCV người ta áp dụng thuật toán Suzuki's Contour tracing. Dưới đây em sẽ trình bày kĩ hơn về 3 phương pháp trên:

a. Thuật toán Square Tracing

Duyệt từ pixel ngoài cùng bên trái phía dưới, đi lên cho tới khi gặp pixel có giá trị bằng 255 (pixel này sẽ được gọi là pixel start) thì bắt đầu di chuyển theo quy tắc sau:

- Nếu gặp Pixel có giá trị bằng 255 thì rẽ trái.
- Nếu gặp Pixel có giá trị bằng 0 thì rẽ phải.
- Di chuyển cho tới khi quay lại pixel start thì dừng lại.

Thuật toán sẽ kết thúc đúng khi di chuyển vào pixel start lần thứ 2 sau khi đi qua n pixel khác và theo đúng hướng đi vào pixel start lần đầu tiên. Và sai khi di chuyển vào pixel start mà không đúng hướng ban đầu. Vậy thuật toán này chỉ chạy đúng trên đối tượng

b. Thuật toán Moore – Neighbor

Thuật toán này có chút khác biệt so với thuật toán Square Tracking, cụ thể: khi gặp pixel có giá trị bằng 255 đầu tiên (pixel start) thì ta sẽ quay lại pixel trước đó, sau đó đi vòng qua các pixel thuộc 8-connected theo chiều kim đồng hồ cho tới khi gặp pixel khác có giá trị bằng 255. Và điều kiện kết thúc cũng giống như thuật toán Square Tracking

c. Thuật toán Suzuki's Tracing

Đây là thuật toán được thư viện OpenCV sử dụng, ngoài khả năng xác định được biên của vật thể như hai phương pháp trên. Phương pháp Suzuki's Tracing còn có khả năng phân biệt được đường biên ngoài (Outer) và đường biên trong (Hole) của vật thể.

Hàm trong OpenCV được biểu diễn như sau:

FindContours (InputOutputArray image, OutputArrayOfArrays contours, OutputArray hierarchy, int mode, int method, Point offset=Point ())

Các tham số:

image: hình ảnh cần tìm biên, là ảnh nhị phân.

contours: lưu trữ các đường biên tìm được, mỗi đường biên được lưu trữ dưới dạng một vector của các điểm.

hierarchy : chứa thông tin về hình ảnh như số đường viền, xếp hạng các đường viền theo kích thước, trong ngoài, ..

mode :

CV_RETR_EXTERNAL : khi sử dụng cờ này nó chỉ lấy ra những đường biên bên ngoài, nhưng biên bên trong của vật thể bị loại bỏ.

CV_RETR_LIST : Khi sử dụng cờ này nó lấy ra tất cả các đường viền tìm được.

CV_RETR_CCOMP : khi sử dụng cờ này nó lấy tất cả những đường biên và chia nó làm 2 level, những đường biên bên ngoài đối tượng, và những đường biên bên trong đối tượng.

CV_RETR_TREE : khi sử dụng cờ này nó lấy tất cả các đường biên và tạo ra một hệ thống phân cấp đầy đủ của những đường lồng nhau.

method:

CV_CHAIN_APPROX_NONE: sử dụng cờ này sẽ lưu trữ tất cả các điểm của đường viền.

CV_CHAIN_APPROX_SIMPLE: Ví dụ: một hình chữ nhật sẽ được mã hoá bằng tọa độ của 4 đỉnh.

CV_CHAIN_APPROX_TC89_L1 or CV_CHAIN_APPROX_TC89_KCOS: Áp dụng thuật toán xấp xỉ Tech-Chin.

2. Lọc biên số

Đầu tiên ta làm xấp xỉ contour thành một hình đa giác và chỉ lấy những đa giác nào chỉ có 4 cạnh. Nghĩa là lúc xấp xỉ contour bộ nhớ chỉ ghi nhớ vị trí các đỉnh của đa giác đó thành một mảng. Số cạnh của đa giác sẽ bằng số đỉnh và bằng chiều dài của mảng đó.

