**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH**

**KHOA CÔNG NGHỆ THÔNG TIN**



**KHÓA LUẬN TỐT NGHIỆP**

**XÂY DỰNG HỆ THỐNG TỰ ĐỘNG PHÁT HIỆN VI PHẠM ĐÈN ĐỎ SỬ DỤNG MÔ HÌNH CNN VÀ MOG2**

**Giảng viên hướng dẫn: ThS. VƯƠNG XUÂN CHÍ**

**Sinh viên thực hiện: NGUYỄN ĐÔNG DUY**

**MSSV: 2100009468**

**Khoá: 2021**

**Ngành/ chuyên ngành: CNTT/ TRÍ TUỆ NHÂN TẠO**

**Tp HCM, tháng 5 năm 2024**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH**

**KHOA CÔNG NGHỆ THÔNG TIN**



**KHÓA LUẬN TỐT NGHIỆP**

**XÂY DỰNG HỆ THỐNG TỰ ĐỘNG PHÁT HIỆN VI PHẠM ĐÈN ĐỎ SỬ DỤNG MÔ HÌNH CNN VÀ MOG2**

**Giảng viên hướng dẫn: ThS. VƯƠNG XUÂN CHÍ**

**Sinh viên thực hiện: NGUYỄN ĐÔNG DUY**

**MSSV: 2100009468**

**Khoá: 2021**

**Ngành/ chuyên ngành: CNTT/ TRÍ TUỆ NHÂN TẠO**

**TPHCM, tháng 5 năm 2025**

|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH  **Khoa Công Nghệ Thông Tin**  ---------- | CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM  **Độc lập – Tự do – Hạnh phúc**  🙜 🙜 🙝 🙝 |

# NHIỆM VỤ KHÓA LUẬN TỐT NGHIỆP

*(Sinh viên phải đóng tờ này vào báo cáo)*

**Họ và tên:** Nguyễn Đông Duy  **MSSV:** 2100009468

**Chuyên ngành:** Trí tuệ nhân tạo **Lớp:** 21DTH2C

**Email:** 2100009468@nttu.edu.vn  **SĐT:** 0906704631

Tên đề tài: Hệ thống tự động phát hiện vi phạm đèn đỏ bằng CNN và MOG2

Giáo viên hướng dẫn: ThS. Vương Xuân Chí

Thời gian thực hiện: 4/3/2025 đến .../5/2025

Nhiệm vụ/nội dung (mô tả chi tiết nội dung, yêu cầu, phương pháp…):

**NỘI DUNG ĐỀ TÀI:**

Xây dựng và phát triển một hệ thống tự động phát hiện phương tiện giao thông vi phạm đèn đỏ, tích hợp các tính năng phát hiện trạng thái đèn giao thông, theo dõi phương tiện và nhận dạng biển số xe.

Hệ thống sẽ phân tích video từ camera tại giao lộ hoặc video có sẵn, đồng thời cho phép lưu trữ dữ liệu vi phạm vào cơ sở dữ liệu SQL Server để tiện quản lý và truy xuất.

**NHIỆM VỤ:**

* Nghiên cứu cách xây dựng và huấn luyện mô hình CNN cho việc phân loại trạng thái đèn giao thông.
* Tìm hiểu các thuật toán Background Subtraction, đặc biệt là MOG2, cho việc phát hiện và theo dõi phương tiện.
* Tìm hiểu về các kỹ thuật xử lý ảnh để cải thiện hiệu suất hệ thống
* Phát hiện và trích xuất contour biển số xe.
* Tích hợp Tesseract OCR để nhận dạng biển số xe.
* Kết nối và lưu trữ dữ liệu vi phạm vào SQL Server.
* Phát triển giao diện quản lý vi phạm bằng ASP.NET
* Đánh giá hiệu suất và tối ưu hóa độ chính xác của việc phát hiện vi phạm.
* Đánh giá hiệu suất nhận dạng biển số xe và tối ưu hóa thời gian xử lý.

**YÊU CẦU KỸ THUẬT:**

* Ngôn ngữ lập trình: Python
* Mô hình và thuật toán:
* CNN để phân loại trạng thái đèn giao thông
* Background Subtraction (MOG2) cho phát hiện và theo dõi phương tiện
* Tesseract OCR cho nhận dạng biển số xe
* Xử lý ảnh bằng các thư viện: OpenCV, NumPy, scikit-image
* Xây dựng và lưu trữ: PyODBC, SQL Server, ASP.NET.

**YÊU CẦU CHỨC NĂNG:**

* Phát hiện và phân loại trạng thái đèn giao thông (đỏ, vàng, xanh)
* Xác định vi phạm vượt đèn đỏ dựa trên quỹ đạo phương tiện
* Trích xuất và nhận dạng biển số xe vi phạm
* Lưu trữ thông tin chi tiết về vi phạm vào cơ sở dữ liệu
* Giao diện quản lý và truy xuất thông tin vi phạm

**PHƯƠNG PHÁP:**

* Phân tích và Thiết kế Hệ thống
* Xử lý Ảnh và Phát hiện Đối tượng
* Phân loại Đèn Giao thông
* Phát hiện và Nhận dạng Biển số xe
* Lưu trữ và Quản lý Dữ liệu
* Triển khai Giao diện và Tương tác Người dùng

**Nội dung và yêu cầu đã được thông qua Bộ môn.**

*TP.HCM, ngày … tháng 5 năm 2025*

|  |  |
| --- | --- |
| **TRƯỞNG BỘ MÔN**  *(Ký và ghi rõ họ tên)* | **GIÁO VIÊN HƯỚNG DẪN**  *(Ký và ghi rõ họ tên)* |

**ThS. Vương Xuân Chí ThS. Vương Xuân Chí**

# LỜI MỞ ĐẦU

Vi phạm an toàn giao thông là một trong những vấn đề quan trọng, có ảnh hưởng lớn đến đời sống xã hội và phát triển kinh tế đất nước. Tuy nhiên, tình trạng vi phạm đèn đỏ tại các giao lộ, vẫn diễn ra phổ biến và gây ra nhiều hậu quả nghiêm trọng. Việc phát hiện và xử lý kịp thời các hành vi vi phạm không chỉ giúp đảm bảo trật tự, an toàn giao thông mà còn nâng cao ý thức chấp hành pháp luật của người tham gia giao thông.

Trong khóa luận này, em ứng dụng các kỹ thuật thị giác máy tính và học sâu để xây dựng hệ thống tự động phát hiện vi phạm đèn đỏ. Hệ thống được thiết kế dựa trên hai công nghệ chính: mạng nơ-ron tích chập (CNN) để phát hiện và phân loại trạng thái đèn giao thông, và phương pháp trừ nền (Background Subtraction) để phát hiện và theo dõi phương tiện tham gia giao thông.

Đề tài "Xây dựng hệ thống tự động phát hiện vi phạm đèn đỏ dựa trên mô hình CNN và Background Subtraction" được lựa chọn nhằm tạo ra một giải pháp công nghệ có khả năng hoạt động liên tục, chính xác và hiệu quả trong việc giám sát giao thông. Đề tài không chỉ tập trung vào nghiên cứu lý thuyết mà còn phát triển hệ thống hoàn chỉnh có thể triển khai trong điều kiện thực tế tại Việt Nam, tích hợp các chức năng như nhận dạng biển số xe để xác định thông tin phương tiện vi phạm.

Hy vọng rằng kết quả của nghiên cứu này sẽ mang lại những đóng góp hữu ích cho công tác quản lý giao thông, góp phần giảm thiểu tai nạn, đảm bảo an toàn giao thông và mở ra hướng phát triển mới trong ứng dụng trí tuệ nhân tạo vào hệ thống giao thông thông minh trong tương lai.

# LỜI CẢM ƠN

Đầu tiên, em xin bày tỏ lòng biết ơn chân thành đến Thầy ThS. Vương Xuân Chí, người đã tận tình hướng dẫn và chia sẻ những kiến thức quý báu trong suốt quá trình em thực hiện khóa luận tốt nghiệp. Thầy không chỉ truyền đạt kiến thức chuyên môn mà còn định hướng và khơi dậy tinh thần học hỏi, giúp em có được những bước tiến vững chắc trên con đường nghiên cứu.

Bên cạnh đó, em xin gửi lời cảm ơn sâu sắc đến các thầy cô trong Khoa Công nghệ Thông tin, Trường Đại học Nguyễn Tất Thành, những người đã xây dựng một nền tảng tri thức vững chắc và tạo ra môi trường học tập đầy cảm hứng. Chính nhờ những nỗ lực của các thầy cô, em đã có cơ hội tiếp cận và ứng dụng kiến thức vào thực tiễn, từng bước hoàn thiện năng lực bản thân.

Quá trình thực hiện đề tài này không tránh khỏi những thiếu sót, và em mong nhận được những góp ý quý báu từ các thầy cô để rút kinh nghiệm và hoàn thiện bản thân hơn nữa.

Em xin kính chúc Thầy Vương Xuân Chí cùng toàn thể thầy cô trong khoa sức khỏe dồi dào, niềm vui và thành công trên hành trình đào tạo và nghiên cứu. Những đóng góp to lớn của thầy cô sẽ luôn là động lực và niềm cảm hứng để các thế hệ sinh viên tiếp tục cố gắng và phát triển.

**Sinh viên ký tên**

(Ký ghi rõ họ, tên)

Nguyễn Đông Duy

# NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

1. Hình thức (Bố cục, trình bày, lỗi, các mục, hình, bảng, công thức, phụ lục…)

2. Nội dung (mục tiêu, phương pháp, kết quả, sao chép, các chương, tài liệu,..)

3. Kết luận

*TPHCM, Ngày … tháng 5 năm 2025*

**Giáo viên hướng dẫn**

(Ký tên,ghi rõ họ tên)

ThS. Vương Xuân Chí

**NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN**

1. Hình thức (Bố cục, trình bày, lỗi, các mục, hình, bảng, công thức, phụ lục…)

2. Nội dung (mục tiêu, phương pháp, kết quả, sao chép, các chương, tài liệu,..)

3. Kết luận

*TPHCM, Ngày … tháng 5 năm 2025*

**Giáo viên phản biện**

(Ký tên,ghi rõ họ tên)

**MỤC LỤC**

[NHIỆM VỤ KHÓA LUẬN TỐT NGHIỆP 1](#_Toc197591924)

[LỜI MỞ ĐẦU 3](#_Toc197591925)

[LỜI CẢM ƠN 4](#_Toc197591926)

[NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN 5](#_Toc197591927)

[CHƯƠNG 1: GIỚI THIỆU 13](#_Toc197591928)

[1.1 Lý do chọn đề tài 13](#_Toc197591929)

[1.2 Mục tiêu nghiên cứu 13](#_Toc197591930)

[1.3 Phương pháp nghiên cứu 14](#_Toc197591931)

[1.3.1 Nghiên cứu lý thuyết và nền tảng 14](#_Toc197591932)

[1.3.2 Phương pháp nghiên cứu thực nghiệm 14](#_Toc197591933)

[1.4 Đối tượng nghiên cứu 14](#_Toc197591934)

[1.5 Phạm vi nghiên cứu 14](#_Toc197591935)

[1.6 Cấu trúc của khóa luận 15](#_Toc197591936)

[CHƯƠNG 2: 16](#_Toc197591937)

[CƠ SỞ LÝ THUYẾT VÀ TỔNG QUAN NGHIÊN CỨU 16](#_Toc197591938)

[2.1 Tổng quan về hệ thống phát hiện vi phạm giao thông 16](#_Toc197591939)

[2.2 Các phương pháp xử lý ảnh trong phát hiện vi phạm giao thông 16](#_Toc197591940)

[2.3 Mô hình CNN và Background Subtraction trong hệ thống 16](#_Toc197591941)

[2.4 Nhận dạng biển số xe với Tesseract OCR 17](#_Toc197591942)

[2.4.1 Phát hiện và trích xuất biển số 17](#_Toc197591943)

[2.4.2 Nhận dạng ký tự với Tesseract OCR 17](#_Toc197591944)

[2.5 Công nghệ lưu trữ dữ liệu với SQL Server 17](#_Toc197591945)

[2.6 Cơ sở lý thuyết về thị giác máy tính 18](#_Toc197591946)

[2.6.1. Tổng quan về thị giác máy tính 18](#_Toc197591947)

[2.6.2. Xử lý ảnh số 19](#_Toc197591948)

[2.6.3. Phát hiện và trích xuất đặc trưng 20](#_Toc197591949)

[2.6.4. Các phép toán hình thái học 20](#_Toc197591950)

[2.7. Mạng nơ-ron tích chập (CNN) 20](#_Toc197591951)

[2.7.1. Kiến trúc cơ bản của CNN 20](#_Toc197591952)

[2.7.2. Cơ chế hoạt động của CNN 22](#_Toc197591953)

[2.7.3. Các kỹ thuật nâng cao trong CNN 24](#_Toc197591954)

[2.7.4. Các kiến trúc CNN hiện đại 25](#_Toc197591955)

[2.8 Thuật toán Mixture of Gaussians v2 (MOG2) 27](#_Toc197591956)

[2.8.1. Tổng quan về phương pháp khử nền 27](#_Toc197591957)

[2.8.2. Các phương pháp Background Subtraction 28](#_Toc197591958)

[2.8.3 Các tham số của thuật toán MOG2 31](#_Toc197591959)

[2.9 Công nghệ nhận dạng ký tự quang học (OCR) 33](#_Toc197591960)

[2.9.1 Tổng quan về OCR 33](#_Toc197591961)

[2.4.2. Tesseract OCR 34](#_Toc197591962)

[2.9.3. Cấu hình và tối ưu Tesseract OCR 35](#_Toc197591963)

[2.9.4 Hậu xử lý kết quả OCR 35](#_Toc197591964)

[CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 37](#_Toc197591965)

[3.1. Phân tích bài toán 37](#_Toc197591966)

[3.1.1. Mô tả bài toán 37](#_Toc197591967)

[3.1.2. Các thách thức kỹ thuật 38](#_Toc197591968)

[3.1.3. Phân tích yêu cầu dữ liệu 40](#_Toc197591969)

[3.2. Yêu cầu hệ thống 43](#_Toc197591970)

[3.3. Kiến trúc tổng thể hệ thống 46](#_Toc197591971)

[3.3.1. Mô hình kiến trúc 46](#_Toc197591972)

[3.4 Thiết kế chi tiết các module 49](#_Toc197591973)

[3.4.1 Module xử lý video đầu vào (Video Input Handler): 49](#_Toc197591974)

[3.4.2 Module phát hiện đèn giao thông (Traffic Light Detector) 49](#_Toc197591975)

[3.4.3 Quy trình chuẩn bị dữ liệu và huấn luyện Thu thập và tiền xử lý dữ liệu: 50](#_Toc197591976)

[3.4.4 Module phát hiện phương tiện và theo dõi phương tiện (Vehicle Detector) 50](#_Toc197591977)

[3.4.5 Module phát hiện vi phạm (Violation Detector) 51](#_Toc197591978)

[3.4.6 Module nhận dạng biển số (License Plate Recognition) 53](#_Toc197591979)

[3.4.7 Module lưu trữ dữ liệu (Data Storage) 55](#_Toc197591980)

[3.5 Sơ đồ tổng chi tiết hệ thống 57](#_Toc197591981)

[3.5.1 Sơ đồ kiến trúc của bộ xử lý chính 57](#_Toc197591982)

[3.5.2 Quy trình xử lý của toàn bộ hệ thống: 58](#_Toc197591983)

[3.6 Thiết kế giao diện diện web người dùng 58](#_Toc197591984)

[CHƯƠNG 4: TRIỂN KHAI VÀ KẾT QUẢ ĐẠT ĐƯỢC 61](#_Toc197591985)

[4.1 Thu thập và chuẩn bị dữ liệu 61](#_Toc197591986)

[4.1.1 Nguồn dữ liệu 61](#_Toc197591987)

[4.1.1 Chuẩn bị video thử nghiệm 61](#_Toc197591988)

[4.2 huấn luyện mô hình CNN 61](#_Toc197591989)

[4.2.1 cấu trúc mô hình CNN 61](#_Toc197591990)

[4.2.2 Huấn luyện mô hình và kết quả 62](#_Toc197591991)

[4.5 Triển khai module nhận dạng biển số 65](#_Toc197591992)

[4.6 Triển khai module phát hiện vi phạm 67](#_Toc197591993)

[4.8. Triển khai module lưu trữ dữ liệu 67](#_Toc197591994)

[4.9. Kết quả đạt được 68](#_Toc197591995)

[CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 70](#_Toc197591996)

[5.1 Tổng kết kết quả đạt được 70](#_Toc197591997)

[5.1.1. Đánh giá tổng quan 70](#_Toc197591998)

[5.1.2 Kết quả cụ thể 70](#_Toc197591999)

[5.1.3 Ưu điểm của hệ thống 71](#_Toc197592000)

[5.2. Hạn chế và thách thức 71](#_Toc197592001)

[5.2.1. Hạn chế về kỹ thuật 71](#_Toc197592002)

[5.2.2. Thách thức triển khai thực tế 71](#_Toc197592003)

[5.3. Hướng phát triển tương lai 72](#_Toc197592004)

[5.3.1. Cải thiện mô hình và thuật toán 72](#_Toc197592005)

[5.3.2. Mở rộng tính năng 72](#_Toc197592006)

[5.3.3. Triển khai quy mô lớn 73](#_Toc197592007)

[5.3.4. Nghiên cứu và phát triển 73](#_Toc197592008)

[5.4. Kết luận 74](#_Toc197592009)

**DANH MỤC BẢNG HÌNH**

[Hình 3.1: Sơ đồ tổng quát hệ thống 46](#_Toc197591875)

[Hình 3-2: Chi tiết quy trình của bộ xử lý chính 55](#_Toc197591876)

[Hình 3-3: Quy trình của hệ thống 56](#_Toc197591877)

[Hình 3-4: giao diện chính trang web 57](#_Toc197591878)

[Hình 3-5: Hiển thị kết quả trên giao diện 58](#_Toc197591879)

[Hình 4-1: kết quả huấn luyện đạt được 60](#_Toc197591880)

[Hình 4-2: Hình báo cáo phân loại 61](#_Toc197591881)

[Hình 4-3: Ma trận nhầm lẫn 61](#_Toc197591882)

[Hình 4-4: Kết quả của MOG2 63](#_Toc197591883)

[Hình 4-5: Trực quan mối liên hệ giữa giá và diện tích 63](#_Toc197591884)

[Hình 4-6: Kết quả nhận diện và cắt biển số xe 64](#_Toc197591885)

[Hình 4-9: Kết quả gửi về Database 66](#_Toc197591886)

[Hình 4-8: Trang chủ của web 66](#_Toc197591887)

[Hình 4-9: Danh sách vi phạm 67](#_Toc197591888)

**DANH MỤC HÌNH**

[Hình 3.1: Sơ đồ tổng quát hệ thống 47](#_Toc197277870)

[Hình 3-3: giao diện chính trang web 56](#_Toc197277871)

[Hình 3-4: Hiển thị kết quả trên giao diện 57](#_Toc197277872)

[Hình 4-1: kết quả huấn luyện đạt được 60](#_Toc197277873)

[Hình 4-2: Hình báo cáo phân loại 61](#_Toc197277874)

[Hình 4-3: Ma trận nhầm lẫn 61](#_Toc197277875)

[Hình 4-4: Kết quả của MOG2 63](#_Toc197277876)

[Hình 4.5: Trực quan mối liên hệ giữa giá và diện tích 63](#_Toc197277877)

[Hình 4-6: Kết quả nhận diện và cắt biển số xe 64](#_Toc197277878)

[Hình 4-9: Kết quả gửi về Database 66](#_Toc197277879)

[Hình 4-8: Trang chủ của web 67](#_Toc197277880)

[Hình 4-9: Danh sách vi phạm 67](#_Toc197277881)

**KÍ HIỆU CÁC CỤM TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| Chữ viết tắt | Ý nghĩa |
| AI | Artificial Intelligence |
| CNN | Convolutional Neural Network |
| OCR | Optical Character Recognition |
| MOG2 | Mixture of Gaussians 2 |
| ROI | Region of Interest |
| RGB | Đỏ Lục Lam |
| HSV | Hue Saturation Value |
| FPS | Frames Per Second |
| GPU | Graphics Processing Unit |

# CHƯƠNG 1: GIỚI THIỆU

1.1 Lý do chọn đề tài

Vi phạm đèn đỏ là một trong những nguyên nhân chính gây ra tai nạn giao thông nghiêm trọng tại các giao lộ. Theo số liệu thống kê, tai nạn do vượt đèn đỏ thường dẫn đến hậu quả nghiêm trọng hơn so với các loại vi phạm khác, với tỷ lệ tử vong và thương tích cao. Tại Việt Nam, tình trạng này ngày càng phổ biến, đặc biệt tại các đô thị lớn, gây áp lực lớn cho công tác quản lý an toàn giao thông.

