TRƯỜNG ĐẠI HỌC HỌC VĂN LANG

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN MÔN HỌC HK233

NHẬP MÔN PHÂN TÍCH DỮ LIỆU VÀ HỌC SÂU (71ITDS30203)

ỨNG DỤNG HỌC MÁY VÀO PHÂN LOẠI VÀ NHẬN DIỆN NGƯỜI ĐEO KHẨU TRANG

Nhóm sinh viên thực hiện (Họ tên - Mã SV):

1. Thái Thị Trà My - 2174802010891

2. Lâm Nguyễn Anh Thư - 2174802010785

3. Võ Quốc Thành – 2174802010886

4. Lê Chí Thiện – 2174802010906

5. Nguyễn Minh Huy – 2174802010934

TP. Hồ Chí Minh – năm 2024

LỜI MỞ ĐẦU

Trong thời đại hiện nay, nhu cầu đeo khẩu trang đã trở thành một phần của cuộc sống hàng ngày trong nhiều bối cảnh khác nhau như bệnh viện, phòng thí nghiệm, nhà máy và những nơi công cộng đông đúc. Đặc biệt, trong bối cảnh đại dịch COVID-19 vừa xảy ra ở năm 2019 và gần đây hơn là sự nghiêm trọng của căn bệnh bạch hầu, việc đeo khẩu trang càng trở nên quan trọng trong việc ngăn ngừa sự lây lan của các bệnh truyền nhiễm. Để giám sát và đảm bảo mọi người tuân thủ quy định đeo khẩu trang, chúng tôi đã phát triển một hệ thống nhận dạng đeo và không đeo khẩu trang sử dụng công nghệ Deep Learning.

Hệ thống này được xây dựng dựa trên nền tảng Python và sử dụng các thư viện mạnh mẽ như PyTorch, Torchvision, PIL (Pillow) để huấn luyện mô hình nhận dạng. Mục tiêu của dự án là tạo ra một công cụ hiệu quả và chính xác để phân biệt giữa người đeo và không đeo khẩu trang trong thời gian thực, từ đó hỗ trợ việc duy trì an toàn và vệ sinh trong các môi trường đòi hỏi điều kiện này.

Đồ án này được thực hiện với mục tiêu xây dựng một chương trình giám sát việc đeo khẩu trang, sử dụng dữ liệu thực tế và các phương pháp phân tích tiên tiến. Chúng tôi xin gửi lời cảm ơn chân thành đến thầy Huỳnh Thái Học và thầy Nguyễn Thái Anh, người đã cung cấp những kiến thức bổ ích và sự hướng dẫn, hỗ trợ tận tình trong quá trình thực hiện đồ án. Chúng tôi rất mong nhận được phản hồi và góp ý từ phía thầy, cô để nâng cao chất lượng của đồ án.

Nhóm báo cáo đồ án

MỤC LỤC

CHƯƠNG 1. TỔNG QUAN VỀ DỮ LIỆU VÀ HỌC SÂU 5
I. Dữ liệu 5
1. Khái niệm 5
2. Các dạng của Dữ liệu 5
3. Xử lý dữ liệu 6
II. Học sâu 7
1. Tổng quan 7
2. Quy trình học sâu 7
3. Các mạng trong Học sâu 8
3.1. Mạng nơ-ron nhân tạo (Artificial Neural Networks - ANN) 8
3.2. Mạng nơ-ron sâu (Deep Neural Networks - DNN) 8
4. Các loại nơ-ron sâu phổ biến 8
4.1. Mạng nơ-ron tích chập (Convolutional Neural Networks - CNN) 8
4.2. Mạng nơ-ron hồi quy (Recurrent Neural Networks - RNN) 9
4.3. Mạng nơ-ron đối kháng sinh học (Generative Adversarial Networks - GAN) 10
CHƯƠNG 2. ỨNG DỤNG HỌC MÁY VÀO PHÂN LOẠI VÀ NHẬN DIỆN NGƯỜI ĐEO KHẨU TRANG 11
I. Tổng quan về đồ án 11
1. Mục tiêu 11
2. Phạm vi của dự án 11
3. Công nghệ sử dụng 11
4. Tiềm năng ứng dụng 12
II. Nguồn cung cấp dữ liệu 12
III. Thực hiện đồ án 12
IV. Kết quả thực hiện và nhận xét 19
KẾT LUẬN VÀ ĐỀ XUẤT 25
BẢNG ĐÁNH GIÁ THÀNH VIÊN 26

MỤC LỤC HÌNH ẢNH

Hình 1: Hình ảnh minh họa về Dữ liệu 5
Hình 2: Minh họa về Xử lý dữ liệu 7
Hình 4: Import thư viện cần thiết 12
Hình 5: Code khởi tạo dataset, lấy kích thước, phần tử của dataset 13
Hình 6: Tạo đường dẫn đến thư mục chứa dữ liệu 13
Hình 7: Biến đổi hình ảnh về cùng một thông số 13
Hình 8: Đọc dữ liệu từ thư mục, áp dụng biến đổi 14
Hình 9: Chia dữ liệu thành tập huấn luyện và tập kiểm tra 14
Hình 10: Tạo DataLoader cho tập huấn luyện và kiểm tra 14
Hình 11: Kiểm tra các số lượng của tập dữ liệu 14
Hình 12: Hiển thị dữ liệu mẫu 15
Hình 13: Xây dựng mô hình CNN 16
Hình 14: Huấn luyện mô hình 16
Hình 15: Huấn luyện mô hình 17
Hình 16: Huấn luyện mô hình 17
Hình 17: Vẽ biểu đồ loss và accuracy 18
Hình 18: Kiểm tra dự đoán mô hình 18
Hình 19: Nhận diện khuôn mặt 19
Hình 20: Kết quả huấn luyện 19
Hình 21: Kết quả vẽ biểu đồ 21
Hình 22: Hình ảnh nhận diện 22
Hình 23: Kết quả nhận diện 22
Hình 24: Kết quả nhận diện 23
Hình 25: Kết quả nhận diện 23
Hình 26: Kết quả nhận diện 24

CHƯƠNG 1. TỔNG QUAN VỀ DỮ LIỆU VÀ HỌC SÂU

I. Dữ liệu

1. Khái niệm

Dữ liệu (Data) là tập hợp các thông tin, giá trị, hoặc ký tự có thể được xử lý và phân tích bởi máy tính. Dữ liệu có thể ở nhiều dạng khác nhau, chẳng hạn như văn bản, số, hình ảnh, âm thanh, và video. Dữ liệu được lưu trữ, quản lý và truyền tải qua các hệ thống máy tính và mạng.

Dữ liệu được đo lường, thu thập, báo cáo và phân tích trước khi được hiển thị dưới dạng đồ thị, bảng hoặc hình ảnh. Dữ liệu là một khái niệm rộng dùng để chỉ một số thông tin hoặc kiến thức hiện có đã được biểu diễn hoặc mã hóa cho phép con người sử dụng hoặc xử lý tốt hơn.

