**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN**

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

**Đề tài số 5: XÂY DỰNG HỆ THỐNG PHÁT HIỆN BUỒN NGỦ KHI LÁI XE BẰNG OPENCV VÀ KERAS**

**Giảng viên hướng dẫn: LƯƠNG THỊ HỒNG LAN**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Mã sinh viên** | **Sinh viên thực hiện** | **Lớp hành chính** |
| **1** | 20210331 | Lương Ngọc Văn | DCCNTT12.10.1 |
| **2** | 20210003 | Nguyễn Thị Thu Hiền | DCCNTT12.10.1 |
| **3** | 20210286 | Trần Trung Hải | DCCNTT12.10.1 |
| **4** | 20210318 | Hùng Thị Cám | DCCNTT12.10.1 |

**Bắc Ninh, năm 2024**

**Bắc Ninh, năm 2024**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÀI TẬP LỚN**

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

**Đề tài số 5: XÂY DỰNG HỆ THỐNG PHÁT HIỆN BUỒN NGỦ KHI LÁI XE BẰNG OPENCV VÀ KERAS**

**Giảng viên hướng dẫn: LƯƠNG THỊ HỒNG LAN**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Mã sinh viên** | **Sinh viên thực hiện** | **Lớp hành chính** |
| **1** | 20210331 | Lương Ngọc Văn | DCCNTT12.10.1 |
| **2** | 20210003 | Nguyễn Thị Thu Hiền | DCCNTT12.10.1 |
| **3** | 20210286 | Trần Trung Hải | DCCNTT12.10.1 |
| **4** | 20210318 | Hùng Thị Cám | DCCNTT12.10.1 |

|  |  |
| --- | --- |
|  |  |

**Bắc Ninh, năm 2024**

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

|  |  |
| --- | --- |
| **PHIẾU CHẤM THI BÀI TẬP LỚN KẾT THÚC HỌC PHẦN**  **Mã đề thi: *05: Xây dựng hệ thống phát hiện buồn ngủ khi lái xe bằng OpenCV và Keras***  **Tên học phần: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**  **Lớp Tín chỉ:**  **XATGMT.03.K12.01.LH.C04.1\_LT** | |
| **Cán bộ chấm thi 1**  *(Ký và ghi rõ họ tên)*  *Lương Thị Hồng Lan* | **Cán bộ chấm thi 2**  *(Ký và ghi rõ họ tên)* |

| **TT** | **TIÊU CHÍ** | **THANG ĐIỂM** | Lương Ngọc Văn | Nguyễn Thị Thu Hiền | Trần Trung Hải | Hùng Thị Cám |
| --- | --- | --- | --- | --- | --- | --- |
| 20210331 | 20210003 | 20210286 | 20210318 |
| **1** | **Nội dung báo cáo trên Word đầy đủ** | **3.5** |  |  |  |  |
| 1.1 | Có bố cục rõ ràng (mục lục, phần mở đầu, nội dung chính, kết luận). | 0,5 |  |  |  |  |
| 1.2 | Nội dung phân tích rõ ràng, logic. | 0,5 |  |  |  |  |
| 1.3 | Có dẫn chứng, số liệu minh họa đầy đủ. | 0,5 |  |  |  |  |
| 1.4 | Ngôn ngữ và trình bày chuẩn, không lỗi chính tả. | 0,5 |  |  |  |  |
| 1.5 | Có trích dẫn tài liệu tham khảo đúng quy cách. | 0,5 |  |  |  |  |
| 1.6 | Được trình bày chuyên nghiệp (canh lề, font chữ, khoảng cách dòng hợp lý). | 0,5 |  |  |  |  |
| 1.7 | Tài liệu đầy đủ, bám sát yêu cầu của đề bài. | 0,5 |  |  |  |  |
| **2** | **Nội dung thuyết trình đầy đủ** | **1.0** |  |  |  |  |
| 2.1 | Trình bày tự tin, phát âm rõ ràng, mạch lạc. | 0,5 |  |  |  |  |
| 2.2 | Nội dung thuyết trình đúng trọng tâm, không lan man. | 0,5 |  |  |  |  |
| **3** | **Slides báo cáo đầy đủ nội dung + Hỏi đáp** | **3.0** |  |  |  |  |
| 3.1 | Slides có bố cục rõ ràng (mở đầu, nội dung, kết luận). | 0,5 |  |  |  |  |
| 3.2 | Thiết kế slides đẹp, chuyên nghiệp (màu sắc, hình ảnh minh họa). | 0,5 |  |  |  |  |
| 3.3 | Nội dung trên slides ngắn gọn, dễ hiểu, súc tích. | 0,5 |  |  |  |  |
| 3.4 | Nội dung slides phù hợp với nội dung báo cáo. | 0,5 |  |  |  |  |
| 3.5 | Trả lời câu hỏi đầy đủ, chính xác. | 0,5 |  |  |  |  |
| 3.6 | Trả lời câu hỏi tự tin, thuyết phục. | 0,5 |  |  |  |  |
| **4** | **Code đầy đủ** | **2.5** |  |  |  |  |
| 1.1 | Code được trình bày rõ ràng, có chú thích đầy đủ. | 0,5 |  |  |  |  |
| 1.2 | Code chạy đúng, không lỗi. | 0,5 |  |  |  |  |
| 1.3 | Code tối ưu, không dư thừa. | 0,5 |  |  |  |  |
| 1.4 | Đáp ứng đầy đủ các yêu cầu chức năng theo đề bài. | 0,5 |  |  |  |  |
| 1.5 | Có tính sáng tạo hoặc cải thiện so với yêu cầu. | 0,5 |  |  |  |  |
| **TỔNG ĐIỂM BẰNG SỐ:** | | **10** |  |  |  |  |
| **TỔNG ĐIỂM BẰNG CHỮ:** | | *Mười tròn* |  |  |  |  |

**MỤC LỤC**

[LỜI CẢM ƠN 9](#_Toc184656155)

[MỞ ĐẦU 10](#_Toc184656156)

[1. Giới thiệu về đề tài 10](#_Toc184656157)

[2. Lý do chọn đề tài 10](#_Toc184656158)

[CHƯƠNG 1. CƠ SỞ LÝ THUYẾT 12](#_Toc184656159)

[1. Xử lý ảnh và thị giác máy tính 12](#_Toc184656160)

[1.1. Khái niệm 12](#_Toc184656161)

[1.2: Tầm quan trọng 12](#_Toc184656162)

[1.3. Ứng dụng 13](#_Toc184656163)

[2. Mô hình học máy 14](#_Toc184656164)

[2.1. Machine learning 14](#_Toc184656165)

[2.2. CNN 15](#_Toc184656166)

[3. Công cụ và thư viện sử dụng 16](#_Toc184656167)

[3.1. Công cụ Visual Studio Code. 16](#_Toc184656168)

[3.2. Ngôn ngữ lập trình Python. 17](#_Toc184656169)

[3.3. Thư viện Keras 18](#_Toc184656170)

[3.4. Thư viện OpenCV 18](#_Toc184656171)

[CHƯƠNG 2. XÂY DỰNG HỆ THỐNG 20](#_Toc184656172)

[1. Phát biểu bài toán 20](#_Toc184656173)

[1.1.Đặt vấn đề 20](#_Toc184656174)

[1.2.Mô tả bài toán 20](#_Toc184656175)

[2. Phân tích hệ thống 21](#_Toc184656176)

[2.1. Mô hình tổng quan hệ thống 21](#_Toc184656177)

[2.2.Thuật toán nhận diện khuôn mặt và mắt 21](#_Toc184656178)

[3. Xây dựng chương trình 24](#_Toc184656179)

[3.1. Xử lý đầu video vào 24](#_Toc184656180)

[3.2. Nhận dạng khuôn mặt 24](#_Toc184656181)

[3.3. Nhận dạng đôi mắt 25](#_Toc184656182)

[3.4. Phát hiện buồn ngủ và cảnh báo 26](#_Toc184656183)

[3.5. Mô tả cấu trúc mã nguồn và giải thích ý nghĩa các file mã nguồn 26](#_Toc184656184)

[CHƯƠNG 3. THỰC NGHIỆM CHƯƠNG TRÌNH 28](#_Toc184656185)

[1. Thu thập dữ liệu 28](#_Toc184656186)

[1.1. Bộ dữ liệu đầu vào(dataset) 28](#_Toc184656187)

[1.2. Tiền xử lý dữ liệu 29](#_Toc184656188)

[1.3. Huấn luyện mô hình 29](#_Toc184656189)

[2. Môi trường thực nghiệm 30](#_Toc184656190)

[2.1. Môi trường triển khai 31](#_Toc184656191)

[2.2. Bộ huấn luyện 31](#_Toc184656192)

[KẾT LUẬN 32](#_Toc184656193)

[1. Kết quả đạt được 32](#_Toc184656194)

[2. Kết quả chưa đạt được 32](#_Toc184656195)

[3. Dự kiến phát triển và cải tiến thuật toán 32](#_Toc184656196)

[TÀI LIỆU THAM KHẢO 34](#_Toc184656197)

# LỜI CẢM ƠN

Lời đầu tiên, chúng em xin được gửi lời cảm ơn chân thành tới các Thầy, các Cô trong trường Đại Học Công Nghệ Đông Á, đặc biệt là các Thầy, Cô trong Viện Kỹ thuật và Công nghệ những người đã trực tiếp giảng dạy, giúp đỡ chúng em trong suốt những năm tháng qua, chúng em xin cảm ơn nhà trường đại học Công Nghệ Đông Á và Viện Kỹ thuật và Công nghệ đã tạo điều kiện cho chúng em điều kiện tốt nhất để được học tập và rèn luyện các kỹ năng cần tại nhà trường.