Trong đó, phương pháp giám sát truyền thống dựa vào lực lượng cảnh sát giao thông trực tiếp có nhiều hạn chế về nhân lực, không thể hiện diện 24/7 tại mọi giao lộ, dẫn đến khó khăn trong việc xử lý triệt để các hành vi vi phạm. Mặt khác, các hệ thống camera giám sát hiện có thường chỉ ghi lại hình ảnh mà chưa có khả năng tự động phát hiện vi phạm và nhận dạng phương tiện.

Với việc chọn đề tài này nhằm mục tiêu góp phần phát triển hệ thống phát hiện vi phạm đèn đỏ tự động, chính xác, phù hợp để ứng dụng trong việc giảm thiểu tai nạn giao thông. Việc xây dựng một hệ thống tự động phát hiện vi phạm đèn đỏ trở nên khả thi và hiệu quả. Kết hợp CNN và Background Subtraction mang lại giải pháp có tính ứng dụng cao, chi phí hợp lý và dễ triển khai, phù hợp với điều kiện thực tế tại Việt Nam. Hệ thống không chỉ giúp cải thiện hiệu quả giám sát giao thông mà còn góp phần nâng cao ý thức chấp hành luật giao thông của người tham gia.

1.2 Mục tiêu nghiên cứu

Nghiên cứu và xây dựng một hệ thống tự động phát hiện vi phạm đèn đỏ dựa trên mô hình CNN và MOG2, có khả năng nhận dạng biển số xe vi phạm, đảm bảo hiệu suất thời gian thực và độ chính xác cao trong điều kiện thực tế. Cụ thể, các mục tiêu bao gồm:

* Xây dựng và huấn luyện mô hình CNN có khả năng phát hiện và phân loại chính xác trạng thái đèn giao thông (đỏ, vàng, xanh).
* Triển khai MOG2 để phát hiện và theo dõi hiệu quả các phương tiện di chuyển trong video giao thông, bao gồm cả trong điều kiện có nhiều phương tiện, ánh sáng thay đổi và bóng đổ.
* Xây dựng thuật toán phát hiện vi phạm vượt đèn đỏ dựa trên sự kết hợp giữa trạng thái đèn và quỹ đạo phương tiện.
* Phát triển thuật toán trích xuất và nhận dạng biển số xe với độ chính xác với OCR.
* Xây dựng Website với ASP.NET MVC và SQL server 2022 làm hệ thống lưu trữ và quản lý thông tin vi phạm, cho phép truy xuất và thống kê dữ liệu.

1.3 Phương pháp nghiên cứu

1.3.1 Nghiên cứu lý thuyết và nền tảng

Nghiên cứu và tổng hợp các công trình nghiên cứu, bài báo khoa học, sách và tài liệu liên quan đến lĩnh vực thị giác máy tính, CNN, MOG2, OCR và các hệ thống giám sát giao thông tự động.

Nghiên cứu cấu trúc và nguyên lý hoạt động của CNN và Background Subtraction.

Phân tích các phương pháp phát hiện phương tiện (Background Subtraction, Haar cascade, HOG+SVM).

Phân tích thuật toán MOG2 (Mixture of Gaussians) và các tham số quan trọng.

Dữ liệu sử dụng là tập dữ liệu “trafficlight Computer Vision Project” trên trang Roboflow bao gồm 5000 ảnh và được chia cho 3 tệp đèn đỏ, đèn vàng và đèn xanh.

1.3.2 Phương pháp nghiên cứu thực nghiệm

Thu thập dữ liệu video giao thông tại các giao lộ có đèn tín hiệu trong nhiều điều kiện ánh sáng (sáng, chiều, trời nắng, trời âm u) để đảm bảo tính đa dạng.

Sử dụng và Triển khai thuật toán MOG2 với các tham số khác nhau (history, varThreshold, detectShadows) để xác định cấu hình tối ưu.

Sử dụng tập dữ liệu “trafficlight Computer Vision Project” trên trang Roboflow cho việc huấn luyện mô hình CNN phân loại đèn giao thông.

Triển khai và phát triển thuật toán phát hiện biển số dựa trên phân tích contour, tỷ lệ kích thước và đặc trưng hình dạng (biển số thường có tỷ lệ chiều rộng/chiều cao đặc trưng).

Đánh giá độ ổn định của hệ thống trong nhiều điều kiện ánh sáng và mật độ giao thông khác nhau.

1.4 Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài này sẽ tập trung vào đèn tín hiệu giao thông như trạng thái màu đèn tín hiệu. Tiếp theo là phương tiện giao thông đặc điểm chuyển động và quỹ đạo của phương tiện.

Đặc điểm và mẫu hình của hành vi vượt đèn đỏ tại giao lộ và Vị trí, hình dạng, màu sắc và các ký tự trên biển số của các phương tiện.

1.5 Phạm vi nghiên cứu

Áp dụng cho camera cố định được lắp đặt tại các ngã tư, có góc nhìn bao quát được vạch dừng, đèn giao thông và một phần của giao lộ

Hệ thống được thiết kế để hoạt động trong điều kiện ánh sáng ban ngày, chưa tối ưu hóa cho điều kiện ánh sáng yếu hoặc ban đêm.

Hệ thống có khả năng xử lý cả video trực tiếp (real-time) từ camera giám sát và video đã được ghi lại trước đó.

Sử dụng mô hình CNN cho phân loại đèn giao thông và sử dụng Background Subtraction cho phát hiện phương tiện thay vì các mô hình học sâu phức tạp hơn.

Đông thời Sử dụng Tesseract OCR cho nhận dạng biển số thay vì các phương pháp học sâu chuyên biệt.

1.6 Cấu trúc của khóa luận

Chương 1: Giới thiệu đề tài

Chương 2: Cơ sở lý thuyết

Chương 3: Phân tích và thiết kế hệ thống

Chương 4: Triển khai và kết quả đạt được

Chương 5: Kết luận và hướng phát triển

# CHƯƠNG 2:

# CƠ SỞ LÝ THUYẾT VÀ TỔNG QUAN NGHIÊN CỨU

2.1 Tổng quan về hệ thống phát hiện vi phạm giao thông

Phát hiện vi phạm giao thông tự động là quá trình sử dụng các kỹ thuật thị giác máy tính và học máy để nhận diện, theo dõi và ghi lại các hành vi vi phạm luật giao thông mà không cần sự can thiệp của con người. Đây là một lĩnh vực quan trọng trong ứng dụng trí tuệ nhân tạo (AI) vào quản lý giao thông thông minh, góp phần nâng cao an toàn và giảm thiểu tai nạn tại các giao lộ.

Các nghiên cứu trong lĩnh vực này đã chỉ ra hiệu quả của kết hợp các phương pháp xử lý ảnh truyền thống với các mô hình học sâu, đặc biệt là mạng nơ-ron tích chập (CNN) và thuật toán Background Subtraction (MOG2) được áp dụng để phát hiện và theo dõi phương tiện. Kết hợp với module nhận dạng biển số xe sử dụng Tesseract OCR, hệ thống có thể xác định được chính xác thông tin phương tiện vi phạm và lưu trữ bằng chứng vào cơ sở dữ liệu SQL Server.

2.2 Các phương pháp xử lý ảnh trong phát hiện vi phạm giao thông

Việc phát hiện vi phạm giao thông chính xác phụ thuộc rất nhiều vào chất lượng xử lý ảnh. Các kỹ thuật tiền xử lý ảnh chính bao gồm:

Cân bằng histogram và CLAHE: Được áp dụng để cải thiện độ tương phản của ảnh, đặc biệt quan trọng khi xử lý hình ảnh biển số xe trong điều kiện ánh sáng không đồng đều.

Gaussian và Bilateral Filtering: Được sử dụng để giảm nhiễu nhưng vẫn bảo toàn các cạnh quan trọng, giúp tăng hiệu quả trong việc phát hiện contour biển số xe.

Phép toán hình thái học: Các phép toán như erosion, dilation, opening và closing được áp dụng để lọc nhiễu sau quá trình Background Subtraction, giúp phân tách rõ ràng hơn phương tiện khỏi nền.

Canny Edge Detection: Thuật toán này được sử dụng để phát hiện cạnh trong quá trình xác định biển số xe, với các ngưỡng được điều chỉnh dựa trên giá trị trung vị của ảnh đầu vào.

2.3 Mô hình CNN và Background Subtraction trong hệ thống

Mạng nơ-ron tích chập (CNN), là một kiến trúc học sâu được tối ưu hóa cho xử lý dữ liệu ảnh, đặc biệt hiệu quả trong việc phân loại và phát hiện đối tượng. Trong hệ thống phát hiện vi phạm đèn đỏ, CNN được ứng dụng để phân loại trạng thái đèn giao thông (đỏ, vàng, xanh) với độ chính xác cao, ngay cả trong điều kiện ánh sáng thay đổi. Cấu trúc đa lớp của CNN giúp trích xuất các đặc trưng phức tạp từ đơn giản đến phức tạp thông qua các lớp tích chập, gộp và kết nối đầy đủ.

Background Subtraction là kỹ thuật phân tách nền tĩnh và đối tượng chuyển động trong video, đặc biệt hiệu quả trong phát hiện và theo dõi phương tiện di chuyển qua giao lộ. Thuật toán MOG2 (Mixture of Gaussians) mô hình hóa mỗi pixel bằng nhiều phân phối Gaussian, cho phép thích ứng với sự thay đổi điều kiện ánh sáng và chuyển động nhỏ trong cảnh nền.

Việc kết hợp CNN để phân loại đèn giao thông và Background Subtraction để phát hiện phương tiện tạo nên một giải pháp mạnh mẽ cho hệ thống phát hiện vi phạm đèn đỏ, cân bằng giữa độ chính xác và hiệu suất xử lý thời gian thực.

2.4 Nhận dạng biển số xe với Tesseract OCR

2.4.1 Phát hiện và trích xuất biển số

Quy trình thực hiện quá trình này với các bước:

* Chuyển đổi ảnh sang không gian màu grayscale
* Phát hiện màu sắc chủ đạo của biển số (trắng hoặc vàng) để điều chỉnh tiền xử lý phù hợp
* Áp dụng CLAHE và Bilateral Filtering để tăng cường chất lượng ảnh
* Sử dụng Canny Edge Detection để phát hiện cạnh
* Áp dụng phép toán closing để kết nối các đường viền
* Phát hiện contour và lọc theo tiêu chí hình dạng, kích thước và tỷ lệ
* Cắt và trả về vùng ảnh chứa biển số

2.4.2 Nhận dạng ký tự với Tesseract OCR

triển khai quá trình OCR với Tesseract:

* Tiền xử lý ảnh biển số (thay đổi kích thước, lọc nhiễu)
* Cấu hình Tesseract với tham số phù hợp cho biển số Việt Nam (whitelist, PSM mode)
* Áp dụng nhiều phương pháp tiền xử lý khác nhau và chọn kết quả tốt nhất
* Hậu xử lý văn bản được nhận dạng để chuẩn hóa định dạng biển số

2.5 Công nghệ lưu trữ dữ liệu với SQL Server

SQL Server là hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) mạnh mẽ, được thiết kế để lưu trữ dữ liệu có cấu trúc với tính toàn vẹn cao. Với khả năng quản lý hiệu quả các giao dịch và bảo mật dữ liệu, SQL Server là lựa chọn phù hợp để lưu trữ thông tin vi phạm giao thông, bao gồm thời gian, địa điểm, biển số xe và hình ảnh vi phạm.

SQL Server cung cấp tính năng FILESTREAM cho phép lưu trữ hiệu quả các tập tin nhị phân lớn như hình ảnh và video vi phạm, đồng thời vẫn duy trì lợi ích của hệ thống quản lý cơ sở dữ liệu quan hệ. Tính năng này đặc biệt hữu ích trong hệ thống phát hiện vi phạm đèn đỏ, nơi cần lưu trữ cả dữ liệu cấu trúc (thông tin vi phạm) và dữ liệu phi cấu trúc (hình ảnh vi phạm) một cách đồng bộ.

Ngoài ra, SQL Server tích hợp tốt với hệ sinh thái Microsoft, bao gồm ASP.NET và các công cụ báo cáo, tạo điều kiện thuận lợi cho việc phát triển giao diện quản lý và hệ thống báo cáo vi phạm giao thông.

2.6 Cơ sở lý thuyết về thị giác máy tính

2.6.1. Tổng quan về thị giác máy tính

Thị giác máy tính (Computer Vision) là lĩnh vực nghiên cứu nhằm phát triển các kỹ thuật giúp máy tính có khả năng "nhìn" và "hiểu" nội dung hình ảnh hay video tương tự như con người. Khác với xử lý ảnh đơn thuần, thị giác máy tính không chỉ thao tác trên dữ liệu hình ảnh mà còn hướng đến việc trích xuất thông tin có ngữ nghĩa cao cấp từ dữ liệu đó.

Phạm vi của thị giác máy tính bao gồm nhiều lĩnh vực từ cơ bản đến nâng cao như: nhận dạng đối tượng, phân loại hình ảnh, phát hiện chuyển động, theo dõi đối tượng, khôi phục hình ảnh 3D từ hình ảnh 2D. Lĩnh vực này đã đạt được những bước tiến vượt bậc trong thập kỷ gần đây nhờ vào sự phát triển của học sâu (Deep Learning) và sự gia tăng về khả năng tính toán.

Các bài toán cơ bản trong thị giác máy tính có thể phân loại thành:

Phân loại (Classification): Xác định một hình ảnh thuộc về lớp nào trong một tập hợp các lớp định trước. Ví dụ: xác định màu của đèn tín hiệu giao thông (đỏ, vàng, xanh).

Phát hiện (Detection): Xác định vị trí và lớp của các đối tượng trong hình ảnh. Ví dụ: phát hiện các phương tiện giao thông trong cảnh giao lộ.

Phân đoạn (Segmentation): Phân chia hình ảnh thành các vùng có ý nghĩa, có thể ở mức pixel (semantic segmentation) hoặc mức đối tượng (instance segmentation). Ví dụ: tách biệt vùng đường đi, vỉa hè, phương tiện, v.v.

Theo dõi (Tracking): Theo dõi vị trí và trạng thái của đối tượng qua thời gian trong video. Ví dụ: theo dõi quỹ đạo di chuyển của xe qua giao lộ.

Trong hệ thống phát hiện vi phạm đèn đỏ, chúng ta sẽ kết hợp nhiều kỹ thuật thị giác máy tính để giải quyết các bài toán con như phát hiện trạng thái đèn giao thông, theo dõi phương tiện, và phát hiện hành vi vi phạm.

2.6.2. Xử lý ảnh số

Xử lý ảnh số là nền tảng cơ bản của thị giác máy tính, tập trung vào việc biến đổi hình ảnh thành dạng phù hợp hơn cho các bước phân tích tiếp theo. Trong phát triển hệ thống phát hiện vi phạm đèn đỏ, xử lý ảnh đóng vai trò quan trọng trong việc cải thiện chất lượng dữ liệu đầu vào và chuẩn bị cho các thuật toán phức tạp hơn.

Ảnh số được biểu diễn dưới dạng ma trận các điểm ảnh (pixel). Mỗi pixel chứa thông tin về giá trị cường độ sáng hoặc màu sắc. Các không gian màu phổ biến bao gồm:

RGB (Red-Green-Blue) là mỗi pixel được biểu diễn bằng ba kênh màu đỏ, xanh lá và xanh dương, mỗi kênh thường có giá trị từ 0-255. Đây là không gian màu phổ biến nhất trong hiển thị kỹ thuật số.

HSV (Hue-Saturation-Value) biểu diễn màu sắc theo ba thành phần: màu sắc, độ bão hòa và giá trị. HSV đặc biệt hữu ích trong việc phát hiện đối tượng dựa trên màu sắc, như đèn giao thông.

Grayscale (Thang xám) mỗi pixel chỉ biểu diễn cường độ sáng, thường có giá trị từ 0 (đen) đến 255 (trắng). Nhiều thuật toán xử lý ảnh hoạt động trên ảnh grayscale để giảm độ phức tạp tính toán.

Binary (Nhị phân) mỗi pixel chỉ có hai giá trị: 0 (đen) hoặc 1 (trắng). Ảnh nhị phân thường được sử dụng sau khi áp dụng phân ngưỡng (thresholding).

Trong đó còn có các phép biến đổi cơ bản:

Resize (Thay đổi kích thước): Điều chỉnh kích thước ảnh, thường sử dụng các thuật toán nội suy như bilinear, bicubic. Trong hệ thống phát hiện vi phạm, việc resize ảnh giúp đồng nhất kích thước đầu vào cho mô hình CNN và giảm tải tính toán.

Crop (Cắt ảnh): Trích xuất một phần của ảnh, thường được sử dụng để tập trung vào vùng quan tâm (ROI - Region of Interest) như khu vực đèn giao thông hoặc biển số xe.

Rotate (Xoay ảnh): Xoay ảnh một góc nhất định, hữu ích trong việc chuẩn hóa hướng của đối tượng như biển số xe.

Tiếp theo là lọc nhiễu và tăng cường ảnh:

Gaussian Blur: Làm mờ ảnh bằng hàm Gaussian, giúp giảm nhiễu và chi tiết không cần thiết. Đây là bước tiền xử lý quan trọng trước khi áp dụng các thuật toán phát hiện cạnh hoặc Background Subtraction.

Bilateral Filter: Loại bỏ nhiễu nhưng vẫn giữ nguyên các cạnh quan trọng, đặc biệt hữu ích trong việc xử lý ảnh biển số trước khi áp dụng OCR.

Histogram Equalization: Cải thiện độ tương phản của ảnh bằng cách phân phối lại các giá trị cường độ sáng. Kỹ thuật này đặc biệt hữu ích khi xử lý ảnh trong điều kiện ánh sáng kém.

2.6.3. Phát hiện và trích xuất đặc trưng

Phát hiện và trích xuất đặc trưng là quá trình xác định các thông tin quan trọng từ hình ảnh, giúp mô tả và phân biệt các đối tượng. Trong bài toán phát hiện vi phạm đèn đỏ, việc trích xuất đặc trưng hiệu quả là nền tảng cho các bước xử lý tiếp theo như phân loại đèn giao thông, phát hiện và theo dõi phương tiện.

Phát hiện cạnh là kỹ thuật xác định ranh giới giữa các vùng có độ tương phản khác nhau trong ảnh. Hai thuật toán phát hiện cạnh phổ biến là:

Sobel Operator: Sử dụng đạo hàm bậc nhất của cường độ ảnh để phát hiện cạnh theo hướng ngang và dọc. Thuật toán này cung cấp thông tin về hướng gradient

Canny Edge Detector: Thuật toán phát hiện cạnh hiệu quả với khả năng loại nhiễu tốt nhờ áp dụng Gaussian filter, tính gradient, tìm cực đại cục bộ và threshold kép. Canny thường được sử dụng trong việc phát hiện biển số xe và đèn giao thông.

2.6.4. Các phép toán hình thái học

Hình thái học toán học (Mathematical Morphology) là một tập hợp các phép toán xử lý ảnh dựa trên lý thuyết tập hợp, chủ yếu áp dụng trên ảnh nhị phân và ảnh thang xám. Các phép toán này đặc biệt hữu ích trong việc xử lý foreground mask từ thuật toán Background Subtraction, cũng như tiền xử lý ảnh biển số trước khi áp dụng OCR.

Phép giãn nở (Dilation): Làm dày các đối tượng trong ảnh bằng cách thêm các pixel vào biên của đối tượng. Phép toán này hữu ích trong việc lấp đầy các lỗ nhỏ và kết nối các thành phần bị đứt đoạn.