Hình 1: Hình ảnh minh họa về Dữ liệu

2. Các dạng của Dữ liệu

Dữ liệu có cấu trúc (Structured Data): Là loại dữ liệu được tổ chức theo định dạng cố định, thường là dạng bảng với các hàng và cột. Các trường dữ liệu trong các cột này có thể được xác định và truy vấn một cách dễ dàng. Các mục trong dữ liệu có cấu trúc có thể được nhóm lại để tạo mối quan hệ với nhau. Dữ liệu có cấu trúc là loại dữ liệu dễ sử dụng nhất cho các doanh nghiệp trong thời gian gần đây vì nó dễ dàng trong việc phân tích, lưu trữ cũng như tìm kiếm. Được lưu trữ trong các cơ sở dữ liệu quan hệ như MySQL, PostgreSQL.

Dữ liệu phi cấu trúc (Unstructured Data): Là loại dữ liệu không có định dạng hoặc cấu trúc cố định, khó truy vấn và phân tích bằng các phương pháp truyền thống. Dữ

liệu không có cấu trúc chiếm một phần đáng kể của tất cả dữ liệu trên thế giới, nó không thể được lưu trữ trong cơ sở dữ liệu theo dạng hàng-cột và cũng không có mô hình dữ liệu liên quan. Thường cần các công cụ và kỹ thuật đặc biệt để xử lý, như xử lý ngôn ngữ tự nhiên (NLP) hoặc học máy (machine learning).

Dữ liệu bán cấu trúc (Semi-structured Data): Là loại dữ liệu không hoàn toàn có cấu trúc như dữ liệu có cấu trúc, nhưng vẫn chứa các thông tin meta hoặc thẻ giúp tổ chức và quản lý. Dữ liệu bán cấu trúc không hoàn toàn tuân theo cấu trúc bảng, nhưng có các yếu tố định danh hoặc thẻ, dễ dàng hơn để phân tích và truy vấn so với dữ liệu phi cấu trúc và thường sử dụng các định dạng như XML, JSON.

3. Xử lý dữ liệu

Xử lý dữ liệu là quá trình thu thập, chuyển đổi, làm sạch, tổ chức và phân tích dữ liệu để biến nó thành thông tin hữu ích và có ý nghĩa. Quá trình này có thể bao gồm nhiều bước và kỹ thuật khác nhau, tùy thuộc vào loại dữ liệu và mục đích sử dụng.

Các bước chính trong xử lý dữ liệu:

- ***Thu thập dữ liệu (Data Collection):*** Là quá trình sử dụng các công cụ và phần mềm để thu thập dữ liệu từ nhiều nguồn khác nhau, bao gồm cảm biến, cơ sở dữ liệu, file, API và các nguồn trực tuyến.
- ***Làm sạch dữ liệu (Data Cleaning):*** Là quá trình sử dụng các phương pháp như lọc, thay thế và xử lý ngoại lệ để loại bỏ hoặc sửa các lỗi, dữ liệu trùng lặp, các giá trị không hợp lệ trong dữ liệu thô để đảm bảo dữ liệu chính xác và đồng nhất.
- ***Chuyển đổi dữ liệu (Data Transformation):*** Sử dụng các phép toán, quy tắc, và thuật toán để biến đổi dữ liệu từ định dạng này sang định dạng khác hoặc tính toán các giá trị mới từ dữ liệu hiện có.
- ***Tổ chức dữ liệu (Data Organization):*** Lưu trữ dữ liệu trong cơ sở dữ liệu, kho dữ liệu hoặc các hệ thống quản lý dữ liệu khác và sắp xếp dữ liệu theo cấu trúc hợp lý để dễ dàng truy cập và phân tích.
- ***Phân tích dữ liệu (Data Analysis):*** Sử dụng phần mềm phân tích dữ liệu như R, Python, SAS, và các công cụ BI (Business Intelligence) với các phương pháp thống kê, toán học, và học máy để khai thác thông tin và tìm ra các mô hình, xu hướng.
- ***Trực quan hóa dữ liệu (Data Visualization):*** Sử dụng các công cụ như Tableau, Power BI, Matplotlib để tạo trực quan hóa bằng cách biểu diễn dữ liệu và kết quả phân tích bằng các đồ thị, biểu đồ, và hình ảnh để dễ dàng hiểu và trình bày.
- ***Lưu trữ và bảo mật dữ liệu (Data Storage and Security):*** Sử dụng các cơ sở dữ liệu, hệ thống lưu trữ đám mây, và các biện pháp bảo mật như mã hóa, kiểm soát truy cập để lưu trữ dữ liệu một cách an toàn và bảo vệ dữ liệu khỏi các mối đe dọa.

Hình 2: Minh họa về Xử lý dữ liệu

II. Học sâu

1. Tổng quan

Học sâu (Deep Learning) là một nhánh của học máy (Machine Learning) tập trung vào việc sử dụng các mạng nơ-ron nhân tạo (Artificial Neural Networks) với nhiều lớp (layers) để mô hình hóa và phân tích dữ liệu. Học sâu đã đạt được những tiến bộ lớn trong nhiều lĩnh vực như xử lý hình ảnh, nhận diện giọng nói, và dịch máy tự động.

Chuyển đổi số không phải đơn thuần thay đổi cách thực hiện công việc từ thủ công truyền thống (ghi chép trong sổ sách, họp trực tiếp,...) sang vận dụng công nghệ để giảm thiểu sức người. Trên thực tế, chuyển đổi số đóng vai trò thay đổi tư duy kinh doanh, phương thức điều hành, văn hóa tổ chức,...Mục tiêu của chuyển đổi số thường là tạo ra giá trị mới cho khách hàng, cải thiện hiệu suất hoạt động, quản lý tổ chức và tạo ra lợi thế cạnh tranh thông qua việc sử dụng công nghệ, kỹ thuật số, tạo ra lợi ích kinh doanh bền vững.

Lợi ích của Chuyển đổi số là tăng cường sự nhanh nhẹn và linh hoạt, cải thiện trải nghiệm của khách hàng, tiếp cận các thị trường và nguồn doanh thu mới, tạo ra một bước chuyển mình tích cực

2. Quy trình học sâu

- ***Thu thập và chuẩn bị dữ liệu:*** Đầu tiên là lấy dữ liệu từ các nguồn khác nhau, như cơ sở dữ liệu, API, và các tệp tin. Sau đó là làm sạch và tiền xử lý dữ liệu, bao gồm việc chuẩn hóa, làm mịn, và chia dữ liệu thành tập huấn luyện, tập kiểm tra, và tập xác nhận

- ***Xây dựng mô hình:***
    - Thiết kế kiến trúc: Quyết định số lượng và loại các lớp, số lượng nơ-ron trong mỗi lớp, và các hàm kích hoạt.
    - Khởi tạo trọng số: Đặt giá trị ban đầu cho các trọng số của mạng.
- ***Huấn luyện mô hình:***
    - Truyền xuôi (Forward Propagation): Tính toán đầu ra của mạng cho một mẫu dữ liệu.
    - Hàm mất mát (Loss Function): Đo lường sự khác biệt giữa đầu ra dự đoán và giá trị thực tế.
    - Truyền ngược (Backward Propagation): Cập nhật các trọng số của mạng để giảm hàm mất mát, sử dụng thuật toán lan truyền ngược và tối ưu hóa, chẳng hạn như Gradient Descent.
- ***Đánh giá và tinh chỉnh:*** Kiểm tra hiệu suất của mô hình trên tập dữ liệu kiểm tra và xác nhận. Sau đó điều chỉnh các tham số và cấu trúc mô hình để cải thiện hiệu suất, có thể bao gồm việc thay đổi số lớp, số nơ-ron, hoặc các kỹ thuật điều chỉnh như dropout, batch normalization.
- ***Triển khai:*** Sử dụng mô hình đã huấn luyện trong các ứng dụng thực tế, chẳng hạn như nhận diện khuôn mặt, dịch máy, hoặc dự đoán hành vi người dùng.