Chúng em là sinh viên ngành công nghệ thông tin tại trường đại học Công Nghệ Đông Á luôn có ý chí phấn đấu để xứng đáng với những gì thầy, cô đã mong đợi với chúng em môn xử lý ảnh và thị giác máy tính là nền tảng cho chúng em biết thêm về kiến thức và kỹ năng tự làm việc và học tập tốt hơn. Và đồng thời cũng là nền tảng cho chúng em về sau khi làm thực tập, và sau khi tốt nghiệp.

Để hoàn thành được bản báo cáo này ngoài sự cố gắng nỗ lực của cả nhóm chúng em trong đó còn có sự tận tình giúp đỡ và giảng dạy của các thầy, cô giáo trong Viện Kỹ thuật và Công nghệ Trường Đại học Công Nghệ Đông Á. Chúng em xin được gửi lời cảm ơn chân thành đến các thầy, cô trong Viện Kỹ Thuật và Công nghệ đã đồng hành và luôn nhiệt tình hướng dẫn chúng em để có thể hoàn thành đề tài như ngày hôm nay.

Do năng lực của chúng em còn có nhiều hạn chế thế nên chúng em sẽ không tránh khỏi những thiếu sót khi hoàn thiện báo cáo. Thế nên chúng em rất mong nhận được những góp ý và chỉ bảo của các thầy, cô để bài báo cáo của chúng em được hoàn thiện và đầy đủ hơn nữa.

Kính chúc cô hạnh phúc và thành công hơn nữa trong sự nghiệp “trồng người”. Kính chúc cô luôn dồi dào sức khỏe để tiếp tục dìu dắt nhiều thế hệ học trò đến những bến bờ tri thức.

MỞ ĐẦU

## ****1. Giới thiệu về đề tài****

- Vấn đề tai nạn giao thông do buồn ngủ

* Buồn ngủ khi lái xe là nguyên nhân hàng đầu gây ra các vụ tai nạn giao thông nghiêm trọng. Tình trạng này thường xảy ra vào ban đêm hoặc trên những chuyến đi dài, khi mà tài xế không thể kiểm soát cơn buồn ngủ.
* Thống kê từ các nghiên cứu về tai nạn giao thông cho thấy rằng số lượng tai nạn do buồn ngủ không hề nhỏ và thường dẫn đến hậu quả nặng nề, như tử vong hoặc chấn thương nghiêm trọng.
* Mục đích của hệ thống
* Hệ thống cảnh báo buồn ngủ là một công cụ nhằm bảo vệ tính mạng con người. Nó theo dõi tình trạng của người lái và phát hiện sớm các dấu hiệu buồn ngủ, từ đó phát ra cảnh báo nhằm giữ cho tài xế luôn tỉnh táo.

****2. Lý do chọn đề tài****

Đề tài **“Xây dựng hệ thống cảnh báo buồn ngủ khi lái xe sử dụng OpenCV và Keras”** Bắt nguồn từ nhu cầu cấp bách trong việc đảm bảo an toàn giao thông. Buồn ngủ khi lái xe là một trong những nguyên nhân chính gây ra tai nạn nghiêm trọng, đặc biệt là đối với các tài xế lái xe đường dài, vào ban đêm hoặc trong điều kiện làm việc căng thẳng. Các tai nạn do buồn ngủ không chỉ gây thiệt hại về tài sản mà còn đe dọa trực tiếp tính mạng của tài xế và những người tham gia giao thông.Từ những vấn đề trên nhóm chúng em đã chọn đề tài này bởi vì thông qua sự tìm hiểu đối với đề tài chúng em nhận thấy khả năng nhận diện kịp thời trạng thái buồn ngủ của tài xế thông qua phân tích hình ảnh khuôn mặt và trạng thái mắt, từ đó đưa ra các cảnh báo an toàn, góp phần giảm thiểu nguy cơ tai nạn giao thông.

Hệ thống này có thể hoạt động liên tục trong thời gian thực, đưa ra cảnh báo kịp thời khi phát hiện tài xế rơi vào trạng thái buồn ngủ. Không cần các thiết bị đeo trên cơ thể, hệ thống chỉ sử dụng camera có sẵn trên xe và phần mềm xử lý ảnh, tạo điều kiện thuận lợi cho việc triển khai trong thực tế.

CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

1. Xử lý ảnh và thị giác máy tính

1.1. Khái niệm

* **Xử lý ảnh**

Xử lý ảnh (Tiếng Anh: digital image processing) hay xử lý ảnh kỹ thuật số là sự sử dụng máy tính số để xử lý các ảnh kỹ thuật số thông qua một thuật toán. Là một phân ngành tổng thể của xử lý tín hiệu số, xử lý ảnh kỹ thuật số có nhiều lợi thế hơn so với xử lý ảnh analog. Nó cho phép áp dụng nhiều thuật toán hơn cho dữ liệu đầu vào và có thể tránh được các vấn đề như sự tích tụ nhiễu và méo hình trong quá trình xử lý. Vì ảnh được định nghĩa trên hai chiều (hoặc nhiều hơn), xử lý ảnh kỹ thuật số có thể được mô hình hóa dưới dạng hệ thống đa chiều. Sự phát triển của xử lý ảnh kỹ thuật số chủ yếu bị ảnh hưởng bởi ba yếu tố: đầu tiên là sự phát triển của máy tính; thứ hai là sự phát triển của toán học (đặc biệt là sáng tạo và cải tiến lý thuyết toán rời rạc); thứ ba là nhu cầu ứng dụng rộng rãi trong môi trường, nông nghiệp, quân sự, công nghiệp và khoa học y tế ngày càng tăng.

* **Thị giác máy tính**

  Thị giác máy tính là một công nghệ mà máy sử dụng để tự động nhận biết và mô tả hình ảnh một cách chính xác và hiệu quả. Ngày nay, các hệ thống máy tính có quyền truy cập vào khối lượng lớn hình ảnh và dữ liệu video bắt nguồn từ hoặc được tạo bằng điện thoại thông minh, camera giao thông, hệ thống bảo mật và các thiết bị khác. Ứng dụng thị giác máy tính sử dụng trí tuệ nhân tạo và máy học (AI/ML) để xử lý dữ liệu này một cách chuẩn xác cho việc xác định đối tượng và nhận diện khuôn mặt, cũng như phân loại, đề xuất, giám sát và phát hiện.

1.2: Tầm quan trọng

Tuy rằng công nghệ xử lý thông tin hình ảnh đã xuất hiện từ lâu nhưng phần lớn quy trình vẫn đòi hỏi sự can thiệp của con người, tốn nhiều thời giờ và dễ bị lỗi. Ví dụ: việc triển khai hệ thống nhận diện khuôn mặt trước đây yêu cầu nhà phát triển phải gắn thẻ thủ công hàng ngàn hình ảnh bằng các điểm dữ liệu chính, chẳng hạn như chiều rộng sống mũi và khoảng cách giữa hai mắt. Tự động hóa các tác vụ này đòi hỏi sức mạnh điện toán rộng lớn vì dữ liệu hình ảnh không có cấu trúc và phức tạp để máy tính có thể sắp xếp. Do đó, ứng dụng thị giác tốn kém và hầu hết các tổ chức không thể tiếp cận.