Tiếp theo ta tính toán tỉ lệ cao/rộng và diện tích của biên số phù hợp, sau đó ta lưu tất cả những biên số có trong hình dưới dạng tọa độ các đỉnh

Từ đây, ta cắt hình ảnh biên số từ các tọa độ vị trí đã biết để phục vụ cho mục đích tiếp theo “Tách các ký tự trong biên số”. Lưu ý ở đây ta cắt từ ảnh nhị phân luôn để máy tính xử lý nhanh hơn, tốn ít thời gian hơn.

3.2. Cơ sở dữ liệu

- Thu thập và chuẩn bị dữ liệu:
 - + Dữ liệu huấn luyện:
 - Hình ảnh ký tự (chữ cái, chữ số) đã được gán nhãn (Ảnh của các chữ cái từ "A" đến "Z" và các chữ số từ "0" đến "9", đã được gán nhãn (mỗi mẫu là một vector 924 chiều cho ảnh 32×32)
 - Biên số xe từ các hình ảnh thực tế, với các ký tự được cắt và chuẩn hóa.
 - + Dữ liệu kiểm tra:
 - Hình ảnh biên số xe cần nhận diện.
 - Biên số xe có ký tự bị bóp méo do góc chụp hoặc ánh sáng, có thể bị che mờ, mất chữ số,...

3.3. Cài đặt chương trình

3.3.1. Thuật toán

Định nghĩa tiêu chí khoảng cách:

- Sử dụng khoảng cách Euclid hoặc khoảng cách Cosine giữa các vector ký tự :

$$d(X, Y) = \sqrt{\sum_{i=1}^n (X_i - Y_i)^2}$$

Huấn luyện:

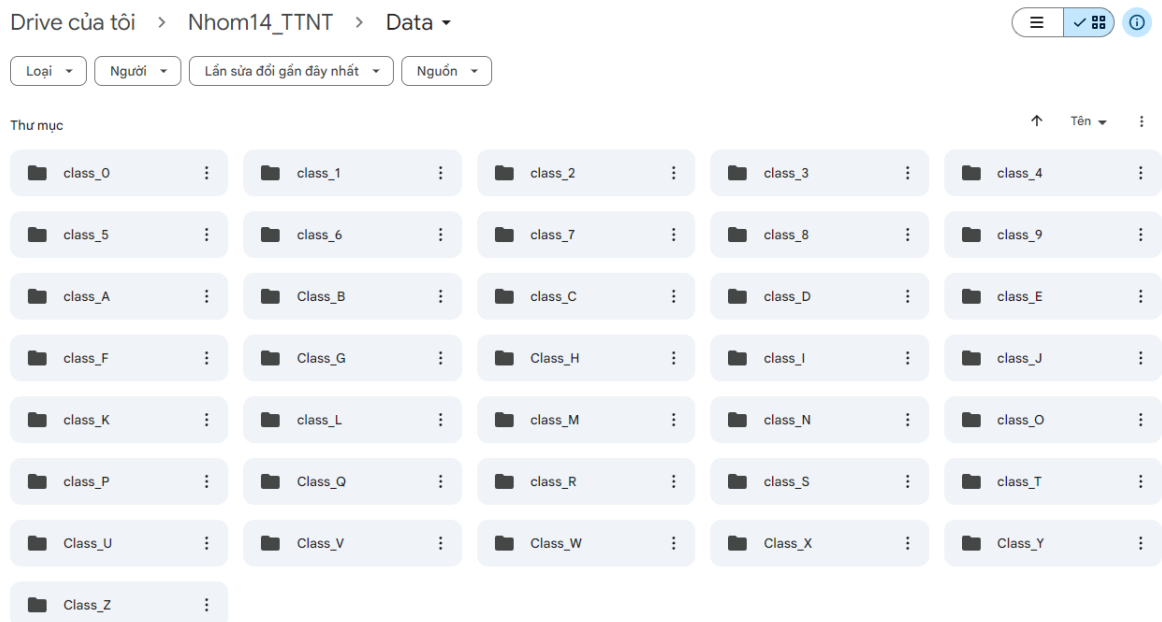
- Lưu trữ toàn bộ tập dữ liệu ký tự huấn luyện (các vector ký tự và nhãn tương ứng).

Dự đoán

- Với mỗi ký tự từ biến số cần nhận diện:
 - + Tính khoảng cách từ ký tự này đến tất cả các mẫu trong tập huấn luyện.
 - + Chọn k ký tự gần nhất.
 - + Phân loại ký tự thành nhãn phổ biến nhất trong k hàng xóm.