3. Các mạng trong Học sâu

3.1. Mạng nơ-ron nhân tạo (Artificial Neural Networks - ANN)

Là mô hình toán học lấy cảm hứng từ cấu trúc và chức năng của não bộ con người. ANN bao gồm các đơn vị xử lý đơn giản gọi là "nơ-ron" (neurons), được sắp xếp thành các lớp (layers) và kết nối với nhau bằng các trọng số (weights).

Các thành phần chính:

- Nơ-ron: Đơn vị cơ bản của mạng, thực hiện các phép toán đơn giản.
- Trọng số: Hệ số quyết định tầm quan trọng của mỗi kết nối giữa các nơ-ron.
- Hàm kích hoạt (Activation Function): Hàm toán học được sử dụng để giới hạn hoặc biến đổi đầu ra của nơ-ron.

3.2. Mạng nơ-ron sâu (Deep Neural Networks - DNN)

Là các mạng nơ-ron nhân tạo có nhiều lớp ẩn (hidden layers) giữa lớp đầu vào (input layer) và lớp đầu ra (output layer). DNN có các lớp ẩn cho phép mạng học các đặc trưng phức tạp và biểu diễn các mối quan hệ phi tuyến tính trong dữ liệu.

4. Các loại nơ-ron sâu phổ biến

4.1. Mạng nơ-ron tích chập (Convolutional Neural Networks - CNN)

Convolutional Neural Network (CNN) là một loại mô hình học sâu (deep learning) được sử dụng chủ yếu trong lĩnh vực nhận diện hình ảnh và thị giác máy tính (computer

vision). CNN có cấu trúc đặc biệt, bao gồm nhiều lớp khác nhau như lớp tích chập (convolutional layer), lớp kích hoạt (activation layer), lớp gộp (pooling layer), và lớp kết nối đầy đủ (fully connected layer).

CNN được sử dụng chủ yếu trong xử lý và phân tích hình ảnh. Đặc điểm của CNN là sử dụng các lớp tích chập (convolutional layers) để tự động trích xuất các đặc trưng từ hình ảnh, giúp giảm số lượng tham số và tăng hiệu quả.

Các thành phần chính của CNN:

- Lớp Tích Chập (Convolutional Layer): Mục đích là trích xuất các đặc trưng từ ảnh đầu vào bằng cách áp dụng các bộ lọc (filters/kernels). Cách hoạt động: Một bộ lọc nhỏ (ví dụ: 3x3 hoặc 5x5) sẽ trượt qua toàn bộ ảnh và tính toán sản phẩm chập (convolution) giữa bộ lọc và các vùng nhỏ của ảnh. Kết quả là một ma trận đầu ra gọi là feature map.
- Lớp Kích Hoạt (Activation Layer): Mục đích để áp dụng các hàm kích hoạt (activation functions) để tạo ra các tính phi tuyến tính cho mạng.
- Lớp Gộp (Pooling Layer): Mục đích: Giảm kích thước của feature maps và do đó giảm số lượng tham số và tính toán trong mạng. Cách hoạt động: Áp dụng một phép gộp như max pooling hoặc average pooling trên mỗi vùng nhỏ của feature map.
- Lớp Kết Nối Đầy Đủ (Fully Connected Layer): Mục đích: Kết hợp các đặc trưng được trích xuất từ các lớp trước đó để dự đoán kết quả cuối cùng. Cách hoạt động: Toàn bộ đầu ra của lớp trước đó được nối với mỗi nút trong lớp này, giống như trong mạng neural thông thường.
- Lớp Dropout (Dropout Layer): Mục đích: Giảm hiện tượng overfitting bằng cách ngẫu nhiên bỏ qua một số nút trong quá trình huấn luyện. Cách hoạt động: Trong mỗi lần huấn luyện, một tỷ lệ ngẫu nhiên của các nút sẽ bị "bỏ qua" (tức là không được kích hoạt).

Quy trình hoạt động của CNN:

- Nhận đầu vào: Một bức ảnh hoặc một phần dữ liệu tương tự.
- Áp dụng các lớp tích chập: Bộ lọc sẽ trích xuất các đặc trưng từ bức ảnh, tạo ra các feature maps.
- Áp dụng các lớp gộp: Giảm kích thước của các feature maps.
- Áp dụng các lớp kích hoạt: Tạo ra các tính phi tuyến tính cho mạng.
- Áp dụng các lớp fully connected: Kết hợp các đặc trưng đã trích xuất để tạo ra kết quả dự đoán cuối cùng.

4.2. Mạng nơ-ron hồi quy (Recurrent Neural Networks - RNN)

- RNN được sử dụng chủ yếu trong xử lý chuỗi dữ liệu như văn bản và tín hiệu âm thanh.

- Đặc điểm: Có các kết nối hồi quy cho phép mạng nhớ thông tin từ các bước trước đó trong chuỗi.
- Biến thể: LSTM (Long Short-Term Memory) và GRU (Gated Recurrent Unit) giúp giải quyết vấn đề vanishing gradient trong RNN truyền thống.

4.3. Mạng nơ-ron đối kháng sinh học (Generative Adversarial Networks - GAN)

- GAN được sử dụng để tạo ra dữ liệu mới giống với dữ liệu đào tạo.
- Đặc điểm: Bao gồm hai mạng: một mạng sinh (generator) và một mạng phân biệt (discriminator) đối kháng nhau trong quá trình huấn luyện.

CHƯƠNG 2. ỨNG DỤNG HỌC MÁY VÀO PHÂN LOẠI VÀ NHẬN DIỆN NGƯỜI ĐEO KHẨU TRANG

I. Tổng quan về đồ án

Đồ án xây dựng một chương trình nhận diện khuôn mặt đeo khẩu trang là một nỗ lực để hướng đến giải quyết những thách thức mà xã hội ngày nay đặt ra . Dưới đây là một tổng quan về dự án:

1. **Mục tiêu**
   - Phát triển một hệ thống nhận diện khuôn mặt phân biệt và xác định người có mang khẩu trang hay không
   - Cải thiện khả năng nhận diện đối với các biến thể khuôn mặt với khẩu trang, từ các góc nhìn và điều kiện ánh sáng khác nhau. Phát triển một hệ thống nhận diện khuôn mặt áp dụng hỗ trợ trong công nghệ thông tin và thị giác máy tính trong tương lai.
2. **Phạm vi của dự án**
   - Thu thập một bộ dữ liệu chứa ảnh các khuôn mặt, một phần có đeo khẩu trang và một phần không đeo. Dữ liệu này cần phải đủ đa dạng về vị trí, ánh sáng, màu sắc và các điều kiện để đảm bảo tính tổng quát của mô hình.
   - Sử dụng các công cụ xử lý ảnh để tiền xử lý ảnh trước khi đưa vào mô hình nhận diện. Đảm bảo dữ liệu được chuẩn hóa và xử lý phù hợp để giảm thiểu nhiễu và cải thiện khả năng nhận diện.
   - Lựa chọn và huấn luyện mô hình Convolutional Neural Networks (CNN) nhận diện khuôn mặt hiệu quả. Chia dữ liệu thành tập huấn luyện và tập kiểm tra để đánh giá hiệu suất của mô hình. Đo lường các độ đo như độ chính xác, độ nhạy, độ chính xác dựa trên các tiêu chí xác định của dự án.
3. **Công nghệ sử dụng**
   - Python và các thư viện: Sử dụng ngôn ngữ lập trình Python là nền tảng chính để phát triển dự án. Các thư viện như PyTorch, torchvision, PIL (Pillow) được sử dụng để xử lý ảnh, xây dựng và huấn luyện mô hình học máy.
   - Convolutional Neural Networks (CNNs): Sử dụng mô hình CNN để nhận diện khuôn mặt và phân loại việc đeo khẩu trang. Mô hình được xây dựng với các lớp convolutional, max pooling, dropout và fully connected layers để học các đặc trưng quan trọng từ dữ liệu ảnh.
   - Transforms và DataLoader: Sử dụng transforms từ torchvision để tiền xử lý dữ liệu ảnh như thay đổi kích thước, chuyển đổi về tensor và chuẩn hóa. DataLoader được sử dụng để tải và chia dữ liệu thành các batch để huấn luyện và kiểm tra mô hình.

- Biểu đồ và visualization: Sử dụng matplotlib để vẽ biểu đồ loss và accuracy của mô hình sau mỗi epoch, giúp hiển thị quá trình huấn luyện và đánh giá mô hình một cách trực quan.
- Jupyter Notebook: Được sử dụng để tổ chức, lưu trữ và chia sẻ mã nguồn, cũng như phân tích và trực quan hóa kết quả từ mô hình.

4. Tiềm năng ứng dụng

- Sau khi hoàn thành mô hình hoàn chỉnh, triển khai nó vào các ứng dụng thực tế như hệ thống an ninh, quản lý ra vào, hoặc các dịch vụ công nghệ như ứng dụng di động để cộng đồng có thể dễ dàng sử dụng.

II. Nguồn cung cấp dữ liệu

Nhóm sử dụng dữ liệu hình ảnh người mang khẩu khang và không mang khẩu trang. Tất cả dữ liệu đó được lấy dưới đường link sau:

https://data-flair.training/blogs/download-face-mask-data/#google_vignette

III. Thực hiện đồ án

1. Import các thư viện cần thiết:

```
import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.data import Dataset, DataLoader, random_split
import numpy as np
import os
from PIL import Image
from torchvision import transforms
import matplotlib.pyplot as plt
```

✓ 0.0s Python

Hình 3: Import thư viện cần thiết

2. Load và chuẩn bị dữ liệu để huấn luyện

Tạo lớp “MaskDataset” để load dữ liệu:

```python
# Bước 1: Load và chuẩn bị data
class MaskDataset(Dataset):
    def __init__(self, data_dir, transform=None):
        self.data_dir = data_dir
        self.transform = transform
        self.images = []
        self.labels = []

        for label, folder in enumerate(['mask', 'no_mask']):
            folder_path = os.path.join(data_dir, folder)
            for img_name in os.listdir(folder_path):
                img_path = os.path.join(folder_path, img_name)
                self.images.append(img_path)
                self.labels.append(label)

    def __len__(self):
        return len(self.images)

    def __getitem__(self, idx):
        img_path = self.images[idx]
        image = Image.open(img_path).convert('RGB')
        label = self.labels[idx]

        if self.transform:
            image = self.transform(image)

        return image, label
```

✓ 0.0s Python

Hình 4: Code khởi tạo dataset, lấy kích thước, phần tử của dataset

```python
# Đường dẫn đến thư mục chứa dữ liệu
data_dir = 'Dataset/train'
```

✓ 0.0s Python

Hình 5: Tạo đường dẫn đến thư mục chứa dữ liệu

```python
# Định nghĩa các phép biến đổi cho dữ liệu ảnh
transform = transforms.Compose([
    transforms.Resize((150, 150)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])
```

✓ 0.0s Python

Hình 6: Biến đổi hình ảnh về cùng một thông số

```python
# Tạo dataset
dataset = MaskDataset(data_dir, transform=transform)
```

✓ 0.0s Python

Hình 7: Đọc dữ liệu từ thư mục, áp dụng biến đổi

```python
# Bước 2: Chia tập dữ liệu thành tập huấn luyện và tập kiểm tra bằng cách chia
train_size = int(0.7 * len(dataset))
test_size = len(dataset) - train_size
train_dataset, test_dataset = random_split(dataset, [train_size, test_size])
```

✓ 0.0s Python

Hình 8: Chia dữ liệu thành tập huấn luyện và tập kiểm tra

```python
# Tạo DataLoader
batch_size = 49
train_loader = DataLoader(train_dataset, batch_size=batch_size, shuffle=True, drop_last=True)
test_loader = DataLoader(test_dataset, batch_size=batch_size, shuffle=False, drop_last=True)
```

✓ 0.0s

Hình 9: Tạo DataLoader cho tập huấn luyện và kiểm tra

```python
# Kiểm tra các số lượng của tập dữ liệu
print("Số lượng hình ảnh trong tập huấn luyện:", len(train_dataset))
print("Số lượng hình ảnh trong tập kiểm tra:", len(test_dataset))
```

✓ 0.0s Python

Hình 10: Kiểm tra các số lượng của tập dữ liệu

3. Hiển thị một số hình ảnh dữ liệu mẫu

```python
# Lấy một số mẫu ngẫu nhiên từ DataLoader
sample_loader = DataLoader(train_dataset, batch_size=16, shuffle=True)
data_iter = iter(sample_loader)
images, labels = next(data_iter)

# Chuẩn bị hiển thị các hình ảnh
fig, axes = plt.subplots(2, 8, figsize=(16, 6))

# Hiển thị từng hình ảnh với nhãn tương ứng
for idx, image in enumerate(images):
    row = idx // 8
    col = idx % 8

    # Chuyển đổi tensor về numpy array và điều chuẩn lại để hiển thị đúng
    image = image.permute(1, 2, 0).numpy()
    mean = np.array([0.485, 0.456, 0.406])
    std = np.array([0.229, 0.224, 0.225])
    image = std * image + mean
    image = np.clip(image, 0, 1)

    axes[row, col].imshow(image)
    axes[row, col].set_title(f'Pic: {labels[idx].item()}')
    axes[row, col].axis('off')

plt.tight_layout()
plt.show()
```