Ngày nay, tiến bộ trong lĩnh vực này kết hợp với sự tăng cường đáng kể của sức mạnh điện toán đã cải thiện cả quy mô và độ chính xác của quy trình xử lý dữ liệu hình ảnh. Các hệ thống thị giác máy tính được hỗ trợ bởi tài nguyên điện toán đám mây hiện giờ trở nên dễ tiếp cận với tất cả mọi người. Bất kỳ tổ chức nào cũng có thể sử dụng công nghệ này để xác minh danh tính, kiểm duyệt nội dung, phân tích video phát trực tuyến, phát hiện lỗi và nhiều tính năng khác.

1.3. Ứng dụng

* **Bảo mật và an toàn**

Chính phủ và doanh nghiệp sử dụng thị giác máy tính để tăng cường bảo mật tài sản, cơ sở và trang thiết bị. Ví dụ: camera và cảm biến giám sát các không gian công cộng, cơ sở công nghiệp và môi trường bảo mật cao. Các thiết bị này gửi cảnh báo tự động nếu có điều gì bất thường xảy ra, chẳng hạn như một người đột nhập trái phép vào khu vực bị hạn chế.

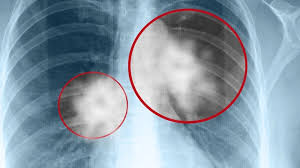
* **Chăm sóc sức khỏe**

Y tế là một trong những ngành đi đầu trong áp dụng công nghệ thị giác máy tính. Đáng chú ý, phân tích hình ảnh y khoa tạo hiển thị trực quan mô và các cơ quan giúp chuyên gia y tế chẩn đoán nhanh và chính xác, từ đó cải thiện kết quả điều trị và kéo dài tuổi thọ. Ví dụ:

-      Phát hiện khối u thông qua phân tích nốt ruồi và tổn thương trên da

-      Phân tích tia X tự động

-      Phát hiện triệu chứng từ phép chụp MRI



*Hình 1: Hình chụp X quang*



*Hình 2: Hình chụp MRI*

* **Phương tiện tự hành**

Công nghệ xe tự hành sử dụng thị giác máy tính để nhận biết hình ảnh theo thời gian thực và xây dựng bản đồ 3D từ nhiều camera được trang bị cho vận chuyển tự động. Phương tiện tự hành có khả năng phân tích hình ảnh và xác định những người đi đường khác, biển báo trên đường, người đi bộ hoặc chướng ngại vật.

Ở các phương tiện bán tự hành, thị giác máy tính sử dụng máy học (ML) để giám sát hành vi của người lái. Ví dụ: nó tìm kiếm các dấu hiệu phân tâm, mệt mỏi và buồn ngủ dựa theo vị trí đầu của người lái xe, theo dõi mắt và chuyển động của phần trên cơ thể. Nếu công nghệ nhận được một số dấu hiệu cảnh báo nhất định, nó sẽ cảnh báo cho người lái xe và giảm nguy cơ xảy ra sự cố khi lái xe.

* **Nông nghiệp**

Ứng dụng thị giác máy tính tăng cường hoạt động tổng thể của ngành nông nghiệp, từ tăng năng suất đến giảm chi phí với công nghệ tự động hóa thông minh. Hình ảnh vệ tinh cũng như cảnh quay UAV giúp phân tích các vùng đất rộng lớn và cải thiện các phương pháp canh tác. Ứng dụng thị giác máy tính tự động hóa các tác vụ như giám sát tình trạng ruộng đồng, xác định bệnh ở cây trồng, kiểm tra độ ẩm của đất và dự đoán thời tiết cũng như năng suất cây trồng. Giám sát động vật bằng thị giác máy tính là một chiến lược canh tác thông minh quan trọng khác

2. Mô hình học máy

2.1. Machine learning

Machine learning là một nhánh của trí tuệ nhân tạo (AI) và khoa học máy tính, tập trung vào việc sử dụng dữ liệu và thuật toán để bắt chước hành động của con người, dần dần cải thiện độ chính xác của nó.

Machine learning còn là một thành phần quan trọng của lĩnh vực khoa học dữ liệu đang phát triển. Thông qua việc sử dụng các phương pháp thống kê, các thuật toán được đào tạo để đưa ra các phân loại hoặc dự đoán và khám phá những thông tin chi tiết từ chính các dự án khai thác dữ liệu.

Thông qua các thông tin chi tiết có được để thúc đẩy việc đưa ra quyết định đối với các ứng dụng và doanh nghiệp, tác động mạnh đến các chỉ số tăng trưởng. Khi dữ liệu lớn tiếp tục nhu cầu mở rộng và phát triển đòi hỏi nhu cầu tuyển dụng các nhà khoa học dữ liệu sẽ tăng lên. Họ sẽ được yêu cầu giúp xác định các câu hỏi kinh doanh có liên quan nhất và dữ liệu để trả lời chúng.

Bài toán của machine learning thường được chia làm hai loại là dự đoán (prediction) và phân loại (classification). Các bài toán dự đoán thường là giá nhà, giá xe, v.v, còn các bài toán phân loại thường là nhận diện chữ viết tay, đồ vật, v.v.

*Các thuật toán phổ biến của machine learning:*

-      Neural networks: Mô phỏng cách thức hoạt động của bộ não con người, với một số lượng khổng lồ các nút xử lý được liên kết. Neural networks là thuật toán được dùng trong việc nhận dạng các mẫu và đóng một vai trò quan trọng trong các ứng dụng bao gồm dịch ngôn ngữ tự nhiên, nhận dạng hình ảnh, nhận dạng giọng nói và tạo hình ảnh.

-  Linear regression: Thuật toán này được sử dụng để dự đoán các giá trị số, dựa trên mối quan hệ tuyến tính giữa các giá trị khác nhau.

-      Logistic regression: Thuật toán giúp đưa ra dự đoán cho các biến phản hồi phân loại, chẳng hạn như câu trả lời “có/không” cho các câu hỏi. Nó có thể được sử dụng cho các ứng dụng như phân loại thư rác và kiểm soát chất lượng trên dây chuyền sản xuất.

-      Clustering: Các thuật toán phân cụm có thể xác định các mẫu trong dữ liệu để nó có thể được nhóm lại. Máy tính có thể giúp các nhà khoa học dữ liệu bằng cách xác định sự khác biệt giữa các mục dữ liệu mà con người đã bỏ qua.

-      Decision trees: Là thuật toán được sử dụng để dự đoán giá trị số (hồi quy) và phân loại dữ liệu. Decision trees sử dụng một chuỗi phân nhánh của các quyết định được liên kết có thể được biểu diễn bằng sơ đồ cây. Một trong những ưu điểm của decision trees là chúng dễ xác thực và kiểm tra, không giống thuật toán Neural networks.

-  Random forests: Trong một khu rừng ngẫu nhiên, thuật toán máy học dự đoán một giá trị hoặc danh mục bằng cách kết hợp các kết quả từ một số cây quyết định. 

2.2. CNN

Convolutional Neural Network (CNN – Mạng nơ-ron tích chập) là một trong  những mô hình Deep Learning tiên tiến. Nó giúp cho chúng ta xây dựng được những  hệ thống thông minh với độ chính xác cao như hiện nay.

Mạng CNN là một tập hợp các lớp Convolution chồng lên nhau và sử dụng các hàm nonlinear activation như ReLU và tanh để kích hoạt các trọng số trong các node. Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo.

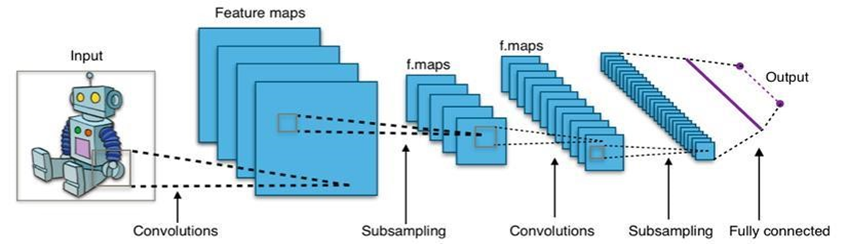
Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu  tượng hơn cho các lớp tiếp theo. Trong mô hình mạng truyền ngược (feedforward  neural network) thì mỗi neural đầu vào (input node) cho mỗi neural đầu ra trong các  lớp tiếp theo.

Mô hình này gọi là mạng kết nối đầy đủ (fully connected layer) hay mạng toàn  vẹn (affine layer). Còn trong mô hình CNN thì ngược lại. Các layer liên kết được với  nhau thông qua cơ chế convolution.