3.3.2. Cài đặt chương trình chi tiết

- Tổ chức dữ liệu:



Hình 3.3 Tổ chức dữ liệu

Dữ liệu được tổ chức theo các thư mục data/train/tên thư mục đại diện cho kí tự/ ảnh các kí tự.

- Đọc dữ liệu từ thư mục data và đưa về size 32*32 sau đó chuyển sang màu xám và chia cho 255.0 để các giá trị màu thuộc khoảng 0-1. Lưu ảnh và nhãn vào 2 list images và labels.

```
imagePath = "/content/drive/MyDrive/Nhom14_TTNT/Data/"
#Đọc ảnh và nhãn
images = []
labels = []
classPaths = os.listdir(imagePath)

for classFolder in classPaths:
    classImagePath = os.path.join(imagePath, classFolder)
    label = classFolder[-1]
    for imageName in os.listdir(classImagePath):
        image = cv2.imread(os.path.join(classImagePath, imageName))
        image = cv2.resize(image, (32, 32))
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        image = image / 255.0
        images.append(image)
        labels.append(label)

imageCount = len(images)
images = np.array(images)
images = images.reshape(-1, 32, 32, 1)
labels = np.array(labels)

print(images)
print(labels)
print(f"Number of images: {imageCount}")
print(images.dtype)
print(images.shape)
```

Hình 3.4 Hàm đọc dữ liệu và chỉnh ảnh

- Dữ liệu của images:

```

[[0.          ]
 [0.00392157]
 [0.          ]
 ...
 [0.          ]
 [0.          ]
 [0.          ]]

...

[[0.00392157]
 [0.56078431]
 [0.67843137]
 ...
 [0.68627451]
 [0.55686275]
 [0.00784314]]

[[0.00784314]
 [0.00392157]
 [0.00784314]
 ...
 [0.          ]
 [0.          ]
 [0.00392157]]

[[0.00784314]
 [0.          ]
 [0.          ]
 ...
 [0.00392157]
 [0.          ]
 [0.00392157]]]]

['3' '3' '3' '3' '3' '3' '3' '3' '3' '3' '3' '3' '3' '3' '3' '3'
 '3' '3' '3' '3' '3' '3' '3' '3' '3' '3' '3' '3' '9' '9' '9' '9'
 '9' '9' '9' '9' '9' '9' '9' '9' '9' '9' '9' '9' '9' '9' '9' '9'
 '9' '9' '9' '9' '9' '9' '7' '7' '7' '7' '7' '7' '7' '7' '7' '7'
 '7' '7' '7' '7' '7' '7' '7' '7' '7' '7' '7' '7' '7' '7' '7' '7']

```

Hình 3.5 Dữ liệu của ảnh

- Chia dữ liệu thành tập huấn luyện và tập kiểm tra cho mô hình nơron để cho mô hình nơron có thể nhận biết đặc trưng.

```

# Split data into training and testing sets

imageTrain, imageTest, labelTrain, labelTest = train_test_split(images, newLabels, test_size=0.2, random_state=42)

trainLabelSet = set(labelTrain)
testLabelSet = set(labelTest)
missingLabels = testLabelSet - trainLabelSet

print("Labels in test set but not in training set:", missingLabels)

```

Hình 3.6 Hàm chia dữ liệu thành tập huấn luyện và tập kiểm tra cho mô hình

- Tạo mô hình nơron và train mô hình để nhận dạng đặc trưng.

```
# Create model
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 1)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(imageCount, activation='softmax'))
```