✓ 0.8s Python

Hình 11: Hiển thị dữ liệu mẫu

4. Xác định mô hình CNN

```python
# Xác định mô hình CNN
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Conv2d(3, 32, 3, 1)
        self.conv2 = nn.Conv2d(32, 64, 3, 1)
        self.conv3 = nn.Conv2d(64, 128, 3, 1)
        self.conv4 = nn.Conv2d(128, 128, 3, 1)
        self.dropout = nn.Dropout(0.5)
        self.fc1 = nn.Linear(128 * 7 * 7, 512)
        self.fc2 = nn.Linear(512, 1)

    def forward(self, x):
        x = torch.relu(self.conv1(x))
        x = torch.max_pool2d(x, 2)
        x = torch.relu(self.conv2(x))
        x = torch.max_pool2d(x, 2)
        x = torch.relu(self.conv3(x))
        x = torch.max_pool2d(x, 2)
        x = torch.relu(self.conv4(x))
        x = torch.max_pool2d(x, 2)
        x = x.view(-1, 128 * 7 * 7)
        x = self.dropout(x)
        x = torch.relu(self.fc1(x))
        x = torch.sigmoid(self.fc2(x))
        return x
```

✓ 0.0s Python

Hình 12: Xây dựng mô hình CNN

5. Huấn luyện mô hình

```python
model = CNN().to(device)
criterion = nn.BCELoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)
```

✓ 0.0s Python

Hình 13: Huấn luyện mô hình

```python
# Huấn luyện mô hình
n_epochs = 50
train_losses = []
test_losses = []
test_accuracies = []

for epoch in range(n_epochs):
    model.train()
    train_loss = 0.0
    for inputs, targets in train_loader:
        inputs, targets = inputs.to(device), targets.to(device)
        optimizer.zero_grad()
        outputs = model(inputs)
        targets = targets.unsqueeze(1).float()
        loss = criterion(outputs, targets)
        loss.backward()
        optimizer.step()
        train_loss += loss.item() * inputs.size(0)

    train_loss /= len(train_loader.dataset)
    train_losses.append(train_loss)

    model.eval()
    test_loss = 0.0
    correct = 0
    total = 0
    with torch.no_grad():
        for inputs, targets in test_loader:
            inputs, targets = inputs.to(device), targets.to(device)
            outputs = model(inputs)
            targets = targets.unsqueeze(1).float()
            loss = criterion(outputs, targets)
            test_loss += loss.item() * inputs.size(0)
```

Hình 14: Huấn luyện mô hình

```python
model.eval()
test_loss = 0.0
correct = 0
total = 0
with torch.no_grad():
    for inputs, targets in test_loader:
        inputs, targets = inputs.to(device), targets.to(device)
        outputs = model(inputs)
        targets = targets.unsqueeze(1).float()
        loss = criterion(outputs, targets)
        test_loss += loss.item() * inputs.size(0)

        predicted = (outputs >= 0.5).float()
        total += targets.size(0)
        correct += (predicted == targets).sum().item()

test_loss /= len(test_loader.dataset)
test_losses.append(test_loss)

accuracy = correct / total
test_accuracies.append(accuracy)

print(f'Epoch {epoch+1}/{n_epochs}, Train Loss: {train_loss:.4f}, Test Loss: {test_loss:.4f}, Test Accuracy: {accuracy:.4f}')
```

✓ 9m 12.7s

Hình 15: Huấn luyện mô hình

6. Vẽ biểu đồ loss và accuracy

```
# Vẽ biểu đồ loss và accuracyx
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(14, 6))

ax1.plot(train_losses, label='Train Loss')
ax1.plot(test_losses, label='Test Loss')
ax1.set_title('Loss vs Epochs')
ax1.set_xlabel('Epoch')
ax1.set_ylabel('Loss')
ax1.legend()

ax2.plot(test_accuracies, label='Test Accuracy')
ax2.set_title('Accuracy vs Epochs')
ax2.set_xlabel('Epoch')
ax2.set_ylabel('Accuracy')
ax2.legend()

plt.show()
```

✓ 0.3s

Hình 16: Vẽ biểu đồ loss và accuracy

7. Kiểm tra dự đoán

```
# Kiểm tra dự đoán của mô hình trên hình ảnh mới
def predict_image(image_path, model, transform):
    image = Image.open(image_path).convert('RGB')
    image = transform(image).unsqueeze(0).to(device)
    model.eval()
    with torch.no_grad():
        output = model(image)
        prediction = output.item()
        return prediction >= 0.5, prediction
```

✓ 0.0s

Hình 17: Kiểm tra dự đoán mô hình

8. Thử dự đoán trên hình ảnh mới

```python
# Đường dẫn đến ảnh mới
test_image_paths = ['Test/image2.jpg', 'Test/image1.jpg']
```
✓ 0.0s

```python
# Tải mô hình đã lưu
model = CNN().to(device)
model.load_state_dict(torch.load('mask_detection_model.pth'))

fig, axes = plt.subplots(1, len(test_image_paths), figsize=(12, 6))
for idx, image_path in enumerate(test_image_paths):
    has_mask, prob = predict_image(image_path, model, transform)
    image = Image.open(image_path).convert('RGB')

    axes[idx].imshow(image)
    axes[idx].set_title(f'No Mask: {has_mask}\nProb: {prob:.2f}')
    axes[idx].axis('off')

plt.tight_layout()
plt.show()
```
✓ 1.0s