Layer tiếp theo là kết quả convolution từ layer trước đó, nhờ vậy mà ta có được các kết nối cục bộ. Như vậy mỗi neuron ở lớp kế tiếp sinh ra từ kết quả của filter áp đặt lên một vùng ảnh cục bộ của neuron trước đó.

Mỗi một lớp được sử dụng các filter khác nhau thông thường có hàng trăm hàng nghìn filter như vậy và kết hợp kết quả của chúng lại. Ngoài ra có một số layer khác như pooling/subsampling layer dùng để chắt lọc lại các thông tin hữu ích hơn (loại bỏ các thông tin nhiễu).

Trong quá trình huấn luyện mạng (training) CNN tự động học các giá trị qua các lớp filter dựa vào cách thức mà bạn thực hiện. Ví dụ trong tác vụ phân lớp ảnh, CNN sẽ cố gắng tìm ra thông số tối ưu cho các filter tương ứng theo thứ tự raw pixel > edges > shapes > facial > high-level features. Layer cuối cùng được dùng để phân lớp ảnh.



Hình 3. Mô hình CNN

Trong mô hình CNN có 2 khía cạnh cần quan tâm là **tính bất biến** (Location Invariance) và **tính kết hợp** (Compositionality). Với cùng một đối tượng, nếu đối tượng này được chiếu theo các góc độ khác nhau (translation, rotation, scaling) thì độ chính xác của thuật toán sẽ bị ảnh hưởng đáng kể.

Pooling layer sẽ cho bạn tính bất biến đối với phép dịch chuyển (translation),  phép quay (rotation) và phép co giãn (scaling). Tính kết hợp cục bộ cho ta các cấp độ  biểu diễn thông tin từ mức độ thấp đến mức độ cao và trừu tượng hơn thông qua  convolution từ các filter.

         Đó là lý do tại sao CNN cho ra mô hình với độ chính xác rất cao. Cũng giống  như cách con người nhận biết các vật thể trong tự nhiên.

3. Công cụ và thư viện sử dụng

3.1. Công cụ Visual Studio Code.

Visual Studio Code (VS Code) là một trình soạn thảo mã nguồn mở, nhẹ và mạnh mẽ do Microsoft phát triển, ra mắt vào năm 2015. Với khả năng chạy trên các hệ điều hành như Windows, macOS, và Linux, VS Code nhanh chóng trở thành công cụ phổ biến của lập trình viên nhờ tính linh hoạt và hiệu suất cao. Trình soạn thảo này hỗ trợ đa dạng ngôn ngữ lập trình như Python, JavaScript, Java, C++, C#, PHP, và nhiều ngôn ngữ khác, cùng với tính năng Intellisense thông minh, giúp hoàn thành mã, gợi ý hàm, và phát hiện lỗi trong khi viết.

VS Code tích hợp Git để quản lý phiên bản mã nguồn và có terminal tích hợp, giúp lập trình viên thao tác nhanh chóng mà không cần chuyển đổi công cụ. Kho tiện ích mở rộng (Marketplace) của VSCode cung cấp hàng ngàn extension hữu ích như Prettier để định dạng mã, Python Extension để hỗ trợ Python, và Live Server cho phát triển web thời gian thực. Giao diện của VS Code cũng dễ dàng tùy chỉnh với nhiều chủ đề và bố cục, mang lại trải nghiệm làm việc tối ưu cho người dùng.

3.2. Ngôn ngữ lập trình Python.

Python là một ngôn ngữ lập trình cấp cao, thường được ứng dụng để phát triển website và phần mềm, tự động hóa công việc cũng như phân tích dữ liệu. Đây là một ngôn ngữ đa năng, cho phép xây dựng nhiều loại chương trình khác nhau mà không bị giới hạn bởi một lĩnh vực cụ thể nào, ví dụ như hình ảnh trong đồ họa có thể thấy một chương trình đang hoạt động.

Tính linh hoạt của Python, kèm theo tính thân thiện với người dùng mới, đã góp phần làm cho nó trở thành một trong những ngôn ngữ lập trình phổ biến nhất hiện nay. Theo khảo sát từ công ty phân tích RedMonk, Python đứng thứ hai trong danh sách ngôn ngữ lập trình được yêu thích nhất trong năm 2021.

Về mặt ứng dụng, Python chủ yếu được dùng trong phát triển website và phần mềm, tự động hóa tác vụ, phân tích và trực quan hóa dữ liệu. Nhờ vào dễ học, nhiều người không chuyên về lập trình, như kế toán viên hay nhà khoa học, đã áp dụng Python trong các công việc hàng ngày liên quan đến quản lý tài chính.

Trong lĩnh vực phân tích dữ liệu và học máy, Python đóng vai trò quan trọng khi cho phép các nhà phân tích thực hiện các phép toán thống kê phức tạp, tạo ra các hình ảnh dữ liệu, xây dựng thuật toán machine learning, xử lý và phân tích dữ liệu, cũng như hoàn thành các nhiệm vụ liên quan khác. Python có khả năng tạo ra nhiều loại biểu đồ khác nhau như biểu đồ đường, thanh, hình tròn và 3D. Đặc biệt, nó còn cung cấp nhiều thư viện hỗ trợ lập trình viên trong việc phân tích dữ liệu và học máy, chẳng hạn như TensorFlow và Keras. Khi bạn phải thực hiện những công việc lặp đi lặp lại, tự động hóa bằng Python sẽ giúp bạn làm việc hiệu quả hơn. Quá trình viết mã để tạo ra các quy trình tự động này được xem là scripting. Trong lĩnh vực lập trình, tự động hóa có thể hỗ trợ kiểm tra lỗi trên nhiều tập tin, chuyển đổi định dạng tệp, thao tác đơn giản và loại bỏ bản sao trong dữ liệu.

3.3. Thư viện Keras

     Keras là một thư viện mã nguồn mở được sử dụng rộng rãi trong lĩnh vực deep learning (học sâu) và mạng nơ-ron. Nó được thiết kế để giúp các nhà phát triển xây dựng và thử nghiệm các mô hình học sâu một cách dễ dàng và linh hoạt.

Mục tiêu chính của Keras là cung cấp một API đơn giản, trực quan và dễ sử dụng, giúp người dùng tập trung vào việc xây dựng mô hình mà không cần lo lắng về chi tiết kỹ thuật phức tạp.

Một trong những điểm mạnh của Keras là khả năng tích hợp với các framework học sâu khác như TensorFlow và Theano, cho phép người dùng tận dụng sức mạnh của cả hai thế giới. Đây chính là điều giúp Keras trở thành một lựa chọn hàng đầu cho các dự án học sâu từ phân loại ảnh, dự đoán chuỗi thời gian, đến các ứng dụng trong lĩnh vực ngôn ngữ tự nhiên và nhiều lĩnh vực khác.

3.4. Thư viện OpenCV

OpenCV, viết tắt của Thư viện Thị giác Máy tính Mở, là một thư viện phần mềm dành cho máy học và thị giác máy tính. Ban đầu được phát triển bởi Intel, hiện tại nó do cộng đồng của OpenCV Foundation duy trì. OpenCV là một thư viện mã nguồn mở lớn phục vụ cho ứng dụng thị giác máy tính, học máy và xử lý hình ảnh. Hiện nay, thư viện này đóng vai trò quan trọng trong các hệ thống thời gian thực, có ý nghĩa tích cực trong nhiều ứng dụng hiện đại. Với OpenCV, người dùng có thể thực hiện việc phân tích hình ảnh và video để phát hiện các đối tượng, khuôn mặt hoặc chữ viết tay.

Khi kết hợp với nhiều thư viện khác như NumPy, Python có khả năng xử lý cấu trúc mảng của OpenCV để thực hiện các phép phân tích. Để nhận diện một mẫu hình ảnh và các đặc điểm khác của nó, ta sử dụng không gian vectơ và thực hiện các phép toán phù hợp.

OpenCV phiên bản đầu tiên là 1.0 và được phát hành dưới giấy phép BSD, do đó có thể sử dụng miễn phí cho cả mục đích học thuật và thương mại. Thư viện này hỗ trợ nhiều ngôn ngữ lập trình như C++, C, Python và Java, cùng với việc hoạt động trên các hệ điều hành như Windows, Linux, Mac OS, iOS và Android. Mục tiêu chính khi thiết kế OpenCV là để phục vụ cho các ứng dụng thời gian thực, với mã nguồn được tối ưu hóa cho khả năng xử lý đa lõi.

