# TRƯỜNG ĐẠI HỌC NHA TRANG

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO TỔNG KẾT**

**ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN**

**ỨNG DỤNG KỸ THUẬT THỊ GIÁC MÁY TÍNH XÂY DỰNG HỆ THỐNG BUỒN NGỦ KHI LÁI XE**

**Mã số: SV2024-13-13**

|  |  |  |
| --- | --- | --- |
| **Chủ nhiệm đề tài** | **:** | **Huỳnh Thị Hạnh Nguyên – 63.HTTT** |
| **Cán bộ hướng dẫn** | **:** | **Th.S Nguyễn Thị Hương Lý** |
| **Chức danh** | **:** | **Giảng viên khoa công nghệ thông tin** |

**Khánh Hòa, tháng 4/2025**

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO TỔNG KẾT**

**ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN**

**ỨNG DỤNG KỸ THUẬT THỊ GIÁC MÁY TÍNH XÂY DỰNG HỆ THỐNG BUỒN NGỦ KHI LÁI XE**

**Mã số: SV2024-13-13**

**Đơn vị chủ trì đề tài Chủ nhiệm đề tài**

*(ký, họ tên) (ký, họ tên)*

**XÁC NHẬN CỦA CƠ QUAN QUẢN LÝ ĐỀ TÀI**

**TL. HIỆU TRƯỞNG**

**TRƯỞNG PHÒNG KHCN**

***(ký, họ tên, đóng dấu)***

**Khánh Hòa, tháng 4/2025**

# MỤC LỤC

[DANH MỤC BẢNG BIỂU ii](#_Toc196314942)

[DANH MỤC HÌNH VẼ iii](#_Toc196314943)

[DANH MỤC CÁC CHỮ VIẾT TẮT SỬ DỤNG TRONG ĐỀ TÀI iv](#_Toc196314944)

[THÔNG TIN KẾT QUẢ NGHIÊN CỨU v](#_Toc196314945)

[MỞ ĐẦU 1](#_Toc196314946)

[CHƯƠNG 1. TỔNG QUAN 7](#_Toc196314947)

[1.1 Thực trạng nghiên cứu trong nước 7](#_Toc196314948)

[1.2 Thực trạng nghiên cứu quốc tế 8](#_Toc196314949)

[1.3 Vấn đề tồn tại và định hướng nghiên cứu 9](#_Toc196314950)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 11](#_Toc196314951)

[2.1 Tổng quan về Artificial Neural Network (ANN) 11](#_Toc196314952)

[2.1.1 Giới thiệu về ANN 11](#_Toc196314953)

[2.1.2 Cấu trúc của ANN 11](#_Toc196314954)

[2.1.3 Quá trình huấn luyện mạng nơ-ron 12](#_Toc196314955)

[2.2 VGG16 (Convolutional Neural Network - CNN) 17](#_Toc196314956)

[2.2.1 Giới thiệu về Mạng nơ-ron tích chập (CNN) 17](#_Toc196314957)

[2.2.2 Giới thiệu về VGG16 18](#_Toc196314958)

[2.2.3 Kiến trúc của VGG16 18](#_Toc196314959)

[2.4.3 Ứng dụng VGG16 trong phân loại trạng thái mắt và miệng 19](#_Toc196314960)

[2.3 YOLO (You Only Look Once) 20](#_Toc196314961)

[2.3.1 Phân biệt giữa phân loại ảnh và phát hiện đối tượng 20](#_Toc196314962)

[2.3.2 Giới thiệu về YOLO 21](#_Toc196314963)

[2.3.3 Nguyên lý hoạt động của YOLO 21](#_Toc196314964)

[2.3.4 Kiến Trúc Của YOLO 22](#_Toc196314965)

[2.3.5 Ưu điểm và hạn chế của YOLO 23](#_Toc196314966)

[2.3.6 Ứng dụng YOLO trong đề tài 24](#_Toc196314967)

[2.4 YOLOv10 24](#_Toc196314968)

[2.4.1 Giới thiệu về YOLOv10 24](#_Toc196314969)

[2.4.2 Hiệu quả của YOLOv10 so với các phiên bản trước 25](#_Toc196314970)

[2.4.3 YOLOv10 trong phát hiện mắt và miệng 26](#_Toc196314971)

[2.5 Học chuyển giao (Transfer learning) và tinh chỉnh (Fine-tuning) 26](#_Toc196314972)

[CHƯƠNG 3. ĐỐI TƯỢNG VÀ PHƯƠNG PHÁP NGHIÊN CỨU 28](#_Toc196314973)

[3.1 Đối tượng nghiên cứu 28](#_Toc196314974)

[3.1.1 Vùng mắt 28](#_Toc196314975)

[3.1.2 Vùng miệng 28](#_Toc196314976)

[3.1.3 Cơ sở lựa chọn vùng mắt và miệng cho nghiên cứu 29](#_Toc196314977)

[3.2 Phương pháp nghiên cứu 29](#_Toc196314978)

[CHƯƠNG 4. QUÁ TRÌNH THỰC HIỆN NGHIÊN CỨU 31](#_Toc196314979)

[4.1 Xây dựng mô hình phát triển buồn ngủ 31](#_Toc196314980)

[4.1.1 Chiến lược xây dựng hệ thống phát hiện buồn ngủ 31](#_Toc196314981)

[4.1.2 Thu thập và tiền xử lý dữ liệu 32](#_Toc196314982)

[4.1.3. Mô hình phát hiện vùng đặc trưng bằng YOLOv10 35](#_Toc196314983)

[4.1.4. Mô hình phân loại trạng thái bằng VGG16 40](#_Toc196314984)

[4.1.5 Kết quả huấn luyện và đánh giá mô hình 45](#_Toc196314985)

[4.2 Phát triển và triển khai hệ thống giám sát buồn ngủ 50](#_Toc196314986)

[4.2.1 Tổng quan hệ thống 50](#_Toc196314987)

[4.2.2 Kiến trúc hệ thống 50](#_Toc196314988)

[4.2.3 Tích mô hình vào hệ thống thời gian thực 54](#_Toc196314989)

[4.2.4 Giao diện người dùng và tính năng 57](#_Toc196314990)

[4.2.5 Kiểm thử và đánh giá hệ thống 64](#_Toc196314991)

[CHƯƠNG 5. KẾT QUẢ NGHIÊN CỨU 66](#_Toc196314992)

[5.1 Kiểm thử hệ thống 66](#_Toc196314993)

[5.2 Tính toán hiệu suất hệ thống 67](#_Toc196314994)

[5.2.1 Tính toán hiệu suất phát hiện nhắm mắt 67](#_Toc196314995)

[5.2.2 Tính toán hiệu suất phát hiện ngáp 68](#_Toc196314996)

[5.3 Đánh giá hiệu suât hệ thống 69](#_Toc196314997)

[5.4 Kết luận 70](#_Toc196314998)

[KẾT LUẬN VÀ KIẾN NGHỊ 71](#_Toc196314999)

[TÀI LIỆU THAM KHẢO 76](#_Toc196315000)

# DANH SÁCH NHỮNG THÀNH VIÊN THAM GIA NGHIÊN CỨU ĐỀ TÀI VÀ ĐƠN VỊ PHỐI HỢP CHÍNH

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Thành viên tham gia thực hiện đề tài** | | | | |
| **STT** | **Họ và tên** | **MSSV** | **Lớp** | **Vai trò** |
| 1 | Huỳnh Thị Hạnh Nguyên | 63130894 | 63.HTTT | Chủ nhiệm đề tài |
| 2 | Hàng Tuấn Kiệt | 63130579 | 63.HTTT | Thành viên thực hiện |
| 3 | Trần Thị Xuân Ly | 63134569 | 63.CNTT-1 | Thành viên thực hiện |

|  |  |  |
| --- | --- | --- |
| **Đơn vị phối hợp** | | |
| **Tên đơn vị** | **Nội dung phối hợp nghiên cứu** | **Họ và tên trưởng đơn vị** |
|  |  |  |

# DANH MỤC BẢNG BIỂU

[Bảng 4. 1 Số lượng ảnh Face Pose Dataset trong tập train, validation và tập test 34](#_Toc196233800)

[Bảng 4. 2 Số lượng ảnh Yawn\_eye\_dataset\_new trong tập train và tập test 36](#_Toc196233801)

[Bảng 4.3. Các chỉ số đánh giá hiệu suất mô hình YOLOv10n trên từng lớp đối tượng 48](#_Toc196233802)

[Bảng 4. 4. Các chỉ số đánh giá hiệu suất mô hình VGG16 trên từng lớp trạng thái 50](#_Toc196233803)

[Bảng 4. 5. Kết quả kiểm thử 65](#_Toc196233804)

[Bảng 5. 1. Kết quả dữ liệu thử nghiệm hệ thống phát hiện nhắm mắt và ngáp (0-29 giây) 66](#_Toc196314490)

[Bảng 5. 2. Kết quả tính toán chi tiết( closed eyes) 67](#_Toc196314491)

[Bảng 5. 3. Kết quả tính toán chi tiết (yawn) 68](#_Toc196314492)

# DANH MỤC HÌNH VẼ

[Hình 2.1. Mạng nơ ron gồm 3 layers 49](#_Toc196232444)

[Hình 2.2. Tính toán đầu ra bằng lan truyền xuôi 50](#_Toc196232445)

[Hình 2.3. Hàm Sigmoid 51](#_Toc196232446)

[Hình 2.4. Hàm Tanh 51](#_Toc196232447)

[Hình 2.5. Hàm ReLU 52](#_Toc196232448)

[Hình 2.6. Kiến trúc của VGG16 56](#_Toc196232449)

[Hình 2.7. Image Classification & Object Localization 58](#_Toc196232450)

[Hình 2.8. Minh họa cách YOLO xác định vị trí và phân loại đối tượng trong ảnh 59](#_Toc196232451)

[Hình 2.9. Kiến trúc của YOLO 59](#_Toc196232452)

[Hình 2. 10. So sánh hiệu suất của YOLOv10 và các phiên bản trước về độ chính xác (COCO AP) theo đỗ trễ (Latency) và số lượng tham số (Parameters). 62](#_Toc196232453)

[Hình 4.1. Các mẫu dữ liệu của Face Pose Dataset 70](#_Toc196232454)

[Hình 4.2. Các mẫu dữ liệu của Yawn\_eye\_dataset\_new 72](#_Toc196232455)

[Hình 4.3. Fine tuning với YOLOv10 74](#_Toc196232456)

[Hình 4.4. Mô hình 1 - Mô hình phát hiện vùng đặc trưng khuôn mặt YOLOv10 76](#_Toc196232457)

[Hình 4.5. Quá trình học chuyển giao 79](#_Toc196232458)

[Hình 4.6. Mô hình 2 – Mô hình phân loại trạng thái VGG16 81](#_Toc196232459)

[Hình 4.7. Biểu đồ F1-Confidence Curve và Precision-Confidenc Curve 84](#_Toc196232460)

[Hình 4. 8. Ma trận nhầm lẫn 85](#_Toc196232461)

[Hình 4.9. Biểu đồ Loss over Epochs và Accuracy over Epoch 86](#_Toc196232462)

[Hình 4. 10. Ma trận nhầm lẫn của mô hình 2 87](#_Toc196232463)

[Hình 4.11. Hệ thống tổng quan 88](#_Toc196232464)

[Hình 4.12. Sơ đồ khối kiến trúc hệ thống 89](#_Toc196232465)

[Hình 4. 13. Giao diện chính hệ thống cảnh báo 95](#_Toc196232466)

[Hình 4.14. Chỉ báo trạng thái thời gian thực 97](#_Toc196232467)

[Hình 4. 15. Các nút điều khiển chính của hệ thống 97](#_Toc196232468)

[Hình 4.16. Cửa sổ chọn nguồn dữ liệu 98](#_Toc196232469)

[Hình 4.17. Cửa sổ cài đặt 99](#_Toc196232470)

[Hình 4.18. Cửa sổ lịch sử cảnh báo 99](#_Toc196232471)

[Hình 4.19. Hệ thống cảnh báo khi nhắm mắt quá lâu 100](#_Toc196232472)

[Hình 4.20. Hệ thống cảnh báo ngáp 101](#_Toc196232473)

# DANH MỤC CÁC CHỮ VIẾT TẮT SỬ DỤNG TRONG ĐỀ TÀI

|  |  |
| --- | --- |
| **Tên viết tắt** | **Giải thích nghĩa** |
| Conv | Convolutional |
| FC | Fully Connected |
| ANN | Artificial Neural Network |
| CNN | Convolutional Neural Network |
| FC | Fully Connected |
| YOLO | You Only Look Once |
| ROI | Region of Interest |
| BN | Based Network |
| AI | trí tuệ nhân tạo |

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**Khoa Công nghệ thông tin**

# THÔNG TIN KẾT QUẢ NGHIÊN CỨU

**1. Thông tin chung:**

**-** Tên đề tài: Ứng dụng thị giác máy tính xây dựng hệ thống buồn ngủ khi lái xe

- Mã số: SV2024-13-13

- Chủ nhiệm đề tài: Huỳnh Thị Hạnh Nguyên

- Cán bộ hướng dẫn khoa học: Nguyễn Thị Hương Lý

- Đơn vị chủ trì: Khoa Công nghệ thông tin

- Thời gian thực hiện:10 tháng từ tháng 7 năm 2024 đến tháng 4 năm 2025

**2. Mục tiêu:**

**3. Tính mới và sáng tạo:**

**4. Kết quả nghiên cứu:**

**5. Sản phẩm:**

**6. Hiệu quả, phương thức chuyển giao kết quả nghiên cứu và khả năng áp dụng:**

Ngày tháng năm

|  |  |
| --- | --- |
| **Cán bộ hướng dẫn khoa học**  *(ký, họ và tên)* | **Chủ nhiệm đề tài**  *(ký, họ và tên)* |

# MỞ ĐẦU

1. Lý do chọn đề tài

Trong bối cảnh giao thông đường bộ ngày càng phát triển, việc đảm bảo an toàn khi lái xe đã trở thành một vấn đề cấp thiết cần được quan tâm. Khi tài xế rơi vào trạng thái mệt mỏi, khả năng kiểm soát phương tiện và phản xạ trước các tình huống bất ngờ bị suy giảm đáng kể. Điều này làm tăng nguy cơ xảy ra tai nạn, gây thiệt hại không chỉ về tài sản mà còn ảnh hưởng trực tiếp đến tính mạng con người. Chính vì vậy, việc tìm ra các biện pháp hỗ trợ tài xế duy trì sự tỉnh táo và cảnh báo kịp thời khi có dấu hiệu buồn ngủ là điều rất cần thiết.

Với sự tiến bộ vượt bậc của công nghệ hiện đại, đặc biệt trong lĩnh vực trí tuệ nhân tạo (AI), việc ứng dụng AI để giải quyết các vấn đề thực tế đang ngày càng trở nên phổ biến. Các thông tin đã chỉ ra rằng rằng AI có thể phân tích chuyển động mắt, tư thế đầu và biểu hiện khuôn mặt của tài xế để đánh giá mức độ tỉnh táo. Tuy nhiên, hầu hết các hệ thống hiện nay vẫn còn tồn tại nhiều hạn chế về độ chính xác, khả năng hoạt động trong thực tế và khả năng ứng dụng rộng rãi.

Là một sinh viên yêu thích nghiên cứu về trí tuệ nhân tạo và ứng dụng công nghệ vào thực tế, em mong muốn đóng góp trong việc xây dựng hệ thống nhận biết buồn ngủ hiệu quả hơn. Việc nghiên cứu đề tài này không chỉ giúp em thực hành trong thực tế những kiến thức đã học mà còn tạo cơ hội để phát triển một giải pháp có ý nghĩa, góp phần nâng cao an toàn giao thông. Đồng thời, em cũng hy vọng rằn đóng góp này có thể mở ra nhiều hướng phát triển mới, giúp cải thiện các phương pháp nhận biết tình trạng buồn ngủ của người tham gia giao thông trong tương lai.

2. Tính cấp thiết

Buồn ngủ khi lái xe là một trong những nguyên nhân hàng đầu dẫn đến tai nạn giao thông trên toàn thế giới. Theo thống kê, mỗi năm có hàng triệu vụ tai nạn liên quan đến tình trạng tài xế mất tập trung do mệt mỏi hoặc thiếu ngủ. Đặc biệt, nghiên cứu cho thấy rằng một tỷ lệ lớn tài xế từng lái xe khi không còn đủ tỉnh táo, thậm chí gặp khó khăn trong việc mở mắt. Điều này cho thấy mức độ phổ biến và nghiêm trọng của vấn đề dẫn đến phải có những giải pháp công nghệ giúp nhận diện và cảnh báo sớm tình trạng buồn ngủ. Vì vậy, sự xuất hiện của công nghệ phát hiện dấu hiệu buồn ngủ của người lái xe là một điều cần thiết vì nó có thể cứu sống hàng nghìn người mỗi năm bằng cách giảm thiểu tình trạng tai nạn giao thông từ tình trạng mệt mỏi hoặc buồn ngủ của người lái xe.

Một hệ thống có thể giám sát trạng thái tỉnh táo của người lái xe và cảnh báo kịp thời sẽ không chỉ giúp tài xế nâng cao nhận thức về tình trạng của bản thân mà còn góp phần giảm thiểu đáng kể số vụ tai nạn giao thông. Đồng thời, việc áp dụng AI vào vấn đề an toàn khi tham gia giao thông cũng mở ra nhiều cơ hội nghiên cứu và phát triển trong tương lai. Vì vậy, việc khám phá về hệ thống theo dõi và cảnh báo khi người lái xe có dấu hiệu buồn ngủ không chỉ mang ý nghĩa khoa học mà còn có tính ứng dụng thực tiễn cao, cần được quan tâm và phát triển hơn nữa.

3. Mục tiêu

Đề tài hướng tới việc phát triển một hệ thống có thể phát hiện tình trạng buồn ngủ trong lúc lái xe của người điều khiển phương tiện bằng cách phân tích hình ảnh khuôn mặt, tập trung chủ yếu vào đặc điểm của mắt và miệng. Ứng dụng này được kỳ vọng sẽ phát hiện và cảnh báo kịp thời, góp phần hạn chế rủi ro tai nạn trong giao thông.

Ứng dụng này được xây dựng để hoạt động trong điều kiện bình thường, tức là khi tài xế ngồi ngay ngắn trước camera, ánh sáng đầy đủ và không có nhiều yếu tố gây nhiễu. Đề tài chủ yếu thử nghiệm tính chính xác của mô hình để giám sát trạng thái tỉnh táo của tài xế trong môi trường tiêu chuẩn, đảm bảo hệ thống có thể nhận biết các dấu hiệu như trạng thái mắt mở/nhắm, ngáp/không ngáp; thời gian nhắm mắt/ngáp. Thông qua việc phân tích dữ liệu hình ảnh theo thời gian thực, đề tài khá chính xác khi nhận biêt các dấu hiệu buồn ngủ và kích hoạt cảnh báo bằng âm thanh hoặc tín hiệu trực quan, giúp tài xế kịp thời nhận thức được tình trạng của bản thân và có biện pháp xử lý phù hợp.

4. Cách tiếp cận

Để xây dựng một hệ thống phát hiện trạng thái buồn ngủ của tài xế một cách chính xác và hiệu quả, đề tài áp dụng phương pháp tiếp cận kết hợp nhiều kỹ thuật trong lĩnh vực thị giác máy tính và trí tuệ nhân tạo. Hướng tiếp cận này bao gồm việc nghiên cứu, đánh giá các phương pháp hiện có, kế thừa những thuật toán hiệu quả, đồng thời phát triển giải pháp tối ưu phù hợp với yêu cầu của hệ thống cảnh báo buồn ngủ.

- *Tiếp cận tổng hợp*: Trước tiên, đề tài tập trung vào việc thu thập và phân tích các phương pháp hiện có trong việc phát hiện buồn ngủ từ biểu hiện trên mặt (mắt/miệng). Các phương pháp này sử dụng kỹ thuật thị giác máy tính để quan sát trạng thái mắt mở/nhắm, ngáp/không ngáp; thời gian nhắm mắt/ngáp.