Hình 3.7 Hàm tạo mô hình nơron

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

# Train the model
model.fit(imageTrain, labelTrain, epochs=60, batch_size=42, validation_data=(imageTest, labelTest))
```

14/14 ————— 0s 8ms/step - accuracy: 0.9669 - loss: 0.0860 - val_accuracy: 0.9489 - val_loss: 0.0741
Epoch 33/60
14/14 ————— 0s 8ms/step - accuracy: 0.9482 - loss: 0.1577 - val_accuracy: 0.9708 - val_loss: 0.0551
Epoch 34/60
14/14 ————— 0s 8ms/step - accuracy: 0.9479 - loss: 0.1570 - val_accuracy: 0.9927 - val_loss: 0.0554
Epoch 35/60
14/14 ————— 0s 8ms/step - accuracy: 0.9481 - loss: 0.1230 - val_accuracy: 0.9854 - val_loss: 0.0483
Epoch 36/60
14/14 ————— 0s 8ms/step - accuracy: 0.9682 - loss: 0.0963 - val_accuracy: 0.9489 - val_loss: 0.0752
Epoch 37/60
14/14 ————— 0s 11ms/step - accuracy: 0.9697 - loss: 0.0927 - val_accuracy: 0.9708 - val_loss: 0.0458
Epoch 38/60
14/14 ————— 0s 9ms/step - accuracy: 0.9611 - loss: 0.1448 - val_accuracy: 0.9854 - val_loss: 0.0479
Epoch 39/60
14/14 ————— 0s 7ms/step - accuracy: 0.9578 - loss: 0.1105 - val_accuracy: 0.9635 - val_loss: 0.0581
Epoch 40/60
14/14 ————— 0s 11ms/step - accuracy: 0.9424 - loss: 0.1270 - val_accuracy: 0.9635 - val_loss: 0.0558
Epoch 41/60
14/14 ————— 0s 8ms/step - accuracy: 0.9702 - loss: 0.0945 - val_accuracy: 0.9708 - val_loss: 0.0554
Epoch 42/60
14/14 ————— 0s 8ms/step - accuracy: 0.9583 - loss: 0.1159 - val_accuracy: 0.9635 - val_loss: 0.0514
Epoch 43/60
14/14 ————— 0s 8ms/step - accuracy: 0.9825 - loss: 0.0580 - val_accuracy: 0.9708 - val_loss: 0.0620
Epoch 44/60
14/14 ————— 0s 8ms/step - accuracy: 0.9588 - loss: 0.1148 - val_accuracy: 0.9708 - val_loss: 0.0583
Epoch 45/60
14/14 ————— 0s 8ms/step - accuracy: 0.9541 - loss: 0.1201 - val_accuracy: 0.9562 - val_loss: 0.0566
Epoch 46/60
14/14 ————— 0s 8ms/step - accuracy: 0.9788 - loss: 0.0813 - val_accuracy: 0.9854 - val_loss: 0.0491
Epoch 47/60
14/14 ————— 0s 8ms/step - accuracy: 0.9520 - loss: 0.1013 - val_accuracy: 0.9708 - val_loss: 0.0619

Hình 3.8 Hàm huấn luyện mô hình

- Xem độ chính xác trên mô hình nơron

```
[8] loss, accuracy = model.evaluate(imageTest, labelTest)
    print(f'Test Accuracy: {accuracy:.2f}')
```

```
5/5 ----- 1s 134ms/step - accuracy: 0.9919 - loss: 0.0584
Test Accuracy: 0.99
```

Hình 3.9 Hàm xem độ chính xác trên mô hình

- Lấy đặc trưng từ mô hình nơron để có thể sử dụng trong thuật toán KNN.

```
# Extract features
features = []
for img in images:
    img = cv2.resize(img, (32, 32))
    img = img.astype(np.float32)
    img = img / 255.0
    imgArray = np.expand_dims(img, axis=0)
    imgArray = np.expand_dims(imgArray, axis=-1)
    features.append(model.predict(imgArray))
```

```
1/1 ----- 0s 46ms/step
1/1 ----- 0s 37ms/step
1/1 ----- 0s 34ms/step
1/1 ----- 0s 38ms/step
1/1 ----- 0s 36ms/step
1/1 ----- 0s 43ms/step
1/1 ----- 0s 40ms/step
1/1 ----- 0s 36ms/step
1/1 ----- 0s 39ms/step
1/1 ----- 0s 33ms/step
1/1 ----- 0s 33ms/step
1/1 ----- 0s 42ms/step
1/1 ----- 0s 37ms/step
1/1 ----- 0s 33ms/step
1/1 ----- 0s 37ms/step
1/1 ----- 0s 38ms/step
1/1 ----- 0s 39ms/step
1/1 ----- 0s 36ms/step
1/1 ----- 0s 41ms/step
1/1 ----- 0s 56ms/step
1/1 ----- 0s 48ms/step
1/1 ----- 0s 45ms/step
1/1 ----- 0s 53ms/step
1/1 ----- 0s 44ms/step
```

Hình 3.10 Hàm lấy đặc trưng

- Đặt lại số chiều của features để phù hợp với mô hình KNN