Một số ứng dụng phổ biến của OpenCV bao gồm:

•        Nhận diện khuôn mặt

•        Giám sát tự động và kiểm tra

•        Đếm số lượng người (như lưu lượng đi bộ trong trung tâm mua sắm)

•        Đếm xe trên đường và theo dõi tốc độ

•        Tác phẩm nghệ thuật tương tác

•        Phát hiện bất thường trong quy trình sản xuất (sản phẩm lỗi)

•        Ghép hình ảnh từ chế độ xem phố

•        Tìm kiếm và truy xuất video/hình ảnh

•        Điều khiển và dẫn đường cho ô tô và robot tự lái

•        Nhận diện đối tượng

•        Phân tích hình ảnh y tế

•        Tạo cấu trúc 3D từ chuyển động trong phim

•        Nhận diện quảng cáo trên truyền hình

Chức năng nổi bật của OpenCV bao gồm:

•        Nhập/xuất hình ảnh/video, xử lý và hiển thị

•        Phát hiện đối tượng và đặc trưng

•        Thị giác máy tính đơn kính hoặc stereo dựa trên hình học

•        Học máy và phân cụm

**CHƯƠNG 2. XÂY DỰNG HỆ THỐNG**

1. Phát biểu bài toán

1.1.Đặt vấn đề

Trong thời đại ngày nay, việc di chuyển bằng ô tô đã trở thành nhu cầu thiết yếu của con người, phục vụ cho công việc, cuộc sống hàng ngày, và các hoạt động giao thông. Tuy nhiên, với sự gia tăng số lượng phương tiện tham gia giao thông, vấn đề an toàn giao thông ngày càng trở nên cấp bách. Một trong những nguyên nhân chính gây ra các vụ tai nạn nghiêm trọng trên đường là do tài xế rơi vào tình trạng mệt mỏi, buồn ngủ khi lái xe. Theo thống kê, tình trạng buồn ngủ là nguyên nhân trực tiếp dẫn đến một số lượng lớn các vụ tai nạn giao thông, đặc biệt là những vụ xảy ra vào ban đêm hoặc trong các chuyến đi dài.

Để giải quyết vấn đề này, các hệ thống cảnh báo buồn ngủ cho tài xế đã ra đời và được áp dụng rộng rãi. Tuy nhiên, các giải pháp hiện tại vẫn còn nhiều hạn chế về độ chính xác, tính linh hoạt, và khả năng triển khai trên nhiều loại phương tiện khác nhau. Nhiều hệ thống sử dụng các thiết bị đo sinh học như cảm biến nhịp tim, những phương pháp này đòi hỏi tài xế phải đeo thiết bị liên tục, gây bất tiện. Do đó, một giải pháp khác tiếp cận thông qua việc sử dụng camera để giám sát hành vi khuôn mặt và cử động mắt của tài xế trở thành một hướng đi mới đầy tiềm năng.

1.2.Mô tả bài toán

    Xây dựng một hệ thống cảnh báo buồn ngủ cho tài xế trong thời gian thực bằng cách sử dụng công nghệ xử lý ảnh và video với OpenCV. Hệ thống sẽ sử dụng camera giám sát hành trình trên xe để liên tục theo dõi khuôn mặt và mắt của tài xế, phân tích trạng thái của mắt (mở/nhắm) nhằm phát hiện dấu hiệu buồn ngủ và cảnh báo kịp thời nếu tài xế có nguy cơ ngủ gật.

* Mục tiêu:

•        Nhận diện khuôn mặt người lái xe: Sử dụng các kỹ thuật xử lý ảnh và video để nhận diện khuôn mặt trong thời gian thực. Đây là bước đầu tiên và quan trọng, vì hệ thống cần phải định vị chính xác khuôn mặt để tiếp tục nhận diện các bộ phận khác.

•        Phát hiện trạng thái của mắt: Sử dụng landmarks trên khuôn mặt để xác định vị trí mắt và tính toán tỷ lệ khía cạnh mắt (EAR). Hệ thống phải phân biệt được mắt đang mở hay đang nhắm với độ chính xác cao.

•        Phát hiện dấu hiệu buồn ngủ: Nếu mắt nhắm liên tục trong một khoảng thời gian nhất định (ví dụ, 2-3 giây), hệ thống sẽ xác định đó là dấu hiệu của buồn ngủ. Việc phát hiện này cần phải dựa vào dữ liệu liên tục từ các khung hình video, đảm bảo hệ thống không bỏ sót các biểu hiện quan trọng.

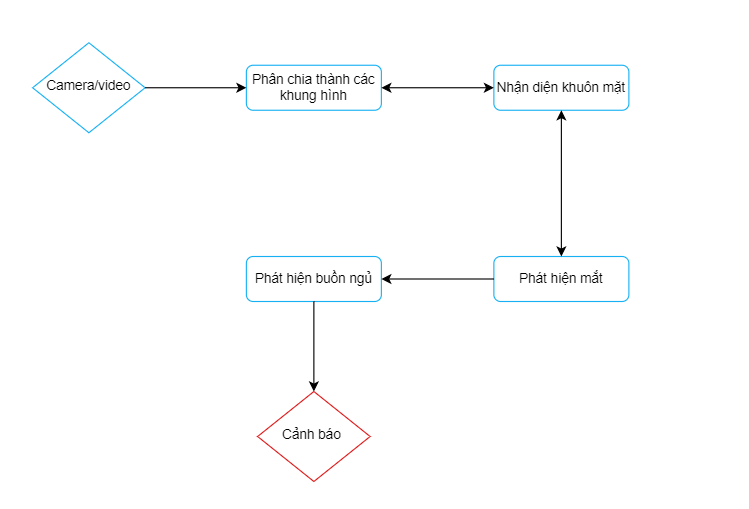
•        Cảnh báo kịp thời: Khi phát hiện dấu hiệu buồn ngủ, hệ thống cần phát ra cảnh báo âm thanh để đánh thức và làm tỉnh táo người lái xe, từ đó giảm thiểu nguy cơ xảy ra tai nạn.

 -   Đầu vào của hệ thống: Luồng video thời gian thực từ camera giám sát

   -   Đầu ra hệ thống: đưa ra cảnh báo nếu phát hiện dấu hiệu buồn ngủ

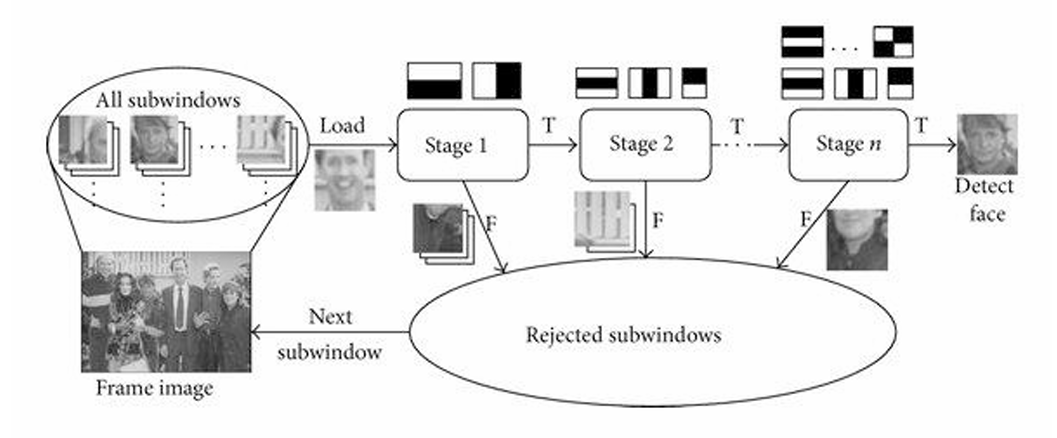
2. Phân tích hệ thống

2.1. Mô hình tổng quan hệ thống



*Hình 4. Mô hình tổng quan hệ thống*

2.2.Thuật toán nhận diện khuôn mặt và mắt



Hình 5. Thuật toán Haar Cascade

Chú thích: Frame Image (Hình ảnh đầu vào):

• Đây là hình ảnh gốc mà thuật toán xử lý. Nó có thể là một khung hình từ video hoặc ảnh tĩnh.

* All Subwindows (Tất cả các cửa sổ con):

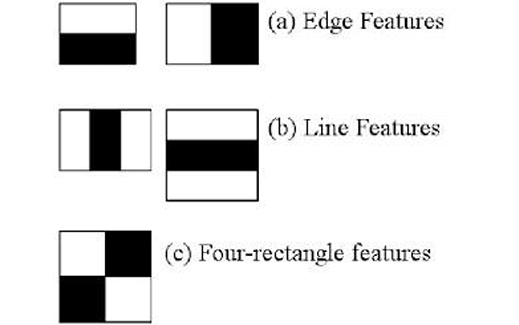
• Hình ảnh đầu vào được chia thành nhiều cửa sổ nhỏ (subwindows) với kích thước khác nhau. Mỗi cửa sổ sẽ được kiểm tra xem có chứa khuôn mặt hay không.