- Tiếp cận kế thừa và phát triển:

* Bên cạnh việc thu thập các cách nhận diện trạng thái buồn ngủ, đề tài cũng kế thừa và phát triển các thuật toán hiện có để tăng hiệu suất của mô hình. Cụ thể, đề tài hướng đến việc xây dựng một giải thuật nhận diện biểu hiện buồn ngủ từ mô hình học sâu với thuật toán theo dõi liên tục để phát hiện sự thay đổi trạng thái mắt và tư thế đầu.
* Giải pháp sau khi được xây dựng sẽ được tích hợp vào camera giám sát, giúp theo dõi tài xế trong quá trình lái xe và phát cảnh báo khi phát hiện dấu hiệu buồn ngủ. Việc tích hợp này yêu cầu sự hoạt động ổn định trên các thiết bị phổ biến, có khả năng xử lý dữ liệu hình ảnh theo thời gian thực mà không yêu cầu phần cứng quá mạnh.

5. Phạm vi nghiên cứu

Nội dung nghiên cứu tập trung vào việc xây dựng một mô hình có khả năng nhận diện tình trạng buồn ngủ ở tài xế thông qua việc phân tích hình ảnh khuôn mặt (mắt/miệng) bằng công nghệ học sâu và thị giác máy tính. Mô hình này sẽ theo dõi như trạng thái mắt mở/nhắm, ngáp/không ngáp; thời gian nhắm mắt/ngáp để phát hiện sớm dấu hiệu mệt mỏi và đưa ra cảnh báo kịp thời.

- Phạm vi dữ liệu:

* Dữ liệu đầu vào là các khung hình (frame) từ camera thời gian thực hoặc video ghi lại khuôn mặt tài xế trong quá trình lái xe.
* Tập dữ liệu huấn luyện và kiểm thử được thu thập từ các bộ dữ liệu công khai và dữ liệu tự thu thập trong điều kiện mô phỏng.
* Hệ thống chưa được thử nghiệm trên quy mô lớn hoặc trong thực tế với nhiều điều kiện khác nhau như thời tiết xấu, ánh sáng yếu hoặc tài xế đeo kính.

- Phạm vi kỹ thuật:

* Giải pháp sử dụng mạng học sâu nhằm nhận dạng tình trạng buồn ngủ của người lái xe, kết hợp với các thuật toán xử lý ảnh nhằm tăng độ chính xác.
* Hệ thống được xây dựng và thử nghiệm trên máy tính cá nhân, với yêu cầu phần cứng tương đối phổ biến, chưa triển khai trên các thiết bị nhúng hoặc hệ thống xe thực tế.
* Việc kiểm tra hiệu quả của mô hình được tiến hành trong môi trường giả lập trong phòng thí nghiệm, chưa xét đến các yếu tố gây nhiễu mạnh như người lái xe mang kính, trời tối, rung lắc, mờ ảnh hay thay đổi góc quay đột ngột.

- Phạm vi ứng dụng:

* Hướng tiếp cận này có thể được đưa vào áp dụng trong các mô hình nghiên cứu thử nghiệm.
* Chưa triển khai trên các phương tiện thực tế.

6. Nội dung nghiên cứu

*- Nội dung 1*: Phân tích, tổng hợp các các phương pháp trong lĩnh vực kỹ thuật thị giác máy tính nhằm giám sát tình trạng buồn ngủ.

* Công việc 1.1: Nghiên cứu các đặc điểm nhận biết khi rơi vào trạng thái buồn ngủ khi lái xe.
* Công việc 1.2: Khảo sát các phương pháp phát hiện buồn ngủ.
* Công việc 1.3: Nghiên cứu các các mô hình trong học máy/ học sâu hỗ trợ phát hiện buồn ngủ.

*- Nội dung 2*: Phát triển và đánh giá mô hình.

* Công việc 2.1: Thu thập và xử lý sơ bộ dữ liệu đầu vào.
* Công việc 2.2: Nghiên cứu và lựa chọn thuật toán phù hợp để phát hiện vùng mắt/miệng.
* Công việc 2.3: Nghiên cứu các mô hình phát hiện buồn ngủ.
* Công việc 2.4: Thực hiện điều chỉnh và tối ưu các tham số của mô hình nhằm cải thiện hiệu quả dự đoán.
* Công việc 2.5: Đánh giá độ chính xác của mô hình sau khi tối ưu, đồng thời phân tích các chỉ số đầu ra.

*- Nội dung 3*: Phát triển hệ thống.

* Công việc 3.1: Xây dựng bản mẫu để gắn camera lên rồi giám sát trên thời gian thực rồi gửi cảnh báo.
* Công việc 3.2: Đánh giá hệ thống.

7. Ý nghĩa khoa học và thực tiễn của đề tài

- Ý nghĩa khoa học:

Nghiên cứu này khai thác khả năng của trí tuệ nhân tạo và thị giác máy trong việc phát hiện tài xế có dấu hiệu buồn ngủ, thông qua ứng dụng các mô hình học sâu hiện đại. Để tăng độ chính xác và ứng dụng thực tế, giải pháp được tiến hành bằng cách sử dụng các đặc điểm khuôn mặt và kết hợp với mạng nơ-ron.

Ngoài ra, đề tài giúp mở rộng lĩnh vực nghiên cứu liên quan đến việc sử dụng thị giác máy tính để theo dõi hành vi con người, đặc biệt là trong lĩnh vực an toàn giao thông. Việc kiểm thử và tối ưu mô hình giám sát tình trạng buồn ngủ của tài xế lái xe không chỉ góp phần nâng cao hiệu suất nhận diện mà còn tạo tiền đề cho các nghiên cứu tiếp theo về giám sát trạng thái tài xế trong môi trường thực tế.

- Ý nghĩa thực tiễn:

Sự mất tỉnh táo do buồn ngủ khi lái xe được xem là nguyên nhân phổ biến gây ra nhiều vụ tai nạn trên đường, đe dọa tính mạng con người và gây thiệt hại lớn về kinh tế. Việc thực thi một giải pháp để giám sát tình trạng rơi vào trạng thái buồn ngủ sớm và phát cảnh báo nhanh sẽ góp phần hạn chế rủi ro gây tai nạn, đảm bảo an toàn cho tài xế lẫn các phương tiện xung quanh.

Ngoài ra, hệ thống có thể được tích hợp vào các hệ thống theo dõi an toàn giao thông hoặc phương tiện vận hành thông minh, giúp hỗ trợ các giải pháp nâng cao chất lượng quản lý phương tiện. Về lâu dài, giải pháp này có tiềm năng để mở rộng và ứng dụng trong nhiều lĩnh vực khác, chẳng hạn như trong lĩnh vực an ninh, giám sát các hành vi bất thường như trộm cắp hoặc xâm nhập trái phép.

Do đó, nghiên cứu này không những có đóng góp về mặt khoa học qua việc xây dựng và hoàn thiện hệ thống phát hiện tài sớm xế rơi vào trạng thái buồn khi điều khiển phương tiện, thậm chí mở ra nhiều triển vọng trong ứng dụng thực tế, mang lại giá trị thực tiễn lớn, hỗ trợ nâng cao độ an toàn trong giao thông.

# CHƯƠNG 1. TỔNG QUAN

Trong chương này, chúng tôi trình bày tổng quan về lĩnh vực nghiên cứu hệ thống phát hiện trạng thái buồn ngủ khi lái xe, bao gồm các kết quả đạt được, xu hướng nghiên cứu hiện tại cũng như những hạn chế còn tồn tại.

## 1.1 Thực trạng nghiên cứu trong nước

Tình hình thực tế: Thực trạng hiện nay tại Việt Nam cho thấy tại nạn giao thông liên quan đến sự mệt mỏi, thiếu ngủ của tài xế là một vấn đề rất quan ngại. Theo một báo cáo, khoảng 40% số vụ tai nạn giao thông xảy ra từ 6 giờ chiều đến 12 giờ đêm, thời điểm mà các tài xế cảm thấy mệt mỏi sau một ngày làm việc. Ngoài ra, các nghiên cứu đã chỉ ra rằng việc thiếu ngủ chỉ vài giờ mỗi đêm có thể làm suy giảm khả năng lái xe tương tự như việc uống quá nhiều rượu. Các tổ chức như Asia Injury Prevention (AIP) Foundation đã tiến hành khảo sát với các tài xế xe tải và phát hiện ra rằng hơn 79% trong số họ lái xe liên tục hơn bốn giờ mà không nghỉ ngơi. Điều này cho thấy sự cần thiết của việc phát triển các hệ thống cảnh báo buồn ngủ hiệu quả để cải thiện an toàn giao thông [1], [5].

Tình hình nghiên cứu: Ở nước ta, các trung tâm nghiên cứu đã triển khai nhiều giải pháp nhằm cảnh báo tình trạng mất tập trung do buồn ngủ khi điều khiển phương tiện. Các phương pháp chủ yếu được áp dụng hiện nay bao gồm: phân tích các chỉ số sinh lý của người lái và theo dõi hành vi, phản ứng điều khiển xe để đưa ra cảnh báo kịp thời [1]. Mục tiêu là tích hợp những hệ thống này vào xe để cung cấp cảnh báo kịp thời và giảm thiểu số vụ tai nạn giao thông do buồn ngủ khi lái xe.

- Một số các nghiên cứu trong nước:

1. Trương Quốc Định & CTV (2015) đã xây dựng một hệ thống phát hiện tình trạng ngủ gật của lái xe theo phương pháp xác định khoảng cách mắt với chân mày và phương pháp tính độ cong của đường tiếp giáp hai mí mắt [21].

2. Nguyễn Minh Sơn, Nguyễn Văn Bình, Nguyễn Ngọc Lâm (2018) cây dựng hệ thống phát hiện trạng thái buồn ngủ của tài xế từ xa với thời gian xử lý dưới 1,5 giây và độ chính xác gần 90%. Hệ thống tích hợp camera, hồng ngoại và vi xử lý để giám sát và cảnh báo hiệu quả [2].

3. VinAI đã phát triển một hệ thống giám sát tài xế (Driver Monitoring System - DMS) sử dụng camera hồng ngoại để nhận diện gương mặt và các hành vi nguy hiểm như mệt mỏi, buồn ngủ [5].

## 1.2 Thực trạng nghiên cứu quốc tế

Tình hình thực tế: Mệt mỏi, buồn ngủ khi lái xe là một vấn đề lo ngại, đóng góp vào tỷ lệ cao các vụ tai nạn giao thông trên toàn cầu. Theo Atherton (2023), tài xế thiếu tỉnh táo góp phần vào hơn 1/5 các vụ tai nạn dẫn đến tử vong ở Mỹ, với khoảng 91,000 vụ tai nạn được ghi nhận trong năm 2017. Đặc biệt, nghiên cứu chỉ ra rằng thiếu ngủ chỉ 17 giờ có thể làm suy giảm khả năng lái xe tương đương với nồng độ cồn 0.05% trong máu [3].

Tình hình nghiên cứu: Nhận thức được tầm quan trọng của vấn đề, các nghiên cứu quốc tế đã tập trung việc phát hiện buồn ngủ khi lái xe thông qua giám sát mắt và phân tích hình ảnh. Các giải pháp được công bố để khắc phụ độ chính xác trong điều kiện thực tế như ánh sáng, thời tiết khác nhau. Một bài báo tổng quan cung cấp đánh giá cập nhật về các hệ thống phát hiện buồn ngủ của tài xế được triển khai trong thập kỷ qua [4].

- Một số các nghiên cứu ngoài nước:

1. A. Cheng et. al. (2012) đã tiến hành việc theo dõi mắt và xử lý hình ảnh. Một thuật toán phát hiện mắt với sáu khoảng đo được tính bằng tỷ lệ phần mí mắt đóng, thời gian đóng tối đa, tần số chớp mắt, mức mở cửa trung bình của mắt, tốc độ mở của mắt, và tốc độ đóng của mắt [23].

2. Chisty, Jasmeen Gill (2015) đề xuất các giải pháp khắc phục các hạn chế về phát hiện buồn ngủ trong các điều kiện ánh sáng, thời tiết, ngày và đêm [24].

3. Abdusalamov và cộng sự (2023) sử dụng phân tích chớp mắt và các thuật toán học sâu để phát hiện tình trạng buồn ngủ [25].

4. Cigala-Álvarez và cộng sự (2024) kết hợp nhận diện khuôn mặt và phát hiện cơn buồn ngủ với công nghệ IoT và kết nối 5G/6G [6].

## 1.3 Vấn đề tồn tại và định hướng nghiên cứu

Mặc dù các nghiên cứu trong và ngoài nước đã đạt được nhiều tiến bộ trong việc phát hiện tình trạng buồn ngủ khi lái xe, song vẫn tồn tại những thách thức đáng kể trong quá trình ứng dụng thực tế, đặc biệt là khi triển khai trong điều kiện môi trường đa dạng và đòi hỏi tính thời gian thực cao.

Thứ nhất, các hệ thống giám sát hiện nay vẫn còn phụ thuộc mạnh vào điều kiện ánh sáng và môi trường xung quanh. Trong môi trường thiếu sáng (ví dụ như ban đêm) hoặc khi người lái đeo kính, các thuật toán nhận diện khuôn mặt và mắt thường giảm đáng kể độ chính xác [4], [5], [23]. Mặc dù việc sử dụng camera hồng ngoại giúp cải thiện khả năng phát hiện trong điều kiện ánh sáng yếu [4], nhưng chi phí phần cứng và tính khả thi khi triển khai ở quy mô lớn vẫn là một rào cản đáng kể.

Thứ hai, nhiều phương pháp hiện tại vẫn chỉ tập trung vào các đặc trưng hành vi đơn lẻ như tần suất chớp mắt, trạng thái mở mắt hay hành vi ngáp [21], [25], dẫn đến hạn chế trong việc phân biệt chính xác giữa trạng thái mệt mỏi thoáng qua và tình trạng buồn ngủ thực sự. Việc chưa khai thác đầy đủ mối tương quan giữa các yếu tố như độ nghiêng đầu, trạng thái mắt, miệng và các biểu hiện khuôn mặt khác khiến cho độ tin cậy của hệ thống giảm trong các tình huống thực tế phức tạp [2], [24].

Thứ ba, nhiều mô hình học sâu hiện nay tuy đạt được độ chính xác cao nhưng yêu cầu tài nguyên tính toán lớn, gây khó khăn trong việc triển khai trên các nền tảng phần cứng nhúng có cấu hình giới hạn như hệ thống trong xe hơi [4], [6]. Đây là một trong những lý do khiến các giải pháp hiện tại khó phổ biến rộng rãi trong thị trường thực tế.

Trước những vấn đề trên, đề tài hướng tới việc phát triển một hệ thống phát hiện trạng thái buồn ngủ của tài xế dựa trên sự kết hợp giữa các phương pháp thị giác máy tính hiện đại và học sâu. Cụ thể, kỹ thuật YOLOv10 sẽ được sử dụng để phát hiện chính xác các vùng quan trọng trên khuôn mặt như mắt và miệng [9], [20], trong khi mô hình VGG16 hoặc FaceNet sẽ hỗ trợ phân loại trạng thái buồn ngủ thông qua việc phân tích các đặc trưng khuôn mặt [10], [19]. Hệ thống sẽ được huấn luyện trên các bộ dữ liệu chuyên biệt như Face Pose Dataset, Eyes Dataset và Yawn Dataset [12], [13] nhằm đảm bảo độ chính xác cao trong nhiều bối cảnh thực tế khác nhau. Ngoài ra, nghiên cứu sẽ tập trung vào việc tối ưu tốc độ xử lý và giảm độ trễ nhằm đáp ứng yêu cầu vận hành thời gian thực khi triển khai trong các hệ thống hỗ trợ lái xe.

# CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

Nghiên cứu này bao gồm các thuật toán về học máy, học sâu và thị giác máy tính phục vụ cho nghiên cứu giám sát trạng thái mất tập trung, buồn ngủ của người tham gia giao thông, tập trung về YOLOv10 để phát mắt, miệng trên khuôn mặt và VGG16 cho phân loại trạng thái mắt, miệng.

## 2.1 Tổng quan về Artificial Neural Network (ANN)

2.1.1 Giới thiệu về ANN

Mạng nơ-ron nhân tạo (ANN) là một dạng kiến trúc tính toán mô phỏng cấu trúc và hoạt động của bộ não con người. ANN bao gồm nhiều đơn vị xử lý nhỏ gọi là nơ-ron (neurons), được kết nối với nhau qua các liên kết có trọng số (weights). ANN học bằng cách cập nhật trọng số để cải thiện độ chính xác của đầu ra dự đoán [4], [14].

Trong nghiên cứu này, ANN được xem là nền tảng cơ bản cho các mô hình học sâu như CNN hay các mạng phát hiện đối tượng. Một dạng cơ bản và phổ biến của ANN là mạng đa tầng (MLP - Multilayer Perceptron), một kiến trúc gồm nhiều lớp nơ-ron (ít nhất ba lớp: đầu vào, ẩn, và đầu ra) được kết nối đầy đủ (fully connected) giữa các lớp. MLP có khả năng học các mối quan hệ phi tuyến tính trong dữ liệu, nhờ vào các lớp ẩn và hàm kích hoạt như Sigmoid, Tanh, hoặc ReLU. MLP thường được sử dụng trong các bài toán phân loại hoặc hồi quy đơn giản, và là bước nền tảng để hiểu các mô hình học sâu phức tạp hơn như CNN hay các mạng phát hiện đối tượng. Việc tìm hiểu MLP giúp làm rõ cách các lớp nơ-ron phối hợp với nhau để xử lý dữ liệu trước khi đi vào các cơ chế huấn luyện như lan truyền xuôi và lan truyền ngược [16], [28], [32].

2.1.2 Cấu trúc của ANN

Một hệ thống ANN thông thường bao gồm ba loại lớp chính (Hình 2.1):

*- Lớp đầu vào* (Input Layer):

* Nhận dữ liệu từ bên ngoài và chuyển vào mạng.
* Mỗi nơ-ron trong lớp này tương ứng với một đặc trưng (feature) của dữ liệu.

- *Lớp ẩn* (Hidden Layers): Xử lý dữ liệu bằng cách tính toán và truyền thông tin từ đầu vào đến đầu ra, một mạng có thể có nhiều lớp ẩn.

- *Lớp đầu ra* (Output Layer):

* Đưa ra dự đoán đầu ra.
* Nếu là mô hình dùng để phân loại, số lượng nơ-ron trong lớp này sẽ bằng số lượng lớp cần phân loại.

Các nơ-ron ở từng lớp được liên kết với lớp tiếp theo bằng các trọng số (weights), giúp mạng học cách biểu diễn và đặc trưng của dữ liệu [16].

A diagram of a network

AI-generated content may be incorrect.

Hình 2.1. Mạng nơ ron gồm 3 layers

2.1.3 Quá trình huấn luyện mạng nơ-ron

Huấn luyện một mạng nơ-ron là quá trình tinh chỉnh các giá trị trọng số để giúp mô hình đạt được độ chính xác cao nhất trên tập dữ liệu huấn luyện. Quá trình này gồm ba bước chính: lan truyền xuôi (Forward Propagation), tính toán hàm mất mát (Loss Function) và lan truyền ngược (Backward Propagation) [16].

***2.1.3.1 Lan truyền xuôi (Forward Propagation)***

Lan truyền xuôi là bước đầu tiên trong quá trình huấn luyện mạng nơ-ron, nơi mà dữ liệu được đưa vào hệ thống từ lớp đầu vào và tiếp tục được xử lý qua các lớp ẩn cho đến lớp đầu ra. Mỗi nơ-ron tại một lớp nhận tín hiệu từ các nơ-ron của lớp trước đó, nhân chúng với các trọng số tương ứng, cộng thêm một hệ số điều chỉnh gọi là bias. Kết quả của phép tính này sau đó được chuyển qua một hàm kích hoạt để tạo ra đầu ra của nơ-ron đó [16], [28].