```

▶ print(features)
features = np.array(features)
print(features.dtype)
print(features.shape)

```

Hình 3.11 Hàm đặt lại số chiều

- Dữ liệu của features:

```

0.00084551, 0.00093605, 0.00081284, 0.00084765, 0.00123983,
0.00104774, 0.00087662, 0.00116793, 0.00162454, 0.00154934,
0.00198553, 0.00060204, 0.00078971, 0.00090099, 0.00149714,
0.00130745, 0.00072051, 0.00054663, 0.00065807, 0.00149906,
0.00101794, 0.00073563, 0.00126299, 0.0010252 , 0.00121101,
0.00092507, 0.00158722, 0.00088215, 0.00080701, 0.00072732,
0.00124527, 0.0028363 , 0.00107038, 0.00101544, 0.0003783 ,
0.00084651, 0.00104971, 0.00128045, 0.00047207, 0.00164297,
0.00083394, 0.00215197, 0.00108984, 0.00141992, 0.00123769,
0.00117339, 0.00106854, 0.0004911 , 0.00109558, 0.00079532,
0.00148332, 0.00120244, 0.00096375, 0.0007618 , 0.00048985,
0.00095574, 0.00079002, 0.00111833, 0.00041722, 0.00074258,
0.00069912, 0.00055262, 0.00152265, 0.00077658, 0.00093745,
0.000841 , 0.00075373, 0.00096864, 0.00090228, 0.00091032,
0.00108177, 0.00098279, 0.00098814, 0.00074998, 0.0012297 ,
0.00062944, 0.00123395, 0.0010348 , 0.00051098, 0.00103345,
0.00221001, 0.00144143, 0.0005174 , 0.00065866, 0.00071717,
0.00103634, 0.00062901, 0.00132468, 0.00128297, 0.00138707,
0.00075491, 0.00058628, 0.00072429, 0.00092552, 0.00105943,
0.00064455, 0.00124508, 0.00072363, 0.00067863, 0.00089944,
0.00089852, 0.00090034, 0.00075299, 0.00088299, 0.00084502,
0.0008817 , 0.00093936, 0.00116495, 0.00099548, 0.00072234,
-----

```

Hình 3.12 Dữ liệu của features

- Tạo mô hình KNN và train mô hình KNN để dự đoán cho biến số mới.

```

from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

featuresTrain, featuresTest, labelsKnnTrain, labelsKnnTest = train_test_split(features, newLabels, test_size=0.2, random_state=42)

knnModel = KNeighborsClassifier(n_neighbors=2)
knnModel.fit(featuresTrain, labelsKnnTrain)
yPred = knnModel.predict(featuresTest)

accuracy = accuracy_score(labelsKnnTest, yPred)
print(f"KNN model accuracy: {accuracy * 100:.2f}%")

```

KNN model accuracy: 94.16%

Hình 3.13 Hàm tạo mô hình KNN và train mô hình

- Hàm xử lý ảnh bị nghiêng trước tiên làm mờ ảnh để dễ nhận dạng các cạnh và loại bỏ nhiễu sử dụng Canny để phát hiện các biên sau đó phát hiện đường thẳng và tính toán góc nghiêng cuối cùng trả về ảnh đã xoay về vị trí đúng.

```
def processSkew(img):
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    _, gray = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)
    # Reduce noise
    blurred = cv2.GaussianBlur(gray, (5, 5), 0)

    # Detect edges
    edges = cv2.Canny(blurred, 50, 150)

    # Detect lines
    lines = cv2.HoughLines(edges, 1, np.pi / 180, 150)

    # Calculate skew angle
    angle = 0
    if lines is not None:
        for rho, theta in lines[:, 0]:
            angle = np.degrees(theta) - 90
            break

    print("Skew angle: " + str(angle))