* Stage Processing (Xử lý từng giai đoạn):

• Mỗi cửa sổ con được gửi qua nhiều giai đoạn (Stages).

    • Tại mỗi giai đoạn, các Haar-like features được tính toán và kiểm tra bằng bộ phân loại (classifier). Những cửa sổ không đạt yêu cầu sẽ bị loại bỏ sớm, giảm khối lượng tính toán.

Trong đó:



 a) Là các bộ lọc bắt các cạnh trong ảnh

 b) Bắt các đường thẳng trong ảnh

 c) Về đặc trưng 4 hình vuông

* Rejected Subwindows (Các cửa sổ bị loại bỏ):

• Tất cả các cửa sổ không đạt tiêu chí ở bất kỳ giai đoạn nào sẽ bị loại bỏ.

* Stage n (Giai đoạn cuối):

• Sau khi vượt qua toàn bộ các giai đoạn, cửa sổ được coi là phát hiện khuôn mặt thành công.

* Detect Face (Kết quả):

• Cuối cùng, các cửa sổ được xác định là chứa khuôn mặt sẽ được trả về.

* Người dùng (User)

- Mục đích: Người dùng là người sử dụng hệ thống để phát hiện buồn ngủ dựa trên trạng thái mắt và nhận cảnh báo khi cần thiết.

 - Tương tác:

+ Khởi động hệ thống bằng cách mở camera.

+ Nhận cảnh báo khi hệ thống phát hiện trạng thái mắt nhắm quá lâu.

* Camera

- Mục đích: Camera được sử dụng để ghi lại khung hình video trong thời gian thực.

- Tương tác:

+ Nhận yêu cầu từ người dùng để mở và bắt đầu ghi hình.

+ Gửi các khung hình video đã ghi đến hệ thống xử lý để chuẩn bị cho quá trình nhận diện.

* Hệ thống nhận diện mắt (Eye Recognition System)

- Mục đích: Sử dụng mô hình đã huấn luyện để phân tích trạng thái mắt trong các khung hình video thời gian thực.

- Tương tác:

+ Nhận các khung hình đã tiền xử lý từ hệ thống tiền xử lý.

+ Sử dụng mô hình đã huấn luyện để dự đoán trạng thái mắt (mở hoặc nhắm).

+ Kích hoạt hệ thống cảnh báo nếu phát hiện mắt nhắm quá lâu.

* Hệ thống cảnh báo (Alert System)

- Mục đích: Gửi cảnh báo cho người dùng khi phát hiện trạng thái mắt nhắm quá lâu, cảnh báo nguy cơ buồn ngủ.

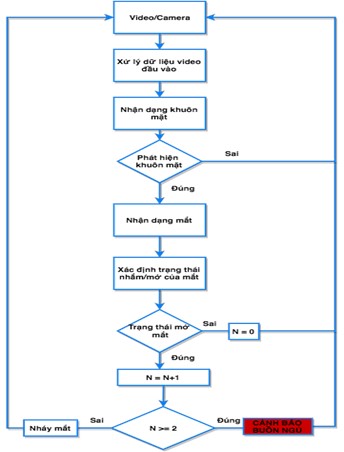
 - Tương tác:

+ Nhận tín hiệu kích hoạt từ hệ thống nhận diện mắt khi phát hiện mắt nhắm quá lâu. + Gửi thông báo hoặc cảnh báo đến người dùng để thông báo về tình trạng buồn ngủ.

* Quy trình hoạt động chính:
* Người dùng mở camera để khởi động hệ thống.
* Camera ghi lại khung hình video và gửi đến hệ thống tiền xử lý.
* Tiền xử lý làm sạch và chuẩn bị khung hình để nhận diện.
* Trong thời gian thực, hệ thống nhận diện mắt sử dụng mô hình đã huấn luyện để phân tích trạng thái mắt trong từng khung hình.
* Nếu hệ thống phát hiện mắt nhắm quá lâu, hệ thống cảnh báo được kích hoạt và gửi cảnh báo đến người dùng.

3. Xây dựng chương trình

Phần này sẽ trình bày chi tiết các bước và thuật toán thực hiện chương trình phát hiện tình trạng buồn ngủ của người lái xe. Sơ đồ các thuật toán của hệ thống đề xuất được trình bày như sau:



3.1. Xử lý đầu video vào

Để phục vụ cho việc rút trích đặc trưng Haar-like, bước tiền xử lý khá đơn giản. Các ảnh đầu vào đòi hỏi phải ở dạng mức xám, do đó, tất cả ảnh màu đầu vào sẽ được chuyển hết về dạng mức xám.

3.2. Nhận dạng khuôn mặt

1. **Khởi tạo bộ dò tìm**

OpenCV đã tích hợp nhiều phân loại (classifiers) để huấn luyện cho các đối tượng nhận dạng như khuôn mặt, đôi mắt, nụ cười... Đó là những file XML được lưu trữ trong thư mục “opencv /data / haarcascades”. Đầu tiên chúng ta cần phải load các phân loại (classifiers) XML cần thiết. Ở đây chúng ta cần các bộ phân loại để huấn luyện cho khuôn mặt và đôi mắt như sau:

* + haarcascade\_frontalface\_alt.xml: bộ dữ liệu huấn luyện (training) cho quá trình xử lý mặt trước.
  + haarcascade\_mcs\_lefteye.xml, haarcascade\_eye\_tree\_eyeglasses.xml, haarcascade\_eye.xml: các bộ dữ liệu huấn luyện (training) cho quá trình xử lý đôi mắt.

1. **Thực hiện dò tìm**

Bước tiếp theo này sẽ là dò tìm khuôn mặt trong ảnh. Nếu đối tượng được tìm thấy, nó sẽ được trả về vị trí của khuôn mặt đã được phát hiện theo cấu trúc **Rect(x,y,w,h)**. Một khi chúng ta nhận được những vị trí đó, chúng ta có thể tạo ROI cho khuôn mặt và xử lý nhận dạng đôi mắt trên ROI này. Phương pháp nhận diện khuôn mặt dựa vào đặc trưng Haarlike kết hợp Adaboost được cài sẵn trong bộ thư viện Opencv. Để sử dụng phương pháp này trong OpenCV, chương trình đã sử dụng hàm **detectMultiScale.**

3.3. Nhận dạng đôi mắt

Hàm detectMultiScale sau khi tìm kiếm xong sẽ trả về bộ giátrị gồm tọa độ gốc của khung chứa khuôn mặt x,y; chiều dài, rộng của khung w,h. Các giá trị này nằm trong mảng faces. Cấu trúc for….in sẽ duyệt qua toàn bộ các bộ giá trị này, với mỗi bộ giá trị ta dùng hàm rectangle để vẽ một hình chữ nhật lên ảnh ban đầu img với tọa độ 2 điểm trái trên và phải dưới: **(x,y),(x+w,y+h). (0,255,0**) là màu sẽ vẽ hình chữ nhật. Như vậy là đã xác định xong các khuôn mặt có trong ảnh. **roi\_gray = gray[y:y+h, x:x+w] và roi\_color = img[y:y+h, x:x+w]** là 2 biến ảnh màu và ảnh xám các khuôn mặt đã tìm thấy.

3.4. Phát hiện buồn ngủ và cảnh báo

Đối với mỗi mắt (mắt trái và mắt phải), lấy vùng mắt từ khung hình, chuyển đổi sang ảnh xám, thay đổi kích thước về (24x24), chuẩn hóa dữ liệu (chia cho 255) và đưa vào mô hình để dự đoán.

Kết quả dự đoán (rpred cho mắt phải và lpred cho mắt trái) sẽ trả về 1 nếu mắt mở và 0 nếu mắt nhắm.