Hình 18: Nhận diện khuôn mặt

IV. Kết quả thực hiện và nhận xét

1. Kết quả huấn luyện:

```
Epoch 1/30, Train Loss: 0.6243, Test Loss: 0.5579, Test Accuracy: 0.6939
Epoch 2/30, Train Loss: 0.4086, Test Loss: 0.2422, Test Accuracy: 0.8929
Epoch 3/30, Train Loss: 0.1745, Test Loss: 0.1571, Test Accuracy: 0.9541
Epoch 4/30, Train Loss: 0.1261, Test Loss: 0.2699, Test Accuracy: 0.8980
Epoch 5/30, Train Loss: 0.0970, Test Loss: 0.2087, Test Accuracy: 0.9337
Epoch 6/30, Train Loss: 0.1193, Test Loss: 0.1426, Test Accuracy: 0.9439
Epoch 7/30, Train Loss: 0.0876, Test Loss: 0.1360, Test Accuracy: 0.9592
Epoch 8/30, Train Loss: 0.0673, Test Loss: 0.1355, Test Accuracy: 0.9592
Epoch 9/30, Train Loss: 0.0518, Test Loss: 0.1285, Test Accuracy: 0.9541
Epoch 10/30, Train Loss: 0.0572, Test Loss: 0.1131, Test Accuracy: 0.9643
Epoch 11/30, Train Loss: 0.0705, Test Loss: 0.1323, Test Accuracy: 0.9541
Epoch 12/30, Train Loss: 0.0533, Test Loss: 0.1327, Test Accuracy: 0.9541
Epoch 13/30, Train Loss: 0.0333, Test Loss: 0.1519, Test Accuracy: 0.9643
Epoch 14/30, Train Loss: 0.0177, Test Loss: 0.1272, Test Accuracy: 0.9592
Epoch 15/30, Train Loss: 0.0177, Test Loss: 0.1443, Test Accuracy: 0.9694
Epoch 16/30, Train Loss: 0.0214, Test Loss: 0.1159, Test Accuracy: 0.9643
Epoch 17/30, Train Loss: 0.0201, Test Loss: 0.1472, Test Accuracy: 0.9592
Epoch 18/30, Train Loss: 0.0201, Test Loss: 0.1488, Test Accuracy: 0.9694
Epoch 19/30, Train Loss: 0.0125, Test Loss: 0.1603, Test Accuracy: 0.9643
Epoch 20/30, Train Loss: 0.0065, Test Loss: 0.1721, Test Accuracy: 0.9592
Epoch 21/30, Train Loss: 0.0040, Test Loss: 0.2386, Test Accuracy: 0.9592
Epoch 22/30, Train Loss: 0.0100, Test Loss: 0.2216, Test Accuracy: 0.9592
Epoch 23/30, Train Loss: 0.0157, Test Loss: 0.2023, Test Accuracy: 0.9694
Epoch 24/30, Train Loss: 0.0813, Test Loss: 0.1721, Test Accuracy: 0.9694
Epoch 25/30, Train Loss: 0.0445, Test Loss: 0.1253, Test Accuracy: 0.9694
...
Epoch 27/30, Train Loss: 0.0144, Test Loss: 0.2179, Test Accuracy: 0.9643
Epoch 28/30, Train Loss: 0.0170, Test Loss: 0.1958, Test Accuracy: 0.9643
Epoch 29/30, Train Loss: 0.0137, Test Loss: 0.1381, Test Accuracy: 0.9694
Epoch 30/30, Train Loss: 0.0046, Test Loss: 0.1569, Test Accuracy: 0.9694
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Hình 19: Kết quả huấn luyện

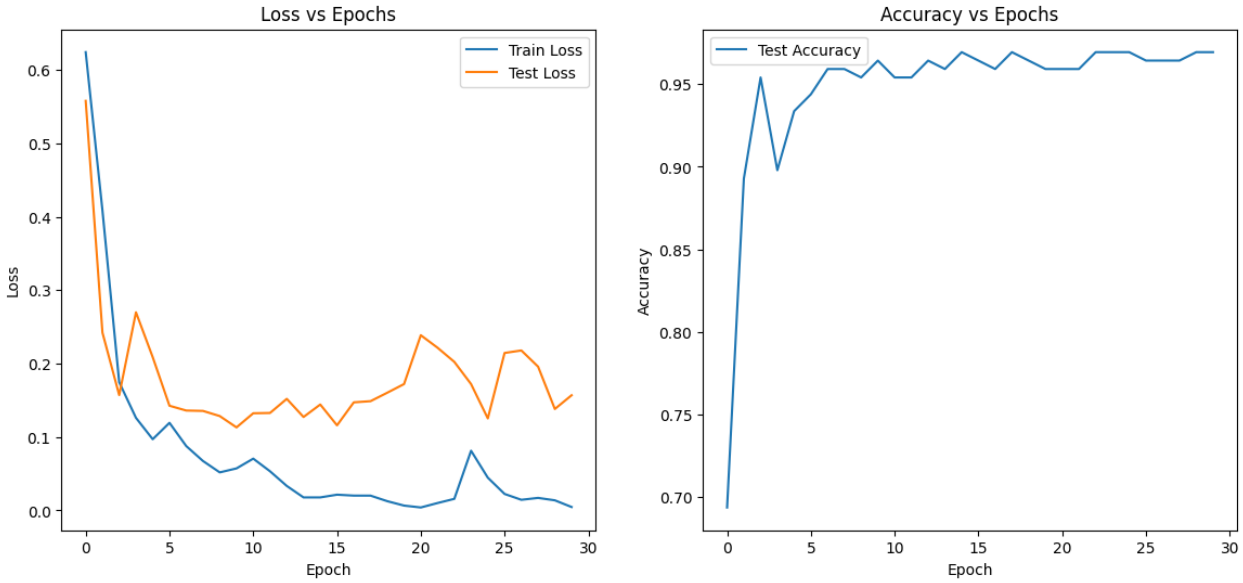
Nhận xét về kết quả theo từng epoch:

- ***Epoch đầu tiên:*** Train Loss: 0.6243, Test Loss: 0.5579, Test Accuracy: 0.6939. cả Train Loss và Test Loss đều cao, và độ chính xác của mô hình trên tập kiểm tra (Test Accuracy) chỉ đạt 69.39%. Điều này cho thấy mô hình vẫn còn nhiều sai số trên dữ liệu huấn luyện
- ***Epoch tiếp theo (2-10):*** Train Loss giảm đáng kể, từ 0.6243 xuống 0.0572, Test Loss cũng giảm từ 0.5579 xuống 0.1131, Test Accuracy tăng từ 0.6939 lên 0.9643. Cho thấy rằng mô hình học nhanh và cải thiện đáng kể trong những epoch đầu. Sự giảm mạnh về loss và tăng về accuracy cho thấy mô hình đang học hiệu quả từ dữ liệu huấn luyện.
- ***Từ epoch 11 đến 30:*** Train Loss dao động quanh các giá trị rất thấp, từ 0.0040 đến 0.0813. Test Loss có dao động nhưng vẫn giữ ở mức thấp, dao động từ 0.1159 đến 0.2386. Test Accuracy duy trì mức cao, từ 0.9541 đến 0.9694. Mô hình tiếp tục học và duy trì độ chính xác cao. Tuy nhiên, có một vài epoch mà Test Loss tăng cao hơn các epoch khác (ví dụ: epoch 21 với Test Loss = 0.2386).