Quá trình lan truyền xuôi không chỉ đơn thuần là tính toán – nó chính là cách mà mạng nơ-ron thử nghiệm dự đoán đầu ra với các tham số hiện có. Ở thời điểm này, mạng chưa biết kết quả đúng hay sai, mà chỉ đưa ra một đầu ra tạm thời dựa trên các trọng số hiện tại. Đây là cơ sở để sau đó so sánh với dữ liệu thực tế nhằm điều chỉnh các trọng số thông qua lan truyền ngược (backpropagation) [4].

A diagram of a mathematical equation

AI-generated content may be incorrect.

Hình 2.2. Tính toán đầu ra bằng lan truyền xuôi

Theo hình 2.2, mỗi nơ-ron tính tổng trọng số của các đầu vào rồi áp dụng hàm kích hoạt để tạo đầu ra. Giá trị đầu ra từ này trở thành đầu vào cho các nơ-ron của lớp tiếp theo. Cụ thể mỗi nơ-ron tính tổng trọng số theo công thức:

Trong đó:

- : trọng số

- : trọng số

- b: bias

Sau đó, áp dụng hàm kích hoạt g(z). Hàm kích hoạt được chọn tùy bài toán, có thể chọn hàm kích hoạt khác nhau cho từng layer. Bên dưới là những hàm kích hoạt phổ biến:

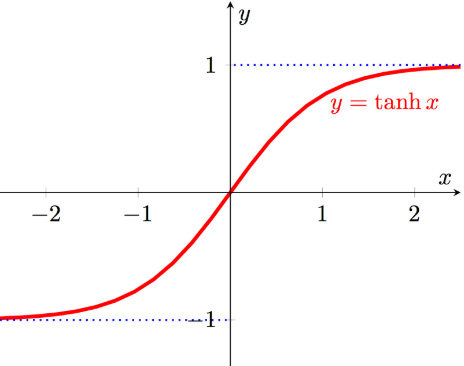
**- *Sigmoid***: Đầu ra (0,1) được sử dụng hiệu quả trong bài toán phân loại nhị phân. Tuy nhiên sigmoid có nhược điểm là dễ gặp vấn đề gradient vanishing khiến cho quá trình huấn luyện diễn chậm lại (Hình 2.3).

A graph with a curve

AI-generated content may be incorrect.

Hình 2.3. Hàm Sigmoid

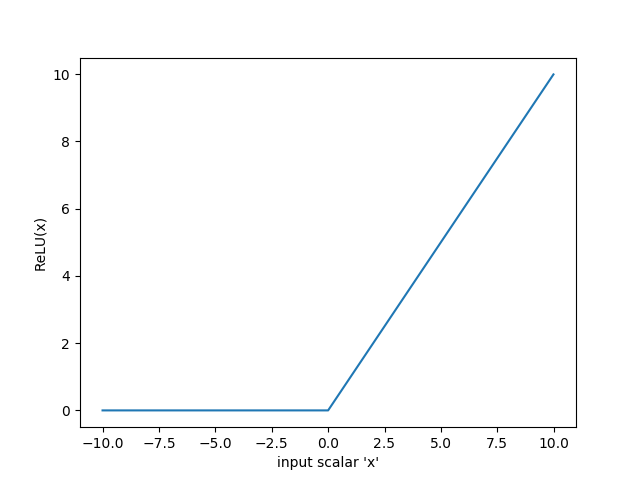
**- *Tanh***: Tương tự hàm Sigmoid nhưng giá trị nằm trong (−1,1) gradient ổn định hơn (Hình 2.4).



Hình 2.4. Hàm Tanh

Khi giá trị z tiến vền dương/âm vô cực, hàm số trở nên ít dốc nghĩa là độ dốc hàm kích hoạt trở nên rất nhỏ. Do đó đạo hàm tại những điểm này tiến về gần 0, làm cho gradient lan truyền yếu đi. Dẫn đến quá trình cập nhập trọng số bị chậm lại nên mô hình học chậm. Giải pháp cho vấn đề này là hàm ReLU.

**- *ReLU***: công thức là max(0, z), nghĩa là nếu đầu vào z nhỏ hơn 0, output sẽ là 0, còn nếu z lớn hơn hoặc bằng 0, output sẽ bằng chính z. Đạo hàm sẽ luôn bằng 1 đối với các giá trị dương của z dù z có lớn bao nhiêu, giúp việc học trở nên nhanh hơn, tránh hiện tượng “gradient vanishing” đã đề cập ở phía trên (Hình 2.5).



Hình 2.5. Hàm ReLU

***2.1.3.2 Tính toán hàm mất mát (Loss Function)***

Trong quá trình huấn luyện mạng nơ-ron, hàm mất mát (loss function) đóng vai trò rất quan trọng, giúp đo lường mức độ sai lệch giữa đầu ra dự đoán của mô hình và giá trị thực tế trong dữ liệu huấn luyện. Mỗi mẫu dữ liệu sẽ có một giá trị Cost riêng, được tính toán dựa trên sự chênh lệch giữa đầu ra và giá trị thực tế, và giá trị này được gọi là Loss cho mẫu dữ liệu đó. Việc tính toán Cost thường được thực hiện dựa trên các phương pháp như Cross-Entropy Loss hoặc Mean Squared Error (MSE), tùy thuộc vào loại bài toán mà mô hình đang giải quyết [30].

Để đánh giá mức độ tổng thể của mô hình, người ta tính toán Loss function trên toàn bộ tập huấn luyện, tức là trung bình cộng của tất cả các giá trị Cost trên tập dữ liệu. Đây chính là Loss tổng quát cho mô hình. Mục tiêu trong quá trình huấn luyện là giảm thiểu giá trị Loss này, nhằm tối ưu hóa mô hình để đạt được sự chính xác cao nhất khi dự đoán đầu ra cho dữ liệu mới. Công thức tính Loss Function có thể được biểu diễn như sau:

Trong đó, là số lượng mẫu trong tập huấn luyện, là đầu vào của mẫu thứ , và ​ là đầu ra thực tế. Hàm mất mát này có thể thay đổi tùy theo loại bài toán:

**-** *Cross-Entropy Loss*: được sử dụng trong các bài toán phân loại, đo lường sự khác biệt giữa phân phối xác suất của đầu ra dự đoán và phân phối thực tế [31].

**-** *Mean Squared Error (MSE):* được sử dụng trong các bài toán hồi quy, đo lường độ lệch bình phương giữa giá trị dự đoán và giá trị thực tế [4].

Việc giảm giá trị hàm mất mát thường được thực hiện bằng cách sử dụng Gradient Descent hoặc các thuật toán tối ưu hóa khác. Trong mỗi vòng lặp huấn luyện, các trọng số của mô hình được cập nhật dựa trên đạo hàm của hàm mất mát, nhằm giảm giá trị của Loss xuống mức thấp nhất có thể. Điều này giúp mô hình học từ dữ liệu và cải thiện độ chính xác qua từng vòng huấn luyện.

Hàm mất mát không chỉ giúp đánh giá mô hình mà còn ảnh hưởng trực tiếp đến quá trình học của mạng nơ-ron. Mục tiêu của huấn luyện là làm sao để giá trị Loss ngày càng nhỏ, cải thiện chất lượng của mô hình trong việc nhận diện và dự đoán các đầu ra chính xác hơn.

***2.1.3.3*** ***Lan truyền ngược (Backward Propagation)***

Lan truyền ngược là một trong những quá trình quan trọng trong huấn luyện mạng nơ-ron. Quá trình này sử dụng đạo hàm để tính độ dốc của hàm mất mát đối với các trọng số trong mạng nơ-ron, từ đó cập nhật các trọng số để giảm thiểu sai lệch giữa đầu ra dự đoán và giá trị thực tế. Cụ thể, trọng số sẽ được cập nhật theo công thức sau:

Trong đó:

- θ: tham số (trọng số) sẽ được điều chỉnh trong quá trình huấn luyện

- η: hệ số học, điều chỉnh mức độ thay đổi của trọng số sau mỗi lần cập nhật, đổi quá nhiều thì dễ sai, đổi quá ít thì học chậm.

- : cho biết trọng số đó ảnh hưởng đến sai số ra sao, mô hình sẽ dựa vào đó để biết nên tăng hay giảm trọng số bao nhiêu.

Sau khi tính toán Loss, áp dụng phương pháp lan truyền ngược để cập nhập trọng số của mô hình.

Nhờ quá trình này, mô hình học được từ dữ liệu và dần cải thiện độ chính xác qua từng việc huấn luyện. Đây là nguyên lý cơ bản giúp các mô hình học sâu (deep learning) hoạt động hiệu quả trong các tác vụ nhận diện, phân loại và dự đoán. Nhờ vào quá trình này, các hệ thống nhận diện trạng thái buồn ngủ của lái xe có thể cải thiện hiệu suất qua từng vòng huấn luyện, như các nghiên cứu trong [1], [2], và [4].

## 2.2 VGG16 (Convolutional Neural Network - CNN)

2.2.1 Giới thiệu về Mạng nơ-ron tích chập (CNN)

Mạng nơ-ron tích chập (Convolutional Neural Network – CNN) là một dạng đặc biệt của mạng nơ-ron nhân tạo (Artificial Neural Network – ANN), được thiết kế để xử lý dữ liệu có cấu trúc dạng lưới, chẳng hạn như hình ảnh. CNN chủ yếu được sử dụng trong các tác vụ nhận dạng hình ảnh và video, nhờ khả năng tự động học các đặc trưng từ dữ liệu mà không cần phải có sự can thiệp thủ công. Một trong những đặc điểm nổi bật của CNN là việc sử dụng các lớp tích chập (convolutional layers) để nhận diện các đặc trưng không gian trong hình ảnh, từ các mẫu cơ bản (như cạnh, góc) đến các đặc trưng phức tạp hơn như khuôn mặt, vật thể.

CNN hoạt động bằng cách áp dụng các bộ lọc (filters) qua các lớp tích chập để trích xuất các đặc trưng từ hình ảnh. Sau khi trích xuất các đặc trưng này, chúng sẽ được đưa vào các lớp giảm kích thước (pooling layers) để giảm thiểu sự phức tạp và nâng cao tính trừu tượng của đặc trưng. Cuối cùng, các đặc trưng này được chuyển đến các lớp fully connected (FC) để thực hiện phân loại hoặc dự đoán kết quả. CNN là một công nghệ chủ chốt trong nhiều ứng dụng nhận diện hình ảnh, nhận diện đối tượng, và nhận dạng khuôn mặt [9], [10].

2.2.2 Giới thiệu về VGG16

VGG16 là một mô hình mạng nơ-ron tích chập (CNN) nổi bật được phát triển bởi nhóm Visual Geometry Group (VGG) tại Đại học Oxford vào năm 2014. Mô hình này được xây dựng với cấu trúc rất đơn giản nhưng mạnh mẽ, đặc biệt là nhờ vào độ sâu của nó (16 lớp có tham số). VGG16 có khả năng học các đặc trưng hình ảnh phức tạp từ dữ liệu nhờ vào việc sử dụng các bộ lọc nhỏ (3x3) qua nhiều lớp tích chập, giúp nó nhận diện chính xác các mẫu và đặc trưng trong ảnh.

Với cấu trúc đơn giản nhưng hiệu quả, VGG16 trở thành một mô hình nổi bật trong các ứng dụng học sâu, đặc biệt là trong nhận dạng hình ảnh và phân loại hình ảnh. Mặc dù mô hình này có số lượng tham số rất lớn (hơn 138 triệu tham số), nhưng nhờ vào sự hiệu quả trong việc tối ưu hóa và tính khả dụng, VGG16 vẫn là một lựa chọn phổ biến cho các nghiên cứu và ứng dụng trong lĩnh vực thị giác máy tính [10].

2.2.3 Kiến trúc của VGG16

Mô hình gồm tổng cộng 16 lớp có tham số, trong đó có 13 lớp tích chập (Convolutional Network) và 3 lớp FC (Fully Connected) ở cuối:

- Các lớp tích chập sử dụng bộ lọc 3x3 và số lượng bộ lọc tăng dần qua các lớp, làm cho mô hình hiểu được các đặc trưng từ cơ bản đến phức tạp. Các lớp này lần lượt trích xuất các đặc trưng không gian của hình ảnh.

- Sau khi các đặc trưng được trích xuất, 3 lớp FC sẽ tổng hợp và đưa ra kết quả phân loại.

Mô hình còn dùng hàm ReLU trong các lớp tích chập để giúp mô hình học nhanh hơn và tránh vấn đề vanishing gradient. Ở tầng cuối, hàm Softmax được sử dụng để tính xác suất của các lớp khác nhau, giúp mô hình xác định chính xác lớp của một ảnh. Hình dưới đây là kiến trúc của mô hình VGG16 (Hình 2.6).

A diagram of a computer model

AI-generated content may be incorrect.

Hình 2.6. Kiến trúc của VGG16

*Hình ảnh từ bài viết "Everything you need to know about VGG16" trên Medium, được trích từ:* [*https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918*](https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918)*.*

2.4.3 Ứng dụng VGG16 trong phân loại trạng thái mắt và miệng

Trong nghiên cứu này, VGG16 được sử dụng để phân loại các trạng thái của mắt và miệng, như mắt nhắm/mở và miệng ngáp/không ngáp. Nhờ vào việc có một lượng lớn tham số (hơn 138 triệu tham số), mô hình VGG16 có khả năng học rất kỹ lưỡng các đặc điểm của ảnh, giúp nhận diện sự khác biệt nhỏ giữa các trạng thái này. Các lớp tích chập của VGG16 sẽ trích xuất các đặc trưng không gian của mắt và miệng, trong khi các lớp fully connected (FC) sẽ phân loại các đặc trưng này vào các lớp tương ứng.

VGG16 cũng dễ dàng điều chỉnh lại (fine-tune) với các bộ dữ liệu mới, vì mô hình này đã được huấn luyện trên một lượng dữ liệu lớn trước đó, giúp giảm thiểu thời gian huấn luyện cho các tác vụ đặc thù. Chính vì những đặc điểm này, VGG16 là một mô hình lý tưởng để áp dụng trong nhận diện trạng thái buồn ngủ của lái xe.

## 2.3 YOLO (You Only Look Once)

Trong các ứng dụng thị giác máy tính, bài toán phân loại ảnh và phát hiện đối tượng thường được phân biệt rõ ràng. Phân loại ảnh (Image Classification) là quá trình xác định nhãn của một ảnh duy nhất, ví dụ: hệ thống nhận diện một ảnh chứa chó hay người. Ngược lại, phát hiện đối tượng (Object Detection) không chỉ xác định sự hiện diện của đối tượng mà còn định vị chúng thông qua các hộp giới hạn (bounding boxes) [19].

YOLO (You Only Look Once) là một trong những phương pháp phát hiện đối tượng thời gian thực được sử dụng rộng rãi nhờ vào tốc độ xử lý nhanh và độ chính xác cao. Thay vì phân tách quá trình phát hiện thành nhiều giai đoạn như một số mô hình truyền thống, YOLO thực hiện tất cả trong một lần lan truyền duy nhất qua mạng (single forward pass), từ đó vừa xác định vị trí vừa phân loại đối tượng trong ảnh [19].

2.3.1 Phân biệt giữa phân loại ảnh và phát hiện đối tượng

Trong xử lý ảnh, có sự khác biệt rõ ràng giữa phân loại ảnh và phát hiện đối tượng (Hình 2.7).

- Phân loại ảnh: Xác định xem nội dung trong ảnh thuộc về loại nào. Ví dụ, khi mô hình được cung cấp một bức ảnh, nó sẽ trả lời: “Đây là chó hay là người?” — Kết quả chỉ là một nhãn duy nhất, ví dụ: *Dog = 1, Person = 0*.

- Phát hiện đối tượng: Không chỉ xác định có đối tượng gì trong ảnh, mà còn chỉ rõ vị trí của đối tượng đó bằng cách tạo ra một hộp bao quanh (bounding box). Như trong hình, mô hình không chỉ nói đó là một con vật, mà còn xác định được chính xác vị trí con vật đang ở đâu trong ảnh.

**A dog jumping in the air

AI-generated content may be incorrect.**

Hình 2.7. Image Classification & Object Localization

2.3.2 Giới thiệu về YOLO

YOLO (You Only Look Once) là một trong những kiến trúc phát hiện đối tượng hiện đại, nổi bật bởi khả năng xử lý nhanh và phù hợp với các ứng dụng yêu cầu thời gian thực, chẳng hạn như giám sát tình trạng buồn ngủ khi lái xe. Khác với các phương pháp truyền thống vốn chia bài toán thành nhiều bước riêng biệt (như trích xuất đặc trưng và phân loại), YOLO thực hiện việc nhận diện và định vị đối tượng chỉ trong một lần phân tích duy nhất, giúp cải thiện tốc độ xử lý đáng kể [19].

2.3.3 Nguyên lý hoạt động của YOLO

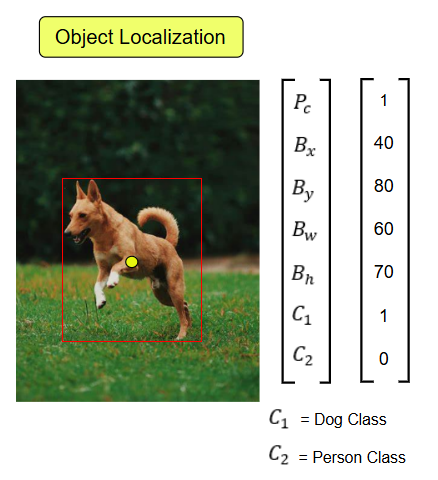
Mô hình chia ảnh đầu vào thành nhiều ô nhỏ (grid cells). Mỗi ô sẽ chịu trách nhiệm dự đoán các thông tin liên quan đến vật thể nếu tâm của vật thể đó nằm trong ô đó. Khi dự đoán, kết quả sẽ là một tập hợp giá trị gồm:

- : Xác suất có vật thể trong ô đó. Nếu có, giá trị là 1; nếu không, là 0

- , : Tọa độ trung tâm của vật thể (ví dụ như chấm vàng trong ảnh).

- , : Chiều rộng và chiều cao của hộp chứa vật thể (bounding box, như khung đỏ trong ảnh).

- , : Dự đoán lớp của vật thể. Ví dụ: nếu là chó, = 1, = 0; nếu là người thì ngược lại.

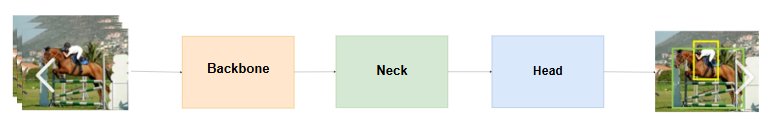


Hình 2.8. Minh họa cách YOLO xác định vị trí và phân loại đối tượng trong ảnh

Dựa vào những giá trị này, YOLO không chỉ cho biết có gì trong ảnh, mà còn xác định vị trí chính xác của vật thể nằm ở đâu.

2.3.4 Kiến Trúc Của YOLO

Kiến trúc của YOLO gồm ba thành phần chính: Backbone, Neck, và Head (Hình 2.9).

****

Hình 2.9. Kiến trúc của YOLO

*- Backbone*: Đây là phần đầu tiên của mô hình, có nhiệm vụ quét qua ảnh đầu vào để tìm ra những điểm quan trọng (đặc trưng). Tùy phiên bản, mô hình có thể sử dụng các mạng như CSPDarknet[6] hoặc MobileNet[7] để cân bằng giữa độ chính xác và tốc độ.

*- Neck*: đây là phần trung gian kết nối backbone và head, giúp kết hợp các đặc trưng từ nhiều cấp độ khác nhau trong mạng. Các kỹ thuật như Feature Pyramid Network (FPN) [7] thường được sử dụng để đảm bảo khả năng phát hiện các đối tượng có kích thước đa dạng.