Phép co (Erosion): Làm mỏng các đối tượng bằng cách loại bỏ các pixel ở biên. Phép toán này giúp loại bỏ các nhiễu nhỏ và tách các đối tượng bị dính nhau.

2.7. Mạng nơ-ron tích chập (CNN)

2.7.1. Kiến trúc cơ bản của CNN

Mạng nơ-ron tích chập (Convolutional Neural Network - CNN) là một kiến trúc mạng nơ-ron được thiết kế đặc biệt để xử lý dữ liệu có cấu trúc lưới như hình ảnh. Khác với mạng nơ-ron truyền thống, CNN sử dụng các phép tích chập để trích xuất đặc trưng từ dữ liệu đầu vào, cho phép mạng học các đặc trưng không gian phân cấp.CNN đã trở thành phương pháp tiêu chuẩn trong lĩnh vực thị giác máy tính nhờ khả năng tự động học các đặc trưng có ý nghĩa từ dữ liệu, thay vì sử dụng các đặc trưng được thiết kế thủ công.

Trong đó:

* weights: Trọng số của các kết nối
* inputs: Dữ liệu đầu vào
* bias: Độ lệch
* activation\_function: Hàm phi tuyến tính

Các lớp trong CNN:

Lớp tích chập (Convolutional Layer) lớp cốt lõi của CNN, thực hiện phép toán tích chập giữa đầu vào và bộ lọc (filter/kernel) để trích xuất các đặc trưng như cạnh, đường nét, và dần dần các mẫu phức tạp hơn.

Phép tích chập được tính toán bằng cách trượt kernel trên ảnh đầu vào và tính tổng tích vô hướng tại mỗi vị trí:

Lớp gộp (Pooling Layer) giảm kích thước không gian của đầu vào, giúp giảm số lượng tham số, tính toán và kiểm soát overfitting. Hai loại phổ biến nhất là Max Pooling (chọn giá trị lớn nhất trong cửa sổ) và Average Pooling (tính trung bình).

Lớp Dropout Kỹ thuật regularization đơn giản nhưng hiệu quả, giúp ngăn ngừa overfitting bằng cách ngẫu nhiên tắt một tỷ lệ nơ-ron trong quá trình huấn luyện.

Lớp fully connected (Dense Layer) kết nối mọi neuron trong lớp này với tất cả các neuron trong lớp trước đó, thường được sử dụng ở các lớp cuối cùng của mạng để đưa ra dự đoán.

Lớp Flatten chuyển đổi đầu ra đa chiều của các lớp tích chập và gộp thành vector một chiều để đưa vào lớp fully connected.

Các hàm kích hoạt:

ReLU (Rectified Linear Unit):

Hàm kích hoạt phổ biến nhất trong CNN, đơn giản và hiệu quả trong việc giải quyết vấn đề vanishing gradient.

Sigmoid sẽ Ánh xạ đầu vào vào khoảng (0,1), thường được sử dụng trong lớp đầu ra của các bài toán phân loại nhị phân.

Softmax sẽ huyển đổi vector đầu ra thành phân phối xác suất, tổng các giá trị bằng 1. Thường được sử dụng trong lớp đầu ra của bài toán phân loại đa lớp, như phân loại trạng thái đèn giao thông (đỏ, vàng, xanh).

2.7.2. Cơ chế hoạt động của CNN

Lan truyền tiến (Forward propagation) là quá trình xử lý dữ liệu đầu vào qua các lớp của mạng neural để tính toán kết quả đầu ra. Trong CNN, quá trình lan truyền tiến được thực hiện như sau:

Lớp tích chập (Convolutional Layer): Mỗi neuron trong lớp tích chập kết nối với một vùng cục bộ của lớp đầu vào thông qua một bộ lọc (kernel). Phép tích chập được tính bằng cách tính tổng của tích các phần tử tương ứng giữa kernel và vùng cục bộ đó:

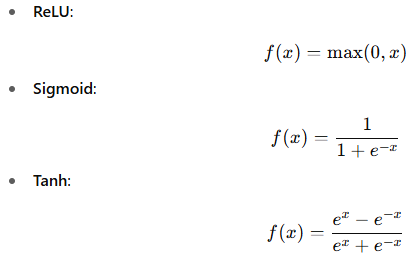
Trong đó:

* I là ma trận đầu vào
* K là kernel với kích thước $M \times N$
* b là bias
* là ma trận đầu ra sau phép tích chập

Lớp gộp (Pooling Layer): Lớp này thực hiện phép gộp (thường là max pooling hoặc average pooling) trên các vùng lân cận không chồng lấp để giảm kích thước không gian của biểu diễn, giảm số lượng tham số và tính toán trong mạng:

Trong đó Rij là vùng lân cận tại vị trí (i, j).

Lớp phi tuyến tính (Activation Layer): Sau mỗi lớp tích chập, một hàm kích hoạt phi tuyến tính (thường là ReLU) được áp dụng để đưa tính phi tuyến tính vào mô hình:



Lớp kết nối đầy đủ (Fully Connected Layer): Sau các lớp tích chập và gộp, các đặc trưng được trích xuất sẽ được đưa qua các lớp kết nối đầy đủ để thực hiện phân loại:

Trong đó:

* x là vector đầu vào
* W là ma trận trọng số
* b là vector bias
* f là hàm kích hoạt
* y là vector đầu ra

Lớp đầu ra (Output Layer): Lớp cuối cùng thường sử dụng hàm kích hoạt Softmax để chuyển đổi các giá trị đầu ra thành phân phối xác suất trên các lớp:

Lan truyền ngược (Backpropagation) là thuật toán cơ bản để huấn luyện mạng neural bằng cách tính toán gradient của hàm mất mát đối với các tham số của mạng. Quá trình lan truyền ngược trong CNN bao gồm các bước sau:

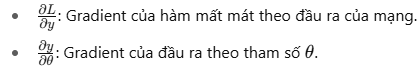
Tính hàm mất mát là tính toán độ lệch giữa đầu ra dự đoán và nhãn thực tế sử dụng một hàm mất mát như Cross-Entropy:

Trong đó:

* yi là nhãn thực tế
*  là xác suất dự đoán

Tính gradient của hàm mất mát: Đối với mỗi tham số $\theta$ (trọng số và bias), tính gradient của hàm mất mát:

Trong đó:



Lan truyền gradient ngược qua mạng:

Tính gradient tại lớp đầu ra:

Trong đó:

* **W** là trọng số của lớp fully connected.
* Gradient được lan truyền qua quy tắc chuỗi.

Lớp tích chập và gộp (Convolution & Pooling Layers)

* Sử dụng quy tắc chuỗi để tính gradient đối với các tham số:
* Áp dụng quy tắc chuỗi để lan truyền gradient qua các lớp này.
* Lan truyền gradient qua lớp gộp và tích chập: Sử dụng quy tắc chuỗi để tính gradient đối với các tham số trong các lớp này.

Hàm mất mát đo lường độ chính xác của mô hình trong quá trình huấn luyện. Các hàm mất mát phổ biến cho các bài toán phân loại trong CNN trong đó có Cross-Entropy Loss, Binary Cross-Entropy Loss, Binary Cross-Entropy Loss, Focal Loss.

Thuật toán tối ưu (Optimization algorithm) là thuật toán tối ưu xác định cách cập nhật trọng số của mạng dựa trên gradient của hàm mất mát.

2.7.3. Các kỹ thuật nâng cao trong CNN

Data augmentation là kỹ thuật tạo ra các biến thể của dữ liệu huấn luyện bằng cách áp dụng các phép biến đổi khác nhau, giúp tăng kích thước tập dữ liệu và cải thiện khả năng tổng quát hóa của mô hình như Phép xoay (Rotation), Phép lật (Flipping), Phép dịch chuyển (Translation) vàPhép cắt (Cropping).

2.7.4. Các kiến trúc CNN hiện đại

**LeNet, AlexNet, VGG**

**LeNet-5 (1998)**

LeNet-5 là kiến trúc CNN tiên phong do Yann LeCun phát triển, ban đầu được sử dụng cho nhận dạng chữ số viết tay.

* **Kiến trúc**: 7 lớp (không tính đầu vào)
  + 2 lớp convolutional (kết hợp với pooling)
  + 3 lớp fully connected
* **Đặc điểm**:
  + Sử dụng hàm kích hoạt tanh
  + Sử dụng average pooling
  + Kích thước nhỏ (~60K tham số)
* **Ứng dụng**: Nhận dạng chữ số, ký tự đơn giản
* **Hạn chế**: Khả năng xử lý ảnh phức tạp thấp, không hiệu quả với dữ liệu lớn

**AlexNet (2012)**

AlexNet đánh dấu bước ngoặt trong lịch sử deep learning khi giành chiến thắng tại cuộc thi ImageNet 2012 với độ chính xác vượt trội.

* **Kiến trúc**: 8 lớp
  + 5 lớp convolutional
  + 3 lớp fully connected
* **Đặc điểm**:
  + Sử dụng hàm kích hoạt ReLU
  + Áp dụng Local Response Normalization
  + Sử dụng Dropout để tránh overfitting
  + Sử dụng max pooling
  + Kích thước lớn (~60M tham số)
* **Ưu điểm**: Khả năng xử lý dữ liệu phức tạp, tăng độ chính xác đáng kể
* **Hạn chế**: Tốn nhiều tài nguyên tính toán, cần GPU để huấn luyện hiệu quả

**VGG (2014)**

VGG (Visual Geometry Group) do Đại học Oxford phát triển, nổi tiếng với kiến trúc đơn giản, thống nhất nhưng hiệu quả.

* **Các biến thể**: VGG16, VGG19 (con số chỉ số lớp có trọng số)
* **Kiến trúc VGG16**:
  + 13 lớp convolutional
  + 3 lớp fully connected
* **Đặc điểm**:
  + Sử dụng các kernel nhỏ 3×3 xếp chồng lên nhau
  + Cấu trúc đồng nhất, dễ hiểu
  + Kích thước rất lớn (~138M tham số)
* **Ưu điểm**: Trích xuất đặc trưng hiệu quả, dễ dàng mở rộng
* **Hạn chế**: Yêu cầu bộ nhớ lớn, tốc độ xử lý chậm, không phù hợp với thiết bị di động

VGG thường được sử dụng làm backbone cho nhiều ứng dụng thị giác máy tính và là lựa chọn phổ biến cho transfer learning.

**ResNet, Inception, MobileNet**

**ResNet (2015)**

ResNet (Residual Network) giải quyết vấn đề "vanishing gradient" trong mạng sâu thông qua kết nối tắt (skip connection), cho phép huấn luyện mạng có độ sâu lên đến hàng trăm lớp.

* **Kiến trúc**: Nhiều biến thể từ ResNet18 đến ResNet152
* **Thành phần cơ bản**: Residual block
  + Input → Conv → BN → ReLU → Conv → BN → + Input → ReLU → Output
* **Đặc điểm**:
  + Sử dụng kết nối tắt (identity mapping)
  + Áp dụng Batch Normalization
  + Sử dụng global average pooling thay vì fully connected
* **Ưu điểm**: Huấn luyện nhanh hơn, độ chính xác cao, giải quyết vấn đề gradient biến mất
* **Ứng dụng**: Phân loại ảnh, phát hiện đối tượng, phân đoạn ngữ nghĩa

**MobileNet (2017)**

MobileNet được Google thiết kế đặc biệt cho các thiết bị di động và edge computing với tài nguyên hạn chế.

* **Kiến trúc**: Nhiều biến thể MobileNetV1, V2, V3
* **Thành phần cơ bản** của MobileNetV2:
  + Depthwise separable convolution
  + Inverted residual blocks với linear bottleneck
* **Đặc điểm**:
  + Tách tích chập thông thường thành depthwise và pointwise convolution
  + Hệ số alpha để điều chỉnh width của mạng
  + Hệ số rho để điều chỉnh độ phân giải đầu vào
* **Ưu điểm**: Kích thước nhỏ, tốc độ nhanh, hiệu suất năng lượng tốt
* **Ứng dụng**: Ứng dụng di động, IoT, hệ thống nhúng, thời gian thực

MobileNet là lựa chọn phù hợp cho hệ thống phát hiện vi phạm đèn đỏ khi cần triển khai trên edge device.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Kiến trúc | Số tham số | Độ chính xác (ImageNet) | Kích thước (MB) | Thời gian suy luận |
| LeNet-5 | ~60K | N/A | <0.5 | Rất nhanh |
| AlexNet | ~60M | 63.3% | ~233 | Chậm |
| VGG16 | ~138M | 71.3% | ~528 | Rất chậm |
| ResNet50 | ~25M | 76.0% | ~98 | Trung bình |
| MobileNetV2 | ~3.5M | 71.8% | ~14 | Nhanh |

Bảng 2-1: So sánh cách kiến trúc

2.8 Thuật toán Mixture of Gaussians v2 (MOG2)

2.8.1. Tổng quan về phương pháp khử nền

Background Subtraction là kỹ thuật xử lý ảnh và video nhằm tách các đối tượng chuyển động (foreground) khỏi phần nền tĩnh (background). Đây là bước tiền xử lý quan trọng trong nhiều ứng dụng thị giác máy tính, đặc biệt là giám sát giao thông.

Background Subtraction (BS) được định nghĩa là quá trình phân đoạn một ảnh thành hai thành phần chính là Background (nền): Những phần tĩnh, không thay đổi hoặc thay đổi định kỳ trong cảnh và Foreground (tiền cảnh): Những đối tượng chuyển động, mới xuất hiện hoặc khác biệt so với mô hình nền

Các ứng dụng chính của Background Subtraction bao gồm:

* Giám sát giao thông và phát hiện phương tiện
* Theo dõi đối tượng trong hệ thống an ninh
* Phân tích hành vi người dùng trong không gian công cộng
* Ứng dụng tương tác người-máy
* Đếm đối tượng và phân tích mật độ đám đông

Mặc dù khái niệm đơn giản, nhưng BS phải đối mặt với nhiều thách thức trong môi trường thực tế như thay đổi ánh sáng như việc thay đổi đột ngột (bật/tắt đèn, mây che nắng). Nhiễu và chất lượng video Ảnh hưởng của thời tiết (mưa, tuyết, sương mù) và thêm đối tượng có màu sắc tương tự nền hoặc chuyển động chậm.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Phương pháp | Nguyên lý | Ưu điểm | Nhược điểm | Phù hợp với |
| Background Subtraction | Mô hình hóa nền và tách đối tượng chuyển động | Nhanh, hiệu quả với camera tĩnh, ít tài nguyên | Nhạy cảm với thay đổi ánh sáng, camera cố định | Camera giám sát cố định, cảnh tĩnh |
| Haar Cascades | Phát hiện dựa trên đặc trưng Haar | Nhanh, đơn giản | Độ chính xác thấp, chỉ phát hiện đối tượng đã học | Phát hiện khuôn mặt, biển số |
| HOG + SVM | Trích xuất HOG và phân loại bằng SVM | Tốc độ khá, mô tả hình dạng tốt | Chậm hơn Haar, khó phát hiện đối tượng nhỏ | Phát hiện người đi bộ, phương tiện |
| CNN-based (YOLO, SSD) | Mạng CNN end-to-end | Độ chính xác cao, phát hiện đa đối tượng | Tốn tài nguyên, yêu cầu GPU | Phát hiện đa đối tượng,môi trường phức tạp |

Bảng 2-2: So sánh với các phương pháp phát hiện đối tượng khác

2.8.2. Các phương pháp Background Subtraction

Background Subtraction có nhiều phương pháp với độ phức tạp và hiệu quả khác nhau. Các phương pháp này được phân loại thành ba nhóm chính: phương pháp cơ bản, phương pháp thống kê và phương pháp mô hình hóa.

Phương pháp cơ bản có Frame differencing là phương pháp đơn giản nhất của Background Subtraction. Phương pháp này dựa trên sự chênh lệch về cường độ pixel giữa hai frame liên tiếp để phát hiện chuyển động.

Trong đó:

* It (x, y) là giá trị cường độ của pixel tại vị trí (x, y) ở frame hiện tại
* It-1(x, y) là giá trị cường độ của pixel tại vị trí (x, y) ở frame trước đó
* T là ngưỡng xác định sự thay đổi

Ưu điểm của phương pháp này là tính đơn giản và hiệu quả tính toán cao. Tuy nhiên, phương pháp này rất nhạy cảm với nhiễu, thay đổi ánh sáng và không thể phát hiện đối tượng đứng yên. Ngoài ra, phương pháp này chỉ phát hiện được phần biên của đối tượng chuyển động, không phát hiện được toàn bộ đối tượng.

Phương pháp thống kê thì có Running Average (Trung bình trượt) phương pháp này cập nhật mô hình nền liên tục bằng cách tính trung bình trượt:

Trong đó:

* Bt (x, y) là giá trị nền tại vị trí (x, y) ở thời điểm t
* It (x, y) là giá trị pixel tại vị trí (x, y) ở frame hiện tại
* Bt-1 (x, y) là giá trị nền tại vị trí (x, y) ở thời điểm t-1
* α là tham số học (learning rate), thường nằm trong khoảng [0,1]

Phương pháp này có ưu điểm là dễ cài đặt, hiệu quả về mặt tính toán và bộ nhớ, và có khả năng thích ứng với những thay đổi chậm trong cảnh. Tuy nhiên, việc chọn tham số α là quan trọng: α quá lớn sẽ khiến nền thích ứng quá nhanh và có thể đưa đối tượng vào nền, trong khi α quá nhỏ sẽ khiến nền thích ứng quá chậm với những thay đổi trong cảnh.

Phương pháp mô hình hóa Gaussian Mixture Model (GMM) phương pháp này mô hình hóa mỗi pixel bằng một hỗn hợp các phân phối Gaussian. Mỗi phân phối Gaussian đại diện cho một "trạng thái" có thể của pixel đó (ví dụ: thuộc nền, thuộc đối tượng chuyển động, bóng, v.v.).

Trong đó:

* P (It​ (x, y)): Xác suất điểm ảnh (x, y) (x, y) (x, y) tại thời điểm **t**.
* K là số lượng phân phối Gaussian
* wi, t là trọng số của phân phối Gaussian thứ i tại thời điểm **t**
* η là hàm mật độ xác suất của phân phối Gaussian
* μi, t là vector trung bình của phân phối Gaussian thứ i tại thời điểm **t**
* ∑i, t là ma trận hiệp phương sai của phân phối Gaussian thứ i tại thời điểm **t**

Mixture of Gaussians (MOG) là một cài đặt cụ thể của GMM, được giới thiệu bởi Stauffer và Grimson. Phương pháp này thường sử dụng từ 3 đến 5 phân phối Gaussian cho mỗi pixel và cập nhật các tham số của từng phân phối dựa trên các pixel mới.

Mixture of Gaussians 2 (MOG2) là phiên bản cải tiến của MOG, được phát triển bởi Zivkovic. MOG2 tự động chọn số lượng phân phối Gaussian phù hợp cho từng pixel, giúp tiết kiệm bộ nhớ và tăng hiệu suất. Phương pháp này cũng có khả năng phát hiện bóng tốt hơn.

Các phương pháp dựa trên GMM có ưu điểm là khả năng thích ứng với những thay đổi trong cảnh, khả năng xử lý nền động (như cây lắc trong gió, mặt nước gợn sóng), và khả năng phân biệt đối tượng với bóng. Tuy nhiên, các phương pháp này có chi phí tính toán cao hơn và yêu cầu điều chỉnh nhiều tham số.

**2.8.3. Thuật toán MOG2 (Mixture of Gaussians 2)**

MOG2 (Mixture of Gaussians 2) là một trong những thuật toán Background Subtraction hiện đại và hiệu quả nhất, được tích hợp sẵn trong thư viện OpenCV. Thuật toán này là phiên bản cải tiến của thuật toán MOG ban đầu, với nhiều ưu điểm vượt trội.

MOG2 mô hình hóa mỗi pixel trong ảnh bằng một hỗn hợp các phân phối Gaussian (K phân phối). Xác suất để một pixel có giá trị X\_t tại thời điểm t được tính.