    # Rotate image to correct skew
    (h, w) = img.shape[:2]
    center = (w // 2, h // 2)
    M = cv2.getRotationMatrix2D(center, angle, 1.0)
    rotated = cv2.warpAffine(img, M, (w, h))
    return rotated

skewedImage = cv2.imread("/content/drive/MyDrive/Nhom14_TTNT/anhNghieng.jpg")

rotatedImage = processSkew(skewedImage)
cv2.imshow(skewedImage)
cv2.imshow(rotatedImage)

cv2.waitKey(0)
cv2.destroyAllWindows()
```


Skew angle: 8.999992



Hình 3.14 Hàm xử lý ảnh nghiêng

- Hàm dự đoán kí tự dựa trên thuật toán KNN

```
def testImage(path):  
    img = cv2.imread(path)  
    img = cv2.resize(img, (32, 32))  
    img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)  
    _, img = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY)  
    img = img / 255.0  
    img = np.array(img)  
    imgArray = np.expand_dims(img, axis=0) # Add batch dimension  
    imgArray = np.expand_dims(imgArray, axis=-1)  
    feature = model.predict(imgArray)  
    feature = np.array(feature)  
    feature = feature.reshape(1, -1)  
    prediction = knnModel.predict(feature)  
    return prediction[0]
```

```
def processImage(img):
    img = img / 255.0
    img = np.array(img)
    imgArray = np.expand_dims(img, axis=0)
    imgArray = np.expand_dims(imgArray, axis=-1)
    feature = model.predict(imgArray)
    feature = np.array(feature)
    feature = feature.reshape(1, -1)
    prediction = knnModel.predict(feature)
    return prediction[0]
```

Hình 3.15 Hàm dự đoán kí tự

- Hàm sắp xếp các kí tự dựa trên danh sách các ký tự được cắt có vị trí x,y sau đó sắp xếp theo x và y để giống với thứ tự biên số.

```
def sortContours(contourList):
    contourList = sorted(contourList, key=lambda cnt: cv2.boundingRect(cnt)[1])
    sortedList = []
    currentY = -1
    currentLineContours = []

    for contour in contourList:
        x, y, w, h = cv2.boundingRect(contour)
        if abs(y - currentY) > h:
            currentLineContours = sorted(currentLineContours, key=lambda cnt: cv2.boundingRect(cnt)[0])
            sortedList.extend(currentLineContours)
            currentLineContours = [contour]
            currentY = y
        else:
            currentLineContours.append(contour)
            currentY = y

    currentLineContours = sorted(currentLineContours, key=lambda cnt: cv2.boundingRect(cnt)[0])
    sortedList.extend(currentLineContours)

    return sortedList
```

Hình 3.16 Hàm sắp xếp các kí tự

- Hàm tách biên số, tách ký tự và thực hiện nhận dạng dựa trên hàm test cho từng ký tự được tách sau đó đưa vào danh sách recognizedChars

```

def extractCharsAndProcess(path):
    image = cv2.imread(path)
    image = processSkew(image)
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    blurred = cv2.GaussianBlur(gray, (5, 5), 0)
    edges = cv2.Canny(blurred, 50, 150)

    # Find contours of license plate
    contours, _ = cv2.findContours(edges, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    for contour in contours:
        approx = cv2.approxPolyDP(contour, 0.02 * cv2.arcLength(contour, True), True)
        # Check if contour has 4 points (quadrilateral)
        if len(approx) == 4:
            x, y, w, h = cv2.boundingRect(contour)
            aspectRatio = w / float(h)
            # Condition to identify license plate area
            if 1 < aspectRatio < 2 and w > 200 and h > 20:
                plateImage = image[y:y+h, x:x+w]
                plateGray = cv2.cvtColor(plateImage, cv2.COLOR_BGR2GRAY)
                # Convert to binary image for better character distinction
                _, plateThresh = cv2.threshold(plateGray, 127, 255, cv2.THRESH_BINARY_INV)
                cv2.imshow(plateThresh)