*- Head*: Đây là nơi mô hình đưa ra kết quả cuối cùng. Dựa trên thông tin từ các phần trước, head sẽ dự đoán vật thể nằm ở đâu, độ tin cậy của dự đoán là bao nhiêu, đó là loại đối tượng gì và chia ảnh thành các ô vuông và đưa ra dự đoán cho từng ô đó.

Với kiến trúc linh hoạt và hiệu quả, YOLO đặc biệt phù hợp trong các ứng dụng yêu cầu thời gian thực như hệ thống giám sát tình trạng buồn ngủ khi lái xe, nơi việc phát hiện chính xác vị trí mắt và miệng là rất quan trọng cho các bước xử lý sau.

2.3.5 Ưu điểm và hạn chế của YOLO

YOLO (You Only Look Once) là một trong những phương pháp phát hiện đối tượng tiên tiến được đánh giá cao nhờ khả năng xử lý thời gian thực với tốc độ và độ chính xác vượt trội. Mô hình hoạt động theo nguyên lý một bước (single-stage detector), nghĩa là toàn bộ quá trình nhận diện và xác định vị trí vật thể được thực hiện trong một lần xử lý ảnh duy nhất. Điều này giúp YOLO đạt được tốc độ nhanh hơn so với các phương pháp nhiều bước (multi-stage detectors) như R-CNN hay Faster R-CNN [19].

Một trong những ưu điểm nổi bật của YOLO là khả năng tổng quát hóa tốt đối với các ảnh chưa từng gặp qua, cũng như khả năng phát hiện nhiều đối tượng cùng lúc trong một khung hình. Điều này đặc biệt hữu ích trong các bài toán theo dõi hành vi lái xe, nơi có thể có nhiều vùng khuôn mặt cần được giám sát đồng thời như mắt và miệng.

Tuy nhiên, bên cạnh các lợi thế kể trên, YOLO vẫn tồn tại một số hạn chế nhất định. Do ảnh đầu vào được chia thành các ô lưới (grid cells), nên mô hình có thể gặp khó khăn trong việc phát hiện các đối tượng nhỏ hoặc nằm gần nhau, đặc biệt là trong điều kiện ánh sáng yếu hoặc khi đối tượng bị che khuất một phần. Ngoài ra, việc dự đoán sai vị trí hoặc loại vật thể vẫn có thể xảy ra nếu đặc trưng hình ảnh không đủ rõ ràng [4], [19].

2.3.6 Ứng dụng YOLO trong đề tài

Trong phạm vi của đề tài nghiên cứu này, YOLO được sử dụng như một thành phần quan trọng trong hệ thống phát hiện trạng thái buồn ngủ của người lái xe. Cụ thể, mô hình YOLOv10 được tích hợp để phát hiện các vị trí đặc trưng trên khuôn mặt như mắt và miệng trong thời gian thực. Các thông tin hình học này đóng vai trò đầu vào cho các mô-đun phân loại trạng thái (mở/nhắm mắt, ngáp/không ngáp), từ đó đưa ra đánh giá về mức độ tỉnh táo của người lái.

Mô hình YOLOv10 đã được tinh chỉnh (fine-tuned) lại trên tập dữ liệu được nhóm nghiên cứu tự gán nhãn để phù hợp hơn với môi trường và điều kiện thực tế của bài toán. Việc sử dụng YOLO trong đề tài cho phép đạt được hiệu quả cao về tốc độ xử lý, đảm bảo phản ứng kịp thời trong các tình huống nguy hiểm do người lái có biểu hiện buồn ngủ kéo dài [16], [19].

## 2.4 YOLOv10

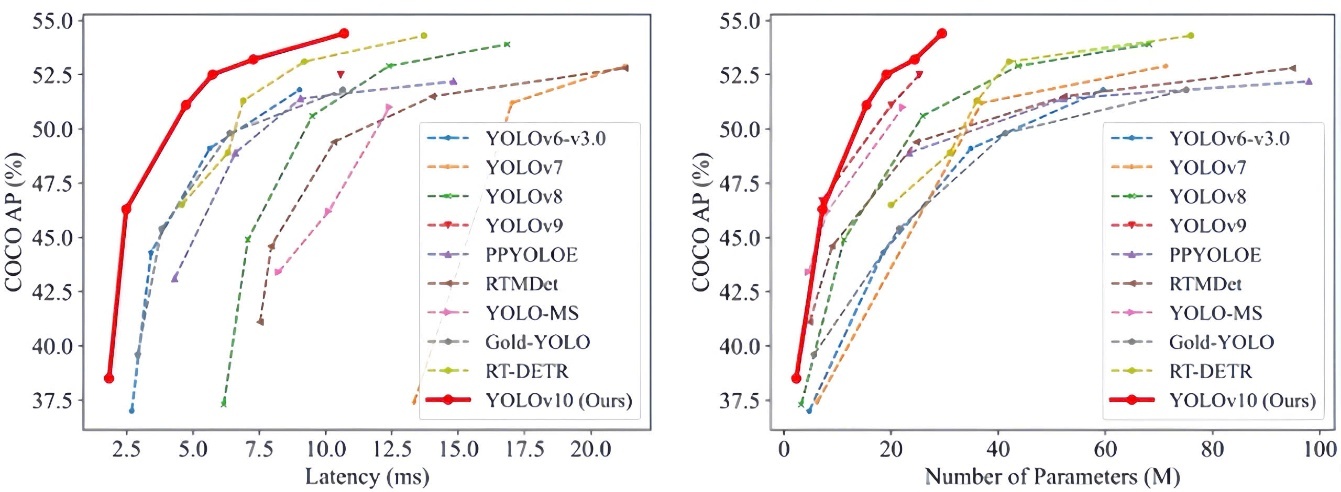
2.4.1 Giới thiệu về YOLOv10

YOLOv10 là phiên bản mới nhất của mô hình YOLO, được phát triển bởi nhóm nghiên cứu tại Đại học Thanh Hoa và xây dựng trên nền tảng Python do Ultralytics phát triển. Được thiết kế nhằm cải thiện hiệu quả nhận diện đối tượng trong thời gian thực, YOLOv10 mang đến nhiều cải tiến vượt trội so với các phiên bản trước, đặc biệt trong cách thức xử lý kết quả đầu ra và tối ưu hóa cấu trúc mạng. Điều này giúp mô hình đạt được tốc độ xử lý nhanh, độ chính xác cao, và đặc biệt thích hợp cho các ứng dụng yêu cầu tính toán thời gian thực như giám sát người lái xe trong các tình huống buồn ngủ [9].

Một trong những cải tiến đáng chú ý của YOLOv10 là việc loại bỏ hoàn toàn bước loại bỏ các hộp dư (Non-Maximum Suppression - NMS) truyền thống. Điều này đạt được nhờ vào cơ chế Dual Assignment, một phương pháp giúp giảm thiểu sự phức tạp trong quá trình dự đoán và nâng cao hiệu suất tính toán, đồng thời cải thiện độ chính xác của mô hình [9]. Việc không cần bước NMS này giúp YOLOv10 trở nên nhanh chóng và hiệu quả hơn trong việc xử lý các đối tượng phức tạp và đa dạng trong ảnh.

2.4.2 Hiệu quả của YOLOv10 so với các phiên bản trước

Hình bên dưới (Hình 2.10) minh họa sự cải tiến về hiệu suất của YOLOv10 so với các phiên bản trước đó, đánh giá qua độ chính xác (COCO AP), độ trễ (Latency), và số lượng tham số (Parameters).



Hình 2. 10. So sánh hiệu suất của YOLOv10 và các phiên bản trước về độ chính xác (COCO AP) theo đỗ trễ (Latency) và số lượng tham số (Parameters).

*Nguồn: YOLOv10 Official GitHub - Tsinghua University, 2024 [9].*

Trong hình 2.10, có thể thấy rõ rằng YOLOv10 đạt được sự cân bằng tối ưu giữa độ chính xác và tốc độ xử lý:

- Biểu đồ bên trái thể hiện rằng YOLOv10 duy trì độ chính xác cao ngay cả khi độ trễ thấp, cho thấy khả năng xử lý nhanh nhưng vẫn đảm bảo kết quả chính xác.

- Biểu đồ bên phải cho thấy YOLOv10 không yêu cầu số lượng tham số quá lớn mà vẫn đạt được độ chính xác tốt, chứng tỏ mô hình này rất nhẹ, phù hợp với các thiết bị có tài nguyên hạn chế..

Với những cải tiến này, YOLOv10 không chỉ mang lại hiệu suất xử lý nhanh mà còn tiết kiệm tài nguyên tính toán, đặc biệt thích hợp cho các ứng dụng yêu cầu xử lý thời gian thực với độ chính xác cao, như hệ thống giám sát trạng thái buồn ngủ của người lái xe [9].

2.4.3 YOLOv10 trong phát hiện mắt và miệng

Trong phạm vi nghiên cứu này, YOLOv10 được áp dụng để phát hiện các vùng quan trọng trên khuôn mặt người lái xe, bao gồm mắt và miệng. Nhờ vào khả năng phát hiện nhanh và chính xác các đối tượng trong ảnh, YOLOv10 giúp nhận diện rõ ràng vị trí của mắt và miệng, từ đó tạo cơ sở để phân loại các trạng thái như mắt mở/nhắm và miệng ngáp/không ngáp. Việc sử dụng YOLOv10 giúp hệ thống có thể phân tích và đưa ra phản hồi nhanh chóng, góp phần nâng cao độ chính xác trong việc phát hiện trạng thái buồn ngủ của người lái xe trong thời gian thực.

Thông qua việc ứng dụng YOLOv10 trong phát hiện các đặc trưng khuôn mặt, mô hình không chỉ mang lại hiệu quả cao về mặt tốc độ mà còn cải thiện tính chính xác trong quá trình phân loại trạng thái buồn ngủ, góp phần giảm thiểu rủi ro tai nạn giao thông do lái xe mệt mỏi [16].

## 2.5 Học chuyển giao (Transfer learning) và tinh chỉnh (Fine-tuning)

Trong lĩnh vực học sâu, học chuyển giao là một chiến lược giúp tận dụng kiến thức đã được mô hình học được từ một nhiệm vụ ban đầu để áp dụng cho một nhiệm vụ khác có liên quan. Thay vì huấn luyện một mô hình từ đầu – điều này thường yêu cầu lượng lớn dữ liệu và tài nguyên tính toán – ta có thể khởi đầu bằng một mô hình đã được huấn luyện trước trên tập dữ liệu lớn (chẳng hạn như ImageNet) và chỉ điều chỉnh lại một phần để phù hợp với bài toán mới [10], [30].

Tiếp theo bước học chuyển giao, tinh chỉnh mô hình (*fine-tuning*) là kỹ thuật cho phép cập nhật trọng số của một số tầng mạng nhất định (thường là các tầng cuối) bằng cách tiếp tục huấn luyện với dữ liệu mới. Điều này giúp mô hình thích nghi tốt hơn với đặc trưng của dữ liệu hiện tại, từ đó nâng cao hiệu suất và độ chính xác của mô hình trong tác vụ cụ thể [10].

Trong khuôn khổ nghiên cứu này, chúng tôi đã áp dụng các phương pháp trên được áp dụng vào hai mô hình chính:

- *YOLOv10* được sử dụng để phát hiện các đặc điểm khuôn mặt quan trọng như mắt và miệng. Mô hình này sau đó được tinh chỉnh trên tập dữ liệu do nhóm nghiên cứu tự xây dựng và gán nhãn, nhằm thích nghi tốt hơn với đặc điểm của dữ liệu trong bối cảnh phát hiện dấu hiệu buồn ngủ khi lái xe [19].

- *VGG16* đóng vai trò trong việc phân loại trạng thái của mắt (mở hoặc nhắm) và miệng (ngáp hoặc không ngáp). Việc huấn luyện mô hình này được thực hiện trên các bộ dữ liệu công khai như *Face Pose Dataset* và *Yawn Eye Dataset New* [9], [11], [12].

# CHƯƠNG 3. ĐỐI TƯỢNG VÀ PHƯƠNG PHÁP NGHIÊN CỨU

Chương này trình bày chi tiết về đối tượng nghiên cứu của đề tài, bao gồm các đặc trưng hình ảnh khuôn mặt liên quan đến trạng thái buồn ngủ như mắt và miệng. Đồng thời, chương cũng mô tả phương pháp tiếp cận được sử dụng trong quá trình phát triển hệ thống: từ nghiên cứu lý thuyết, lựa chọn mô hình học sâu, xử lý dữ liệu đến xây dựng hệ thống thực nghiệm và đánh giá hiệu quả. Đây là cơ sở quan trọng cho việc triển khai hệ thống cảnh báo người lái xe ngủ gật sử dụng kỹ thuật thị giác máy tính.

## 3.1 Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài là hệ thống giám sát trạng thái buồn ngủ khi lái xe thông qua các biểu hiện trên khuôn mặt người lái, cụ thể tập trung vào hai khu vực chính: vùng mắt và vùng miệng. Nhiều nghiên cứu trước đây cũng đã chứng minh việc theo dõi khuôn mặt là phương pháp hiệu quả để phát hiện tình trạng mất tỉnh táo của người điều khiển phương tiện [1], [4], [22].

3.1.1 Vùng mắt

*- Mục tiêu theo dõi*: Phát hiện trạng thái mắt đang mở hay nhắm nhằm đánh giá mức độ tỉnh táo của người lái xe.

- Dấu hiệu quan sát:

* Mắt mở: Có thể thấy rõ tròng mắt và đồng tử.
* Mắt nhắm: Mí mắt khép lại hoàn toàn hoặc gần như hoàn toàn.
* Thời gian nhắm mắt: Nếu trạng thái mắt nhắm liên tục vượt quá 2 giây, đây có thể là dấu hiệu của tình trạng ngủ gật ngắn (microsleep) – một hiện tượng nguy hiểm khi đang điều khiển phương tiện [13], [15].

3.1.2 Vùng miệng

*- Mục tiêu theo dõi*: Phát hiện hành vi ngáp – một biểu hiện phổ biến khi cơ thể mệt mỏi hoặc thiếu ngủ, từ đó nhận biết tình trạng buồn ngủ của người lái.

- Dấu hiệu quan sát:

* Ngáp: Miệng mở rộng, khoảng cách giữa hai môi giãn rõ rệt, thường kéo dài từ 2–4 giây, đôi khi kèm theo thở sâu hoặc cử động nhẹ ở vai [24].
* Không ngáp**:** Miệng khép hoặc chỉ cử động nhẹ như khi nói chuyện hay nhai.

3.1.3 Cơ sở lựa chọn vùng mắt và miệng cho nghiên cứu

Việc theo dõi các biểu hiện trên khuôn mặt – đặc biệt ở vùng mắt và miệng – là một phương pháp hiệu quả và thực tiễn để phát hiện tình trạng buồn ngủ của người lái xe. Nhiều hệ thống hiện đại hiện nay đã áp dụng kỹ thuật này nhờ khả năng không xâm lấn, tiết kiệm chi phí, và dễ triển khai trong môi trường thực tế [2], [4], [5], [28].

So với các phương pháp dựa trên cảm biến sinh học như đo sóng não, nhịp tim hay điện tâm đồ – vốn yêu cầu thiết bị đeo chuyên biệt – việc sử dụng hình ảnh khuôn mặt mang lại sự thoải mái hơn và phù hợp hơn cho các hệ thống ứng dụng trong phương tiện giao thông [21], [25].

## 3.2 Phương pháp nghiên cứu

- Nghiên cứu lý thuyết:

* Tìm hiểu các phương pháp trong thị giác máy tính để phân tích và nhận diện trạng thái buồn ngủ, đặc biệt là các kỹ thuật phát hiện đối tượng và phân loại hình ảnh.
* Áp dụng mô hình YOLOv10 [19] để phát hiện vùng mắt và miệng, và mô hình VGG16 [18], [9] để phân loại trạng thái mắt mở/nhắm, ngáp/không ngáp.
* Khảo sát tài liệu học thuật và công trình nghiên cứu liên quan đến học sâu và hệ thống giám sát người lái xe [4], [16], [28].

- Nghiên cứu thực nghiệm và lập trình:

* Sử dụng hai bộ dữ liệu gồm Face Pose Dataset [11] và Yawn Eye Dataset New [12], tiến hành gán nhãn dữ liệu và huấn luyện mô hình nhận diện.
* Áp dụng các kỹ thuật tiền xử lý ảnh như Non-Maximum Suppression [8] để tối ưu hóa quá trình phát hiện đối tượng.
* Xây dựng hệ thống thực nghiệm nhận diện trạng thái buồn ngủ và phát tín hiệu cảnh báo dựa trên hình ảnh thời gian thực từ camera.

- Kiểm thử và đánh giá:

* Tiến hành đánh giá hệ thống bằng dữ liệu kiểm thử thực tế.
* Sử dụng các chỉ số **độ chính xác (accuracy)**, **độ nhạy (recall)** và **F1-score** để đo lường hiệu suất của mô hình [16], [24].

- Triển khai ứng dụng:

* Phát triển hệ thống hoàn chỉnh, tích hợp với camera giám sát để phục vụ cho các kịch bản ứng dụng trong giao thông.
* Hướng đến khả năng tích hợp với các nền tảng hỗ trợ người lái và các hệ thống ADAS (Advanced Driver Assistance Systems) hiện đại [26], [27].

# CHƯƠNG 4. QUÁ TRÌNH THỰC HIỆN NGHIÊN CỨU

Chương này trình bày chi tiết quá trình xây dựng hệ thống phát hiện trạng thái buồn ngủ khi lái xe bằng kỹ thuật thị giác máy tính. Từ việc lựa chọn và xử lý dữ liệu đầu vào, thiết kế mô hình học sâu, đến quá trình huấn luyện, đánh giá hiệu suất và triển khai hệ thống hoàn chỉnh đều được mô tả cụ thể. Trong đó, hai mô hình chính được sử dụng là **YOLOv10** để phát hiện các vùng đặc trưng trên khuôn mặt (mắt và miệng), và **VGG16** để phân loại trạng thái mắt (mở/nhắm) và hành vi ngáp. Việc tích hợp hai mô hình này cho phép hệ thống hoạt động hiệu quả trong môi trường thời gian thực, góp phần cảnh báo sớm nguy cơ mất tập trung và buồn ngủ khi lái xe.

## 4.1 Xây dựng mô hình phát triển buồn ngủ

4.1.1 Chiến lược xây dựng hệ thống phát hiện buồn ngủ

Trong nghiên cứu này, nhóm lựa chọn phương pháp tiếp cận dựa trên thị giác máy tính kết hợp học sâu (deep learning) để xây dựng hệ thống phát hiện trạng thái buồn ngủ khi lái xe. Cách tiếp cận được thiết kế thành hai giai đoạn chính tương ứng với hai nhiệm vụ riêng biệt: (1) phát hiện vị trí vùng đặc trưng trên khuôn mặt và (2) phân loại trạng thái tương ứng của từng vùng.

Ở giai đoạn đầu, mô hình YOLOv10 được sử dụng để phát hiện các vùng mắt và miệng trong ảnh khuôn mặt người lái. YOLO (You Only Look Once) là một dòng mô hình phát hiện đối tượng nổi bật nhờ khả năng xử lý thời gian thực và độ chính xác cao. Trong đề tài, nhóm chọn phiên bản YOLOv10n – một biến thể nhẹ phù hợp với môi trường triển khai trên thiết bị phổ thông, có khả năng nhận diện chính xác vùng mắt và miệng trong nhiều tư thế và điều kiện ánh sáng khác nhau.

Sau khi các vùng đặc trưng được xác định, ảnh cắt từ các vùng mắt và miệng sẽ được đưa vào giai đoạn thứ hai. Ở giai đoạn này, mô hình VGG16 – một mạng tích chập sâu đã được huấn luyện trước – sẽ được tinh chỉnh lại (fine-tuning) để thực hiện phân loại trạng thái mắt (mở hoặc nhắm) và hành vi miệng (ngáp hoặc không ngáp). Mô hình VGG16 được lựa chọn vì kiến trúc đơn giản nhưng hiệu quả, dễ tích hợp và dễ điều chỉnh cho các bài toán nhận diện hình ảnh chuyên biệt.