Nhận xét chung:

- ***Hiệu suất mô hình:*** Mô hình đạt được độ chính xác cao trên tập kiểm tra (Test Accuracy), dao động từ 95.41% đến 96.94% sau epoch 10. Sự giảm mạnh của Train Loss và Test Loss trong những epoch đầu cho thấy mô hình học hiệu quả.
- ***Overfitting:*** Sau epoch 10, Train Loss tiếp tục giảm mạnh trong khi Test Loss dao động, cho thấy mô hình có thể đang bị overfitting. Mô hình quá khớp với dữ liệu huấn luyện nhưng không tổng quát tốt trên dữ liệu kiểm tra.

Kết quả vẽ biểu đồ:



Hình 20: Kết quả vẽ biểu đồ

Nhận xét:

- Biểu đồ Loss vs Epochs:
  - Training loss bắt đầu cao (khoảng 0.6243) và giảm nhanh chóng trong những epoch đầu tiên. Đến epoch 5, nó đã giảm xuống khoảng 0.0970. Sau đó, training loss tiếp tục giảm dần và ổn định ở mức thấp, quanh mốc 0.0040 đến 0.0201 trong các epoch sau. Điều này cho thấy mô hình đang học khá tốt từ dữ liệu huấn luyện.
  - Test loss cũng bắt đầu cao (khoảng 0.5579) và giảm nhanh trong các epoch đầu tiên, nhưng không đều như training loss. Nó có xu hướng dao động sau khoảng epoch 10, tăng và giảm không ổn định nhưng nhìn chung duy trì ở mức dưới 0.3. Dù dao động nhưng xu hướng giảm và duy trì ở mức thấp cho thấy mô hình có khả năng tổng quát hóa tốt trên dữ liệu kiểm tra
- Biểu đồ Accuracy vs Epochs:
  - Test accuracy tăng nhanh trong vài epoch đầu, từ khoảng 0.6939 ở epoch 1 lên khoảng 0.9541 ở epoch 3. Sau đó, nó tiếp tục tăng nhẹ và dao động quanh mức 0.9643 đến 0.9694 từ epoch 10 trở đi. Test accuracy tăng nhanh chóng và duy trì ở mức cao (>95%) cho thấy mô hình có độ chính xác cao trong việc phân loại dữ liệu kiểm tra.

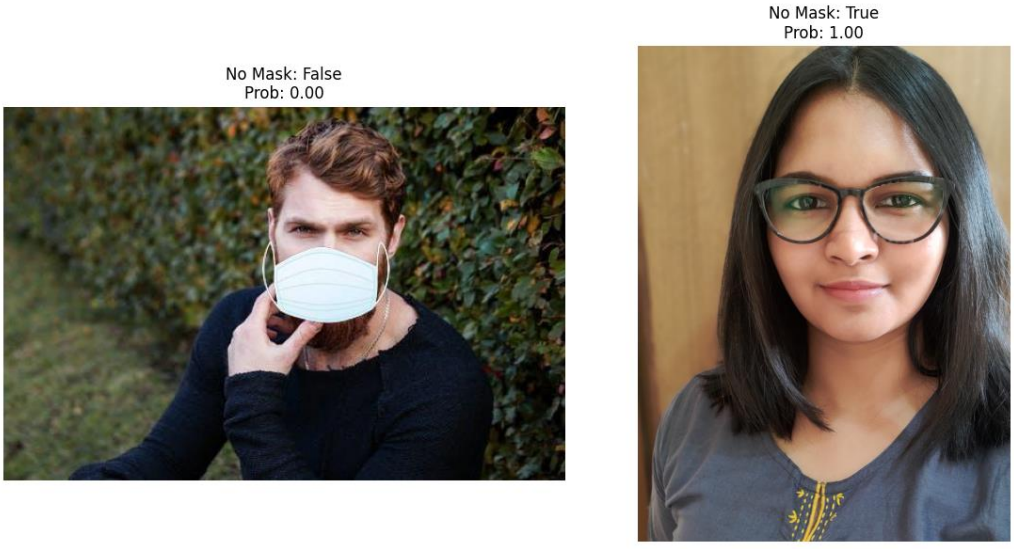
Kết luận: Mô hình hiện tại đạt hiệu suất khá cao nhưng có dấu hiệu overfitting. Nên áp dụng các kỹ thuật regularization, tăng cường dữ liệu, và dừng sớm có thể giúp cải thiện hiệu suất và tính tổng quát của mô hình trên dữ liệu mới.

2. Kết quả nhận diện

Thử nhận diện 2 hình ảnh cho vào thư mục Test, 1 hình ảnh người có mang khẩu trang và 1 hình ảnh người không mang khẩu trang như sau:

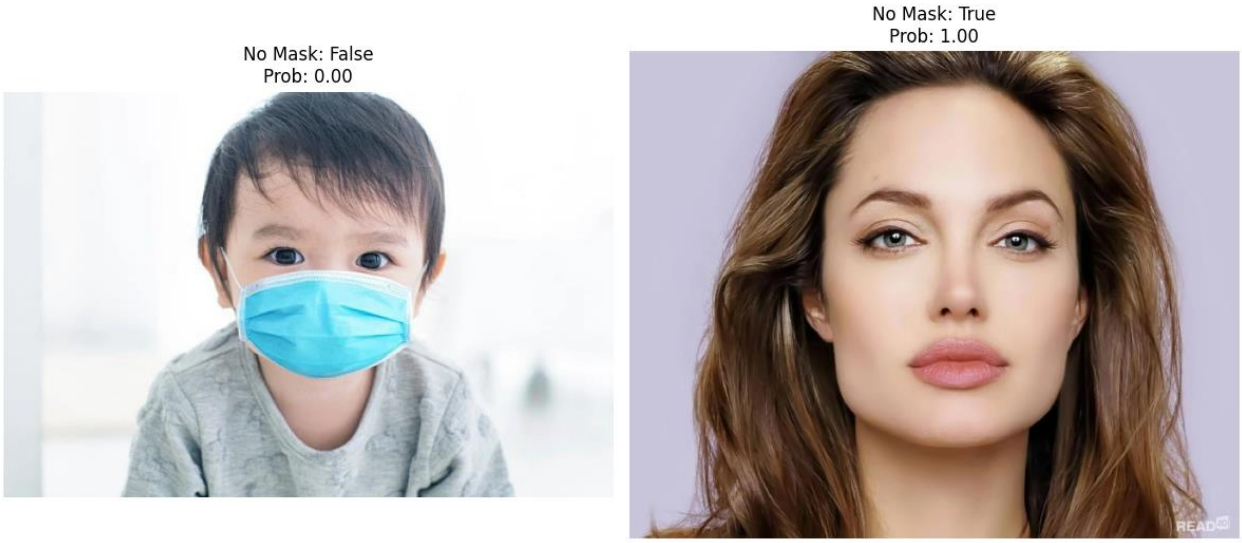
Hình 21: Hình ảnh nhận diện

Kết quả chương trình đã nhận diện được mặt đeo khẩu trang với Prob: là 1.00, mặt còn lại là không có khẩu trang với Prob 0.00:



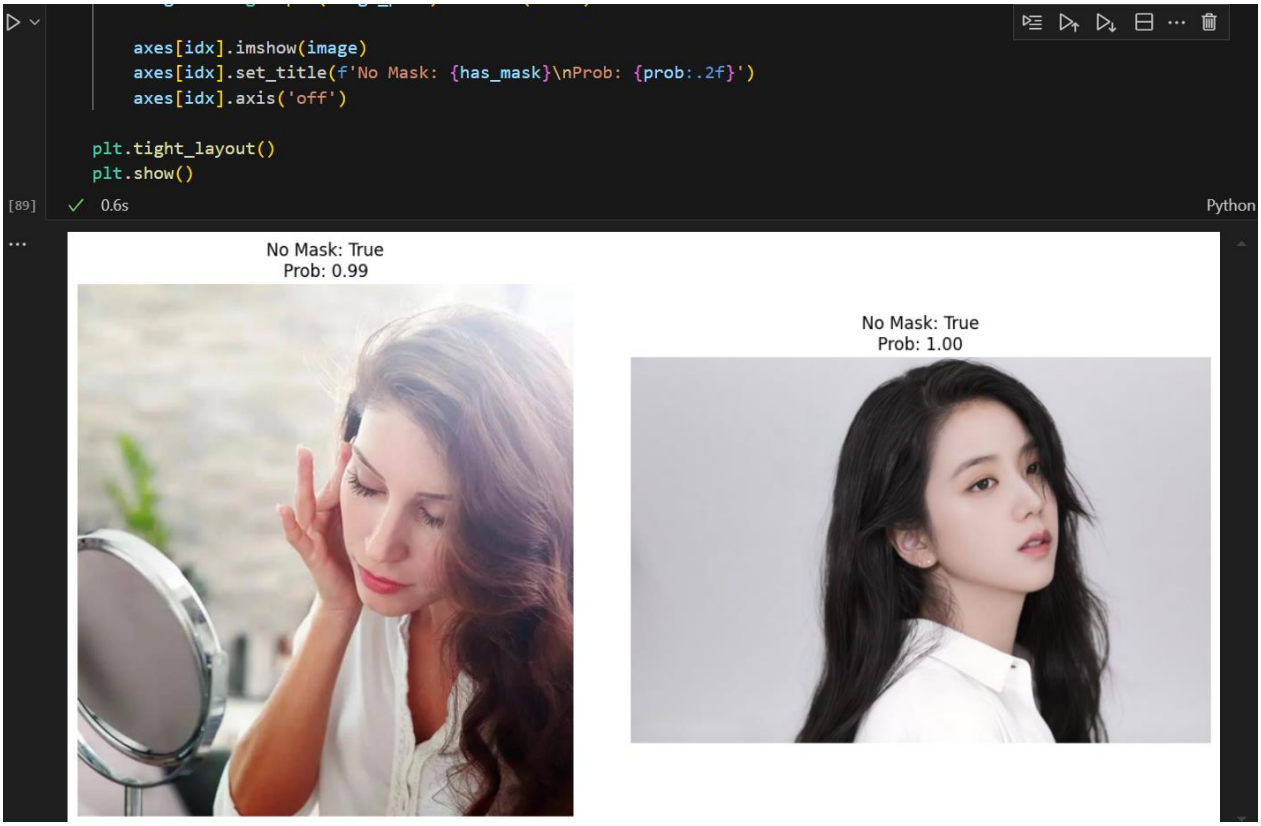
Hình 22: Kết quả nhận diện

Thử nhận diện các khuôn mặt tải về trên trang Google (các khuôn mặt mơi skhoong có trong tập dữ liệu train) thì kết quả nhận diện vẫn đúng:



Hình 23: Kết quả nhận diện

Thử cho chương trình nhận diện với 2 khuôn mặt đều không mang khẩu trang được tải về từ Google và từ các góc độ khác nhau, chương trình vẫn nhận diện được khuôn mặt không mang khẩu trang:



Hình 24: Kết quả nhận diện

Cho 2 tệp hình ảnh có nhiều người với các góc độ chụp khác thì chương trình vẫn nhận diện được:

Hình 25: Kết quả nhận diện

Kết quả nhận dạng người có mang khẩu trang hay không có tỉ lệ chính xác rất cao. Chương trình có thể nhận diện được khuôn mặt có mang khẩu trnag hay không từ các góc độ khác nhau chứ không bắt buộc phải góc chính diện. Điều đó cho thấy chương trình đã đạt được một hiệu quả cao trong quá trình xây dựng và ra kết quả.