                # Find contours of characters inside license plate
                contours, _ = cv2.findContours(plateThresh, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
                # Sort contours by y then x to maintain reading order
                contours = sortContours(contours)

                for i, charContour in enumerate(contours):
                    x, y, w, h = cv2.boundingRect(charContour)
                    aspectChar = h / float(w)
                    # Conditions to filter valid character contours
                    if 2 < aspectChar < 5 and w > 5 and h > 30:
                        charImage = plateImage[y:y+h, x:x+w]
                        charImage = cv2.resize(charImage, (32, 32))
                        charImage = cv2.cvtColor(charImage, cv2.COLOR_BGR2GRAY)
                        _, charImage = cv2.threshold(charImage, 127, 255, cv2.THRESH_BINARY_INV)
                        print("Coordinates:", x, y, w, h)
                        recognizedChars.append(processImage(charImage))
                    break

    cv2.destroyAllWindows()

```

Hình 3.17 Hàm tách biến số, kí tự

- Hàm chuyển đổi mã hoá chữ ký tự chữ sang số để phù hợp với mô hình nơron

```

def stringToInt(char):
    charToInt = {
        "A": 10, "B": 11, "C": 12, "D": 13, "E": 14, "F": 15, "G": 16, "H": 17,
        "I": 18, "J": 19, "K": 20, "L": 21, "M": 22, "N": 23, "O": 24, "P": 25,
        "Q": 26, "R": 27, "S": 28, "T": 29, "U": 30, "V": 31, "W": 32, "X": 33,
        "Y": 34, "Z": 35
    }
    return charToInt[char]

def convertLabels(labelList):
    convertedLabels = []
    for label in labelList:
        if label.isalpha():
            convertedLabels.append(stringToInt(label))
        else:
            convertedLabels.append(int(label))
    return convertedLabels

newLabels = convertLabels(labels)
newLabels = np.array(newLabels, dtype=int)

print(newLabels)

```

Hình 3.18 Hàm chuyển đổi mã hoá chữ ký tự

- Dữ liệu của label khi chuyển sang số

```

[ 3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3
  3  3  3  3  3  3  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9
  9  9  9  9  9  9  9  9  9  9  9  9  7  7  7  7  7  7  7  7  7  7
  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  2  2  2  2  2  2
  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2
  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6
  6  6  6  6  6  6  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8
  8  8  8  8  8  8  8  8  8  8  8  8  4  4  4  4  4  4  4  4  4  4
  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  5  5  5  5  5
  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5
  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
  1  1  1  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  12 12 12 12 12 12 12 12 12 12
 12 12 12 12 12 12 12 12 12 12 12 12 14 14 14 14 14 14 14 14 14 14
 14 14 14 14 14 14 14 14 14 14 14 14 10 10 10 10 10 10 10 10 10 10
 10 10 10 10 10 10 10 10 10 10 10 10 19 19 19 19 19 19 19 19 19 19
 19 19 19 19 19 19 19 19 19 19 19 19 18 18 18 18 18 18 18 18 18 18
 18 18 18 18 18 18 18 18 18 18 18 18 13 13 13 13 13 13 13 13 13 13
 13 13 13 13 13 13 13 13 13 13 13 13 15 15 15 15 15 15 15 15 15 15
 15 15 15 15 15 15 15 15 15 15 15 15 28 28 28 28 28 28 28 28 28 28

```

Hình 3.19 Hình ảnh của label khi chuyển sang số

- Hàm chuyển đổi ngược lại từ số sang ký tự chữ:

```
def intToString(num):
    intToChar = {
        10: "A", 11: "B", 12: "C", 13: "D", 14: "E", 15: "F", 16: "G", 17: "H",
        18: "I", 19: "J", 20: "K", 21: "L", 22: "M", 23: "N",
        24: "O", 25: "P", 26: "Q", 27: "R", 28: "S",
        29: "T", 30: "U", 31: "V", 32: "W", 33: "X", 34: "Y", 35: "Z"
    }
    return intToChar[num]
```

Hình 3.20 Hàm chuyển đổi ngược lại từ số sang ký tự chữ

- Mã hoá ngược lại từ số sang chữ và đưa ra kết quả.