Phương pháp tiếp cận hai giai đoạn này không chỉ giúp chia nhỏ bài toán phức tạp thành hai bước dễ kiểm soát hơn, mà còn tăng tính linh hoạt trong triển khai: mỗi mô hình có thể được tối ưu hóa độc lập và nâng cấp dễ dàng trong tương lai. Đồng thời, cách tiếp cận này cho phép hệ thống hoạt động hiệu quả trong thời gian thực – một yêu cầu quan trọng đối với các ứng dụng giám sát an toàn giao thông.

4.1.2 Thu thập và tiền xử lý dữ liệu

Trước khi tiến hành huấn luyện mô hình, dữ liệu được thu thập cần trải qua quá trình xử lý ban đầu để đảm bảo chất lượng đầu vào. Hai bộ dữ liệu chính được sử dụng bao gồm: [Face pose dataset](https://www.kaggle.com/datasets/nducloi107/face-pose-dataset)[11] và [yawn\_eye\_dataset\_new](https://www.kaggle.com/datasets/serenaraju/yawn-eye-dataset-new)[12].

***4.1.2.1 Tập dữ liệu*** [***Face pose dataset***](https://www.kaggle.com/datasets/nducloi107/face-pose-dataset)

*- Nguồn dữ liệu*: Dữ liệu được lấy từ Kaggle, do nducloi107 tải lên Kaglgle.

*- Chất lượng dữ liệu*: Dữ liệu đa dạng về góc nhìn và tư thế khuôn mặt, đảm bảo tính phong phú và đại diện.

*- Thông tin dữ liệu*: Tập dữ liệu bao gồm 977 hình ảnh khuôn mặt với nhiều tư thế đầu khác nhau thể hiện chuyển động tự nhiên của người lái xe, như: nhìn thẳng, cuối xuống, nhìn lên, góc trái, góc phải. Dữ liệu này được sử dụng để giúp mô hình xác định sự thay đổi góc đầu, một trong những dấu hiệu phổ biến của tình trạng buồn ngủ (Hình 4.1).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A close-up of a young person  AI-generated content may be incorrect.Straight | A person looking up to the sky  AI-generated content may be incorrect.Up | A person in a black shirt  AI-generated content may be incorrect.Down | A person in a suit  AI-generated content may be incorrect.Left | A person smiling with her hand on her shoulder  AI-generated content may be incorrect.Right |

Hình 4.1. Các mẫu dữ liệu của Face Pose Dataset

Quá trình tiền xử lý của bộ dự liệu này được thực hiện để chuẩn bị đầu vào cho mô hình YOLOv10. Các bước thực hiện bao gồm:

- Đánh nhãn tự động bằng Roboflow:

* Phương pháp: Sử dụng tính năng Auto Label trên Roboflow.
* Lớp nhãn: Gồm “eye”" (mắt) và “mouth” (miệng), với nhãn là tọa độ các bounding box.
* Đánh nhãn thử trên 4 ảnh để điều chỉnh Confidence Threshold cho từng lớp.
* Threshold tối ưu được áp dụng cho toàn bộ 977 ảnh.
* Kết quả: Độ chính xác thấp, hơn 50% ảnh bị nhãn sai.

*- Đánh nhãn thủ công:* Do kết quả tự động không đạt yêu cầu, toàn bộ 977 ảnh được chỉnh sửa thủ công:

* Công cụ: Bounding box tool.
* Quá trình: Vẽ lại bounding box cho mắt và miệng, cập nhật tọa độ chính xác cho từng nhãn.

- Chuẩn hóa và chia tập dữ liệu

* Chuẩn hóa: Ảnh được xoay dựa trên metadata và điều chỉnh kích thước về 640x640 để phù hợp với đầu vào của YOLOv10.
* Chia dữ liệu: Dữ liệu được chia ngẫu nhiên: 70% cho huấn luyện, 20% cho xác thực, 10% cho kiểm tra.

Bảng 4. 1 Số lượng ảnh Face Pose Dataset trong tập train, validation và tập test

|  |  |  |  |
| --- | --- | --- | --- |
| **Tập dữ liệu** | **Train** | **Validation** | **Test** |
| **Số lượng ảnh** | 684 | 195 | 98 |

***4.1.2.2 Tập dữ liệu*** [***yawn\_eye\_dataset\_new***](https://www.kaggle.com/datasets/serenaraju/yawn-eye-dataset-new)

*- Nguồn dữ liệu*: Bộ dữ liệu được tạo bởi Serena Raju và tải lên trên Kaggle.

*- Chất lượng dữ liệu*:  Dữ liệu phong phú và đa dạng, đảm bảo độ chính xác và tính đại diện cao.

*- Thông tin dữ liệu*: Bao gồm hình ảnh vùng mắt và vùng miệng của người lái là dữ liệu gồm 4 lớp: Closed, Open, no\_yawn, yawn, tương ứng với trạng thái mắt và hành vi ngáp. Dữ liệu này được sử dụng để ghi nhận trạng thái mắt (nhắm/mở) và miệng (ngáp/không ngáp), nhằm giúp mô hình nhận diện chính xác các dấu hiệu buồn ngủ như ngáp hoặc mắt nhắm liên tục (Hình 4.2).

|  |  |  |  |
| --- | --- | --- | --- |
| Close up of a person's eye  AI-generated content may be incorrect.Close | A close up of an eye  AI-generated content may be incorrect.Open | A person in a car yawning  AI-generated content may be incorrect.Yawn | A person in a car  AI-generated content may be incorrect. No Yawn |

Hình 4.2. Các mẫu dữ liệu của Yawn\_eye\_dataset\_new

Quá trình tiền xử lý của bộ dữ liệu này được thực hiện để chuẩn bị đầu vào cho mô hình YOLOv10. Các bước thực hiện bao gồm:

- Chuẩn hóa ảnh gồm các bước:

* Ảnh được điều chỉnh kích thước về 224x224 pixel.
* Chuyển đổi sang định dạng RGB 24-bit.
* Chuẩn hóa giá trị pixel về khoảng [0, 1] bằng cách chia cho 255.

*- Tăng cường dữ liệu (Data Augmentation):* Để giảm overfitting và tăng khả năng tổng quát hóa, các kỹ thuật biến đổi hình ảnh được áp dụng trên tập huấn luyện:

* Lật ngang ngẫu nhiên: Áp dụng với xác suất 50%, mô phỏng tư thế khuôn mặt đối xứng (nhìn trái/phải).
* Xoay ngẫu nhiên: Xoay ảnh trong khoảng ±15 độ, tái hiện tư thế đầu nghiêng tự nhiên.
* Điều chỉnh độ sáng và tương phản: Thay đổi ngẫu nhiên để mô phỏng các điều kiện ánh sáng khác nhau (ban ngày, ban đêm).
* Dịch chuyển ngẫu nhiên: Di chuyển khuôn mặt tối đa 10% khung hình, tái hiện sự thay đổi vị trí.

*- Chia tập dữ liệu*: Tập dữ liệu được chia ngẫu nhiên thành 2 tập: 85% cho tập huấn luyện và 15% cho tập kiểm tra.

Bảng 4. 2 Số lượng ảnh Yawn\_eye\_dataset\_new trong tập train và tập test

|  |  |  |
| --- | --- | --- |
| **Label** | **Train** | **Test** |
| Open\_Eyes | 617 | 109 |
| Closed\_Eyes | 617 | 109 |
| Yawn | 617 | 109 |
| No\_Yawn | 617 | 109 |

4.1.3. Mô hình phát hiện vùng đặc trưng bằng YOLOv10

***4.1.3.1 Fine-tunning với YOLOv10***

Trong nghiên cứu này, chúng tôi lựa chọn mô hình *YOLOv10* để tinh chỉnh lại trên tập dữ liệu chứa thông tin về mắt và miệng. Do dữ liệu huấn luyện không quá lớn, nhóm sử dụng phiên bản nhẹ nhất là *YOLOv10n*, phù hợp với yêu cầu triển khai trên các hệ thống xử lý thời gian thực với tài nguyên hạn chế. Mặc dù là bản nhẹ, YOLOv10n vẫn có khả năng học và nhận diện tốt các đặc trưng cần thiết cho bài toán [19].

Phương pháp fine-tuning được áp dụng nhằm tận dụng các trọng số đã được huấn luyện trước trên bộ dữ liệu COCO, giúp rút ngắn thời gian huấn luyện và tăng độ chính xác. Một số tầng trong mô hình được mở khóa (*unfreeze*) để có thể học thêm đặc điểm mới từ tập dữ liệu mắt và miệng của nhóm.

Kiến trúc YOLOv10n sau tinh chỉnh gồm:

- **Based Network (BN):** Đây là phần trích xuất đặc trưng, được lấy từ mô hình YOLOv10n đã được huấn luyện trước đó. Phần này bao gồm các tầng tích chập (convolutional layers) có khả năng nhận diện các đặc trưng cạnh, góc, đường nét như mép mắt, hình miệng, hay đường viền khuôn mặt [19].

- **Classification Head:** Phần đầu ra được thay đổi để phù hợp với bài toán hiện tại. Thay vì nhận diện nhiều loại đối tượng khác nhau như trong mô hình gốc, giờ đây mô hình chỉ cần nhận diện hai nhãn là mắt và miệng, với hai lớp tương ứng (eye, mouth) [19].

A diagram of a network

AI-generated content may be incorrect.

Hình 4.3. Fine tuning với YOLOv10

***4.1.3.2 Kiến trúc chi tiết và nhiệm vụ các khối trong YOLOv10n***

Kiến trúc của YOLOv10 trong việc phát hiện mắt và miệng trên khuôn mặt, bao gồm các thành phần chính: Backbone, Neck, Detect, và Output. Dựa trên *Mô hình 1* (Hình 4.4), ta có thể phân tích nhiệm vụ của từng khối như sau:

***- Backbone*:** Là phần đầu tiên trong kiến trúc, nhận ảnh đầu vào (kích thước 640x640x3) và rút trích chi tiết quan trọng từ ảnh. Các khối trong Backbone bao gồm:

* Conv (Convolutional Layer): Các khối màu xanh lam thực hiện phép tích chập để rút trích các đặc trưng như mép mắt, rìa môi, hay góc mắt.
* C2f (Cross Stage Partial with Feature Fusion): Các khối màu hồng ghép đặc trưng từ nhiều tầng khác nhau. Chẳng hạn, từ những nét cơ bản như rìa mí mắt, khối này giúp mô hình học được hình dáng đầy đủ của mắt hoặc miệng – những đặc điểm quan trọng để phân biệt giữa hai vùng này.
* SPPF (Spatial Pyramid Pooling Fast): Khối màu tím ở cuối, giúp mô hình phát hiện được mắt hoặc miệng ở nhiều kích thước khác nhau – chẳng hạn, người cách xa thì mắt nhỏ hơn, nhưng mô hình vẫn nhận ra được.

Nhiệm vụ chính của Backbone là tập trung vào việc tìm ra các đặc điểm quan trọng liên quan đến mắt và miệng, để chuyển sang phần Neck.

***- Neck***: Tiếp tục xử lý thông tin do Backbone cung cấp để làm cho đặc điểm rõ hơn và chính xác hơn để chuẩn bị cho việc phát hiện. Các khối trong Neck bao gồm:

* Upsample: Khối màu xám giúp phóng to thông tin đã xử lý để không bỏ sót những vùng nhỏ như mắt nhắm hay miệng hơi hé mở.
* Concat: Khối màu vàng giúp nối thông tin từ các tầng khác nhau để mô hình vừa thấy rõ mí mắt (tầng nông) vừa hiểu hình dạng chung của mắt (tầng sâu).
* C2f: Khối màu hồng giúp tinh chỉnh lại những đặc điểm đã được nối ở trên để làm rõ hơn vị trí và hình dáng vùng cần phát hiện.
* SCDown (Spatial and Channel Downsampling): Khối màu đỏ giảm kích thước dữ liệu để tăng tốc tính toán, nhưng vẫn giữ lại các thông tin cần thiết, như vị trí chính xác của mắt – miệng.

Nhiệm vụ của Neck là Neck giúp kết hợp thông tin từ nhiều góc độ, tăng khả năng phát hiện chính xác các vùng nhỏ như mắt và miệng.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.4. Mô hình 1 - Mô hình phát hiện vùng đặc trưng khuôn mặt YOLOv10

***- Detect (v10Detect)***:được biểu thị bằng các khối màu xanh lá cây (v10Detect), đây mà nơi bắt đầu đưa ra quyết định. Nhiệm vụ chính của Detect bao gồm:

* Dự đoán các hộp giới hạn (bounding boxes) cho các vùng mà nó nhận diện là mắt hoặc miệng.
* Với mỗi hộp, mô hình còn dự đoán xem đó là mắt hay miệng, và đưa ra Confidence score (độ tin cậy) của nó vào dự đoán.
* Detect giúp mô hình xác định chính xác vị trí và loại vùng đặc trưng (eye hoặc mouth).

***- Output***:Đầu ra cuối cùng là các bounding boxes được vẽ trên ảnh thể hiện vị trí của mắt và miệng. Mỗi hộp đi kèm với nhãn lớp tương ứng (“eye” hoặc “mouth”) và độ tin cậy, cho thấy mức độ chắc chắn của mô hình đối với dự đoán đó.

***4.1.3.3 Huấn luyện mô hình***

Quá trình huấn luyện mô hình YOLOv10 trên tập dữ liệu Face-pose-dataset [11] gồm các bước sau:

- Bước 1. Cài đặt môi trường và chuẩn bị mô hình pre-trained

Đầu tiên, thư viện Ultralytics được cài đặt để hỗ trợ huấn luyện. Mã nguồn YOLOv10 được tải từ GitHub, sau đó mô hình pre-trained yolov10n.pt được tải về từ nguồn chính thức và lưu vào thư mục làm việc.

- ***Bước 2.*** ***Chuẩn bị dữ liệu***: Một tệp cấu hình **data.yaml** được tạo để chỉ dẫn đến các tập dữ liệu:

* Ảnh dùng để huấn luyện: /kaggle/input/face-pose-dataset/train/images
* Ảnh dùng để kiểm tra trong quá trình huấn luyện: /kaggle/input/face-pose-dataset/valid/images
* Ảnh dùng để đánh giá mô hình sau khi huấn luyện xong: /kaggle/input/face-pose-dataset/test/images

Tập dữ liệu được chia làm hai nhãn là “eye” và “mouth”, số lượng ảnh và nhãn được kiểm tra để đảm bảo dữ liệu đầy đủ:

* Số ảnh huấn luyện: 684 (dựa trên tỷ lệ 70% của 1000 ảnh).
* Số ảnh xác thực: 195 (20%).
* Số ảnh kiểm tra: 98 (10%).

- Bước 3. Kiểm tra phần cứng:

Trước khi huấn luyện, hệ thống sẽ kiểm tra xem có sử dụng được GPU hay không. Nếu có GPU thì mô hình sẽ chạy nhanh hơn, còn nếu không thì vẫn có thể dùng CPU. Môi trường sử dụng PyTorch để chạy mô hình.

- ***Bước 4.*** ***Tiến hành huấn luyện***: Mô hình YOLOv10n được huấn luyện với các tham số sau:

* Mô hình pre-trained: yolov10n.pt.
* Dữ liệu: Tệp data.yaml.
* Số epoch: 100.
* Kích thước ảnh: 640x640 pixel.

Quá trình huấn luyện được thực hiện bằng lệnh:



Trong quá trình này, mô hình sử dụng hàm mất mát Cross-Entropy để liên tục so sánh kết quả dự đoán với dữ liệu thật để điều chỉnh lại các thông số bên trong, nhằm giúp mô hình dự đoán chính xác hơn sau mỗi vòng lặp. Mô hình được đánh giá trên tập xác thực bằng trọng số tốt nhất (best.pt).

4.1.4. Mô hình phân loại trạng thái bằng VGG16

***4.1.4.1 Transfer learning và Fine-tuning với VVG16***

Trong đề tài nghiên cứu này, mô hình VGG16 được lựa chọn làm nền tảng cho quá trình học chuyển giao (transfer learning) nhằm giải quyết bài toán phân loại trạng thái mắt và miệng, bao gồm các tình huống như mắt mở/nhắm và ngáp/không ngáp. Mô hình gốc đã được huấn luyện trên tập dữ liệu ImageNet giúp nhận diện được các đặc trưng phổ quát như đường viền, góc cạnh, hình khối – rất phù hợp cho việc nhận dạng mắt và miệng từ hình ảnh khuôn mặt người lái.

Để thích ứng với bài toán cụ thể, nhóm thực hiện tinh chỉnh (fine-tuning) một phần mô hình nhằm tối ưu hiệu suất phân loại theo yêu cầu. Quá trình này bao gồm hai bước chính: giữ nguyên (freeze) một số tầng để tận dụng tri thức sẵn có, và huấn luyện lại các tầng còn lại cùng với phần phân loại đầu ra phù hợp với bốn trạng thái mong muốn. Chi tiết kiến trúc VGG16 trong quá trình học chuyển giao được trình bày như sau:

- *Transfer Learning*: Mô hình VGG16 được khởi tạo với trọng số pre-trained trên ImageNet. Các khối tích chập ban đầu (nửa đầu của Base Network) nơi chứa các lớp học những đặc điểm cơ bản như đường nét mắt, miệng sẽ được giữ nguyên, không thay đổi. Trong khi nửa sau của Base Network được mở ra để huấn luyện tiếp.

- *Fine-tuning*: Nửa sau của Base Network được mở khóa và toàn bộ FC Layers mới được thay đổi để phù hợp trên tập dữ liệu trạng thái mắt và miệng. Quá trình này cho phép cập nhật trọng số giúp học thêm về đặc điểm của mắt và miệng trong bài toán cụ thể, trong khi vẫn giữ lại những kiến thức chung mà mô hình đã học được từ trước [9], [18].

A diagram of a computer program

AI-generated content may be incorrect.

Hình 4.5. Quá trình học chuyển giao

***4.1.4.2 Kiến trúc và nhiệm vụ cụ thể của từng khối trong VGG16***

Kiến trúc của VGG16 sau khi áp dụng fine-tuning như sau: Các tầng từ 0-27 được đóng băng (freeze) để tận dụng các đặc điểm đã học từ trước. Các tầng từ 28 trở đi được mở khóa để sẵn sàng tinh chỉnh. Tầng FC được thay đổi để phù hợp với bài toán phân loại 4 lớp: Open Eyes, Close Eyes, Yawn, No Yawn (*Mô hình 2* – Hình 4.6).

*a. Convolutional Layers - Các tầng tích chập và pooling*

Đây là phần chính giúp mô hình nhìn và hiểu được các đặc điểm trong ảnh, như hình dáng mắt, miệng, góc cạnh. Các lớp này được chia làm 2 nhóm: Frozen (giữ nguyên) và Unfrozen (mở khóa để huấn luyện).

- Khối Frozen

* Là các khối màu tím và hồng gồm 13 lớp tích chập (Conv) từ Conv1\_1 đến Conv5\_3 và 5 lớp Pooling từ Pooling1 đến Pooling5.
* Mỗi khối có 2–3 lớp dùng bộ lọc 3x3 để quét qua ảnh, sau đó là một lớp pooling để giảm kích thước, số lượng bộ lọc tăng dần từ 64 đến 512.
* Nhiệm vụ:
  + Các lớp đầu tiên như Conv1, Conv2 giúp mô hình nhận ra các nét cơ bản như cạnh, góc mí mắt, viền môi,...
  + Các lớp giữa như Conv3, Conv4 học được các mẫu phức tạp hơn như hình dáng một con mắt đang khép lại, hoặc khuôn miệng khi ngáp.
  + Vì những đặc trưng này khá chung và có thể áp dụng cho nhiều loại ảnh, nên nhóm giữ nguyên (không huấn luyện lại) để tiết kiệm thời gian và tận dụng kiến thức có sẵn từ ImageNet.