Trong đó:

* K là số lượng phân phối Gaussian, MOG2 tự động điều chỉnh K cho từng pixel
* wi, t là trọng số của phân phối Gaussian thứ i tại thời điểm t
* η là hàm mật độ xác suất của phân phối Gaussian
* μi, t là vector trung bình của phân phối Gaussian thứ i
* ∑i, t là ma trận hiệp phương sai của phân phối Gaussian thứ i

Khác với MOG, MOG2 áp dụng phương pháp "maximization likelihood" để xác định liệu một pixel có khớp với một trong các phân phối Gaussian hiện có hay không. Nếu khớp, các tham số của phân phối đó sẽ được cập nhật. Nếu không khớp, một phân phối mới sẽ được tạo ra hoặc phân phối ít quan trọng nhất sẽ bị thay thế.

Các phân phối Gaussian được sắp xếp theo giá trị w/σ (trọng số / độ lệch chuẩn), và B phân phối đầu tiên được coi là mô hình nền

Trong đó T là ngưỡng xác định tỷ lệ dữ liệu nên thuộc về nền.

MOG2 cũng giới thiệu phương pháp phát hiện bóng dựa trên việc phân tích sự thay đổi về cường độ sáng mà không thay đổi màu sắc.

Ưu điểm và hạn chế của MOG2

Ưu điểm:

* Khả năng thích ứng với thay đổi trong cảnh (như thay đổi ánh sáng, chuyển động của các đối tượng nền)
* Tự động điều chỉnh số lượng phân phối Gaussian cho từng pixel, tiết kiệm bộ nhớ và tăng hiệu suất
* Có khả năng phát hiện và phân biệt bóng với đối tượng chuyển động
* Xử lý tốt các trường hợp nền động (như cây lắc trong gió, mặt nước gợn sóng)
* Đã được tối ưu hóa trong thư viện OpenCV

Hạn chế:

* Chi phí tính toán cao hơn so với các phương pháp đơn giản
* Có thể gặp khó khăn trong điều kiện ánh sáng thay đổi đột ngột
* Hiệu suất phụ thuộc vào việc điều chỉnh các tham số
* Có thể gặp khó khăn với đối tượng di chuyển chậm (có thể bị đưa vào mô hình nền)
* Không hiệu quả trong trường hợp camera di chuyển

2.8.3 Các tham số của thuật toán MOG2

MOG2 có nhiều tham số ảnh hưởng đến hiệu suất và độ chính xác của thuật toán. Dưới đây là các tham số quan trọng:

1. history (Lịch sử)

* Mô tả: Độ dài lịch sử (số khung hình) được sử dụng để huấn luyện mô hình nền
* Giá trị mặc định: 500
* Ảnh hưởng:
  + Giá trị lớn: Mô hình ổn định hơn, nhưng chậm thích ứng với thay đổi
  + Giá trị nhỏ: Thích ứng nhanh với thay đổi, nhưng dễ bị nhiễu

2. varThreshold (Ngưỡng phương sai)

* Mô tả: Ngưỡng Mahalanobis để quyết định pixel có khớp với mô hình hay không
* Giá trị mặc định: 16
* Ảnh hưởng:
  + Giá trị lớn: Ít phát hiện chuyển động hơn (ít false positive)
  + Giá trị nhỏ: Phát hiện nhiều chuyển động hơn (nhiều false positive)

3. detectShadows (Phát hiện bóng đổ)

* Mô tả: Bật/tắt khả năng phát hiện bóng đổ
* Giá trị mặc định: true
* Ảnh hưởng:
  + true: Phân biệt bóng đổ khỏi đối tượng chuyển động, giảm false positive
  + false: Tăng tốc độ xử lý nhưng có thể coi bóng đổ là đối tượng chuyển động

4. shadowThreshold (Ngưỡng bóng đổ)

* Mô tả: Ngưỡng để xác định bóng đổ
* Giá trị mặc định: 0.5
* Ảnh hưởng:
  + Giá trị lớn: Phát hiện ít bóng đổ hơn
  + Giá trị nhỏ: Phát hiện nhiều bóng đổ hơn

5. nMixtures (Số phân phối)

* Mô tả: Số lượng phân phối Gaussian tối đa cho mỗi pixel
* Giá trị mặc định: 5
* Ảnh hưởng:
  + Giá trị lớn: Mô hình phức tạp hơn, xử lý nền đa dạng tốt hơn nhưng tốn bộ nhớ
  + Giá trị nhỏ: Mô hình đơn giản, tốc độ nhanh hơn nhưng kém chính xác hơn

6. backgroundRatio (Tỷ lệ nền)

* Mô tả: Ngưỡng để xếp hạng phân phối nền (giá trị càng nhỏ, càng ít phân phối được coi là nền)
* Giá trị mặc định: 0.9
* Ảnh hưởng:
  + Giá trị lớn: Nhiều phân phối được coi là nền
  + Giá trị nhỏ: Ít phân phối được coi là nền

7. learningRate (Tốc độ học)

* Mô tả: Tốc độ cập nhật mô hình nền
* Giá trị mặc định: -1 (tự động)
* Ảnh hưởng:
  + Giá trị lớn: Mô hình thích ứng nhanh với thay đổi
  + Giá trị nhỏ: Mô hình ổn định hơn, thích hợp với cảnh tĩnh

2.9 Công nghệ nhận dạng ký tự quang học (OCR)

Nhận dạng ký tự quang học (Optical Character Recognition - OCR) là công nghệ cho phép chuyển đổi các hình ảnh chứa văn bản (đã được in, viết tay, hoặc khắc) thành dạng văn bản số có thể chỉnh sửa được. Trong hệ thống phát hiện vi phạm đèn đỏ, OCR được sử dụng để nhận dạng biển số xe từ hình ảnh của phương tiện vi phạm.

2.9.1 Tổng quan về OCR

OCR có lịch sử phát triển lâu dài, bắt đầu từ những năm 1870 với bằng sáng chế của C.R. Carey về thiết bị truyền hình độ phân giải thấp. Tuy nhiên, OCR hiện đại được phát triển từ những năm 1950 với việc ra đời của máy tính điện tử. Những hệ thống OCR đầu tiên được thiết kế để nhận dạng một số ký tự cố định với font chữ cụ thể.

Sự phát triển của OCR có thể chia thành các giai đoạn:

* Giai đoạn 1 (1950-1970): OCR cơ học dựa trên template matching, chỉ nhận dạng được font chữ cụ thể.
* Giai đoạn 2 (1970-1990): OCR dựa trên các thuật toán phân loại thống kê, nhận dạng được nhiều font chữ hơn.
* Giai đoạn 3 (1990-2010): OCR dựa trên học máy truyền thống như SVM, Random Forest.
* Giai đoạn 4 (2010-nay): OCR dựa trên deep learning, đặc biệt là CNN và RNN/LSTM, đạt được độ chính xác cao và khả năng nhận dạng văn bản phức tạp, thậm chí là văn bản viết tay.

Một hệ thống OCR hiện đại thường bao gồm các thành phần sau:

Tiền xử lý ảnh: Bao gồm các bước như chuyển đổi sang ảnh grayscale, loại bỏ nhiễu, chuẩn hóa độ tương phản, xoay ảnh, v.v.

* Phân đoạn văn bản: Xác định vùng chứa văn bản trong ảnh.
* Phân đoạn dòng và từ: Tách văn bản thành các dòng và từ riêng biệt.
* Phân đoạn ký tự: Tách mỗi từ thành các ký tự riêng biệt.
* Nhận dạng ký tự: Phân loại mỗi ký tự thành ký tự cụ thể trong bảng chữ cái.
* Hậu xử lý: Sử dụng thông tin ngữ cảnh, từ điển, hoặc các quy tắc ngôn ngữ để sửa lỗi và cải thiện kết quả nhận dạng.

Trong các hệ thống OCR hiện đại dựa trên deep learning, một số bước trên có thể được thực hiện đồng thời bởi một mô hình end-to-end.

2.4.2. Tesseract OCR

Tesseract OCR là một trong những công cụ OCR mã nguồn mở phổ biến và mạnh mẽ nhất, được phát triển ban đầu bởi HP vào những năm 1980 và 1990, sau đó được Google tiếp quản và phát triển từ năm 2006. Tesseract hỗ trợ nhận dạng hơn 100 ngôn ngữ và có thể được huấn luyện để nhận dạng các font chữ và ngôn ngữ mới.

Kiến trúc và nguyên lý hoạt động của Tesseract OCR hoạt động theo quy trình như sau:

1. Phân tích bố cục trang: Tesseract phân tích bố cục trang để xác định các vùng văn bản.
2. Phân đoạn dòng và từ: Tesseract tách văn bản thành các dòng và từ.
3. Nhận dạng từ: Tesseract thực hiện nhận dạng ký tự cho mỗi từ, sử dụng một quy trình hai giai đoạn:
   * Giai đoạn 1: Nhận dạng mỗi từ mà không có thông tin ngữ cảnh.
   * Giai đoạn 2: Thực hiện nhận dạng lần thứ hai, sử dụng thông tin từ các từ được nhận dạng trong giai đoạn 1 để cải thiện độ chính xác.
4. Phân tích ngôn ngữ: Tesseract sử dụng các mô hình ngôn ngữ để cải thiện kết quả nhận dạng.

**2.9.3. Tiền xử lý ảnh cho OCR**

Tiền xử lý ảnh là bước quan trọng trong quy trình OCR, đặc biệt khi làm việc với biển số xe trong điều kiện thực tế. Quá trình này giúp nâng cao chất lượng ảnh đầu vào, từ đó cải thiện đáng kể độ chính xác của kết quả nhận dạng.

Phương pháp cải thiện chất lượng ảnh

Chất lượng ảnh biển số xe thường bị ảnh hưởng bởi nhiều yếu tố môi trường như ánh sáng không đồng đều, mờ do chuyển động, hay nhiễu. Các kỹ thuật cải thiện chất lượng ảnh phổ biến bao gồm:

* Điều chỉnh độ tương phản: Tăng cường độ tương phản giữa ký tự và nền biển số sử dụng các phương pháp như Histogram Equalization hoặc CLAHE (Contrast Limited Adaptive Histogram Equalization).
* Loại bỏ nhiễu: Áp dụng các bộ lọc như Gaussian, Median hoặc Bilateral để giảm nhiễu mà vẫn bảo toàn cạnh của ký tự.
* Điều chỉnh độ sáng: Chuẩn hóa độ sáng của ảnh để đạt được giá trị tối ưu cho OCR, đặc biệt quan trọng khi xử lý ảnh chụp trong điều kiện thiếu sáng hoặc quá sáng.
* Hiệu chỉnh độ nghiêng: Phát hiện và điều chỉnh góc nghiêng của biển số để các ký tự nằm trên cùng một đường thẳng, giúp OCR nhận dạng tốt hơn.

2.9.3. Cấu hình và tối ưu Tesseract OCR

Tesseract OCR là một công cụ OCR mã nguồn mở mạnh mẽ, tuy nhiên để đạt được hiệu suất tối ưu cho bài toán nhận dạng biển số xe, cần có sự cấu hình phù hợp.

PSM là tham số quan trọng trong Tesseract, xác định cách Tesseract phân tích bố cục của ảnh đầu vào. Đối với nhận dạng biển số xe, một số chế độ PSM phù hợp bao gồm:

* PSM 7 (Treat the image as a single text line): Phù hợp khi biển số đã được căn chỉnh và chỉ có một dòng text.
* PSM 8 (Treat the image as a single word): Phù hợp với biển số ngắn không có khoảng cách.
* PSM 10 (Treat the image as a single character): Sử dụng khi đã phân đoạn từng ký tự riêng biệt.
* PSM 13 (Raw line. Treat the image as a single text line, bypassing hacks that are Tesseract-specific): Phù hợp với một số biển số đặc biệt.

2.9.4 Hậu xử lý kết quả OCR

Kết quả OCR thô thường chứa các lỗi và cần được hậu xử lý để đạt được độ chính xác cao hơn. Đặc biệt với biển số xe, có thể áp dụng kiến thức về định dạng và cấu trúc cụ thể của biển số Việt Nam để cải thiện kết quả.

Kỹ thuật sửa lỗi dựa trên từ điển

Phương pháp này sử dụng kiến thức về cấu trúc biển số xe để sửa các lỗi nhận dạng phổ biến:

* Bảng ánh xạ ký tự: Sửa các lỗi nhận dạng thường gặp, ví dụ '0' thành 'O', '1' thành 'I', '8' thành 'B'...
* Sửa lỗi vị trí: Dựa vào vị trí của ký tự trong biển số để xác định khả năng cao nhất (ví dụ: vị trí đầu tiên của biển số xe Việt Nam thường là số)

Regular expressions (regex) có thể được sử dụng để xác thực và chuẩn hóa kết quả OCR theo định dạng biển số xe Việt Nam:

* Biển số xe máy: Có định dạng NN-AA-NNNNN (N: số, A: chữ)
* Biển số ô tô: Có định dạng NNA-NNNN hoặc NNA-NNN.NN

Hậu xử lý kết quả OCR là bước quan trọng cuối cùng để cải thiện độ chính xác nhận dạng biển số xe.

Bằng cách kết hợp kiến thức về cấu trúc biển số, sử dụng regex và đánh giá độ tin cậy, có thể đạt được kết quả nhận dạng chính xác hơn so với kết quả OCR thô.

# CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

3.1. Phân tích bài toán

3.1.1. Mô tả bài toán

Bài toán phát hiện vi phạm đèn đỏ và nhận dạng biển số xe tập trung vào việc xây dựng một hệ thống tự động có khả năng giám sát giao thông tại các giao lộ có đèn tín hiệu. Mục tiêu chính của hệ thống là phát hiện, ghi nhận và xử lý các trường hợp phương tiện vượt đèn đỏ một cách chính xác và kịp thời. Đây là một bài toán quan trọng trong lĩnh vực an toàn giao thông và ứng dụng trí tuệ nhân tạo vào quản lý đô thị thông minh.

Hệ thống cần theo dõi đồng thời ba đối tượng chính trong môi trường giao thông:

Đèn giao thông: Hệ thống cần có khả năng nhận diện và phân loại trạng thái của đèn giao thông (đỏ, vàng, xanh) theo thời gian thực. Việc xác định chính xác trạng thái đèn là điều kiện tiên quyết để xác định vi phạm.

Phương tiện giao thông: Hệ thống phải phát hiện và theo dõi các phương tiện xuất hiện trong khung hình video. Điều này bao gồm việc xác định vị trí, hướng di chuyển và theo dõi quỹ đạo của từng phương tiện.

Vạch dừng (stop line): Hệ thống cần xác định chính xác vị trí của vạch dừng trên mặt đường, đây là ranh giới mà phương tiện không được vượt qua khi đèn đỏ.

Một vi phạm đèn đỏ được xác định khi thỏa mãn đồng thời các điều kiện sau:

* Đèn giao thông đang ở trạng thái đỏ.
* Một phương tiện vượt qua vạch dừng (điểm đặc trưng của phương tiện, thường là phần dưới, vượt quá tọa độ y của vạch dừng theo hướng từ trên xuống dưới trong khung hình).
* Phương tiện chưa được ghi nhận vi phạm trong chu kỳ đèn đỏ hiện tại, nhằm tránh ghi nhận trùng lặp.

Khi phát hiện một vi phạm, hệ thống sẽ thực hiện chuỗi các hành động sau:

* Chụp và lưu trữ ảnh của phương tiện vi phạm từ khung hình video.
* Định vị và cắt ảnh biển số xe từ ảnh phương tiện vi phạm.
* Sử dụng kỹ thuật nhận dạng ký tự quang học (OCR) để đọc và trích xuất thông tin biển số xe.
* Lưu trữ đầy đủ thông tin vi phạm vào cơ sở dữ liệu, bao gồm: thời gian vi phạm, ảnh phương tiện, ảnh biển số, kết quả nhận dạng biển số.

Ngoài ra, hệ thống cần được thiết kế để xử lý hiệu quả các tình huống khác như:

* Nhiều phương tiện vi phạm cùng một lúc.
* Điều kiện ánh sáng môi trường thay đổi (ngày/đêm, nắng/mưa).
* Phương tiện di chuyển với tốc độ cao khiến việc chụp và nhận dạng biển số trở nên khó khăn.
* Các trường hợp biển số bị che khuất do góc camera hoặc vật cản.

Hệ thống cần đảm bảo khả năng hoạt động liên tục, độ chính xác cao và xử lý thời gian thực để đáp ứng yêu cầu của một hệ thống giám sát giao thông hiện đại.

3.1.2. Các thách thức kỹ thuật

Việc xây dựng một hệ thống phát hiện vi phạm đèn đỏ và nhận dạng biển số xe tự động đối mặt với nhiều thách thức kỹ thuật phức tạp. Những thách thức này đòi hỏi các giải pháp chuyên biệt để đảm bảo hệ thống hoạt động hiệu quả trong điều kiện thực tế. Dưới đây là phân tích chi tiết về các thách thức chính:

Thách thức về điều kiện ánh sáng:

* Thay đổi ánh sáng tự nhiên: Ánh sáng thay đổi liên tục theo thời gian trong ngày (sáng, trưa, chiều, tối) ảnh hưởng đến độ sáng, màu sắc và độ tương phản của hình ảnh thu được. Đặc biệt, các thời điểm chuyển giao như bình minh và hoàng hôn tạo ra những thay đổi đột ngột về điều kiện ánh sáng.
* Phản xạ ánh sáng: Ánh sáng mặt trời hoặc đèn đường có thể tạo ra hiệu ứng phản xạ trên mặt đường hoặc bề mặt phương tiện, gây nhiễu cho quá trình nhận diện.
* Điều kiện thời tiết bất lợi: Mưa, sương mù, hoặc nắng chói tạo ra các hiệu ứng quang học như đốm sáng, mờ ảnh, hoặc giảm độ tương phản, ảnh hưởng đến chất lượng hình ảnh.
* Độ tương phản giữa đèn giao thông và nền: Trong một số điều kiện ánh sáng, có thể khó phân biệt giữa màu sắc của đèn giao thông và nền xung quanh, đặc biệt khi có ánh nắng chiếu trực tiếp vào đèn.

Thách thức về góc camera và vị trí lắp đặt:

* Góc nhìn hạn chế: Camera có thể không bao quát được toàn bộ khu vực cần giám sát, hoặc một phần quan trọng (như đèn giao thông) có thể bị che khuất bởi các vật cản như cây xanh, biển báo.
* Khoảng cách xa: Khoảng cách từ camera đến đèn giao thông và biển số xe có thể quá xa, làm giảm độ phân giải và độ chi tiết của đối tượng quan tâm.
* Độ rung camera: Yếu tố môi trường như gió mạnh hoặc rung chấn từ phương tiện lưu thông có thể gây ra hiện tượng rung lắc camera, làm mờ hình ảnh và ảnh hưởng đến chất lượng xử lý.
* Biến dạng hình ảnh: Khi camera được lắp đặt với góc nghiêng quá lớn, hình ảnh thu được có thể bị biến dạng, khó khăn cho việc nhận diện chính xác đối tượng và đặc biệt là việc đọc biển số xe.

Thách thức về mật độ giao thông:

* Hiệu ứng che khuất: Trong điều kiện giao thông đông đúc, các phương tiện có thể che khuất lẫn nhau, khiến hệ thống khó theo dõi liên tục một phương tiện cụ thể hoặc xác định chính xác vị trí biển số.
* Xác định đúng phương tiện vi phạm: Khi nhiều phương tiện di chuyển gần nhau và gần vạch dừng, việc xác định chính xác phương tiện nào thực sự vượt đèn đỏ trở nên phức tạp.
* Theo dõi phương tiện trong điều kiện đông đúc: Thuật toán theo dõi (tracking) phải đối mặt với thách thức khi nhiều đối tượng xuất hiện, di chuyển, và biến mất khỏi khung hình liên tục.
* Xử lý các trường hợp dừng/đỗ: Phương tiện dừng hoặc đỗ trong khung hình có thể gây nhiễu cho hệ thống, đặc biệt nếu chúng nằm gần vạch dừng hoặc che khuất các đối tượng khác.