```
extractCharsAndProcess(imagePath)
for i in range(len(recognizedChars)):
    if recognizedChars[i] > 9:
        recognizedChars[i] = intToString(recognizedChars[i])
    else:
        recognizedChars[i] = str(recognizedChars[i])

print("Recognized license plate characters:")
print(recognizedChars)
```

Hình 3.21 Hàm mã hóa từ số sang chữ

- Thử nghiệm

```

def uploadAndRecognizeImage():
    uploaded = files.upload()
    if not uploaded:
        print("No file uploaded. Exiting.")
        return

    imagePath = list(uploaded.keys())[0]
    print(f"Uploaded image: {imagePath}")
    extractCharsAndProcess(imagePath)
    for i in range(len(recognizedChars)):
        if recognizedChars[i] > 9:
            recognizedChars[i] = intToString(recognizedChars[i])
        else:
            recognizedChars[i] = str(recognizedChars[i])

    print("Recognized license plate characters:")
    print(recognizedChars)

uploadAndRecognizeImage()


```



Hình 3.22 Ảnh ví dụ minh họa

- Kết quả

Skew angle: 0



Coordinates: 36 14 26 64
 1/1 ————— 0s 35ms/step
 Coordinates: 67 13 27 64
 1/1 ————— 0s 34ms/step
 Coordinates: 130 12 27 66
 1/1 ————— 0s 34ms/step
 Coordinates: 166 12 19 65
 1/1 ————— 0s 34ms/step
 Coordinates: 16 89 27 66
 1/1 ————— 0s 42ms/step
 Coordinates: 53 89 28 65
 1/1 ————— 0s 30ms/step
 Coordinates: 92 89 26 66
 1/1 ————— 0s 33ms/step
 Coordinates: 145 89 26 66
 1/1 ————— 0s 32ms/step
 Coordinates: 182 89 27 65
 1/1 ————— 0s 35ms/step
 Recognized license plate characters:
 ['9', '9', 'F', '1', '4', '8', '9', '9', '8']

Hình 3.23 Ảnh kết quả của ví dụ

KẾT LUẬN

Trong báo cáo này, nhóm đã xây dựng thành công một hệ thống nhận dạng biển số xe sử dụng thuật toán K-Nearest Neighbors (KNN) làm phương pháp phân loại chính. Hệ thống hoạt động bằng cách xử lý ảnh đầu vào, tách và chuẩn hóa các ký tự trên biển số, sau đó phân loại các ký tự này dựa trên khoảng cách đến các mẫu đã được huấn luyện trước đó.

Để tăng độ chính xác của mô hình, nhóm đã sử dụng mạng nơ-ron tích chập (CNN) như một bước hỗ trợ để tự động trích xuất đặc trưng hình ảnh từ các ký tự. Việc kết hợp giữa CNN và KNN giúp cải thiện đáng kể chất lượng dữ liệu đầu vào cho mô hình phân loại, từ đó nâng cao hiệu quả nhận dạng, đặc biệt trong các trường hợp ảnh có nhiễu hoặc không rõ nét.

Thử nghiệm thực tế cho thấy hệ thống nhận dạng hoạt động hiệu quả với các biển số rõ ràng và điều kiện ánh sáng ổn định. Tuy nhiên, hệ thống vẫn còn gặp khó khăn trong các tình huống như biển số bị nghiêng, mờ, hoặc các ký tự bị biến dạng. Đây là một trong những hạn chế mà nhóm hướng tới cải thiện trong tương lai.

Trong các bước tiếp theo, nhóm có thể nghiên cứu sử dụng các kiến trúc học sâu tích hợp như CNN kết hợp Softmax, hoặc áp dụng các mô hình hiện đại như CRNN, YOLO để tăng độ chính xác, tốc độ xử lý và khả năng tổng quát hóa trong các môi trường thực tế phức tạp hơn.

Nhìn chung, việc sử dụng KNN trong bài toán nhận dạng ký tự trên biển số xe cho thấy đây là một giải pháp đơn giản, dễ cài đặt, và đạt hiệu quả tốt khi được kết hợp với các kỹ thuật trích đặc trưng phù hợp như CNN.

Tài liệu tham khảo

- [1] Nguyễn Phương Nga, Trần Hùng Cường(2021), Giáo trình Trí tuệ nhân tạo - Trường Đại học Công Nghiệp Hà Nội, NXB Thống Kê.
- [2] Vũ, H. T. (n.d.). *Machine Learning cơ bản – Các thuật toán phổ biến*. Tài liệu học tập trên trang Machine Learning Cơ Bản.
- [3] Raschka, S. (2015). *Python Machine Learning*. Birmingham, UK: Packt Publishing.
- [4] Bipin Kumar. (n.d.). *Lung Cancer Example Dataset*. Dữ liệu thực hành cho thuật toán KNN.
- [5] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21–27.
- [6] Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.