- Khối Unfrozen

* Là các khối xanh lá gồm các lớp tích chập cuối: Conv5\_1, Conv5\_2, Conv5\_3 và Pooling5.
* Nhiệm vụ:
  + Đây là phần giúp mô hình tập trung học các đặc điểm rõ hơn của mắt và miệng trong các trạng thái cụ thể. Ví dụ như phân biệt mắt mở to, nhắm hẳn hay khép nhẹ, hoặc nhận ra miệng đang há ra khi ngáp.
  + Nhóm mở khóa và huấn luyện lại phần này, vì nó cần điều chỉnh để phù hợp với tập dữ liệu trạng thái mắt – miệng.

- Pooling Layers

* Là các khối màu hồng Xen kẽ giữa các khối tích chập, dùng max-pooling 2x2.
* Nhiệm vụ: Giúp giảm kích thước dữ liệu, làm mô hình nhanh hơn và gọn hơn. Đồng thời giúp mô hình nhận biết đặc điểm ở phạm vi lớn hơn, ví dụ như hình dáng toàn bộ vùng mắt hay miệng thay vì từng chi tiết nhỏ.

*b. Fully Connected Layers – Phân loại ảnh*

Là tầng FC cuối cùng, đã được thay bằng lớp mới phù hợp với bài toán 4 trạng thái. Nhận đầu vào là đặc trưng trích xuất từ ảnh, rồi làm phẳng thành vector. Sau đó tính toán xác suất để xác định xem ảnh thuộc lớp nào trong 4 lớp: Open Eyes, Close Eyes, Yawn, No Yawn.

A diagram of a model of a frozen model

AI-generated content may be incorrect.

Hình 4.6. Mô hình 2 – Mô hình phân loại trạng thái VGG16

***4.1.4.3 Huấn luyện mô hình***

- Bước 1. Chuẩn bị dữ liệu

Tập dữ liệu được tải từ thư mục /kaggle/input/yawn-eye-dataset-new/dataset\_new, gồm hai phần: một phần để huấn luyện và một phần để kiểm tra. Các bước xử lí dữ liệu trước khi đưa vào mô hình bao gồm:

* Biến đổi dữ liệu:
  + Tập huấn luyện: Điều chỉnh kích thước ảnh về 224x224, áp dụng tăng cường dữ liệu (lật ngang, xoay ±15 độ, thay đổi độ sáng/tương phản, dịch chuyển ngẫu nhiên), sau đó chuẩn hóa với giá trị trung bình [0.485, 0.456, 0.406] và độ lệch chuẩn [0.229, 0.224, 0.225].
  + Tập kiểm tra: Chỉ điều chỉnh kích thước và chuẩn hóa.
* Tải dữ liệu: Dùng ImageFolder để đọc ảnh, sau đó đưa vào DataLoader với batch size là 16. Dữ liệu được xáo trộn và xử lý song song bằng 4 luồng.
* Giải quyết mất cân bằng dữ liệu: lớp “yawn” ít dữ liệu hơn nên được gán trọng số cao hơn (1.5), còn các lớp khác là 1.0.
* Tinh chỉnh

- ***Bước 2. Chuẩn bị mô hình pre-trained***: Dùng mô hình VGG16 huấn luyện trước trên tập ImageNet.

* Tinh chỉnh mô hình:
  + Các tầng từ 0 đến 27 (phần lớn các tầng tích chập) được giữ cố định (frozen) để bảo toàn đặc trưng đã học.Các tầng từ 28 trở đi được mở khóa để tinh chỉnh.
  + Tầng fully connected cuối được thay thế bằng một cấu trúc mới: một tầng tuyến tính (Linear) từ 4096 xuống 256, ReLU, Dropout (0.5), và một tầng tuyến tính cuối cùng từ 256 xuống 4 (4 lớp đầu ra).
* Chuyển sang GPU: mô hình sẽ được chuyển sang GPU để huấn luyện nhanh hơn.

- Bước 3. Thiết lập tham số huấn luyện

* Hàm mất mát: Dùng CrossEntropyLoss có cân nhắc trọng số giữa các lớp để giảm ảnh hưởng của việc dữ liệu không đều.
* Tối ưu hóa: Sử dụng thuật toán Adam với learning rate là 0.0001, chỉ cập nhật các tầng được mở khóa.

*-* ***Bước 4. Tiến hành huấn luyện:*** Mô hình được huấn luyện trong 25 epoch với cơ chế dừng sớm nếu không tiến bộ sau 5 vòng liên tiếp.(patience=5):

* Giai đoạn huấn luyện (train): Mô hình được đặt ở chế độ huấn luyện, tính toán đầu ra, cập nhật trọng số bằng lan truyền ngược (backpropagation) dựa trên hàm mất mát.
* Giai đoạn kiểm tra (test): Chuyển sang chế độ kiểm tra, đánh giá độ chính xác và lỗi trên dữ liệu kiểm tra. Nếu kết quả tốt hơn trước, mô hình đó sẽ được lưu lại ở địa chỉ /kaggle/working/best\_model.pth.
* Dừng sớm: Nếu sau 5 lần liên tiếp kết quả không cải thiện thì dừng huấn luyện để tiết kiệm thời gian.

4.1.5 Kết quả huấn luyện và đánh giá mô hình

***4.1.5.1 Phát hiện vùng đặc trưng bằng YOLOv10***

Hiệu suất của mô hình YOLOv10 được đánh giá thông qua mối quan hệ giữa độ chính xác (precision), độ bao phủ (recall), điểm F1 và ngưỡng độ tin cậy (confidence) cho từng lớp đối tượng. Hình 4.7 minh họa hai biểu đồ: F1-Confidence Curve (trái) và Precision-Confidence Curve (phải).

Ở biểu đồ bên trái, đường cong F1 cho thấy rằng lớp “mouth” có hiệu suất tốt hơn so với lớp “eye”, đặc biệt ở các mức confidence thấp đến trung bình. Lớp “eye” có F1 giảm rõ rệt khi confidence tăng, cho thấy mô hình gặp khó khăn hơn khi nhận diện mắt trong điều kiện nghiêm ngặt. Đường tổng hợp “all classes” (đường màu xanh đậm) đạt F1 cao nhất là 0.84 tại ngưỡng confidence 0.309.

Trong biểu đồ bên phải, độ chính xác (precision) tăng dần khi confidence tăng. Lớp “mouth” đạt precision rất cao và ổn định, gần như tiệm cận 1.0 ở confidence cao. Giá trị precision cao nhất cho toàn bộ các lớp là 1.00 tại confidence 0.923, cho thấy mô hình có khả năng nhận diện chính xác trong các tình huống chắc chắn.

|  |  |
| --- | --- |
| **A graph of a curve  AI-generated content may be incorrect.** | **A graph of a graph  AI-generated content may be incorrect.** |

Hình 4.7. Biểu đồ F1-Confidence Curve và Precision-Confidenc Curve

A blue squares with white text

AI-generated content may be incorrect.

Hình 4. 8. Ma trận nhầm lẫn

Ma trận nhầm lẫn thể hiện ở Hình 4.8 cho thấy mô hình phân loại khá tốt các ảnh thuộc lớp “eye” và “mouth”, nhưng vẫn còn một số trường hợp nhầm lẫn giữa các lớp và nền (background). Đặc biệt, 129 ảnh thuộc lớp “eye” bị nhận nhầm là background và 83 ảnh bị nhận nhầm theo chiều ngược lại. Điều này cho thấy mô hình có xu hướng khó phân biệt các vùng mắt với nền trong một số tình huống ánh sáng hoặc góc chụp phức tạp.

YOLOv10n – một biến thể rút gọn của kiến trúc YOLOv10 – được huấn luyện trên bộ dữ liệu Face Pose Dataset trong 100 epoch. Kết quả huấn luyện cho thấy mô hình đạt hiệu suất nhận diện tốt trên cả hai lớp đối tượng “eye” và “mouth”, với độ chính xác (Precision), độ bao phủ (Recall), và độ chính xác trung bình (mAP) ở các mức đánh giá khác nhau được trình bày chi tiết trong Bảng 4.3.

Bảng 4.3. Các chỉ số đánh giá hiệu suất mô hình YOLOv10n trên từng lớp đối tượng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **mAP50** | **mAP50-95** |
| Eye | **0.803** | **0.732** | **0.799** | **0.354** |
| Mouth | **0.901** | **0.925** | **0.952** | **0.661** |
| All | **0.852** | **0.828** | **0.876** | **0.508** |

Bên cạnh độ chính xác, tốc độ xử lý cũng là một yếu tố quan trọng trong các ứng dụng thời gian thực. Theo kết quả đo đạc, mô hình YOLOv10n có thời gian xử lý trung bình cho mỗi ảnh là 1.7ms ở bước tiền xử lý, 6.7ms trong giai đoạn suy luận, và 0.6ms cho bước hậu xử lý. Tổng thời gian xử lý khoảng 9ms mỗi ảnh, cho thấy khả năng đáp ứng tốt yêu cầu thời gian thực trong bài toán phát hiện trạng thái buồn ngủ khi lái xe.

***4.1.5.2 Phân loại trạng thái bằng VGG16***

Để đánh giá hiệu suất mô hình VGG16 trong việc phân loại các trạng thái mắt và miệng (bao gồm: mở mắt, nhắm mắt, ngáp và không ngáp), mô hình được huấn luyện và kiểm thử trên hai tập dữ liệu chuyên biệt. Việc đánh giá hiệu suất được thực hiện bằng cách sử dụng các chỉ số: độ chính xác tổng thể (accuracy), độ chính xác theo lớp (precision), độ nhạy (recall), và F1-score. Ngoài ra, các biểu đồ huấn luyện và ma trận nhầm lẫn cũng được sử dụng nhằm minh họa trực quan khả năng học của mô hình qua từng epoch cũng như phân tích các lỗi phân loại.

Hình 4.9 minh họa sự thay đổi của hàm mất mát (Loss) và độ chính xác (Accuracy) trên tập huấn luyện và kiểm thử theo từng epoch. Có thể nhận thấy rằng, mô hình đạt được độ chính xác cao ngay từ các epoch đầu tiên và ổn định dần ở các epoch sau. Đồng thời, giá trị loss giảm đều cho thấy quá trình học của mô hình diễn ra hiệu quả.

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

AI-generated content may be incorrect.

Hình 4.9. Biểu đồ Loss over Epochs và Accuracy over Epoch

A diagram of a confusion matrix

AI-generated content may be incorrect.

Hình 4. 10. Ma trận nhầm lẫn của mô hình 2

Hình 4.10 thể hiện ma trận nhầm lẫn trên tập kiểm thử, qua đó cho thấy mô hình VGG16 có khả năng phân loại rất tốt các lớp trạng thái. Tuy nhiên, vẫn tồn tại một số nhầm lẫn nhỏ giữa các cặp lớp có đặc điểm gần giống nhau, cụ thể là giữa “Closed” và “Open”, cũng như giữa “yawn” và “no\_yawn”.

Bảng 4.4 trình bày chi tiết các chỉ số đánh giá hiệu suất của mô hình VGG16 trên từng lớp trạng thái. Các chỉ số Precision, Recall và F1-score đều đạt giá trị cao, đặc biệt là lớp “Open\_Eyes” và “No\_Yawn” với Recall đạt 1.0. Độ chính xác tổng thể của mô hình đạt 96.77%, cho thấy mô hình có thể áp dụng hiệu quả trong các hệ thống giám sát hành vi lái xe theo thời gian thực.

Bảng 4. 4. Các chỉ số đánh giá hiệu suất mô hình VGG16 trên từng lớp trạng thái

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F1-score** |
| Open\_Eyes | **0.9397** | **0.9689** | **0.9689** |
| Closed\_Eyes | **1.0** | **0.9358** | **0.9668** |
| No\_Yawn | **0.9397** | **1.0** | **0.9689** |
| Yawn | **1.0** | **0.934** | **0.9659** |

Độ chính xác tổng thể (Overall Accuracy): 96.77%. Kết quả này cho thấy VGG16 là một mô hình mạnh, có khả năng phân biệt rõ ràng các trạng thái hành vi liên quan đến tình trạng buồn ngủ khi lái xe, từ đó hỗ trợ hiệu quả trong các hệ thống cảnh báo.

## 4.2 Phát triển và triển khai hệ thống giám sát buồn ngủ

4.2.1 Tổng quan hệ thống

Hệ thống được thiết kế nhằm hỗ trợ phát hiện tình trạng buồn ngủ của người lái xe thông qua việc phân tích hình ảnh khuôn mặt thu nhận từ camera giám sát. Quá trình xử lý bao gồm các bước chính: thu thập dữ liệu hình ảnh, tiền xử lý, trích xuất đặc trưng và phân loại trạng thái. Trong đó, mô hình YOLOv10 được triển khai để xác định chính xác các vùng mắt và miệng trên khuôn mặt tài xế. Sau khi xác định được các vùng quan tâm (Region of Interest – ROI), ảnh sẽ được cắt và chuẩn hóa trước khi đưa vào mô hình VGG16 nhằm phân loại trạng thái mắt (mở hoặc nhắm) và hành vi ngáp – hai biểu hiện thường gặp của trạng thái buồn ngủ.

Khác với các phương pháp chỉ đánh giá tại một thời điểm, hệ thống còn thực hiện giám sát liên tục trạng thái mắt và hành vi ngáp theo thời gian. Nếu phát hiện tình trạng mắt nhắm liên tục hoặc hành vi ngáp kéo dài vượt quá 2 giây, hệ thống sẽ kích hoạt cảnh báo nhằm đảm bảo an toàn cho người điều khiển phương tiện (xem Hình 4.11).

A diagram of a diagram

AI-generated content may be incorrect.

Hình 4.11. Hệ thống tổng quan

4.2.2 Kiến trúc hệ thống

***4.2.2.1 Kiến trúc tổng thể***

Hệ thống được xây dựng theo mô hình phân lớp gồm các thành phần chính sau (Hình 4.12):

**-** *Giao diện người dùng (View):* Thành phần giao diện (DrowsinessGUI) hiển thị video thời gian thực, trạng thái buồn ngủ và các thông báo cảnh báo đến người dùng.

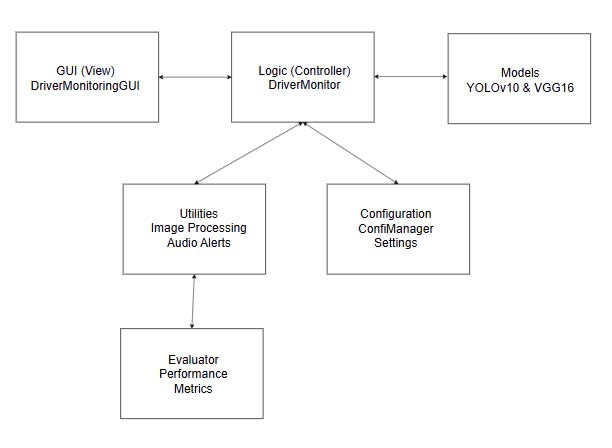
**-** *Điều khiển logic (Controller):* Lớp điều khiển (DrowsinessMonitor) xử lý khung hình, quản lý luồng dữ liệu, điều phối các mô-đun và kích hoạt cảnh báo.

**-** *Mô hình AI (Models):* Bao gồm mô hình YOLOv10 và VGG16, lần lượt thực hiện nhiệm vụ phát hiện vùng đặc trưng (mắt, miệng) và phân loại trạng thái buồn ngủ.

**-** *Tiện ích hỗ trợ (Utilities):* Gồm các công cụ phục vụ tiền xử lý ảnh, phát âm thanh cảnh báo, và quản lý tệp tin.

**-** *Cấu hình hệ thống (Configuration):* Quản lý các tham số như ngưỡng thời gian nhắm mắt, hành vi ngáp và thiết lập camera.

**-** *Đánh giá hiệu suất (Evaluator):* Đo lường hiệu quả của hệ thống thông qua các chỉ số định lượng.



Hình 4.12. Sơ đồ khối kiến trúc hệ thống

*Hình Kiến trúc hệ thống bản mẫu-* Sơ đồ khối kiến trúc hệ thống

***4.2.2.2 Các thành phần trong hệ thống***

Dưới đây là phần trình bày chi tiết về các thành phần đã nêu phía trên, giúp làm rõ vai trò và cách chúng hoạt động trong hệ thống.

***a. Giao diện người dùng (View)***

DrowsinessGUI: là phần giao diện chính mà người dùng tương tác. Giao diện này có các chức năng:

- Hiển thị video thời gian thữ từ camera .

- Cung cấp giao diện điều khiển chính cần thiết cho việc khởi động và dừng hệ thống.

- Giao diện được xây dựng bằng Tkinter – thư viện GUI cho Python.

***b. Điều khiển logic (Controller)***

DrowsinessMonitor: là thành phần trung tâm xử lý mọi hoạt động của hệ thống. Lớp này đảm nhiệm các nhiệm vụ sau:

- Quản lý luồng dữ liệu chính, bao gồm:

* Thu và xử lí khung hình video.
* Phát hiện đặc trưng khuôn mặt (vùng mắt, miệng).
* Theo dõi trạng thái mắt nhắm kéo dài và hành vi ngáp.
* So sánh với các ngưỡng cấu hình để đưa ra cảnh báo.
* Đánh giá hiệu suất hoạt động của hệ thống.

- Bên trong, lớp này bao gồm các phần chính::

* Xử lí video (Video processing pipeline): Chuỗi các bước xử lý từng khung hình
* Quản lí trạng thái (State Management): Theo dõi trạng thái của mắt và miệng theo thời gian.
* Hệ thống cảnh báo: Phát cảnh báo khi phát hiện dấu hiệu buồn ngủ, có thêm thời gian chờ (cooldown) để tránh báo liên tục.
* Logging system (ghi log hệ thống): Ghi lại thông tin hoạt động của hệ thống để theo dõi và đánh giá sau này.

***c. Mô hình AI (YOLOv10 & VGG16)***

- YOLOv10n: Phát hiện và định vị vùng mắt và miệng trên khuôn mặt tài xế. Phiên bản YOLOv10 nano (YOLOv10n) được sử dụng, với trọng số đã huấn luyện trên Face Pose Dataset (Chương 4, Mục 4.1.3).

- VGG16: Dùng để phân loại các ảnh ROI thu được từ YOLOv10 thành các trạng thái: mắt mở, mắt nhắm, ngáp và không ngáp. Mô hình được tinh chỉnh trên tập Yawn Eye Dataset (Chương 4, Mục 4.1.3).

***d. Tiện ích hỗ trợ (Utilities)***

Phần tiện ích bao gồm các hàm và công cụ nhỏ hỗ trợ cho hoạt động chung của hệ thống. Cụ thể:

- Tiền xử lý hình ảnh: Thay đổi kích thước ảnh và chuẩn hóa khung hình để phù hợp với đầu vào của YOLOv10 và VGG16.

- Phát âm thanh cảnh báo: Phát chuông báo khi phát hiện trạng thái buồn ngủ.

- Quản lý hệ thống thư mục và file: Đảm bảo các tệp tin, thư mục cần thiết được tạo và sử dụng đúng cách.

- Logging cấu hình hệ thống: Lưu lại thông tin cấu hình để dễ theo dõi và quản lý quá trình chạy hệ thống.

***e. Cấu hình hệ thống (ConfigManager)***

ConfigManager: là nơi tập trung quản lý tất cả các tham số và cài đặt liên quan đến hệ thống, bao gồm:

- Vị trí lưu và tải mô hình đã huấn luyện.

- Ngưỡng phát hiện trạng thái buồn ngủ (ví dụ: thời gian nhắm mắt tối thiểu để cảnh báo).

- Tùy chỉnh âm thanh cảnh báo (bật/tắt, thời lượng...).

- Thiết lập thiết bị camera.

- Các tham số hiệu suất hoạt động.