Cập nhật điểm số dựa trên trạng thái mắt:

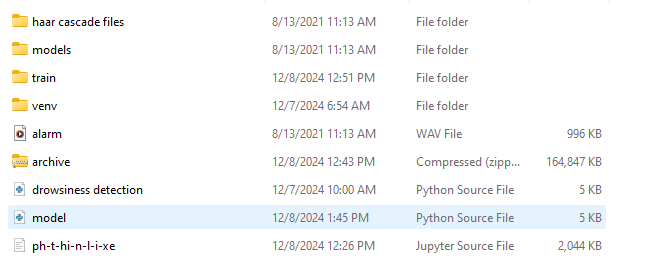
* Nếu cả hai mắt đều nhắm (cả rpred và lpred đều là 0), điểm số sẽ tăng lên (người có khả năng buồn ngủ).
* Nếu ít nhất một trong hai mắt mở, điểm số sẽ giảm xuống (người vẫn tỉnh táo).

Hiển thị điểm số:

* cv2.putText để hiển thị trạng thái "Closed" hoặc "Open" và điểm số trên khung hình.
* Cảnh báo khi điểm số quá cao (người có thể buồn ngủ):
* Nếu điểm số vượt quá 15, một bức ảnh của khung hình hiện tại được lưu lại (cv2.imwrite).
* Đồng thời, âm thanh cảnh báo sẽ phát ra, và viền màn hình được vẽ bằng màu đỏ (cv2.rectangle), với độ dày thicc thay đổi.

3.5. Mô tả cấu trúc mã nguồn và giải thích ý nghĩa các file mã nguồn

3.5.1. Cấu trúc file mã nguồn



3.5.2. Giải thích ý nghĩa các file mã nguồn

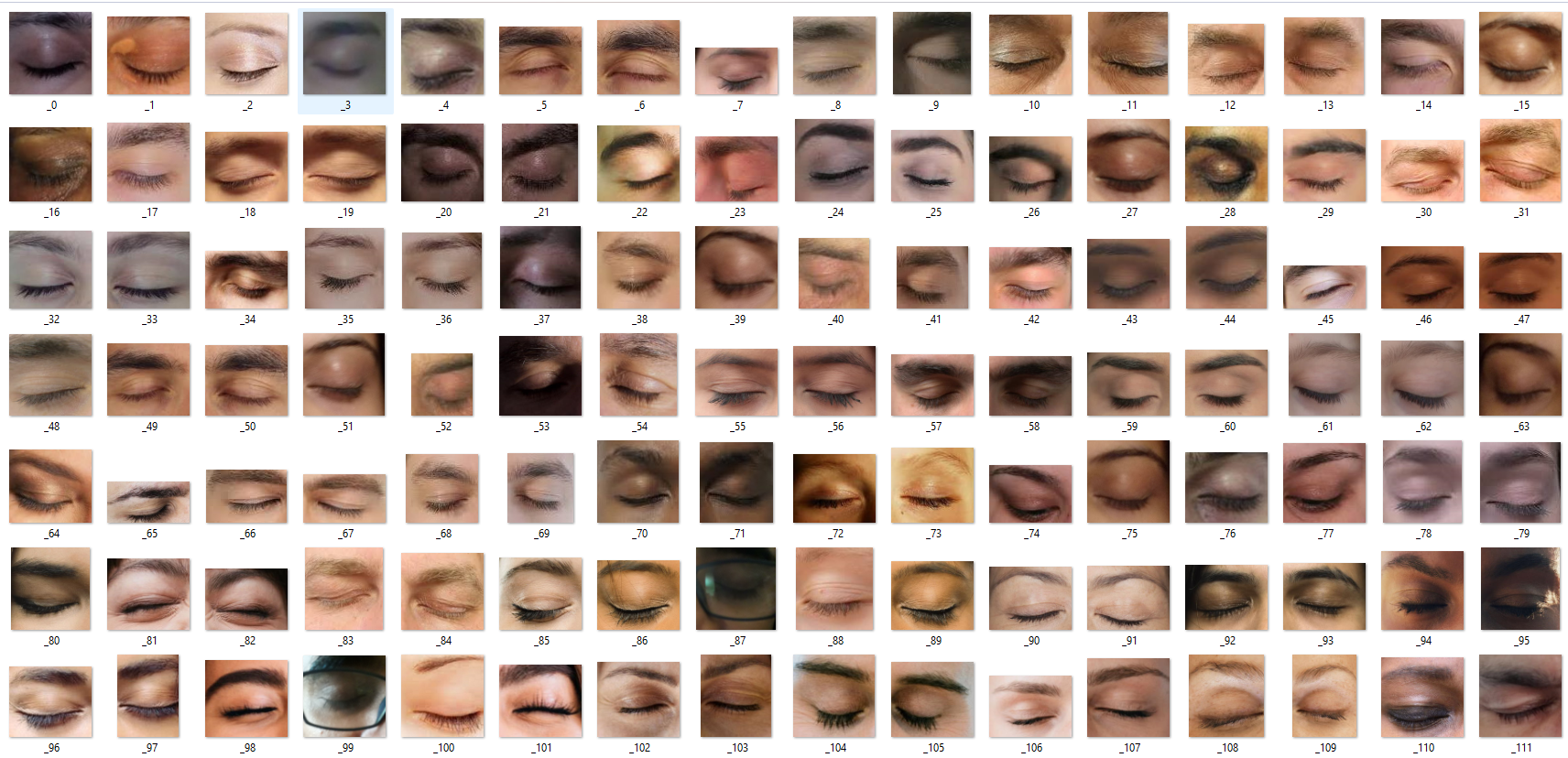
|  |  |
| --- | --- |
| Tên file | Ý nghĩa file mã nguồn |
| Haar cascade files | Tệp chứa các bộ huấn luyện HaarCascade |
| models | chứa mô hình máy học đã được huấn luyện |
| train | Tệp chứa ảnh mô tả trạng thái mắt |
| Alarm.wav | File âm thanh này có thể được phát ra khi hệ thống phát hiện thấy dấu hiệu buồn ngủ. |
| Drowsiness detection.py | file hệ thống cảnh báo buồn ngủ |
| Model.py | file chứa mô hình máy học |
| Ph-t-hi-n-l-i-xe | File kết quả quá trình chạy mô hình |

CHƯƠNG 3. THỰC NGHIỆM CHƯƠNG TRÌNH

1. Thu thập dữ liệu

1.1. Bộ dữ liệu đầu vào(dataset)

Để thực hiện huấn luyện mô hình sử dụng CNN, trước hết cần chuẩn bị dữ liệu dầu vào là ảnh và nhãn dán tương ứng với ảnh đó. Mô hình bao gồm 2 mô hình nhỏ dược huấn luyện một cho phát hiện buồn ngủ với dataset ảnh và nhãn dán tương ứng, một cho việc nhận diện các ký tự trên mắt người với 4776 ảnh đã được augmentation.





Hình 6:Ảnh dataset

1.2. Tiền xử lý dữ liệu

**- Chia tập dữ liệu**

def get\_split\_data(ds, train\_split=0.8, test\_splt=0.1, val\_split=0.1, shuffle=True, shuffle\_size=10000):

assert (train\_split + test\_splt + val\_split) == 1

* Chia tệp dữ liệu thành 8-1-1(80% train, 10% test và 10% validation)

**- Chuẩn hóa và resize ảnh**

resize\_and\_rescale = tf.keras.Sequential([

layers.Rescaling(1./255),

layers.Resizing(IMAGE\_SIZE, IMAGE\_SIZE)

* Rescaling(1./255), : Đây là bước chuẩn hóa giá trị pixel của ảnh từ khoảng 0-255 về khoảng 0-1.
* Resizing(IMAGE\_SIZE, IMAGE\_SIZE) : Resize ảnh về kích thước cố định