Thách thức về nhận dạng biển số xe:

* Kích thước nhỏ của biển số: Trong khung hình video tổng thể, biển số xe thường chiếm một phần nhỏ, hạn chế độ phân giải và chi tiết có thể thu được.
* Phụ thuộc vào chất lượng camera: Độ sắc nét của hình ảnh biển số phụ thuộc rất lớn vào chất lượng camera sử dụng và điều kiện ánh sáng.
* Biển số nghiêng hoặc biến dạng: Góc nhìn camera có thể khiến biển số xuất hiện với góc nghiêng hoặc biến dạng hình học, gây khó khăn cho OCR.
* Đa dạng về kiểu dáng biển số: Sự đa dạng về kích thước, màu sắc, định dạng và font chữ của biển số theo vùng miền hoặc loại phương tiện đòi hỏi hệ thống phải có khả năng thích ứng.

Thách thức về độ chính xác:

* Giảm thiểu false positive: Hệ thống cần tránh việc báo cáo sai vi phạm không tồn tại, điều này có thể gây ra hậu quả pháp lý và làm giảm niềm tin vào hệ thống.
* Giảm thiểu false negative: Hệ thống không được bỏ sót các vi phạm thực sự xảy ra, đảm bảo tính công bằng và hiệu quả của việc giám sát.
* Duy trì độ chính xác trong điều kiện thay đổi: Độ chính xác cần được duy trì dù có sự thay đổi về điều kiện môi trường, thời tiết hoặc mật độ giao thông.
* Xử lý các trường hợp ngoại lệ: Hệ thống cần có khả năng xử lý các tình huống đặc biệt như xe cứu thương vượt đèn đỏ, các tình huống khẩn cấp hoặc điều kiện đặc biệt khác.

Những thách thức kỹ thuật này đòi hỏi cách tiếp cận đa chiều, kết hợp nhiều kỹ thuật xử lý ảnh, thị giác máy tính và học sâu để xây dựng một hệ thống hoàn chỉnh, hiệu quả và đáng tin cậy.

3.1.3. Phân tích yêu cầu dữ liệu

Để xây dựng và vận hành một hệ thống phát hiện vi phạm đèn đỏ và nhận dạng biển số xe hiệu quả, việc thu thập, xử lý và quản lý dữ liệu đóng vai trò then chốt. Các loại dữ liệu cần thiết và phương pháp xử lý được phân tích chi tiết như sau:

Đầu tiên dữ liệu huấn luyện cho mô hình phân loại đèn giao thông được sử dụng là “trafficlight Computer Vision Project” bởi Gio 5 tháng trước được đăng tải trên Roboflow với định dạng .COCO (Common Objects in Context) và đã được chia thành 3 tệp test, train, valid phù hợp cho việc huấn luyện mô hình.

Mô hình phân loại đèn giao thông là thành phần cốt lõi của hệ thống, đòi hỏi tập dữ liệu chất lượng cao để huấn luyện:

* Tập dữ liệu hình ảnh đèn giao thông thu thập đa dạng hình ảnh đèn giao thông trong các trạng thái khác nhau (đỏ, vàng, xanh) từ nhiều góc độ, khoảng cách và điều kiện ánh sáng.
* Mỗi hình ảnh đều có chú thích dữ liệu (annotation) được gán nhãn với bounding box xác định vị trí của đèn trong hình và nhãn trạng thái tương ứng (đỏ, vàng, xanh).

Phân chia dữ liệu: Áp dụng phương pháp phân chia dữ liệu theo tỷ lệ tiêu chuẩn:

* Tập huấn luyện (train): 70% tổng số mẫu
* Tập kiểm định (validation): 15% tổng số mẫu
* Tập kiểm thử (test): 15% tổng số mẫu

Gán nhãn cho các lớp: Mỗi ảnh được gán nhãn với các lớp tương ứng:

* "den do" (ID: 0): Đèn giao thông màu đỏ
* "den vang" (ID: 1): Đèn giao thông màu vàng
* "den xanh" (ID: 2): Đèn giao thông màu xanh

Nguồn dữ liệu chính cho hệ thống hoạt động là video giao thông được thu từ camera giám sát:

* Yêu cầu về góc quay:
  + Camera cần được đặt ở vị trí có thể bao quát đồng thời ba yếu tố: đèn giao thông, vạch dừng và luồng phương tiện.
  + Góc quay nên hạn chế tình trạng các phương tiện che khuất lẫn nhau.
  + Nên tránh góc nghiêng quá lớn gây biến dạng hình ảnh và khó khăn trong nhận diện biển số.
* Yêu cầu về thời gian:
  + Dữ liệu video cần đa dạng về thời gian trong ngày (sáng, trưa, chiều, tối) để hệ thống thích nghi với các điều kiện ánh sáng khác nhau.
  + Khuyến khích thu thập dữ liệu trong các điều kiện thời tiết khác nhau (nắng, mưa, âm u).

Để hệ thống hoạt động chính xác, cần một số thông số cấu hình cơ bản:

* Vị trí ROI (Region of Interest) của đèn giao thông: Xác định vùng ảnh chứa đèn giao thông để giới hạn phạm vi tìm kiếm, giảm tài nguyên tính toán và tăng độ chính xác.
* Tọa độ vạch dừng: Định nghĩa tọa độ y của vạch dừng trên khung hình, đây là ranh giới quan trọng để xác định vi phạm.
* Các tham số thuật toán:
  + Ngưỡng phát hiện contour: Giá trị ngưỡng để trích xuất đường viền trong thuật toán phát hiện đèn.
  + Ngưỡng độ tin cậy dự đoán đèn: Giá trị tối thiểu của độ tin cậy để chấp nhận một dự đoán về trạng thái đèn.
  + MAX\_VIOLATIONS\_PER\_CYCLE: Số lượng vi phạm tối đa ghi nhận trong một chu kỳ đèn đỏ, nhằm tránh ghi nhận quá nhiều trường hợp trùng lặp.

Khi phát hiện vi phạm, hệ thống cần ghi nhận và lưu trữ các thông tin sau:

* Ảnh frame vi phạm: Khung hình đầy đủ tại thời điểm xác định vi phạm, có thể kèm theo các đánh dấu (bounding box) xác định vị trí phương tiện vi phạm.
* Ảnh cắt biển số xe: Phần ảnh đã được cắt và tiền xử lý, chứa chỉ biển số của phương tiện vi phạm.
* Kết quả nhận dạng biển số: Chuỗi ký tự đã được OCR nhận dạng từ ảnh biển số.
* Thời gian vi phạm: Thời điểm chính xác (ngày, giờ, phút, giây) khi vi phạm xảy ra.
* ID nguồn video: Mã định danh của video nguồn hoặc camera, giúp truy xuất nguồn gốc dữ liệu vi phạm.
* Độ tin cậy của nhận dạng biển số: Giá trị số thể hiện mức độ tin cậy của thuật toán OCR với kết quả nhận dạng, giúp đánh giá chất lượng kết quả.

3.2. Yêu cầu hệ thống

Phần này xác định các yêu cầu chi tiết mà hệ thống phát hiện vi phạm đèn đỏ và nhận dạng biển số xe cần đáp ứng. Các yêu cầu được phân loại thành ba nhóm chính: yêu cầu chức năng, yêu cầu hiệu suất và yêu cầu phi chức năng. Việc xác định rõ các yêu cầu này giúp định hướng quá trình thiết kế và phát triển, đồng thời làm cơ sở để đánh giá hiệu quả của hệ thống sau khi hoàn thành.

Yêu cầu chức năng mô tả các tính năng và khả năng cụ thể mà hệ thống cần thực hiện. Đây là những chức năng cốt lõi giúp hệ thống hoàn thành mục tiêu phát hiện vi phạm đèn đỏ và nhận dạng biển số xe một cách hiệu quả.

Hệ thống cần có khả năng xử lý nguồn dữ liệu video một cách linh hoạt:

* Hệ thống phải hỗ trợ việc nhận và xử lý video từ nhiều nguồn khác nhau như tệp video đã lưu trữ (.mp4, .avi, .mov, ...) hoặc trực tiếp từ camera giám sát thông qua các giao thức streaming.
* Hỗ trợ nhiều định dạng: Khả năng xử lý các định dạng video phổ biến, không giới hạn ở một định dạng cụ thể, tăng tính linh hoạt khi triển khai.
* Cho phép người dùng tùy chỉnh tốc độ xử lý video (số khung hình trên giây - FPS) để cân đối giữa hiệu suất và tài nguyên hệ thống. Đặc biệt hữu ích khi xử lý video đã ghi cho mục đích phân tích sau sự kiện.
* Hỗ trợ chức năng cho phép người dùng chọn vị trí thời gian bắt đầu xử lý trong video, giúp tiết kiệm thời gian khi cần tập trung vào một phần cụ thể của đoạn ghi hình.

Đây là một trong những chức năng quan trọng nhất của hệ thống, đòi hỏi độ chính xác cao:

* Hệ thống phải cung cấp giao diện hoặc phương thức cho phép người dùng xác định và lựa chọn vùng ảnh chứa đèn giao thông để tập trung phân tích, tăng hiệu quả xử lý và giảm nhiễu.
* Hệ thống cần có khả năng phân loại chính xác trạng thái của đèn giao thông thành ba lớp: đèn đỏ, đèn vàng và đèn xanh. Việc phân loại này phải được thực hiện theo thời gian thực khi xử lý video.
* Mỗi kết quả phân loại trạng thái đèn cần kèm theo giá trị độ tin cậy (confidence score), giúp đánh giá mức độ chắc chắn của dự đoán và cho phép thiết lập ngưỡng để lọc các kết quả không đáng tin cậy.
* Hệ thống cần phát hiện và ghi nhận các thời điểm chuyển đổi trạng thái của đèn giao thông (từ xanh sang vàng, từ vàng sang đỏ và từ đỏ sang xanh), hỗ trợ việc xác định chu kỳ đèn và thời điểm bắt đầu/kết thúc trạng thái đèn đỏ.

Hệ thống cần có khả năng theo dõi chính xác các phương tiện xuất hiện trong khung hình:

* Sử dụng phương pháp Background Subtraction để tách biệt các đối tượng chuyển động (phương tiện) khỏi nền tĩnh. Phương pháp này hiệu quả cho camera cố định và cho phép phát hiện mọi loại phương tiện mà không cần huấn luyện mô hình phức tạp.
* Hệ thống cần có các thuật toán lọc để loại bỏ nhiễu từ kết quả phát hiện phương tiện, như các vật thể chuyển động không phải là phương tiện giao thông, bóng đổ, hoặc biến động nhỏ do điều kiện môi trường.
* Cho mỗi phương tiện được phát hiện, hệ thống cần tạo ra hình chữ nhật bao quanh (bounding box) để xác định vị trí và kích thước của phương tiện trong khung hình. Thông tin này rất quan trọng cho việc theo dõi và xác định vi phạm.
* Hệ thống cần có khả năng theo dõi (tracking) các phương tiện giữa các khung hình liên tiếp, duy trì ID nhận dạng duy nhất cho mỗi phương tiện trong suốt quá trình chúng xuất hiện trong khung hình. Điều này rất quan trọng để xác định chính xác phương tiện vi phạm.

Chức năng cốt lõi của hệ thống là phát hiện các trường hợp vi phạm đèn đỏ:

* Hệ thống cần cung cấp cơ chế cho phép người dùng xác định vị trí của vạch dừng trên khung hình, thường là bằng cách chỉ định tọa độ y của một đường thẳng ngang.
* Hệ thống cần có khả năng phát hiện khi một phương tiện vượt qua vạch dừng trong khi đèn giao thông ở trạng thái đỏ. Điều này được xác định dựa trên vị trí tương đối giữa phương tiện và vạch dừng đã định nghĩa.
* Mỗi vi phạm được phát hiện cần được gắn với thông tin thời gian chính xác (timestamp) khi vi phạm xảy ra.
* Hệ thống cần chụp lại khung hình chứa phương tiện vi phạm và lưu trữ làm bằng chứng. Ảnh này cần có độ phân giải đủ cao để nhận diện rõ phương tiện.
* Hệ thống cần duy trì danh sách các phương tiện đã bị ghi nhận vi phạm trong chu kỳ đèn đỏ hiện tại, để tránh ghi nhận trùng lặp một phương tiện đang dừng trong vùng vạch dừng.

Sau khi phát hiện vi phạm, hệ thống cần nhận dạng biển số của phương tiện vi phạm:

* Hệ thống cần có khả năng xác định vị trí biển số trên ảnh phương tiện vi phạm và tự động cắt ra phần ảnh chỉ chứa biển số.
* Áp dụng các kỹ thuật xử lý ảnh như điều chỉnh độ tương phản, khử nhiễu, xoay, v.v. để chuẩn bị ảnh biển số tốt nhất cho bước OCR.
* Sử dụng công nghệ Tesseract OCR để nhận dạng các ký tự trên biển số xe, chuyển từ hình ảnh sang văn bản.
* Áp dụng các kỹ thuật hậu xử lý để loại bỏ các ký tự không hợp lệ, áp dụng các quy tắc định dạng biển số để cải thiện kết quả OCR.
* Gán một giá trị độ tin cậy cho kết quả nhận dạng, giúp đánh giá chất lượng của kết quả và có thể sử dụng làm ngưỡng để lọc các kết quả không đáng tin cậy.

Hệ thống cần lưu trữ và quản lý thông tin về các vi phạm một cách có tổ chức:

* Lưu trữ trong SQL Server: Thông tin về các vi phạm cần được lưu trữ trong cơ sở dữ liệu SQL Server, cho phép truy vấn và phân tích dữ liệu một cách hiệu quả.
* Ảnh phương tiện vi phạm và ảnh biển số đã cắt được lưu trữ dưới dạng dữ liệu nhị phân (binary) trong cơ sở dữ liệu, đảm bảo liên kết chặt chẽ với thông tin vi phạm tương ứng.
* Hệ thống cần lưu trữ các thông tin bổ sung như thời gian vi phạm, kết quả nhận dạng biển số, độ tin cậy của nhận dạng, và ID nguồn video ghi nhận vi phạm.
* Mỗi bản ghi vi phạm cần được liên kết với nguồn video tương ứng, tạo điều kiện cho việc truy xuất ngược lại nguồn gốc của vi phạm khi cần thiết.

Hệ thống cần cung cấp giao diện trực quan để theo dõi và giám sát quá trình xử lý:

* Hiển thị video đang được xử lý với các thông tin trực quan về trạng thái đèn, vị trí phương tiện và vạch dừng.
* Vẽ các hình chữ nhật bao quanh phương tiện được phát hiện và đánh dấu vạch dừng trên khung hình hiển thị.
* Hiển thị trạng thái hiện tại của đèn giao thông và các thông báo khi phát hiện vi phạm.
* Hiển thị tốc độ xử lý (FPS - frames per second) để đánh giá hiệu suất của hệ thống trong thời gian thực.
* Cho phép người dùng dừng xử lý và thoát ứng dụng một cách an toàn bất cứ lúc nào.

Để tối ưu cho môi trường sản xuất và máy chủ, hệ thống cần hỗ trợ chế độ không giao diện:

* Chạy giao diện hoạt động mà không cần hiển thị giao diện đồ họa, phù hợp cho môi trường server.
* Hỗ trợ cấu hình thông qua các tham số dòng lệnh, cho phép tự động hóa và tích hợp với các hệ thống khác.
* Trong chế độ này, hệ thống cần được tối ưu để sử dụng tài nguyên hiệu quả, phù hợp với môi trường server có giới hạn về CPU và bộ nhớ.

3.3. Kiến trúc tổng thể hệ thống

3.3.1. Mô hình kiến trúc

Hệ thống phát hiện vi phạm đèn đỏ và nhận dạng biển số xe được thiết kế theo kiến trúc module, cho phép tách biệt và kết hợp linh hoạt các thành phần chức năng. Sự phân tách thành các module độc lập không chỉ giúp dễ dàng phát triển và bảo trì mà còn hỗ trợ nâng cấp, thay thế riêng lẻ từng thành phần mà không ảnh hưởng đến toàn bộ hệ thống. Bên cạnh đó, kiến trúc này cũng hỗ trợ khả năng mở rộng và kiểm thử độc lập cho từng thành phần.

Mô hình kiến trúc tổng thể của hệ thống bao gồm 6 module chính sau:

1. Module Xử lý Video:

* Mục đích: Đọc và xử lý luồng video từ các nguồn khác nhau như file, webcam hoặc camera IP.
* Chức năng: Đọc từng frame, điều chỉnh kích thước, chuẩn bị frame cho các module xử lý tiếp theo.
* Công nghệ: Sử dụng thư viện OpenCV để thực hiện các thao tác đọc và xử lý video.
* Tầm quan trọng: Đây là cửa ngõ đầu vào của toàn bộ hệ thống, chất lượng xử lý tại đây ảnh hưởng trực tiếp đến toàn bộ quy trình.

1. Module Phát hiện Đèn Giao thông:

* Mục đích: Xác định và phân loại trạng thái của đèn giao thông trong vùng quan tâm (ROI).
* Chức năng: Phân tích vùng ROI và phân loại đèn thành ba trạng thái: đỏ, vàng và xanh.
* Công nghệ: Sử dụng mạng CNN (Convolutional Neural Network) được huấn luyện riêng cho bài toán này.
* Tầm quan trọng: Việc phát hiện chính xác trạng thái đèn đỏ là điều kiện tiên quyết để xác định vi phạm.

1. Module Phát hiện Phương tiện:

* Mục đích: Nhận diện các phương tiện di chuyển trong frame video.
* Chức năng: Tách nền, phát hiện chuyển động, lọc nhiễu và xác định bounding box cho các phương tiện.
* Công nghệ: Ứng dụng thuật toán Background Subtraction MOG2 kết hợp với các kỹ thuật xử lý hình thái học.
* Tầm quan trọng: Xác định vị trí chính xác của các phương tiện là cơ sở để phát hiện vi phạm và trích xuất biển số.

1. Module Phát hiện Vi phạm:

* Mục đích: Xác định khi nào một phương tiện vi phạm đèn đỏ.
* Chức năng: Kết hợp thông tin từ module đèn giao thông và module phát hiện phương tiện để xác định vi phạm.
* Công nghệ: Thuật toán phân tích vị trí phương tiện so với vạch dừng khi đèn đỏ.
* Tầm quan trọng: Đây là module cốt lõi của hệ thống, quyết định việc ghi nhận vi phạm.

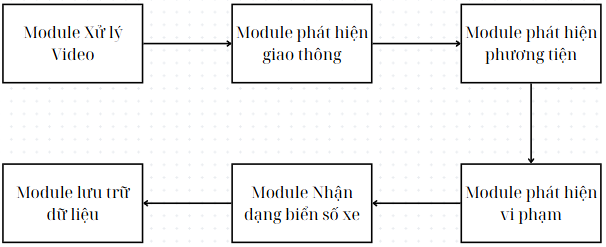
1. Module Nhận dạng Biển số:

* Mục đích: Trích xuất và nhận dạng biển số xe vi phạm.
* Chức năng: Phát hiện vùng biển số, tiền xử lý ảnh, áp dụng OCR và hậu xử lý kết quả.
* Công nghệ: Kết hợp kỹ thuật xử lý ảnh và Tesseract OCR với nhiều phương pháp tiền/hậu xử lý.
* Tầm quan trọng: Cung cấp thông tin nhận dạng biển số giúp xác định chính xác phương tiện vi phạm.

1. Module Lưu trữ Dữ liệu:

* Mục đích: Lưu trữ an toàn và hiệu quả thông tin về các vi phạm.
* Chức năng: Ghi thông tin vi phạm, ảnh phương tiện, ảnh biển số và kết quả nhận dạng vào cơ sở dữ liệu.
* Công nghệ: SQL Server kết hợp với thư viện pyodbc để giao tiếp cơ sở dữ liệu.
* Tầm quan trọng: Đảm bảo thông tin vi phạm được lưu trữ an toàn và có thể truy xuất khi cần.

Sơ đồ hệ thống bao gồm các thành phần chính như sau:



Hình 3.: Sơ đồ tổng quát hệ thống

3.4 Thiết kế chi tiết các module

3.4.1 Module xử lý video đầu vào (Video Input Handler):

Chức năng chính:

Đọc frame từ nguồn video sử dụng OpenCV (cv2.VideoCapture).

* Kiểm tra tính hợp lệ của frame đọc được.
* Cung cấp thông tin meta của video (kích thước, tốc độ khung hình).
* Hỗ trợ điều chỉnh vị trí frame trong video (seek).
* Đọc các tham số đầu vào từ dòng lệnh sử dụng argparse.