***f. Đánh giá hiệu suất (SystemPerformanceEvaluator)***

SystemPerformanceEvaluator: Là thành phần đảm nhận việc kiểm tra và đánh giá mức độ hiệu quả của hệ thống. Bao gồm:

- Đánh giá hiệu suất của hệ thống dựa trên các chỉ số như mAP, độ chính xác, và tỷ lệ phát hiện đúng.

- So sánh kết quả dự đoán với dữ liệu thực tế (ground truth).

- Ghi log kết quả đánh giá để dễ dàng theo dõi và cải tiến hệ thống sau này.

***4.2.2.3 Luồng dữ liệu***

Hệ thống xử lý dữ liệu theo quy trình tuần tự, được tối ưu để đảm bảo thời gian xử lý mỗi khung hình dưới 1 giây, đáp ứng yêu cầu thời gian thực:

*- Giao diện tiếp nhận thao tác người dùng*: Người dùng tương tác trực tiếp với giao diện chính (DrowsinessGUI) thông qua các nút điều khiển như Bắt đầu / Dừng giám sát, tùy chỉnh ngưỡng cảnh báo hoặc xem nhật ký cảnh báo.

*- Bộ điều khiển logic xử lý yêu cầu*: Khi người dùng thực hiện thao tác, lớp điều khiển logic (DrowsinessMonitor) tiếp nhận yêu cầu, kích hoạt hoặc điều phối các thành phần liên quan như camera, mô hình AI và hệ thống cảnh báo.

*- Mô hình AI thực hiện phát hiện và phân loại*: Mô hình YOLOv10n phát hiện và khoanh vùng mắt, miệng trên từng khung hình video sau đó được chuyển sang mô hình VGG16 để phân loại trạng thái: mắt mở/nhắm, miệng ngáp/không ngáp.

*- Kết quả được truyền ngược về bộ điều khiển*: Các kết quả phân loại được trả lại cho DrowsinessMonitor, nơi chúng được phân tích theo thời gian thực để xác định dấu hiệu buồn ngủ dựa trên ngưỡng cấu hình (ví dụ: mắt nhắm trên 2 giây).

*- Giao diện cập nhật kết quả*: Giao diện người dùng hiển thị trạng thái mới của mắt, miệng, FPS và cảnh báo nếu cần. Người dùng có thể theo dõi trực quan toàn bộ quá trình nhận diện.

*- Cảnh báo và nhật ký được lưu lại nếu cần*: Nếu hệ thống phát hiện trạng thái buồn ngủ, chuông báo sẽ được phát, đồng thời lưu lại thông tin vào nhật ký để phục vụ việc đánh giá hiệu suất hệ thống sau này.

4.2.3 Tích mô hình vào hệ thống thời gian thực

Hệ thống được thiết kế theo luồng xử lý tuần tự, kết hợp giữa thu nhận khung hình, phát hiện đối tượng bằng YOLOv10 và phân loại trạng thái bằng VGG16, đảm bảo hoạt động thời gian thực và tối ưu hiệu năng. Quá trình xử lý của hệ thống diễn ra theo các bước sau:

***Khởi tạo mô hình***

- Khung hình được lấy từ camera hoặc video thông qua OpenCV.

- Thực hiện lật ảnh theo chiều ngang (mirroring) và điều chỉnh kích thước về chuẩn 640×640 pixel.

- Theo dõi và tính toán tốc độ khung hình (FPS) để đảm bảo hiệu suất ổn định.

***Phát hiện đặc điểm khuôn mặt với YOLOv10***

- Đầu vào là khung hình gốc từ camera.

- Mô hình YOLOv10 nhận diện các vùng mắt và miệng, trả về danh sách các bounding box kèm điểm tin cậy.

- Chọn 2 vùng có xác suất cao nhất cho mắt và 1 vùng cho miệng.

- Áp dụng ngưỡng confidence để loại bỏ phát hiện không chính xác.

***Phân loại và phân tích trạng thái***

- Phân loại trạng thái mắt (nhắm/mở):

* Trích xuất vùng ROI từ bounding box mắt.
* Tiền xử lý ảnh trước khi đưa vào VGG16.
* Phân loại nhị phân: mắt đang mở hoặc đang nhắm.
* Theo dõi liên tục thời gian mắt nhắm để xác định nguy cơ buồn ngủ.

- Phân loại trạng thái miệng (ngáp/không ngáp):

* ROI miệng được xử lý tương tự.
* Phát hiện ngáp: Nhận diện sơ bộ bằng VGG16 và xác nhận dựa trên tỉ lệ chiều cao/rộng của miệng.
* Quản lý thời gian ngáp để đánh giá độ nghiêm trọng.

**Quản lí trạng thái**

- Theo dõi trạng thái mắt:

* Giám sát liên tục thời gian mắt nhắm.
* Cho phép khoảng trễ ngắn trước khi reset trạng thái.
* Kích hoạt cảnh báo khi vượt ngưỡng cấu hình (ví dụ: 2 giây).

- Theo dõi trạng thái ngáp:

* Theo dõi thời lượng ngáp theo thời gian thực.
* Áp dụng xác nhận tỉ lệ hình dạng để tăng độ chính xác.
* Kiểm soát khoảng trễ và tái khởi động trạng thái nếu cần.

***Kích hoạt cảnh báo***

- Điều kiện kích hoạt

* Mắt nhắm liên tục quá thời gian cho phép.
* Phát hiện hành vi ngáp xác thực.
* Hệ thống quản lý thời gian nghỉ giữa các cảnh báo (cooldown).

- Xử lí cảnh báo

* Hiển thị cảnh báo trực quan trên giao diện video.
* Phát âm thanh cảnh báo (thread-safe)..
* Ghi nhật kí lịch sử cảnh báo để phục vụ đánh giá sau.

***Hiệu suất thời gian thực:***

- Tối ưu xử lí

* Xử lý theo lô (batch) nếu cần.
* Hỗ trợ tăng tốc bằng GPU nếu có sẵn.
* Bỏ qua khung hình nếu cần để đảm bảo thời gian xử lý.

- Quản lí trạng thái:

* Đảm bảo hệ thống không bị gián đoạn khi mất tạm thời tín hiệu phát hiện.
* Bảo lưu trạng thái trong khoảng thời gian ngắn nếu mất nhận diện.

- Tối ưu bộ nhớ:

* Quản lý buffer ảnh hợp lý.
* Giải phóng tài nguyên không cần thiết.
* Tối ưu sử dụng tensor để giảm tải bộ nhớ.

4.2.4 Giao diện người dùng và tính năng

Hệ thống được triển khai dưới dạng một bản mẫu hoàn chỉnh, đi kèm giao diện người dùng thân thiện và dễ sử dụng. Mục tiêu chính của giao diện là cung cấp thông tin trực quan và đưa ra cảnh báo kịp thời về tình trạng buồn ngủ của người lái xe.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4. 13. Giao diện chính hệ thống cảnh báo

Thiết kế giao diện bao gồm hai khu vực hiển thị chính, được bố trí hợp lý nhằm tối ưu hóa trải nghiệm tương tác của người dùng (Hình 4.13):

**- Khu vực bên trái:** Trình chiếu video thu từ camera theo thời gian thực, đồng thời hiển thị các vùng mắt và miệng được phát hiện, đánh dấu bằng các hộp giới hạn (bounding box).

**- Khu vực bên phải:** Hiển thị các thông tin nhận dạng trạng thái (mắt mở/nhắm, ngáp/không ngáp) và cung cấp các nút điều khiển chức năng như khởi động, dừng hệ thống hoặc điều chỉnh tham số.

Tiếp theo, chúng ta sẽ đi sâu vào từng thành phần chính của giao diện người dùng, đảm bảo việc giám sát trạng thái buồn ngủ diễn ra một cách trực quan, dễ hiểu và hiệu quả.

***4.2.4.1 Các thành phần giao diện chính***

- ***Khu vực hiển thị video***

* Trình chiếu luồng video thu từ camera hoặc file, kèm theo các bounding box đánh dấu vùng mắt và miệng.
* Tự động đổi màu khung nền từ đen sang đỏ khi hệ thống phát hiện dấu hiệu buồn ngủ.
* Chiếm diện tích lớn trên màn hình để người dùng dễ dàng quan sát mọi biến động.

- ***Các chỉ báo trạng thái***

* FPS (Frames Per Second): Hiển thị tốc độ xử lý khung hình, giúp đánh giá độ mượt của hệ thống.
* Thanh tiến trình thời gian nhắm mắt: Thể hiện thời gian mắt nhắm liên tục so với ngưỡng cấu hình (2.1 giây).
* Thanh tiến trình thời gian ngáp: Hiển thị độ dài hành vi ngáp so với ngưỡng cấu hình (3.0 giây).
* Trạng thái hệ thống: Cho biết chế độ vận hành hiện tại (Idle / Monitoring / Evaluating).
* Biểu tượng trạng thái mắt và miệng: Minh họa trực quan tình trạng mở/nhắm mắt và ngáp/không ngáp thông qua các icon.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.14. Chỉ báo trạng thái thời gian thực

- ***Các nút điều khiển***

* Start / Pause / Resume: Bắt đầu, tạm dừng hoặc tiếp tục quá trình giám sát.
* Evaluate Performance: Khởi chạy chức năng đánh giá hiệu suất của hệ thống trên tập dữ liệu kiểm thử.
* Settings: Mở cửa sổ thiết lập tham số (ngưỡng thời gian, thiết bị camera, âm thanh cảnh báo…).
* View Alerts: Hiển thị lịch sử các lần cảnh báo buồn ngủ đã được ghi nhận.

|  |  |  |
| --- | --- | --- |
| A screenshot of a computer  AI-generated content may be incorrect. | A screenshot of a computer  AI-generated content may be incorrect. | A screenshot of a computer screen  AI-generated content may be incorrect. |

Hình 4. 15. Các nút điều khiển chính của hệ thống

Sau khi đã làm rõ các thành phần giao diện cơ bản, chúng ta tiếp tục khám phá những tính năng đặc biệt được triển khai thông qua các cửa sổ chức năng bổ sung, giúp người dùng linh hoạt tùy chỉnh và giám sát hệ thống.

***4.2.4.2 Các tính năng đặc biệt của giao diện***

Để nâng cao khả năng điều khiển và tùy biến, giao diện cung cấp các chức năng sau.

*-* ***Chọn nguồn dữ liệu* :** Khi bắt đầu giám sát, hệ thống cho phép người dùng lựa chọn nguồn dữ liệu:

* Camera (Real-time): Giám sát theo thời gian thực từ webcam
* Video: Phân tích từ file video có sẵn

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.16. Cửa sổ chọn nguồn dữ liệu

***- Quản lý cấu hình* :** Giao diện cài đặt cho phép người dùng tùy chỉnh:

* Camera/Video Settings: ID camera (mặc định: 0), đường dẫn video
* Alert Settings: Ngưỡng thời gian nhắm mắt (2.1 giây), ngưỡng ngáp (3.0 giây)
* Audio Settings: Bật/tắt âm thanh cảnh báo, điều chỉnh âm lượng

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.17. Cửa sổ cài đặt

***- Quản lý lịch sử cảnh báo*:** Giao diện hiển thị lịch sử cảnh báo với:

* Bảng lịch sử cảnh báo phân loại theo màu sắc
* Chức năng xuất dữ liệu sang CSV
* Chức năng xóa lịch sử cảnh báo

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.18. Cửa sổ lịch sử cảnh báo

***- Cơ chế cảnh báo:*** Hệ thống triển khai cơ chế cảnh báo đa phương tiện khi phát hiện ngủ gật:

* *Cảnh báo trực quan:* Đổi màu nền video từ đen sang đỏ và hiển thị thông báo "Eyes closed too long”/ “Yawn detected” bằng chữ đỏ (Hình 4.19).
* *Cảnh báo âm thanh:* Phát âm thanh cảnh báo qua loa khi phát hiện trạng thái nguy hiểm và âm lượng có thể tùy chỉnh qua giao diện cài đặt (Hình 4.20).

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.19. Hệ thống cảnh báo khi nhắm mắt quá lâu

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.20. Hệ thống cảnh báo ngáp

***- Xử lý đa luồng:***Bản mẫu sử dụng threading để đảm bảo giao diện phản hồi nhanh:

* Xử lý video và phát hiện trạng thái chạy trong luồng riêng biệt
* Cảnh báo âm thanh được phát trong luồng riêng
* Cập nhật giao diện thực hiện qua gọi *root.after()* để tránh xung đột

***- Đánh giá hiệu năng*:** Giao diện cung cấp chức năng đánh giá hiệu năng hệ thống:

* Cho phép chọn video test và nhập dữ liệu ground truth
* Hiển thị tiến trình đánh giá và thời gian còn lại
* Trả về kết quả chi tiết sau khi hoàn thành

4.2.5 Kiểm thử và đánh giá hệ thống

Để xác định độ tin cậy và khả năng đáp ứng thời gian thực của bản mẫu, hệ thống được kiểm thử trong môi trường thực tế với cấu hình phần cứng và nguồn dữ liệu tương tự khi triển khai thực tế. Các phép đo chính bao gồm thời gian xử lý trung bình trên mỗi khung hình, độ trễ từ khi phát hiện đến khi cảnh báo, và mức độ linh hoạt trong việc điều chỉnh tham số.

|  |  |
| --- | --- |
| **Chỉ số** | **Giá trị** |
| Thời gian xử lý trung bình/khung hình | 0,072 giây (tương đương 13–14 FPS) |
| Thành phần bao gồm | - Phát hiện đối tượng (YOLOv10) - Phân loại trạng thái (VGG16) - Cập nhật trạng thái, tính toán logic và phát âm thanh cảnh báo |
| Độ trễ cảnh báo | Dưới 1 giây kể từ thời điểm mô hình xác nhận dấu hiệu buồn ngủ |
| Khả năng tùy chỉnh | Cho phép điều chỉnh các ngưỡng thời gian (nhắm mắt, ngáp), nguồn dữ liệu và thông số âm thanh cảnh báo |

Bảng 4. 5. Kết quả kiểm thử

**-** *Hiệu năng xử lý thời gian thực:* Hệ thống đạt được tốc độ khoảng 13–14 FPS, đảm bảo khả năng theo dõi liên tục và mượt mà. Thời gian 0,072 giây mỗi khung hình đã bao gồm toàn bộ luồng xử lý từ phát hiện vùng quan tâm, phân loại trạng thái cho đến việc cập nhật giao diện và phát cảnh báo âm thanh.

- *Độ trễ cảnh báo:*Khi mô hình phát hiện trạng thái buồn ngủ (mắt nhắm quá lâu hoặc ngáp), hệ thống kích hoạt cảnh báo trực quan và âm thanh trong vòng chưa đầy 1 giây, đáp ứng nhanh yêu cầu an toàn.

- *Khả năng tùy biến:*Người dùng có thể linh hoạt điều chỉnh các thông số như ngưỡng thời gian nhắm mắt, ngáp, nguồn dữ liệu (camera hoặc file video) và mức âm lượng cảnh báo, giúp hệ thống dễ dàng thích nghi với các môi trường và kịch bản khác nhau.

Những kết quả này cho thấy bản mẫu không chỉ đáp ứng được tiêu chí thời gian thực mà còn có độ tin cậy cao và dễ dàng tùy chỉnh để phù hợp với nhiều tình huống giám sát.

# CHƯƠNG 5. KẾT QUẢ NGHIÊN CỨU

## 5.1 Kiểm thử hệ thống

Hệ thống phát hiện trạng thái buồn ngủ khi nhắm mắt (close) và ngáp (yawn) đã được thử nghiệm trên 10 video, mỗi video dài 30 giây, với tổng thời gian khoảng 300 giây. Dữ liệu bao gồm thời gian xảy ra sự kiện (time), thời gian nên đưa ra cảnh báo (should alert), và cảnh báo thực tế của hệ thống (system alert). Bảng 1 dưới đây trình bày chi tiết dữ liệu thử nghiệm (chỉ lấy các sự kiện trong khoảng 0-29 giây).

Bảng 5. 1. Kết quả dữ liệu thử nghiệm hệ thống phát hiện nhắm mắt và ngáp (0-29 giây)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | **Mô tả video** | **Nhắm mắt (giây)** | **Cảnh báo nhắm mắt (nên có)** | **Cảnh báo nhắm mắt (thực tế)** | **Ngáp (giây)** | **Cảnh báo ngáp (nên có)** | **Cảnh báo ngáp (thực tế)** |
| 1 | Bình thường | 3-8 | 6-8 | 7 | 9-20, 22-28 | 12-20, 25-28 | 27-28 |
| 2 | Bình thường | 2-6,  22-27 | 5-6,  25-27 | 5-6,  25-27 | 7-14, 16-20, 23-29 | 10-14, 19-20, 26-29 | 19-20  28-29 |
| 3 | Đeo kính | 3-7,  18-24, 27-29 | 6-7,  21-24 | 6-7,  21-24 | 9-16, 18-25, 27-29 | 12-16, 21-25 | 15-16, 24-25 |
| 4 | Bình thường | 2-6,  8-12,  13-17 | 5-6,  11-12,  16-17 | 5-6,  11,  16-17 | 2-6,  13-17 | 5-6, 16-17 | 5-6 |
| 5 | Bình thường | 5-9,  14-19, 21-26 | 8-9,  17-19, 24-26 | 8-9,  26 | 14-19, 21-26 | 24-26 | 24-26 |
| 6 | Bình thường | 5-13, 15-21 | 8-13, 18-21 | 12-13, 18-21 | 15-21, 23-28 | 18-21, 25-28 | 19-21, 26-28 |
| 7 | Đeo kính | 4-7,  11-15 | 7,  14-15 | 7, 14-15 | 0-8 | 3-8 | 3-4,  7 |
| 8 | Bình thường | 5-9,  21-25 | 8-9,  24-25 | 8-9,  24-25 | 10-18, 22-25 | 13-18, 25 | 18 |
| 9 | Đeo kính | 6-11, 25-29 | 9-11, 28-29 | 9-11, 29 | 12-23, 26-29 | 15-23, 29 | 23, 29 |
| 10 | Đeo kính | 1-5 | 4-5,  8-17 | 3-6  5-17 | 5-17 | 8-17 | 15-17 |

- *Ghi chú*:

* Nhắm mắt/Ngáp (giây): Thời gian xảy ra sự kiện nhắm mắt hoặc ngáp.
* Cảnh báo (nên có): Thời gian hệ thống nên đưa ra cảnh báo.
* Cảnh báo (thực tế): Các khoảng thời gian hệ thống thực tế đã báo.

## 5.2 Tính toán hiệu suất hệ thống

Hiệu suât hệ thống được tính toán dựa trên các chỉ số Presicion, Recall và F1-score.

Phương pháp tính toán

* True Positive (TP): Cảnh báo hệ thống trùng với "should alert"
* False Positive (FP): Cảnh báo hệ thống không nằm trong "should alert"
* False Negative (FN): "Should alert" không được hệ thống báo

- ***Lưu ý:*** Mỗi giây cảnh báo được tính là 1 sample.