Trong tương lai, nếu có cơ hội, nhóm sẽ cố gắng phát triển dự án này hơn với dữ liệu được áp dụng các phép biến đổi như xoay, cắt, lật, thay đổi độ sáng, và thêm nhiễu vào ảnh để làm phong phú thêm tập dữ liệu, sát với thực tế. Chương trình này rất có triển vọng trong tương lai, có thể ứng dụng tích hợp mô hình vào các hệ thống giám sát video để tự động phát hiện người đeo khẩu trang tại các khu vực công cộng, bệnh viện, hoặc nơi làm việc.

Kết quả và quá trình thực hiện đồ án được lưu trên link github:
https://github.com/tramy28/Recognize-Faces-Wearing-Masks-G2/tree/master
Nhiệm vụ và quá trình làm việc của mỗi thành viên trong đồ án:
https://github.com/users/tramy28cts/2/views/1

KẾT LUẬN VÀ ĐỀ XUẤT

Dự án nhận dạng đeo và không đeo khẩu trang bằng Deep Learning đã chứng minh được tính khả thi và hiệu quả của công nghệ này trong việc giải quyết các vấn đề thực tế. Mô hình đã đạt được độ chính xác cao trong việc phân loại hình ảnh, giúp tăng cường khả năng giám sát và đảm bảo an toàn trong cộng đồng.

Trong quá trình nghiên cứu và phát triển hệ thống này, nhóm đã nhận thấy được sự phức tạp và đa dạng của việc giám sát việc đeo khẩu trang trong nhiều bối cảnh khác nhau. Việc áp dụng công nghệ Deep Learning và quản trị thông tin số đã mở ra những cánh cửa mới, cho phép không chỉ nhóm mà cả người dùng có thể dễ dàng tiếp cận và hiểu sâu hơn về các yếu tố ảnh hưởng đến việc nhận dạng và giám sát này.

Đồ án đã phát triển một mô hình nhận dạng sử dụng các thuật toán học máy và phân tích dữ liệu, giúp đưa ra các dự đoán chính xác về việc đeo khẩu trang với độ chính xác tương đối cao. Tuy nhiên, việc nhận dạng đeo khẩu trang luôn là một thách thức, và có nhiều yếu tố khác nhau có thể ảnh hưởng đến kết quả nhận dạng mà chưa thể khai triển hoàn toàn, nên vẫn chưa thể đạt được độ chính xác 100%.

Dựa trên đồ án này, nhóm đã đề xuất một số hướng phát triển tiếp theo để cải thiện và mở rộng hiệu suất của hệ thống nhận dạng đeo khẩu trang như: Tăng cường dữ liệu huấn luyện: Tiếp tục thu thập và tích hợp nhiều dữ liệu đa dạng hơn từ các nguồn khác nhau, bao gồm các tình huống thực tế và các yếu tố môi trường. Ngoài ra, cần phải hiểu rõ hơn về các yếu tố ảnh hưởng đến việc nhận dạng, đặc biệt là trong những tình huống phức tạp. Sử dụng các phương pháp tiên tiến hơn trong học máy và trí tuệ nhân tạo: Áp dụng các mô hình và thuật toán mới để cải thiện độ chính xác của hệ thống nhận dạng. Sau khi thực hiện các sửa đổi, nhóm sẽ cố gắng đưa hệ thống vào ứng dụng thực tế để hỗ trợ các cơ quan và tổ chức trong việc giám sát và đảm bảo an toàn cộng đồng.

Nhóm chúng tôi tin rằng việc tiếp tục nghiên cứu và phát triển trong những hướng này sẽ cung cấp những cơ hội mới để cải thiện việc giám sát và nhận dạng đeo khẩu trang, góp phần vào cuộc chiến chống lại đại dịch và bảo vệ sức khỏe cộng đồng không chỉ trong bối cảnh hiện tại mà còn trong tương lai.

BẢNG ĐÁNH GIÁ THÀNH VIÊN

STT	MSSV	Họ và tên	Công việc	Đánh giá đóng góp theo thang 10
1	2174802010891	Thái Thị Trà My	Hoàn thiện code, chỉnh sửa file word	10
2	2174802010785	Lâm Nguyễn Anh Thư	Code nhận diện, nhận xét kết quả, file ppt	10
3	2174802010886	Võ Quốc Thành	Tìm kiếm nội dung, hoàn thiện chương 1	9
4	2174802010906	Lê Chí Thiện	Tìm kiếm nội dung, hoàn thiện chương 2	9
5	2174802010934	Nguyễn Minh Huy	Tìm kiếm dữ liệu, source code demo	10