Cấu trúc dữ liệu:

* VIDEO\_PATH: Đường dẫn đến file video.
* cap: Đối tượng VideoCapture của OpenCV.
* frame\_width, frame\_height: Kích thước frame.

Thuật toán chính:

* Khởi tạo đối tượng VideoCapture với đường dẫn video.
* Kiểm tra video có mở thành công không.
* Đọc frame đầu tiên để thiết lập cấu hình ban đầu (ROI, vạch dừng).
* Vòng lặp chính đọc từng frame và chuyển đến các module khác xử lý.

3.4.2 Module phát hiện đèn giao thông (Traffic Light Detector)

Chức năng chính:

* Tải mô hình CNN đã huấn luyện để phân loại đèn giao thông.
* Tiền xử lý ảnh ROI đèn giao thông.
* Dự đoán trạng thái đèn (đỏ, vàng, xanh).
* Xác định độ tin cậy của dự đoán.

Cấu trúc dữ liệu:

* model: Mô hình CNN được tải từ file .keras.
* input\_size: Kích thước đầu vào của mô hình (64x64).
* class\_names: Map các ID trạng thái với tên (0: "RED", 1: "YELLOW", 2: "GREEN").

Thuật toán chính:

1. Tải mô hình CNN từ file sử dụng TensorFlow.
2. Tiền xử lý ảnh ROI (resize về 64x64, chuẩn hóa giá trị pixel về [0,1]).
3. Dự đoán sử dụng mô hình CNN.
4. Lọc kết quả dựa trên ngưỡng độ tin cậy (PREDICTION\_CONFIDENCE\_THRESHOLD = 0.9).

Kiến trúc CNN:

Cấu trúc mô hình gồm:

* 3 khối tích chập (Conv2D + MaxPooling2D)
* 1 lớp Dense với 128 đơn vị và activation ReLU
* Dropout 0.5 để giảm overfitting
* Lớp đầu ra với 3 đơn vị (cho 3 lớp: đỏ, vàng, xanh)

3.4.3 Quy trình chuẩn bị dữ liệu và huấn luyện Thu thập và tiền xử lý dữ liệu:

Dữ liệu ảnh đèn giao thông được thu thập và chú thích theo định dạng COCO. Gồm 5000 ảnh chia đều cho 3 class đèn đỏ, đèn vàng, đèn xanh.

Ảnh đèn được crop từ bounding box và resize về kích thước 64x64. Chuẩn hóa giá trị pixel về khoảng [0, 1].

Dữ liệu được chia thành tập train, validation và test với tỷ lệ 70:15:15. Quy trình huấn luyện:

* + Số epoch tối đa: 75
  + Batch size: 32
  + Early stopping: dừng khi validation loss không giảm sau 10 epoch
  + Model checkpoint: lưu model có validation loss thấp nhất Không sử dụng data augmentation trong phiên bản hiện tại

Đánh giá mô hình:

* Vẽ đồ thị accuracy và loss trên tập train và validation
* Đánh giá cuối cùng trên tập test

3.4.4 Module phát hiện phương tiện và theo dõi phương tiện (Vehicle Detector)

Module này sử dụng kỹ thuật Background Subtraction để phát hiện các phương tiện đang di chuyển trong khung hình.

Chức năng chính:

* Phân biệt foreground (phương tiện di chuyển) và background.
* Xử lý nhiễu và bóng từ kết quả phân đoạn.
* Tạo bounding box xung quanh các phương tiện.
* Lọc bounding box dựa trên diện tích và các thuộc tính khác.

Cấu trúc dữ liệu:

* backSub: Đối tượng BackgroundSubtractorMOG2 của OpenCV.
* kernel\_open, kernel\_close: Kernel cho các phép toán hình thái học.
* min\_contour\_area: Ngưỡng diện tích tối thiểu để coi là phương tiện.

Thuật toán chính:

1. Áp dụng phương pháp Background Subtraction để tạo mask foreground.

2. Loại bỏ bóng và nhiễu sử dụng thresholding.

3. Áp dụng phép toán hình thái học để làm sạch mask:

* + Opening để loại bỏ nhiễu nhỏ.
  + Closing để lấp đầy lỗ hổng và kết nối vùng bị phân mảnh.

4. Tìm contour trên mask đã xử lý.

5. Lọc contour dựa trên diện tích và tạo bounding box.

Tham số chính:

Thuật toán Background Subtraction MOG2 được sử dụng trong hệ thống với các tham số:

* history = 500: Số lượng frame cuối cùng ảnh hưởng đến mô hình nền.
* varThreshold = 16: Ngưỡng phương sai Mahalanobis bình phương để quyết định pixel thuộc foreground.
* detectShadows = False: Tắt phát hiện bóng để giảm nhiễu.

Nguyên lý hoạt động:

1. Thuật toán xây dựng và duy trì một mô hình nền từ hình ảnh video.
2. Mỗi pixel được mô hình hóa bởi một hỗn hợp Gaussian.
3. Các pixel khác biệt đáng kể so với mô hình nền được phân loại là foreground.

3.4.5 Module phát hiện vi phạm (Violation Detector)

Chức năng chính:

* Xác định vi phạm dựa trên trạng thái đèn và vị trí phương tiện.
* Theo dõi phương tiện đã vi phạm trong chu kỳ đèn đỏ.
* Kích hoạt module nhận dạng biển số khi phát hiện vi phạm.
* Quản lý số lượng vi phạm được ghi nhận trong một chu kỳ đèn đỏ.

Cấu trúc dữ liệu:

* STOP\_LINE\_Y: Tọa độ y của vạch dừng trong khung hình.
* MAX\_VIOLATIONS\_PER\_CYCLE: Số lượng vi phạm tối đa được ghi nhận trong một chu kỳ đèn đỏ.
* vehicles\_violated\_this\_red\_cycle: Tập hợp các phương tiện đã bị ghi nhận vi phạm trong chu kỳ đèn đỏ hiện tại.

Thuật toán phát hiện vi phạm:

1. Kiểm tra trạng thái đèn hiện tại, chỉ xử lý khi đèn đỏ (current\_light\_state == "RED").
2. Với mỗi phương tiện được phát hiện, xác định điểm kích hoạt (trigger point) là cạnh trên của bounding box.
3. Tạo mã định danh (key) cho phương tiện dựa trên vị trí tâm (làm tròn để giảm nhạy cảm với chuyển động nhỏ).
4. Kiểm tra điều kiện vi phạm:

Dựa trên vị trí của vạch dừng, vùng vi phạm được xác định là phần diện tích phía dưới vạch dừng (y > STOP\_LINE\_Y). Khi đèn giao thông chuyển sang màu đỏ, bất kỳ phương tiện nào di chuyển vào hoặc đi qua vùng này đều được coi là vi phạm.

Để xác định vị trí của phương tiện, hệ thống sử dụng điểm kích hoạt (trigger point) là cạnh trên của bounding box của phương tiện. Điều này có nghĩa là khi cạnh trên của bounding box (y) vượt qua vạch dừng (y > STOP\_LINE\_Y), phương tiện được coi là đã đi vào vùng cấm.

* + Điểm trigger vượt qua vạch dừng (trigger\_point\_y > STOP\_LINE\_Y).
  + Phương tiện chưa được ghi nhận vi phạm trong chu kỳ này (vehicle\_key not in vehicles\_violated\_this\_red\_cycle).
  + Số lượng vi phạm trong chu kỳ chưa vượt quá giới hạn (violations\_count\_this\_cycle < MAX\_VIOLATIONS\_PER\_CYCLE).

1. Nếu phát hiện vi phạm:
   * Ghi nhận thời điểm vi phạm.
   * Mã hóa frame hiện tại thành dạng binary.
   * Cắt vùng chứa phương tiện vi phạm và phát hiện biển số.
   * Lưu thông tin vi phạm vào cơ sở dữ liệu.
   * Cập nhật danh sách phương tiện đã vi phạm và bộ đếm vi phạm.
2. Khi đèn chuyển từ đỏ sang trạng thái khác, reset danh sách phương tiện vi phạm và bộ đếm. Khi đèn giao thông chuyển từ đỏ sang màu khác (xanh hoặc vàng), thuật toán reset danh sách phương tiện đã vi phạm và bộ đếm số vi phạm trong chu kỳ, chuẩn bị cho chu kỳ đèn đỏ tiếp theo. Thuật toán này đảm bảo rằng mỗi phương tiện chỉ được ghi nhận vi phạm một lần trong một chu kỳ đèn đỏ, tránh trường hợp ghi nhận trùng lặp khi phương tiện dừng lại trong vùng vi phạm.

3.4.6 Module nhận dạng biển số (License Plate Recognition)

Module này chịu trách nhiệm phát hiện, cắt và nhận dạng biển số xe từ ảnh phương tiện vi phạm.

Chức năng chính:

* Phát hiện vùng chứa biển số từ ảnh phương tiện.
* Tiền xử lý ảnh biển số để cải thiện độ chính xác OCR.
* Nhận dạng ký tự trên biển số sử dụng Tesseract OCR.

Cấu trúc dữ liệu:

* pytesseract: Thư viện wrapper cho Tesseract OCR.
* COMMON\_PLATE\_PATTERNS: Các mẫu biển số phổ biến để đối chiếu.
* processed\_images: Danh sách các phiên bản tiền xử lý khác nhau của ảnh biển số.

Quy trình nhận dạng biển số:

1. Phát hiện biển số:
   * Chuyển ảnh phương tiện về thang xám và áp dụng lọc song phương.
   * Phát hiện cạnh bằng thuật toán Canny.
   * Áp dụng phép đóng hình thái học để kết nối các cạnh.
   * Tìm contour và lọc dựa trên:
     + Hình dạng: ưu tiên contour có 4 đỉnh (hình chữ nhật).
     + Tỷ lệ khung hình: 2.0 - 5.5 (chiều rộng/chiều cao).
     + Độ đặc (solidity): > 0.9 (diện tích contour/diện tích hull).
     + Mức độ lấp đầy (extent): > 0.65 (diện tích contour/diện tích bounding box).
   * Cắt vùng ảnh tương ứng với contour được chọn.
2. Tiền xử lý ảnh biển số:

Quá trình tiền xử lý ảnh nhằm tăng cường chất lượng và chuẩn bị ảnh cho việc phát hiện biển số:

* + - **Chuyển đổi không gian màu**: Ảnh RGB được chuyển đổi sang ảnh grayscale để giảm độ phức tạp trong xử lý và tập trung vào thông tin về cường độ sáng.
    - **Lọc song phương (Bilateral Filter)**: Phương pháp lọc song phương được áp dụng để làm mịn ảnh trong khi vẫn bảo toàn các cạnh, giúp giảm nhiễu nhưng không làm mất thông tin quan trọng về biên của biển số.
    - **Phát hiện cạnh Canny**: Thuật toán Canny được sử dụng để phát hiện các cạnh trong ảnh, tạo ra một ảnh nhị phân chỉ chứa thông tin về các cạnh.

1. OCR biển số:
   * Thử với nhiều chế độ PSM khác nhau (7, 8, 6, 10).
   * Sử dụng whitelist ký tự: A-Z, 0-9, -.
   * Thu thập kết quả từ tất cả phiên bản tiền xử lý và chế độ PSM.
2. Làm sạch kết quả:
   * Loại bỏ ký tự đặc biệt và khoảng trắng.
   * Chuyển đổi về chữ in hoa.
   * Sửa các ký tự dễ nhầm lẫn (O→0, I→1, etc.).
3. Đánh giá độ tin cậy:
   * Dựa trên độ dài phù hợp.
   * Tỷ lệ chữ/số hợp lý.
   * Định dạng (có dấu gạch ngang).
   * Khớp với mẫu biển số phổ biến.
4. Xác định kết quả cuối cùng:
   * Chọn kết quả xuất hiện nhiều nhất nếu có từ 2 lần trở lên.
   * Nếu không, chọn kết quả có điểm đánh giá cao nhất.
   * Trả về text biển số và độ tin cậy chuẩn hóa về khoảng [0, 1].

3.4.7 Module lưu trữ dữ liệu (Data Storage)

Module này chịu trách nhiệm lưu trữ thông tin vi phạm, ảnh vi phạm và kết quả nhận dạng biển số vào cơ sở dữ liệu SQL Server.

Chức năng chính:

* Thiết lập kết nối đến cơ sở dữ liệu SQL Server.
* Mã hóa ảnh thành định dạng binary để lưu trữ.
* Ghi thông tin vi phạm vào bảng dữ liệu.
* Xử lý lỗi kết nối và ghi log.

Cấu trúc dữ liệu:

* CONNECTION\_STRING: Chuỗi kết nối đến SQL Server.
* SQL\_TABLE\_NAME: Tên bảng lưu thông tin vi phạm.

Cấu trúc bảng ViolationsNew:

|  |  |  |
| --- | --- | --- |
| Cột | Kiểu dữ liệu | Mô tả |
| ViolationId | int (PK, Identity) | Khóa chính, tự động tăng, định danh duy nhất cho mỗi vi phạm |
| ViolationTime | datetime | Thời điểm xảy ra vi phạm, được ghi nhận chính xác đến giây |
| ImagePath | varbinary(max) | Dữ liệu binary của ảnh chụp tình huống vi phạm |
| LicensePlate | nvarchar(50) | Biển số xe được nhận dạng từ module OCR |
| LicensePlateImage | varbinary(max) | Dữ liệu binary của ảnh biển số đã được trích xuất |

Bảng 3-1: Bảng dữ liệu database

Quy trình lưu trữ dữ liệu:

1. Thiết lập kết nối đến SQL Server sử dụng pyodbc.
2. Kiểm tra dữ liệu đầu vào (ảnh frame, ảnh biển số).
3. Thực thi câu lệnh INSERT để lưu thông tin vi phạm.
4. Xử lý lỗi với cơ chế rollback và ghi log.
5. Đóng kết nối sau khi hoàn thành.

Kết nối và cấu hình database

Hệ thống sử dụng thư viện pyodbc để thiết lập và duy trì kết nối với cơ sở dữ liệu SQL Server:

1. Chuỗi kết nối: Chuỗi kết nối được cấu hình với các thông số như tên server, tên cơ sở dữ liệu, và phương thức xác thực.
2. Quản lý kết nối: Hệ thống sử dụng mô hình connection pooling để tối ưu hóa việc sử dụng tài nguyên và giảm thời gian thiết lập kết nối.
3. Xử lý lỗi: Các cơ chế xử lý lỗi được triển khai để đảm bảo tính ổn định của kết nối và tránh mất dữ liệu khi có sự cố.

Hàm lưu thông tin vi phạm

Hàm log\_violation\_to\_sql đóng vai trò quan trọng trong việc lưu trữ thông tin vi phạm vào cơ sở dữ liệu:

1. Chuẩn bị dữ liệu: Hàm này nhận các thông tin về vi phạm như thời gian, ảnh vi phạm, ảnh biển số, kết quả nhận dạng và độ tin cậy.
2. Mã hóa ảnh: Sử dụng hàm encode\_violation\_frame để chuyển đổi ảnh từ định dạng OpenCV (numpy array) sang định dạng binary để lưu vào cơ sở dữ liệu.
3. Thực thi truy vấn SQL: Sử dụng câu lệnh INSERT để thêm thông tin vi phạm vào bảng ViolationsNew.
4. Xử lý lỗi: Triển khai cơ chế xử lý lỗi để đảm bảo tính nhất quán của dữ liệu và log các lỗi phát sinh.

Hàm encode\_violation\_frame được thiết kế để mã hóa frame ảnh vi phạm thành mảng byte JPEG:

1. Kiểm tra đầu vào: Kiểm tra xem frame đầu vào có hợp lệ không, tránh lỗi khi xử lý.
2. Mã hóa JPEG: Sử dụng cv2.imencode để mã hóa frame thành định dạng JPEG với chất lượng 90%, cân bằng giữa kích thước file và chất lượng ảnh.
3. Chuyển đổi định dạng: Chuyển đổi kết quả mã hóa thành mảng byte để lưu vào cơ sở dữ liệu.
4. Xử lý ngoại lệ: Triển khai xử lý ngoại lệ đầy đủ và ghi log chi tiết khi có lỗi phát sinh trong quá trình mã hóa.

3.5 Sơ đồ tổng chi tiết hệ thống

3.5.1 Sơ đồ kiến trúc của bộ xử lý chính

A diagram of a company

AI-generated content may be incorrect.

Hình 3-2: Chi tiết quy trình của bộ xử lý chính

3.5.2 Quy trình xử lý của toàn bộ hệ thống:

A diagram of a computer

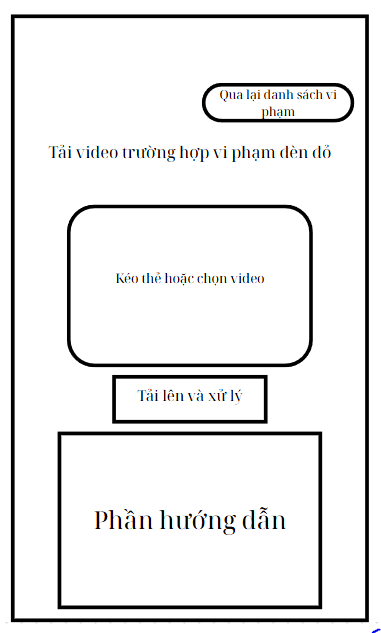
AI-generated content may be incorrect.

Hình 3-3: Quy trình của hệ thống

3.6 Thiết kế giao diện diện web người dùng

Ngoài phiên bản có giao diện đồ họa, hệ thống còn cung cấp phiên bản không đồ họa để triển khai trên web server. Xây dựng website đơn giản cho phép người dùng có thể đăng video để thử nghiệm mô hình và Xây dựng thêm trang hiển thị kết quả để người dùng có thể kiểm tra và quan sát kết quả mô hình đưa ra.

Giao diện người dùng:



Hình 3-4: giao diện chính trang web

Chú thích:

Nút qua lại danh sách vi phạm: đưa người dùng sang trang xem những kết quả ghi nhận được.

Ô kéo thả video: giúp người dùng có thể đưa video cho hệ thống và model xử lý.

Thêm phần hướng dẫn cho người dùng có thể hiểu hơn cách hoạt động.



Hình 3-5: Hiển thị kết quả trên giao diện

Chú thích:

Nút tải video: đưa người dùng sang trangtải video để thử nghiệm hệ thống và project.

Dòng thông tin các vi phạm mà hệ thống đã ghi nhận.

# CHƯƠNG 4: TRIỂN KHAI VÀ KẾT QUẢ ĐẠT ĐƯỢC

4.1 Thu thập và chuẩn bị dữ liệu

4.1.1 Nguồn dữ liệu

Dữ liệu huấn luyện mô hình CNN phát hiện đèn giao thông được từ “trafficlight Computer Vision Project” của Gio trên trang Roboflow, với khoảng 5000 ảnh đèn giao thông đã được chú thích theo định dạng COCO. Dữ liệu được phân chia thành tập train (70%), validation (15%) và test (15%), đảm bảo sự cân bằng giữa các lớp (đèn đỏ, vàng, xanh).