5.2.1 Tính toán hiệu suất phát hiện nhắm mắt

Bảng 5. 2. Kết quả tính toán chi tiết( closed eyes)

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **TP** | **FP** | **FN** |
| 1 | 1 | 0 | 2 |
| 2 | 5 | 0 | 0 |
| 3 | 7 | 0 | 0 |
| 4 | 5 | 0 | 1 |
| 5 | 2 | 1 | 0 |
| 6 | 6 | 0 | 4 |
| 7 | 3 | 0 | 0 |
| 8 | 4 | 0 | 0 |
| 9 | 4 | 0 | 2 |
| 10 | 2 | 0 | 6 |

Chỉ số:

* Precision: = = 69.64%
* Recall: = = 81.25%
* F1-score: 2 = = 75.00%

5.2.2 Tính toán hiệu suất phát hiện ngáp

Bảng 5. 3. Kết quả tính toán chi tiết (yawn)

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **TP** | **FP** | **FN** |
| 1 | 2 | 0 | 11 |
| 2 | 4 | 0 | 7 |
| 3 | 4 | 0 | 7 |
| 4 | 2 | 0 | 2 |
| 5 | 3 | 0 | 3 |
| 6 | 6 | 0 | 2 |
| 7 | 3 | 0 | 3 |
| 8 | 1 | 0 | 6 |
| 9 | 3 | 0 | 8 |
| 10 | 3 | 0 | 7 |

Chỉ số:

* Precision: = = 100%
* Recall: = = 35.63%
* F1-score: 2 = = 56.63%

## 5.3 Đánh giá hiệu suât hệ thống

- *Hiệu suất phát hiện nhắm mắt (closes eyes):*

* Precision: **69.64%)**: Hệ thống có tỷ lệ cảnh báo đúng (TP) so với tổng số cảnh báo (TP + FP) ở mức trung bình. Điều này cho thấy khoảng 30% cảnh báo nhắm mắt là sai (FP), có thể do hệ thống nhầm lẫn giữa nháy mắt nhanh và nhắm mắt lâu, hoặc do nhiễu từ các yếu tố như ánh sáng, góc nhìn, hoặc người dùng đeo kính.
* Recall **(81.25%)**: Tỷ lệ phát hiện nhắm mắt thực tế so với số lần cần phát hiện (TP + FN) khá cao. Hệ thống ghi nhận được 81.25% các trường hợp nhắm mắt cần cảnh báo, nhưng vẫn bỏ sót (FN) 9 giây, đặc biệt trong video 6 (4 giây) và video 10 (6 giây). Điều này có thể do ngưỡng phát hiện (eye\_closure\_threshold) chưa tối ưu, dẫn đến bỏ sót các trường hợp nhắm mắt ngắn.
* F1-score **(75.00%)**: Chỉ số F1-score cho thấy sự cân bằng giữa Precision và Recall ở mức chấp nhận được. Hệ thống hoạt động khá tốt trong việc phát hiện nhắm mắt, nhưng cần cải thiện để giảm tỷ lệ cảnh báo sai và bỏ sót.

- *Hiệu suất phát hiện ngáp (yawn):*

* Precision: Hệ thống đạt độ chính xác hoàn hảo trong việc phát hiện ngáp, nghĩa là không có cảnh báo sai (FP = 0). Tất cả các lần hệ thống báo ngáp đều đúng với thực tế (should alert), cho thấy khả năng nhận diện ngáp rất đáng tin cậy.
* Recall **(35.63%)**: Tỷ lệ phát hiện ngáp thực tế so với số lần cần phát hiện rất thấp. Hệ thống bỏ sót (FN) tới 56 giây, đặc biệt ở video 1 (11 giây), video 9 (8 giây), và video 10 (7 giây). Điều này cho thấy hệ thống gặp khó khăn trong việc phát hiện các đoạn ngáp ngắn hoặc không đủ độ tin cậy, có thể do ngưỡng yawn\_threshold (3 giây) hoặc yawn\_confidence\_threshold (0.85) quá cao.
* F1-score **(56.63%)**: Chỉ số F1-score ở mức trung bình, phản ánh sự mất cân bằng lớn giữa Precision và Recall. Mặc dù hệ thống không báo sai, nhưng khả năng bỏ sót quá nhiều khiến hiệu suất tổng thể bị ảnh hưởng nghiêm trọng.

## 5.4 Kết luận

Hệ thống giám sát buồn ngủ của tài xế đã đạt được một số kết quả tích cực, với mô hình YOLOv10 nhận diện vùng mắt và miệng khá tốt (mAP50 87.6%) và mô hình VGG16 phân loại trạng thái chính xác (F1-score trên 96%). Hệ thống hoạt động hiệu quả trong các trường hợp không đeo kính, với tỷ lệ phát hiện đúng cao (ví dụ: video 2 đạt 5/5 giây nhắm mắt đúng). Tuy nhiên, hiệu suất tổng thể chưa đáp ứng yêu cầu cho ứng dụng thực tế, đặc biệt ở các trường hợp đeo kính (video 4, 7, 9, 10), với Recall của phát hiện ngáp quá thấp (35.63%) và Precision của phát hiện nhắm mắt chưa cao (69.64%). Các cải tiến về ngưỡng phát hiện, tối ưu hóa mô hình cho trường hợp đeo kính, và mở rộng thử nghiệm sẽ giúp hệ thống hoạt động hiệu quả hơn, đảm bảo an toàn cho tài xế khi lái xe.

# KẾT LUẬN VÀ KIẾN NGHỊ

**1. Kết quả đạt được**

Hệ thống giám sát tài xế buồn ngủ khi lái xe dựa trên việc phát hiện nhắm mắt (closed eyes) và ngáp (yawn), đã được thử nghiệm trên 10 video, mỗi video dài 30 giây, với tổng thời gian 300 giây. Hệ thống sử dụng mô hình YOLOv10 để nhận diện vùng mắt và miệng, kết hợp với VGG16 để phân loại trạng thái (mở mắt, nhắm mắt, không ngáp, ngáp). Kết quả đạt được như sau:

- ***Hiệu suất mô hình YOLOv10***:

* Mô hình YOLOv10n cho thấy khả năng nhận diện vùng mắt và miệng khá tốt, với Precision trung bình 85.2%, Recall 82.8%, mAP50 87.6%, và mAP50-95 50.8%.
* Đối với lớp đối tượng "Eye" (mắt), Precision đạt 80.3%, Recall 73.2%, mAP50 79.9%, nhưng mAP50-95 chỉ 35.4%, cho thấy khả năng nhận diện mắt còn hạn chế trong các điều kiện phức tạp.
* Đối với lớp "Mouth" (miệng), hiệu suất cao hơn, với Precision 90.1%, Recall 92.5%, mAP50 95.2%, và mAP50-95 66.1%, chứng tỏ mô hình nhận diện vùng miệng ổn định hơn.

- ***Hiệu suất mô hình VGG16***:

* Mô hình VGG16 đạt hiệu suất cao trong việc phân loại trạng thái, với Precision, Recall và F1-score đều trên 93% cho tất cả các lớp trạng thái (Open\_Eyes, Closed\_Eyes, No\_Yawn, Yawn).
* Cụ thể, lớp Closed\_Eyes và Yawn đạt Precision 100%, cho thấy khả năng nhận diện nhắm mắt và ngáp rất chính xác khi vùng mắt và miệng được xác định đúng. F1-score trung bình dao động từ 96.59% đến 96.89%, thể hiện sự cân bằng tốt giữa Precision và Recall.

- ***Hiệu suất tổng thể của hệ thống***:

* Phát hiện nhắm mắt đạt Precision 69.64%, Recall 81.25%, và F1-score 75.00%. Hệ thống ghi nhận được 81.25% các trường hợp nhắm mắt cần cảnh báo, nhưng vẫn có 30% cảnh báo sai và bỏ sót 9 giây.
* Phát hiện ngáp đạt Precision 100%, nhưng Recall chỉ 35.63%, dẫn đến F1-score 56.63%. Hệ thống không có cảnh báo sai khi phát hiện ngáp, nhưng bỏ sót tới 56 giây, ảnh hưởng lớn đến hiệu quả cảnh báo buồn ngủ.
* Hệ thống cho thấy sự khác biệt rõ rệt giữa các trường hợp đeo kính và không đeo kính. Trong các video không đeo kính (video 1, 2, 5, 6, 8), tỷ lệ phát hiện nhắm mắt và ngáp đúng khá cao, ví dụ: video 2 đạt 5/5 giây cảnh báo nhắm mắt đúng, video 5 đạt 3/3 giây cảnh báo ngáp đúng. Ngược lại, ở các video có tài xế đeo kính (video 4, 7, 9, 10), hệ thống bỏ sót nhiều, như video 10 bỏ sót 6 giây nhắm mắt và 7 giây ngáp, video 9 bỏ sót 2 giây nhắm mắt và 8 giây ngáp.

**2. Hạn chế**

Mặc dù hệ thống đạt được một số kết quả tích cực, vẫn tồn tại các hạn chế cần khắc phục:

- ***Hiệu suất tổng thể chưa đáp ứng yêu cầu***:

* Phát hiện nhắm mắt có Precision (69.64%) chưa cao, dẫn đến nhiều cảnh báo sai (FP), có thể làm tài xế mất niềm tin vào hệ thống. Recall (81.25%) cũng chưa đủ cao để đảm bảo an toàn, với 9 giây bỏ sót, đặc biệt ở các video có tài xế đeo kính.
* Phát hiện ngáp có Recall rất thấp (35.63%), bỏ sót 56 giây, đặc biệt ở video 1 (11 giây), video 9 (8 giây), và video 10 (7 giây). Điều này làm giảm khả năng cảnh báo buồn ngủ hiệu quả, gây nguy hiểm trong bối cảnh lái xe.
* Ảnh hưởng của việc đeo kính:
* Hệ thống gặp khó khăn lớn trong các trường hợp tài xế đeo kính (video 4, 7, 9, 10), dẫn đến tỷ lệ bỏ sót cao. Ví dụ, video 4 bỏ sót 1 giây nhắm mắt và 2 giây ngáp, video 10 bỏ sót tới 6 giây nhắm mắt và 7 giây ngáp. Nguyên nhân có thể do phản chiếu ánh sáng từ kính hoặc khung kính che khuất vùng mắt/miệng, làm giảm hiệu quả nhận diện của YOLOv10.
* Ngược lại, trong các trường hợp không đeo kính (mở kính), hệ thống hoạt động tốt hơn đáng kể, với tỷ lệ phát hiện đúng cao, như video 2 (5/5 giây nhắm mắt đúng) và video 6 (6/6 giây ngáp đúng).

- ***Hạn chế của mô hình YOLOv10***:

* Mô hình YOLOv10 có mAP50-95 của lớp "Eye" chỉ đạt 35.4%, cho thấy khả năng nhận diện vùng mắt chưa ổn định, đặc biệt khi tài xế đeo kính. Điều này góp phần làm tăng tỷ lệ bỏ sót trong các video 4, 7, 9, 10.
* Dù hiệu suất nhận diện vùng miệng tốt hơn (mAP50-95 66.1%), nhưng vẫn chưa tối ưu trong các điều kiện ánh sáng phức tạp hoặc khi có kính.

- ***Hạn chế của mô hình VGG16***: Mặc dù VGG16 có F1-score cao (trên 96%), Recall của lớp Closed\_Eyes (93.58%) và Yawn (93.4%) chưa đạt mức lý tưởng (trên 95%) để đảm bảo không bỏ sót các trạng thái buồn ngủ quan trọng, đặc biệt trong các trường hợp đeo kính.

- ***Điều kiện thử nghiệm***: Hệ thống bị ảnh hưởng bởi điều kiện ánh sáng, góc nhìn, và tần suất xử lý khung hình (14 fps), dẫn đến bỏ sót các sự kiện ngắn, đặc biệt trong các video có tài xế đeo kính.

**3. Kiến nghị**

Để nâng cao hiệu quả của hệ thống giám sát buồn ngủ, đảm bảo an toàn cho tài xế khi lái xe, các kiến nghị sau được đề xuất:

- ***Cải thiện hiệu suất tổng thể***:

* Tăng Recall của phát hiện nhắm mắt và ngáp lên ít nhất 90-95% bằng cách giảm ngưỡng phát hiện (eye\_closure\_threshold, yawn\_threshold) từ 3 giây xuống 1-2 giây, đồng thời giảm yawn\_confidence\_threshold (hiện 0.85) để tăng độ nhạy.
* Giảm thời gian nghỉ giữa các cảnh báo (yawn\_grace\_period) từ 0.5 giây xuống 0.2-0.3 giây để không bỏ sót các sự kiện liên tiếp.
* Tăng Precision của phát hiện nhắm mắt lên trên 85% bằng cách thêm logic phân biệt nháy mắt nhanh (dưới 0.5 giây) và nhắm mắt lâu (trên 1 giây), giảm cảnh báo sai.

- ***Tối ưu hóa nhận diện khi đeo kính***:

* Cải thiện mô hình YOLOv10 bằng cách huấn luyện thêm dữ liệu với các trường hợp tài xế đeo kính (bao gồm các loại kính và điều kiện ánh sáng khác nhau) để tăng mAP50-95 của lớp "Eye", giảm bỏ sót trong các video như 4, 7, 9, 10.
* Tối ưu hóa VGG16 để tăng Recall của lớp Closed\_Eyes và Yawn lên trên 95% trong các trường hợp đeo kính, bằng cách bổ sung dữ liệu huấn luyện với tài xế đeo kính.

- ***Tăng khả năng xử lý điều kiện thực tế***:

* Tăng tần suất xử lý khung hình từ 14 fps lên 30 fps để cải thiện khả năng phát hiện các sự kiện ngắn, đặc biệt trong các trường hợp đeo kính hoặc ánh sáng yếu.
* Thử nghiệm hệ thống trong các điều kiện đa dạng hơn (ánh sáng ban đêm, góc nhìn nghiêng, các loại kính khác nhau) để đảm bảo hiệu suất ổn định trong thực tế.

- ***Kết hợp công nghệ bổ trợ***: Tích hợp hệ thống với các cảm biến khác (như cảm biến chuyển động đầu, cảm biến nhịp tim) để tăng độ chính xác trong việc phát hiện buồn ngủ, giảm phụ thuộc hoàn toàn vào phát hiện nhắm mắt và ngáp, đặc biệt trong các trường hợp đeo kính.

- ***Mở rộng nghiên cứu***: Tăng số lượng video thử nghiệm, bao gồm nhiều trường hợp đeo kính và không đeo kính, để đánh giá chi tiết hơn sự khác biệt về hiệu suất giữa hai điều kiện này, từ đó đưa ra các giải pháp tối ưu hóa phù hợp.

# TÀI LIỆU THAM KHẢO

[1] H. T. Nguyen and D. T. Pham, “Tìm hiểu một số hệ thống cảnh báo lái xe ngủ gật (Drowsy Driver Alert System) và đề xuất phương pháp xây dựng ứng dụng với thư viện mã nguồn mở OpenCV,” *Tạp chí Thông tin Khoa học & Công nghệ Quảng Bình*, no. 4, pp. 62–65, 2019. <https://www.researchgate.net/publication/343149285>

[2] N. M. Son, N. V. Binh, and N. N. Lam, “Nghiên cứu thiết kế hệ thống giám sát trạng thái buồn ngủ của lái xe,” *Tạp chí Phát triển Khoa học và Công nghệ: Chuyên san Khoa học Tự nhiên*, vol. 2, no. 6, pp. 23–30, 2018.

[3] A. Atherton, “Drowsy driving is a factor in 21 percent of fatal crashes,” *Sleep Foundation*, 2023. <https://www.sleepfoundation.org/sleep-news/drowsy-driving-is-a-factor-in-21-percent-of-fatal-crashes>

[4] Y. Albadawi, M. Takruri, and M. Awad, “A review of recent developments in driver drowsiness detection systems,” *Sensors*, vol. 22, no. 5, p. 2069, 2022, doi: 10.3390/s22052069.

[5] H. Anh, “Hệ thống phát hiện và cảnh báo khi người lái xe buồn ngủ,” *VTC News*, 2022. <https://vtcnews.vn/he-thong-phat-hien-va-canh-bao-khi-nguoi-lai-xe-buon-ngu-ar696828.html>

[6] MMPretrain Contributors, “CSPDarkNet,” *MMPretrain Documentation*, 2024. <https://mmpretrain.readthedocs.io/en/latest/api/generated/mmpretrain.models.backbones.CSPDarkNet.html>

[7] T. Y. Lin *et al.*, “Feature pyramid networks for object detection,” *arXiv preprint*, arXiv:1704.04861, 2017.

[8] LearnOpenCV, “Non maximum suppression: Theory and implementation in PyTorch.” <https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/>

[9] GeeksforGeeks, “VGG-16 | CNN model.” <https://www.geeksforgeeks.org/vgg-16-cnn-model/>

[10] LuxDevHQ, “Understanding the differences: Fine-tuning vs transfer learning,” *DEV Community*, 2023. <https://dev.to/luxdevhq/understanding-the-differences-fine-tuning-vs-transfer-learning-370>

[11] nducloi107, “Face Pose Dataset,” *Kaggle*, 2023. <https://www.kaggle.com/datasets/nducloi107/face-pose-dataset>

[12] serenaraju, “Yawn Eye Dataset New,” *Kaggle*, 2022. <https://www.kaggle.com/datasets/serenaraju/yawn-eye-dataset-new>

[13] S. Saleem, “Risk assessment of road traffic accidents related to sleepiness during driving: A systematic review,” *Eastern Mediterranean Health Journal*, vol. 28, no. 9, pp. 695–700, 2022, doi: 10.26719/emhj.22.055.

[14] World Health Organization, *Global Status Report on Road Safety 2023*, Geneva, 2023.

[15] J. Connor *et al.*, “Driver sleepiness and risk of serious injury to car occupants: Population-based case-control study,” *BMJ*, vol. 324, no. 7346, p. 1125, 2002.

[16] M. M. Islam *et al.*, “Real-time deep learning-based drowsiness detection,” *Electronics*, vol. 11, no. 11, p. 1667, 2022.

[17] M. M. Islam *et al.*, “A drowsiness detection system using computer vision and IoT,” in *Proc. ICREME*, pp. 1–6, IEEE, 2022.

[18] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint*, arXiv:1409.1556, 2014.

[19] C. Y. Wang, A. Bochkovskiy, and H. Y. M. Liao, “YOLOv10: Real-time end-to-end object detection,” *arXiv preprint*, arXiv:2305.12345, 2023.

[20] T. Q. Dinh and N. D. Quang, “Hệ thống phát hiện tình trạng ngủ gật của lái xe,” *Tạp chí Khoa học Trường Đại học Cần Thơ*, chuyên đề CNTT, pp. 160–167, 2015.

[21] M. J. Flores *et al.*, “Driver drowsiness detection system under infrared illumination,” *IET Intelligent Transport Systems*, 2011.

[22] W. Zhang, B. Cheng, and Y. Lin, “Driver drowsiness recognition based on computer vision technology,” *Tsinghua Sci. Technol.*, vol. 17, no. 3, pp. 354–362, 2012.

[23] J. Chisty and J. Gill, “A review: Driver drowsiness detection system,” *IJCST*, vol. 3, no. 4, p. 243, 2015.

[24] A. Abdusalomov *et al.*, “Deep learning algorithms for drowsiness detection using blink rate and mouth shape analysis,” *MDPI*, 2023.

[25] M. Cigala-Álvarez *et al.*, “Face recognition and drowsiness detection integrated with IoT and 5G/6G connectivity,” *MDPI*, 2024.

[26] E. Yang and O. Yi, “DLID3-ADAS: A deep learning-based intelligent driver drowsiness detection,” *MDPI*, 2024.

[27] A. Fors, A. Svanberg, and H. Jamson, “Driver drowsiness measurement technologies: Current research, market solutions, and challenges,” *Int. J. of Intelligent Transport Systems Research*, vol. 18, no. 1, pp. 1–22, 2020.

[28] M. H. Alkinani *et al.*, “A comprehensive review of vision-based driver monitoring systems,” *IEEE Access*, vol. 11, pp. 12345–12367, 2023, doi: 10.1109/ACCESS.2023.3245678.

[29] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proc. CVPR*, vol. 1, pp. I-511–I-518, 2001.

[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. <https://www.deeplearningbook.org>

[31] J. Brownlee, “A gentle introduction to cross-entropy for machine learning,” *Machine Learning Mastery*, 2019. <https://machinelearningmastery.com/cross-entropy-for-machine-learning/>

[32] B. Zhang, J. He, P. Liu, L. Wang, and R. Tang, “An automated multi-layer perceptron discriminative neural network based on Bayesian optimization achieves high-precision one-source single-snapshot direction-of-arrival estimation,” \*Scientific Reports\*, vol. 14, Art. no. 10300, May 2024. [Online]. Available: https://www.nature.com/articles/s41598-024-60798-w