**- Cấu hình mạng CNN**

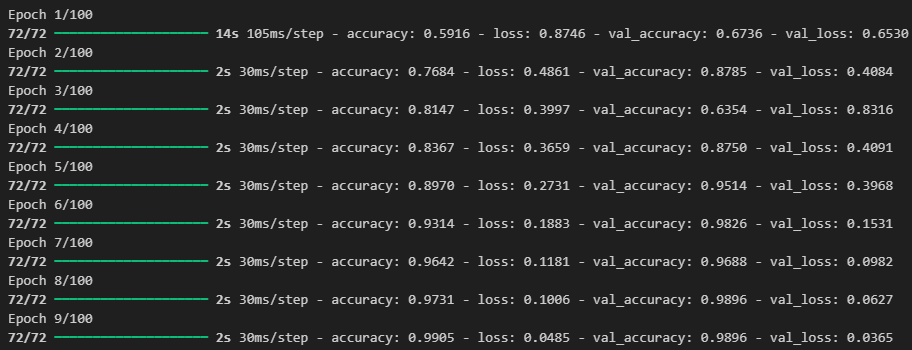
input\_shape = (BATCH\_SIZE, IMAGE\_SIZE, IMAGE\_SIZE, CHANNELS)

n\_classes = 4

* Input\_shape : Cấu hình ảnh
* n\_classes : Số lượng lớp

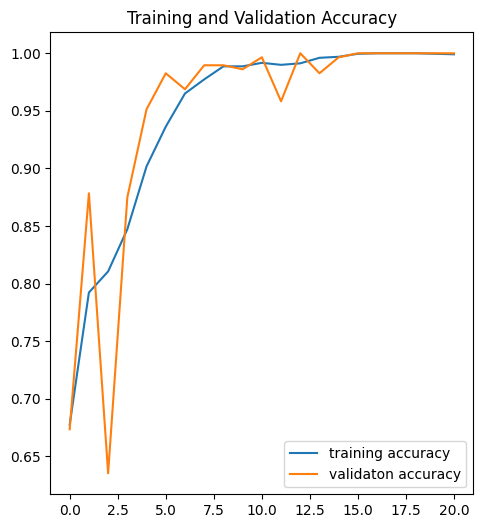
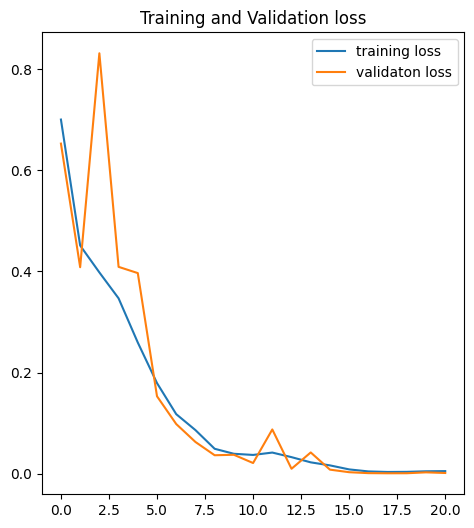
1.3. Huấn luyện mô hình

Gọi hàm fit để huấn luyện mô hình với dữ liệu huấn luyện và kiểm tra, trải qua 100 epoch. Và lưu mô hình đã huấn luyện vào file CnnCat2.h5 để sử dụng sau này.



Hình 7:Quá trình training

1.4 Kết quả quá trình huấn luyện chương trình

Hình 8:Biểu đồ huấn luyện

2. Môi trường thực nghiệm

Quá trình thực nghiệm được chia làm hai giai đoạn. Giai đoạn thứ nhất xây dựng mô hình nhận dạng với dữ liệu video đầu vào trong các điều kiện khác nhau. Dữ liệu huấn luyện được chọn và sử dụng trong quá trình xây dựng mô hình. Ở giai đoạn này, một số giá trị tham số đầu vào và điều kiện ánh sáng xung quanh được thay đổi bằng thực nghiệm để tìm ra giá trị phù hợp cho kết quả (độ chính xác) cao nhất ứng với mỗi phương pháp. Kết quả thu được ở giai đoạn thực nghiệm này là độ chính xác của chương trình tương ứng với các dữ liệu video đầu vào khác nhau.

2.1. Môi trường triển khai

Quá trình thực nghiệm được thực hiện trên một chiếc laptop sử dụng hệ điều hành Window 11 có cấu hình tương đương một máy tính phổ thông hiện nay và sử dụng camera trực tiếp trên máy để thu video đầu vào.

Ngôn ngữ sử dụng trong chương trình thực nghiệm là ngôn ngữ lập trình Python. Bên cạnh đó, chương trình thực nghiệm có sử dụng thư viện mã nguồn mở OpenCV.

2.2. Bộ huấn luyện

Bao gồm 3 bộ huấn luyện HaarCascade được hổ trợ trong bộ cài thư viện mở OpenCV. Đó là các file XML được lưu trữ trong thư mục “haarcascades”:

* haarcascade\_frontalface\_alt.xml
* haarcascade\_lefteye\_2splits.xml
* haarcascade\_righteye\_2splits.xml

KẾT LUẬN

1. Kết quả đạt được

Hệ thống đã thành công trong việc phát hiện dấu hiệu buồn ngủ thông qua việc phân tích tỷ lệ của mắt. Các thử nghiệm cho thấy hệ thống có thể chính xác trong việc xác định trạng thái buồn ngủ dựa trên dấu hiệu của mắt và thời gian kiểm tra. Hệ thống phát cảnh báo âm thanh đúng lúc giúp người ngủ dễ dàng tỉnh dậy và giảm nguy cơ tai nạn.

Hệ thống hoạt động ổn định trong điều kiện ánh sáng khác nhau và với các góc nhìn khác nhau của khuôn mặt.

2. Kết quả chưa đạt được

* **Khả năng xử lý trong điều kiện ánh sáng cực đoan**: Mặc dù hệ thống hoạt động tốt trong điều kiện ánh sáng bình thường, nhưng nó vẫn gặp khó khăn trong các điều kiện ánh sáng rất yếu hoặc quá sáng. Đặc biệt, việc phát hiện khuôn mặt và mắt có thể bị ảnh hưởng khi ánh sáng quá yếu hoặc quá sáng. - **Hiệu suất với các vật cản và góc nhìn lệch**: Hệ thống có thể gặp khó khăn trong việc phát hiện mắt khi có vật cản hoặc khi tài xế nghiêng đầu quá nhiều.
* **Đáp ứng với các tình huống đặc biệt**: Trong một số tình huống như tài xế đeo kính râm hoặc có đặc điểm khuôn mặt không điển hình, khả năng phát hiện có thể bị ảnh hưởng.

3. Dự kiến phát triển và cải tiến thuật toán

* **Áp dụng thuật toán vào các lĩnh vực khác nhau như :** Nhận diện học sinh ngủ gật,…
* **Cải thiện khả năng phát hiện trong điều kiện ánh sáng cực đoan**: Nghiên cứu và áp dụng các kỹ thuật xử lý ảnh tiên tiến hơn như cân bằng ánh sáng tự động và cải thiện thuật toán nhận diện khuôn mặt và mắt để tăng cường khả năng hoạt động trong các điều kiện ánh sáng khác nhau.
* **Tăng cường độ chính xác với vật cản và góc nhìn lệch**: Phát triển và áp dụng các mô hình học máy mạnh mẽ hơn như mạng nơ-ron sâu (Deep Learning) để nhận diện khuôn mặt và mắt chính xác hơn trong các điều kiện không lý tưởng.

Cập nhật các thuật toán để xử lý các góc nhìn lệch và vật cản hiệu quả hơn.

* **Tích hợp cảm biến phụ trợ**: Kết hợp các cảm biến khác như cảm biến chuyển động, cảm biến độ nghiêng, hoặc cảm biến nhịp tim để tăng cường khả năng phát hiện tình trạng mệt mỏi của tài xế và giảm thiểu các lỗi phát hiện sai.
* **Nâng cao khả năng tùy chỉnh**: Cung cấp các tùy chọn điều chỉnh nâng cao cho người dùng, bao gồm khả năng cá nhân hóa ngưỡng EAR và thời gian cảnh báo dựa trên các yếu tố cá nhân và điều kiện lái xe cụ thể.
* **Tăng cường hiệu suất và tính tương thích**: Tối ưu hóa thuật toán để giảm thiểu mức tiêu tốn tài nguyên và nâng cao hiệu suất xử lý thời gian thực. Đảm bảo hệ thống tương thích với các loại camera và phần cứng khác nhau.
* **Thực hiện thử nghiệm và đánh giá mở rộng**: Tiến hành thêm các thử nghiệm trong điều kiện và môi trường khác nhau để xác định điểm yếu và cải thiện tính chính xác và đáng tin cậy của hệ thống

TÀI LIỆU THAM KHẢO

[1] **Mitchell, T. M. (1997).** Machine Learning. McGraw-Hill.

[2] **Goodfellow, I., Bengio, Y., & Courville, A. (2016).** Deep Learning. MIT Press.

[3] **LeCun, Y., Bengio, Y., & Hinton, G. (2015).** "Deep Learning". Nature, 521(7553), 436-444.

[4] **Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012).** "ImageNet Classification with Deep Convolutional Neural Networks". Advances in Neural Information Processing Systems (NIPS), 25, 1097-1105.

[5] **Zhang, C., & Ma, Y. (Eds.) (2021).** Deep Learning for Computer Vision. Springer.

[6] **Brownlee, J. (2016).** Machine Learning Mastery with Python. Machine Learning Mastery.

[7] **Russell, S., & Norvig, P. (2021).** Artificial Intelligence: A Modern Approach. Pearson.