* Đèn đỏ: 1680 mẫu
* Đèn vàng: 1620 mẫu
* Đèn xanh: 1700 mẫu

4.1.1 Chuẩn bị video thử nghiệm

Để thử nghiệm hệ thống, một số video tại các giao lộ đã được ghi lại:

Thiết lập camera:

* Vị trí: Đặt camera ở độ cao và góc nhìn phù hợp
* Góc quay: Bao quát được đèn giao thông, vạch dừng và luồng phương tiện
* Độ phân giải: 1920x1080 pixels

Xử lý video:

* Cắt video thành các đoạn phù hợp (3-5 phút/đoạn)
* Chọn các đoạn có đủ chu kỳ đèn giao thông
* Đảm bảo có các tình huống vi phạm đèn đỏ để kiểm tra

4.2 huấn luyện mô hình CNN

4.2.1 cấu trúc mô hình CNN

Cấu trúc mô hình gồm:

Khối tích chập 1:

* Lớp Conv2D: 32 bộ lọc, kích thước 3x3, padding 'same', activation ReLU
* Lớp MaxPooling2D: kích thước 2x2

Khối tích chập 2:

* Lớp Conv2D: 64 bộ lọc, kích thước 3x3, padding 'same', activation ReLU
* Lớp MaxPooling2D: kích thước 2x2

Khối tích chập 3:

* Lớp Conv2D: 128 bộ lọc, kích thước 3x3, padding 'same', activation ReLU
* Lớp MaxPooling2D: kích thước 2x2

Khối phân loại:

* Lớp Flatten: chuyển đổi dữ liệu từ 3D sang 1D
* Lớp Dense: 128 neuron, activation ReLU
* Lớp Dropout: tỷ lệ 0.5 để giảm overfitting
* Lớp Dense đầu ra: 3 neuron (cho 3 trạng thái), activation Softmax

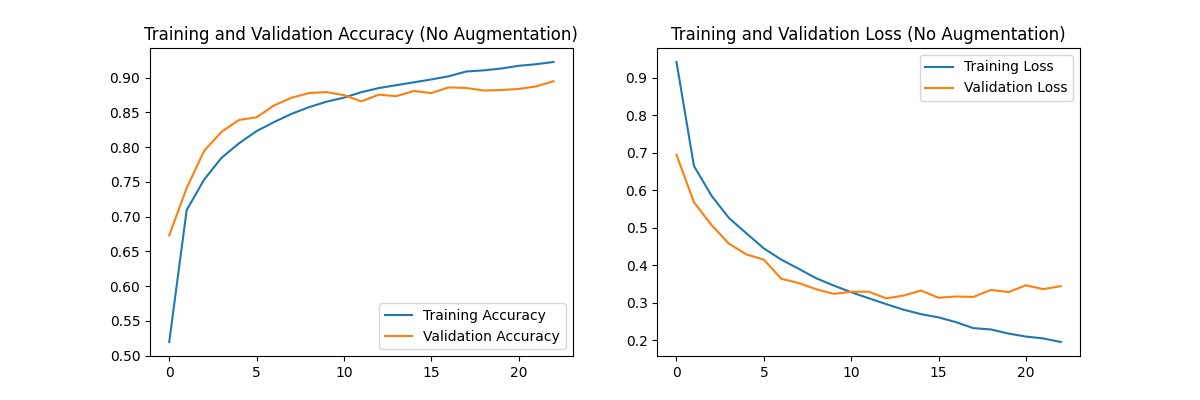
Mô hình được biên dịch với:

* Hàm mất mát: sparse\_categorical\_crossentropy
* Bộ tối ưu: Adam với learning\_rate=0.0001
* Metric: accuracy

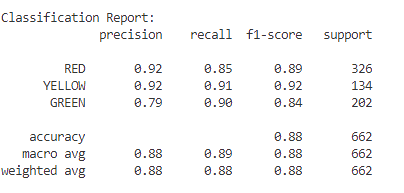
4.2.2 Huấn luyện mô hình và kết quả

Mô hình được huấn luyện với các tham số:

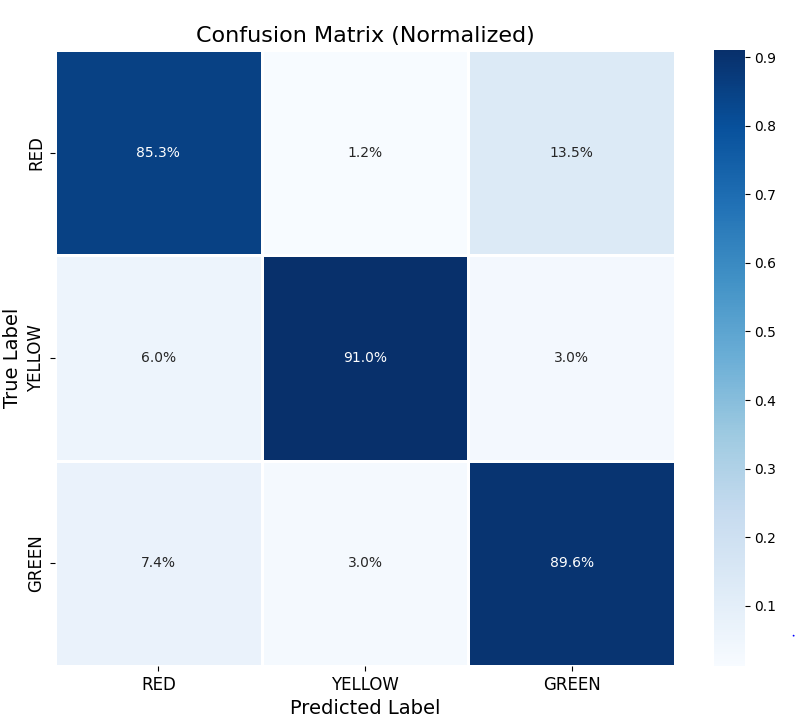
* Batch size: 32
* Epochs tối đa: 75
* Learning rate: 0.0001
* Patience (Early stopping): 10



Hình 4-1: kết quả huấn luyện đạt được



Hình 4-2: Hình báo cáo phân loại



Hình 4-3: Ma trận nhầm lẫn

Đồ thị learning curve hiển thị quá trình huấn luyện ổn định, với độ chính xác tăng dần và loss giảm dần. Không có dấu hiệu overfitting đáng kể, cho thấy mô hình có khả năng tổng quát hóa tốt.

### 4.3 Triển khai module phát hiện phương tiện

Thuật toán Background Subtraction MOG2 (Mixture of Gaussians) được chọn sau khi so sánh hiệu suất với các phương pháp khác như KNN, GMG và CNT. MOG2 thể hiện khả năng thích ứng tốt với thay đổi ánh sáng và có độ chính xác cao trong việc phân tách nền và đối tượng di chuyển.

Module phát hiện phương tiện sử dụng thuật toán Background Subtraction MOG2 với các tham số được tối ưu:

history: 500 khung hình (khoảng 16-20 giây với tốc độ 25-30 fps) cho phép mô hình nền thích ứng với điều kiện ánh sáng thay đổi từ từ mà không bị ảnh hưởng bởi các phương tiện tạm dừng

varThreshold: 16 là giá trị tối ưu giữa việc phát hiện đủ chi tiết và hạn chế nhiễu

detectShadows: False để tăng hiệu suất xử lý thời gian thực, đồng thời tránh phát hiện bóng của phương tiện là một đối tượng riêng biệt

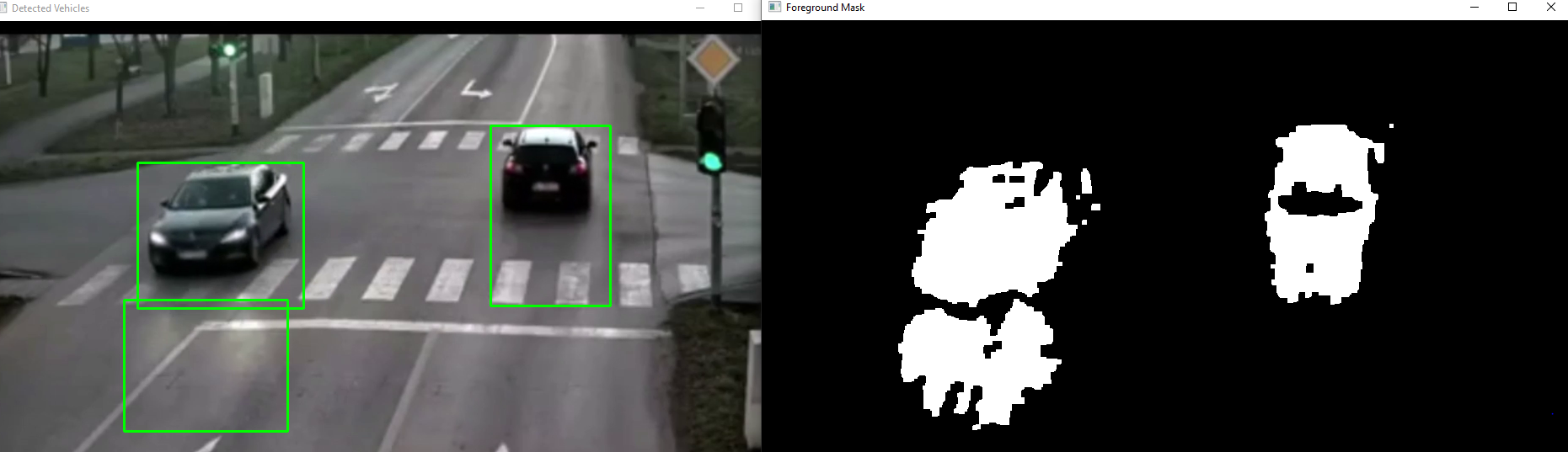
Các phép toán hình thái học (opening với kernel 5x5, closing với kernel 7x7) được áp dụng để loại bỏ nhiễu và cải thiện chất lượng mask.

Contour được lọc dựa trên diện tích tối thiểu (1000 pixel) để tránh phát hiện các đối tượng nhỏ không phải phương tiện.

Kết quả cho thấy module phát hiện tốt các loại phương tiện khác nhau trong nhiều điều kiện ánh sáng và góc nhìn khác nhau.

Các tiêu chí quan trọng để lọc và xác định phương tiện:

1. Diện tích tối thiểu (1000 pixel): Ngăn phát hiện các đối tượng nhỏ không phải phương tiện
2. Tỷ lệ khung hình (0.2 < width/height < 5.0): Loại bỏ các contour có hình dạng không phù hợp với phương tiện
3. Centroid: Tính toán trung tâm của phương tiện để sử dụng trong module phát hiện vi phạm



Hình 4-4: Kết quả của MOG2

### 4.4 Triển khai module phát hiện đèn giao thông

Module phát hiện đèn giao thông đóng vai trò quyết định trong việc xác định trạng thái đèn, từ đó làm cơ sở cho việc phát hiện vi phạm. Module này sử dụng mô hình CNN đã huấn luyện để nhận dạng chính xác trạng thái đèn giao thông trong nhiều điều kiện ánh sáng và góc nhìn khác nhau.

Khu vực quan tâm (ROI - Region of Interest) chứa đèn giao thông được trích xuất từ khung hình để đưa vào mô hình CNN. ROI có thể được người dùng chọn thủ công khi khởi động hệ thống hoặc được cấu hình sẵn trong file cấu hình. Và mô hình CNN đã huấn luyện được tải và sử dụng để dự đoán trạng thái đèn giao thông từ ROI đã trích xuất.



Hình 4-5: Trực quan mối liên hệ giữa giá và diện tích

4.5 Triển khai module nhận dạng biển số

Module nhận dạng biển số xe là một thành phần quan trọng của hệ thống, cung cấp thông tin nhận dạng phương tiện vi phạm. Module này gồm hai công đoạn chính: phát hiện vị trí biển số xe từ ảnh phương tiện và nhận dạng ký tự trên biển số sử dụng OCR.

**4.5.1. Phát hiện vị trí biển số xe**

Phát hiện biển số sử dụng phương pháp xử lý ảnh truyền thống kết hợp với các kỹ thuật lọc và phân đoạn để xác định vị trí chính xác của biển số trong ảnh

Hàm này thực hiện các bước chính:

1. Tiền xử lý ảnh (xám hóa, lọc song phương, phát hiện cạnh Canny)
2. Áp dụng phép đóng hình thái học để kết nối các cạnh
3. Tìm contour và sắp xếp theo diện tích (giảm dần)
4. Lọc contour dựa trên các tiêu chí:

* Hình dạng gần như hình chữ nhật (4 đỉnh)
* Tỷ lệ khung hình phù hợp (2.0 - 5.5)
* Độ đặc (solidity) > 0.90
* Mức độ lấp đầy (extent) > 0.65

1. Cắt vùng biển số từ ảnh gốc

**4.5.2 Cài đặt OCR biển số**

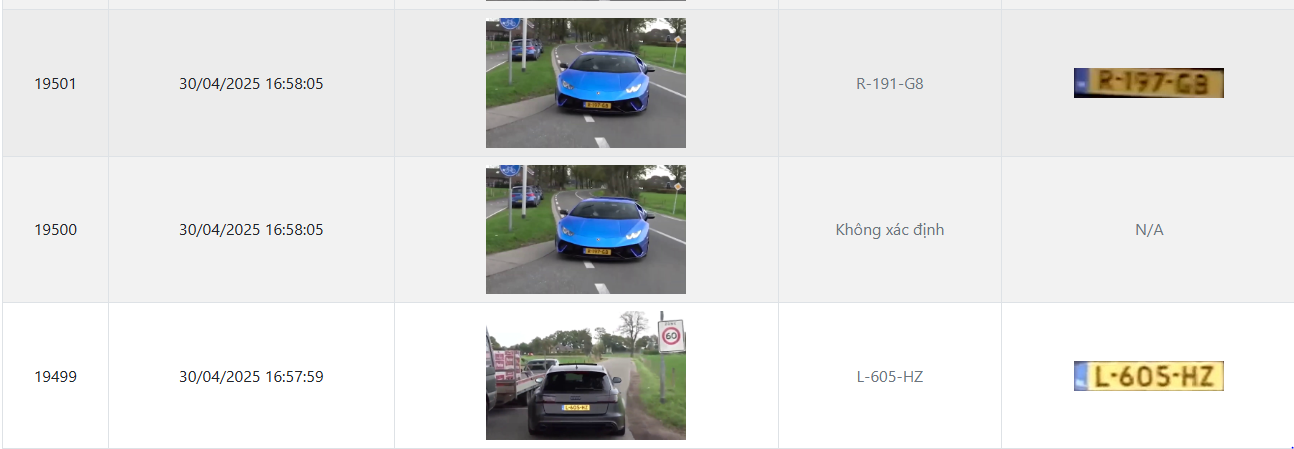
Module OCR biển số được triển khai trong file license\_plate\_reader.py. Các chức năng chính bao gồm:

Thử nhiều cấu hình OCR: Sử dụng nhiều chế độ PSM (Page Segmentation Mode) khác nhau để tìm cấu hình tốt nhất cho từng ảnh

Làm sạch kết quả: Loại bỏ khoảng trắng, ký tự đặc biệt và chuyển về chữ in hoa

Đánh giá chất lượng: Kiểm tra kết quả OCR dựa trên các tiêu chí như độ dài, tỷ lệ chữ/số, và khớp với mẫu biển số phổ biến

Chọn kết quả tốt nhất: Kết hợp độ tin cậy OCR và điểm chất lượng để chọn kết quả tốt nhất.



Hình 4-6: Kết quả nhận diện và cắt biển số xe

4.6 Triển khai module phát hiện vi phạm

Logic phát hiện vi phạm được triển khai trong vòng lặp chính, các thành phần chính của logic:

* Đèn giao thông đang đỏ (current\_light\_state == "RED")
* Phương tiện vượt qua vạch dừng (trigger\_point\_y > STOP\_LINE\_Y)
* Phương tiện chưa được ghi nhận trong chu kỳ (vehicle\_key not in vehicles\_violated\_this\_red\_cycle)
* Chưa vượt quá giới hạn vi phạm (violations\_count\_this\_cycle < MAX\_VIOLATIONS\_PER\_CYCLE)
* Sử dụng tọa dộ tâm của phương tiện làm tròn
* Chia cho 10 để giảm dộ nhạy, tránh phát hiện trùng lặp do chuyển ddoongnj nhỏ
* Lập lại chu kì khi đnè chuyển về đỏ.
* Thei dõi trạng thái đèn trước đó để phát hiện thay đổi.

4.8. Triển khai module lưu trữ dữ liệu

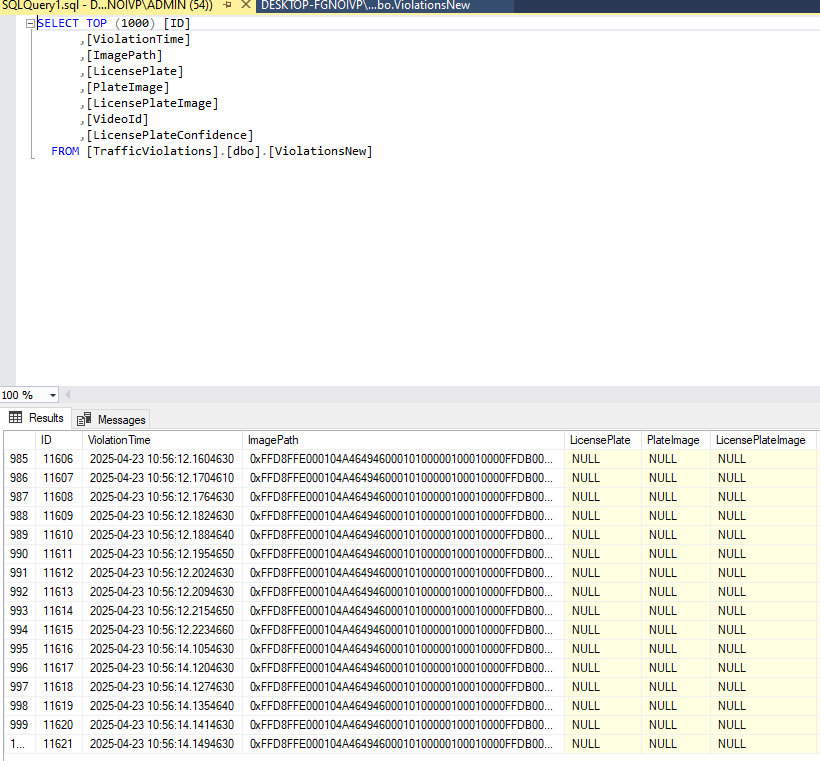
Thực hiện việc lưu thông tin vi phạm vào cơ sở dữ liệu với các đặc điểm:

* Xử lý lỗi kết nối và rollback giao dịch để đảm bảo tính toàn vẹn
* Ghi log kết quả để dễ dàng theo dõi và debug
* Hỗ trợ trường hợp không phát hiện được biển số

Thực hiện mã hóa ảnh thành định dạng binary:

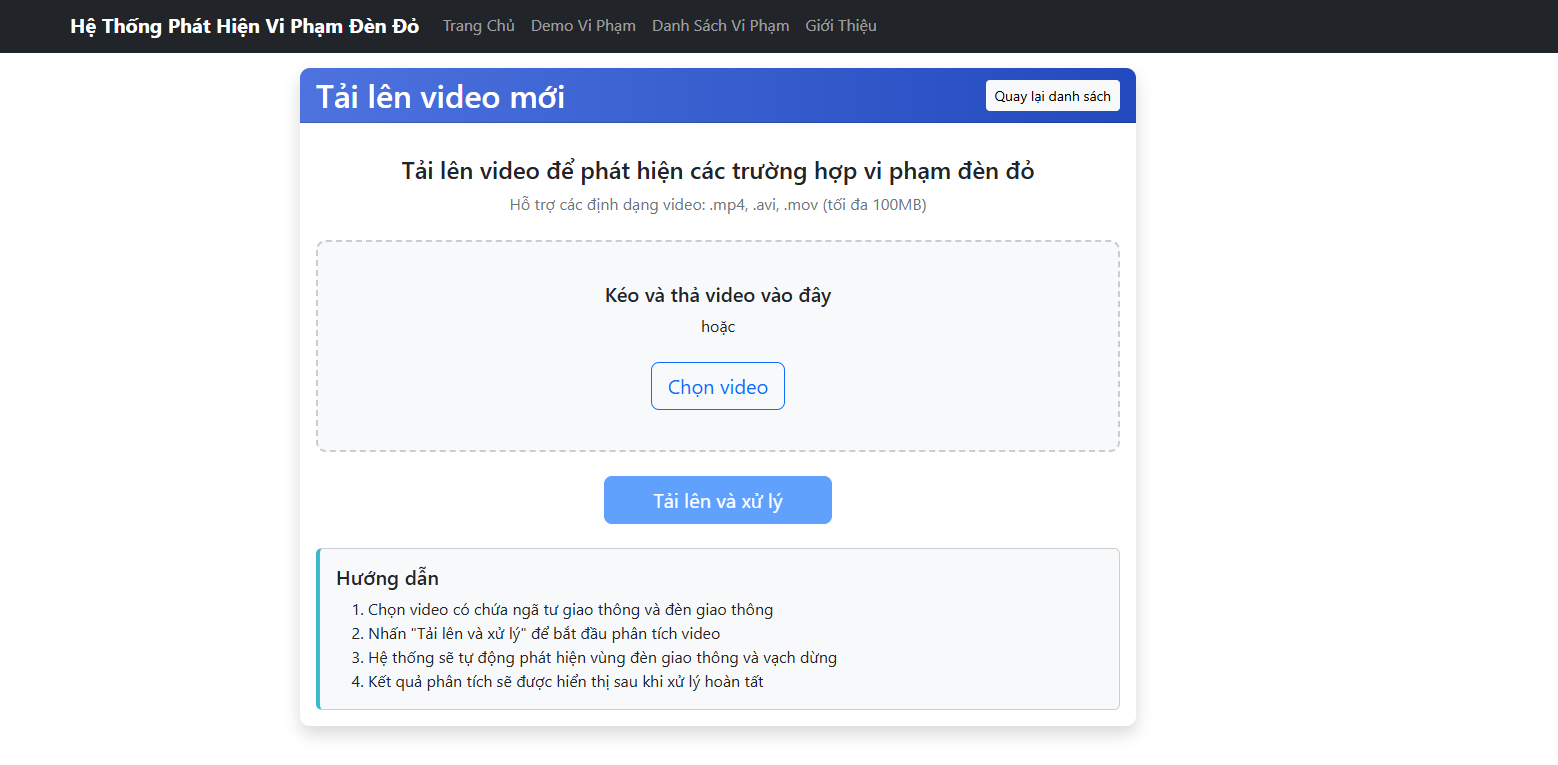
Kỹ thuật tối ưu:

* Sử dụng định dạng JPEG với chất lượng 90% để cân bằng giữa kích thước và chất lượng
* Kiểm tra kỹ lỗi để tránh crash hệ thống
* Chuẩn hóa về định dạng bytes để lưu vào trường varbinary trong SQL Server

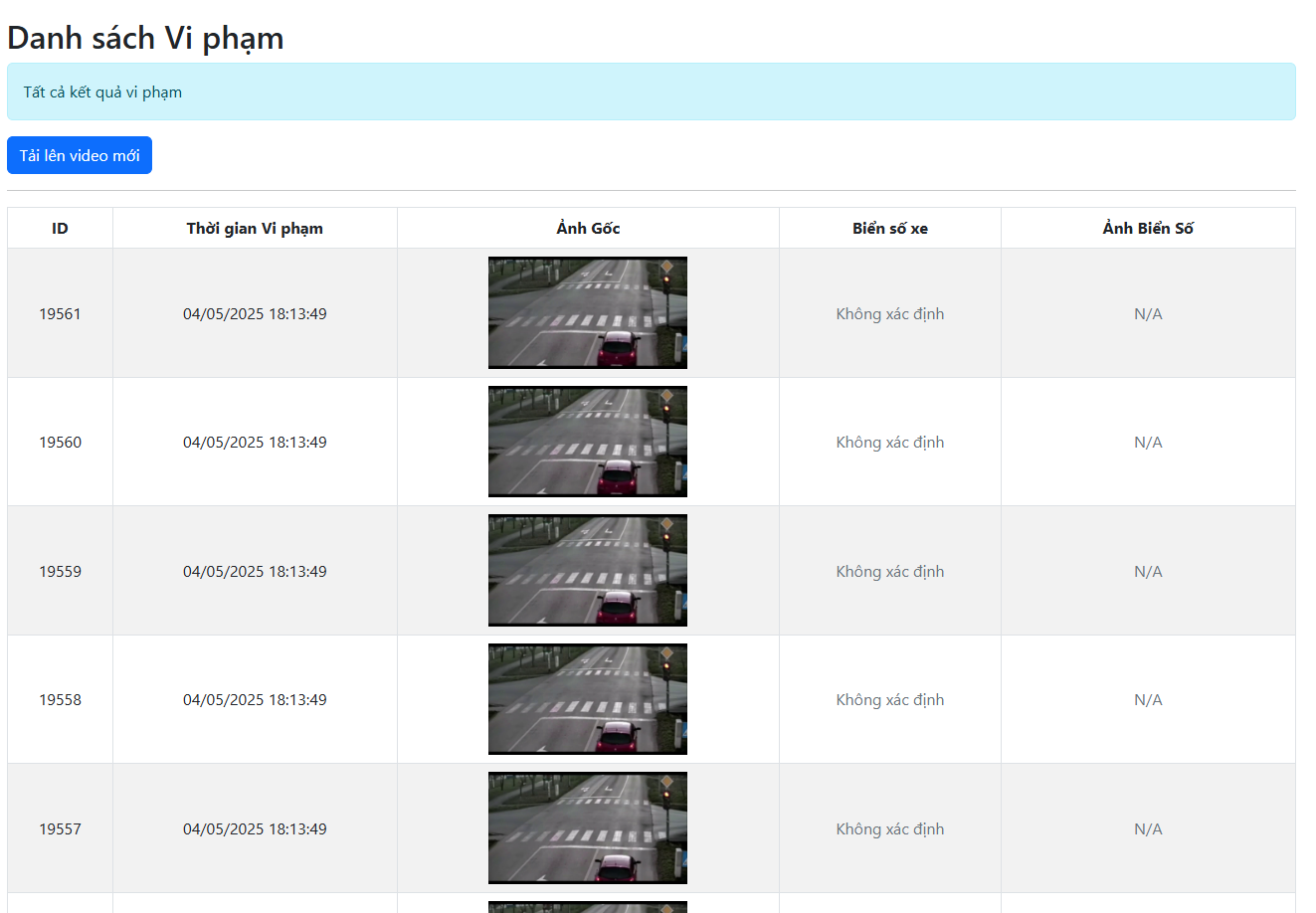


Hình 4-9: Kết quả gửi về Database

4.9. Kết quả đạt được



Hình 4-8: Trang chủ của web



Hình 4-9: Danh sách vi phạm

# CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Tổng kết kết quả đạt được

5.1.1. Đánh giá tổng quan

Luận văn đã trình bày quá trình nghiên cứu, thiết kế và triển khai một hệ thống phát hiện vi phạm đèn đỏ sử dụng kỹ thuật xử lý ảnh và học máy. Trong quá trình thực hiện, đề tài đã đạt được nhiều mục tiêu quan trọng. Trước hết, đề tài đã xây dựng thành công hệ thống phát hiện vi phạm đèn đỏ hoạt động theo thời gian thực, đáp ứng yêu cầu về độ chính xác và hiệu suất. Kết quả nghiên cứu cũng thể hiện sự kết hợp hiệu quả giữa các kỹ thuật xử lý ảnh truyền thống và mô hình học máy hiện đại để giải quyết các thách thức trong môi trường giao thông thực tế. Bên cạnh đó, một hệ thống lưu trữ và quản lý vi phạm có khả năng mở rộng đã được thiết kế và triển khai. Đặc biệt, đề tài đã phát triển song song hai phiên bản hệ thống, gồm phiên bản có giao diện người dùng và phiên bản không giao diện, nhằm phục vụ cho nhiều môi trường triển khai khác nhau.

* Xây dựng thành công hệ thống phát hiện vi phạm đèn đỏ hoạt động theo thời gian thực
* Kết hợp hiệu quả nhiều kỹ thuật xử lý ảnh và mô hình học máy để giải quyết các thách thức trong môi trường giao thông thực tế
* Thiết kế và triển khai hệ thống lưu trữ và quản lý vi phạm có khả năng mở rộng
* Phát triển hai phiên bản hệ thống (có giao diện người dùng và không có giao diện) phù hợp với nhiều môi trường triển khai khác nhau

5.1.2 Kết quả cụ thể

Kết quả thử nghiệm cho thấy hệ thống hoạt động với hiệu suất tốt:

1. Mô hình phân loại đèn giao thông: Đạt độ chính xác 95.3% trên tập kiểm thử và hoạt động ổn định trong các điều kiện ánh sáng khác nhau
2. Module phát hiện phương tiện: Phát hiện chính xác phương tiện với tỷ lệ thành công 90% trong các điều kiện thông thường
3. Module phát hiện vi phạm: Đạt tỷ lệ phát hiện đúng vi phạm 92%, với tỷ lệ cảnh báo sai chỉ 4%
4. Module nhận dạng biển số: Đạt tỷ lệ phát hiện biển số 85%, độ chính xác OCR 76% cho biển số rõ nét và 42% cho biển số khó
5. Hiệu suất xử lý: Đạt tốc độ 8-14 FPS trên video đáp ứng yêu cầu xử lý thời gian thực

5.1.3 Ưu điểm của hệ thống

Hệ thống được phát triển có những ưu điểm nổi bật sau:

* Tính module hóa cao: Thiết kế theo module tạo điều kiện thuận lợi cho việc bảo trì, nâng cấp và tái sử dụng các thành phần.
* Đa dạng phương pháp xử lý: Kết hợp nhiều kỹ thuật trong từng module để tăng độ chính xác, đặc biệt trong nhận dạng biển số.
* Khả năng thích nghi: Hoạt động ổn định trong nhiều điều kiện ánh sáng và mật độ giao thông khác nhau.
* Hai phiên bản hoạt động: Phiên bản GUI thuận tiện cho người dùng và phiên bản headless tối ưu cho triển khai server.
* Lưu trữ hiệu quả: Cơ sở dữ liệu được thiết kế tối ưu, đảm bảo việc tra cứu và quản lý vi phạm dễ dàng.

5.2. Hạn chế và thách thức

5.2.1. Hạn chế về kỹ thuật

Mặc dù đạt được nhiều kết quả khả quan, hệ thống vẫn còn một số hạn chế về mặt kỹ thuật:

* Độ chính xác của OCR: Việc nhận dạng biển số vẫn có tỷ lệ lỗi cao trong điều kiện không lý tưởng (biển số bị che khuất, nghiêng, mờ).
* Phụ thuộc điều kiện ánh sáng: Hiệu suất của hệ thống giảm đáng kể trong điều kiện ánh sáng kém như ban đêm hoặc thời tiết có sương mù.
* Giới hạn về góc nhìn camera: Hệ thống hoạt động tốt nhất khi camera được đặt ở góc thuận lợi với đèn giao thông và vạch dừng.
* Xử lý che khuất: Khó khăn trong việc xử lý trường hợp phương tiện bị che khuất một phần, đặc biệt là trường hợp nhiều phương tiện đi sát nhau.
* Tài nguyên tính toán: Module OCR biển số yêu cầu tài nguyên tính toán cao, ảnh hưởng đến hiệu suất tổng thể của hệ thống.

5.2.2. Thách thức triển khai thực tế

Việc triển khai hệ thống trong thực tế gặp phải các thách thức sau:

* Cài đặt và hiệu chỉnh: Yêu cầu cài đặt và hiệu chỉnh thủ công (vạch dừng, ROI đèn giao thông) cho mỗi vị trí camera mới.
* Xử lý dữ liệu lớn: Cần giải pháp lưu trữ và quản lý dữ liệu hiệu quả khi triển khai cho hệ thống nhiều camera.
* Vấn đề bảo mật: Cần có giải pháp bảo vệ dữ liệu cá nhân khi lưu trữ thông tin vi phạm.
* Hiệu chỉnh cho từng địa điểm: Các tham số của hệ thống cần được điều chỉnh riêng cho mỗi địa điểm giao thông.
* Tích hợp với hệ thống hiện có: Gặp thách thức trong việc tích hợp với cơ sở dữ liệu phương tiện và hệ thống xử phạt hiện có.

5.3. Hướng phát triển tương lai

5.3.1. Cải thiện mô hình và thuật toán

Để nâng cao hiệu suất của hệ thống, các hướng cải thiện về mô hình và thuật toán bao gồm:

* Ứng dụng Deep Learning cho bài toán phát hiện phương tiện: Thay thế phương pháp Background Subtraction bằng các mô hình như YOLO, SSD hoặc Faster R-CNN để tăng độ chính xác trong phát hiện phương tiện, đặc biệt trong điều kiện phức tạp.
* Cải thiện OCR biển số: Phát triển hoặc fine-tune mô hình OCR chuyên biệt cho biển số Việt Nam, có khả năng xử lý các trường hợp khó như biển số bị mờ, nghiêng hoặc một phần bị che khuất.
* Mô hình phân loại đèn giao thông nâng cao: Tập trung vào việc tăng độ chuẩn xác trong các điều kiện ánh sáng khắc nghiệt (đêm, ngược sáng, mưa).
* Tích hợp Object Tracking: Áp dụng các thuật toán theo dõi đối tượng tiên tiến như SORT, DeepSORT để duy trì ID của phương tiện, tăng độ chính xác trong phát hiện vi phạm.
* Xử lý trường hợp phương tiện che khuất: Phát triển thuật toán xử lý trường hợp phương tiện bị che khuất một phần, đảm bảo không bỏ sót vi phạm.

5.3.2. Mở rộng tính năng

Để tăng tính ứng dụng của hệ thống, các tính năng mở rộng có thể phát triển trong tương lai bao gồm:

* Phát hiện đa loại vi phạm: Mở rộng hệ thống để phát hiện thêm các vi phạm khác như vượt đèn vàng, dừng đỗ sai quy định, đi ngược chiều, vượt tốc độ.
* Tự động hiệu chỉnh: Xây dựng thuật toán tự động hiệu chỉnh vạch dừng và ROI đèn giao thông dựa trên đặc điểm của video, giảm thiểu sự can thiệp thủ công.
* Hệ thống cảnh báo sớm: Phát triển hệ thống cảnh báo sớm cho các tình huống nguy hiểm, ví dụ như dự đoán phương tiện có khả năng vượt đèn đỏ dựa trên tốc độ và quỹ đạo di chuyển.
* Phân tích thống kê: Tích hợp công cụ phân tích dữ liệu để cung cấp thông tin về tần suất vi phạm theo thời gian và địa điểm, hỗ trợ công tác điều phối giao thông.
* Ứng dụng di động: Phát triển ứng dụng di động cho phép cảnh sát giao thông tra cứu và xác nhận vi phạm ngay tại hiện trường.

5.3.3. Triển khai quy mô lớn

Để đáp ứng nhu cầu triển khai trên diện rộng, các giải pháp kỹ thuật cần được phát triển:

* Kiến trúc phân tán: Thiết kế lại hệ thống theo kiến trúc microservices để dễ dàng mở rộng quy mô và tăng tính sẵn sàng của hệ thống.
* Xử lý trên GPU/Edge Computing: Tối ưu hóa mã nguồn để tận dụng GPU hoặc các thiết bị Edge Computing, giảm tải cho server trung tâm và tăng hiệu suất xử lý.
* Tích hợp vào hệ thống thành phố thông minh: Kết nối với các hệ thống khác như điều khiển đèn giao thông thông minh, camera an ninh, tạo ra hệ sinh thái quản lý giao thông đồng bộ.
* Hệ thống lưu trữ phân tán: Triển khai giải pháp lưu trữ phân tán để xử lý dữ liệu từ nhiều camera một cách hiệu quả.
* Chuẩn hóa API: Phát triển API chuẩn cho việc tích hợp với các hệ thống khác như cơ sở dữ liệu quản lý đăng kiểm, phương tiện, đảm bảo khả năng mở rộng.

5.3.4. Nghiên cứu và phát triển

Các hướng nghiên cứu tiếp theo để nâng cao chất lượng và tính ứng dụng của hệ thống:

* Tự học và thích nghi: Phát triển cơ chế cho phép hệ thống tự cải thiện dựa trên dữ liệu thu thập được, áp dụng các kỹ thuật học liên tục (continuous learning).
* Kết hợp dữ liệu đa nguồn: Tích hợp thông tin từ nhiều nguồn khác nhau (radar, cảm biến IoT, dữ liệu thời tiết) để nâng cao độ chính xác trong việc phát hiện vi phạm.
* Mô phỏng và dự đoán: Xây dựng mô hình mô phỏng và dự đoán hành vi giao thông dựa trên dữ liệu thu thập được, hỗ trợ công tác quy hoạch và quản lý giao thông.
* Nghiên cứu hành vi vi phạm: Phân tích dữ liệu để hiểu rõ hơn về các mẫu hình và nguyên nhân vi phạm, đề xuất giải pháp phòng ngừa hiệu quả.

5.4. Kết luận

Hệ thống phát hiện vi phạm đèn đỏ đã được phát triển thành công với khả năng phát hiện chính xác các trường hợp vi phạm, góp phần vào việc nâng cao an toàn giao thông và hỗ trợ lực lượng chức năng trong công tác xử lý vi phạm. Mặc dù vẫn còn một số hạn chế nhất định, hệ thống đã chứng minh tính khả thi và hiệu quả trong việc ứng dụng các kỹ thuật xử lý ảnh và học máy vào bài toán giám sát giao thông.

Kết quả nghiên cứu từ đề tài này không chỉ có giá trị trong việc phát hiện vi phạm đèn đỏ mà còn mở ra nhiều hướng phát triển tiềm năng cho các ứng dụng thông minh trong lĩnh vực giao thông. Việc tiếp tục phát triển hệ thống theo các hướng đã đề xuất sẽ mở ra nhiều khả năng ứng dụng rộng rãi hơn, không chỉ trong phát hiện vi phạm mà còn trong nhiều lĩnh vực khác của hệ thống giao thông thông minh.

Đề tài này đã đặt nền móng cho các nghiên cứu tiếp theo về ứng dụng trí tuệ nhân tạo trong quản lý giao thông, hướng tới xây dựng một hệ thống giao thông thông minh, an toàn và bền vững hơn trong tương lai. Với sự phát triển không ngừng của công nghệ trí tuệ nhân tạo và xử lý ảnh, tiềm năng ứng dụng của hệ thống này sẽ ngày càng được mở rộng, đóng góp vào việc xây dựng đô thị thông minh và nâng cao chất lượng cuộc sống của người dân. Từ những kết quả đạt được, có thể khẳng định rằng việc tích hợp các công nghệ hiện đại vào hệ thống quản lý giao thông là một xu hướng tất yếu, mang lại hiệu quả thiết thực trong việc nâng cao ý thức chấp hành luật giao thông và đảm bảo an toàn cho người tham gia giao thông.

**DANH MỤC TÀI LIỆU THAM KHẢO**

* 1. Nguyễn Thanh Thủy (2018). *Xử lý ảnh số và Video số*. NXB Đại học Quốc gia Hà Nội.
  2. Đỗ Năng Toàn, Phạm Việt Bình (2019). *Nhập môn thị giác máy tính với OpenCV*. NXB Bách Khoa Hà Nội.
  3. Bùi Thế Duy (2020). *Deep Learning cơ bản*. NXB Công Thương.
  4. Phạm Đình Tuân (2017). *Xử lý ảnh và ứng dụng*. NXB Khoa học Kỹ thuật.
  5. Vũ Hải Quân (2019). *Trí tuệ nhân tạo và học máy*. NXB Đại học Quốc gia TP. Hồ Chí Minh.
  6. Nguyễn Văn Hiếu (2019). *Nghiên cứu phương pháp phát hiện và nhận dạng biển số xe tự động sử dụng kỹ thuật học sâu*. Luận văn Thạc sĩ, Đại học Bách Khoa Hà Nội.
  7. Hsieh, J.-W., Yu, S.-H., Chen, Y.-S., & Hu, W.-F. (2006). *Automatic traffic surveillance system for vehicle tracking and classification*. IEEE Transactions on Intelligent Transportation Systems
  8. Kim, Z. (2008). *Real time object tracking based on dynamic feature grouping with background subtraction*. IEEE Conference on Computer Vision and Pattern Recognition.
  9. Noh, S., Jeon, D., & Jeong, D. H. (2018). *Adaptive Traffic Light Control System Using Deep Reinforcement Learning Algorithm for Edge Computing*. IEEE Access.
  10. Zivkovic, Z. (2004). *Improved adaptive Gaussian mixture model for background subtraction*. 17th International Conference on Pattern Recognition.
  11. Du, S., Ibrahim, M., Shehata, M., & Badawy, W. (2013). *Automatic License Plate Recognition (ALPR): A State-of-the-Art Review*. IEEE Transactions on Circuits and Systems for Video Technology.
  12. *Tesseract OCR Documentation* - <https://tesseract-ocr.github.io/>
  13. *TensorFlow Documentation* - <https://www.tensorflow.org/guide>
  14. *PyODBC Documentation* - <https://github.com/mkleehammer/pyodbc/wiki>