

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN 1



BÁO CÁO BÀI TẬP LỚN
HỌC PHẦN: THỰC TẬP CƠ SỞ

ĐỀ TÀI: NHẬN DIỆN CẢM XÚC KHUÔN MẶT
THEO THỜI GIAN THỰC

Giảng viên hướng dẫn: Đào Ngọc Phong

Sinh viên

Mã sinh viên

Đặng Thị Hà

B22DCCN253

Nguyễn Thị Hiền

B22DCCN289

Hà Nội - 5/2025

LỜI CẢM ƠN

Trước tiên, em xin bày tỏ lòng biết ơn sâu sắc đến thầy **Đào Ngọc Phong**, giảng viên Khoa Công nghệ Thông tin – Học viện Công nghệ Bưu chính Viễn thông, người đã trực tiếp hướng dẫn em trong quá trình thực hiện môn **Thực tập cơ sở**.

Trong suốt quá trình học tập và thực hiện nội dung thực tập, chúng em luôn nhận được sự quan tâm, chỉ bảo tận tình, cẩn thận và nghiêm túc từ phía thầy. Thầy không chỉ giúp chúng em định hướng phương pháp tiếp cận vấn đề một cách khoa học, mà còn luôn khuyến khích chúng em phát huy tinh thần chủ động, sáng tạo và tự học trong nghiên cứu. Những góp ý quý báu từ thầy đã giúp chúng em có thêm kiến thức chuyên môn, đồng thời rèn luyện được tư duy logic, kỹ năng làm việc độc lập và khả năng giải quyết vấn đề – những yếu tố vô cùng quan trọng trong lĩnh vực Công nghệ thông tin.

Tuy đã cố gắng hết sức, nhưng do thời gian thực hiện có hạn và năng lực của bản thân còn nhiều hạn chế, dự án thực tập không thể tránh khỏi những thiếu sót nhất định. Chúng em rất mong tiếp tục nhận được sự góp ý, chỉ bảo từ thầy để chúng em có thể hoàn thiện hơn trong các học phần và dự án tiếp theo.

Một lần nữa, em xin chân thành cảm ơn thầy và kính chúc thầy cùng gia đình dồi dào sức khỏe, hạnh phúc và thành công trong công việc cũng như trong cuộc sống.

Chúng em xin chân thành cảm ơn!

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Điểm:(bằng chữ:).....

Hà Nội, ngày ... tháng 5 năm 2025

CÁN BỘ - GIẢNG VIÊN HƯỚNG DẪN

(ký, họ tên)

MỤC LỤC

LỜI CẢM ƠN	2
NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN	3
DANH MỤC CÁC BẢNG BIỂU	6
DANH MỤC CÁC HÌNH ẢNH	6
DANH MỤC CÁC TỪ VIẾT TẮT	6
CHƯƠNG 1: TỔNG QUAN HỆ THỐNG VÀ CÁC CÔNG NGHỆ SỬ DỤNG	8
I. TỔNG QUAN HỆ THỐNG	8
1. Mục tiêu và ý nghĩa	8
2. Phạm vi	8
3. Bài toán cần giải quyết.....	9
4. Phương hướng giải quyết bài toán.....	10
5. Các nghiên cứu liên quan.....	10
II. GIỚI THIỆU CÁC CÔNG NGHỆ SỬ DỤNG TRONG HỆ THỐNG ..	12
1. Backend:	12
2. Frontend	17
3. Machine Learning/Deep Learning:.....	18
4. Công cụ phát triển:.....	26
5. Các thành phần khác:.....	27
CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG.....	29
I. LẤY YÊU CẦU HỆ THỐNG	29
1. Mô tả hệ thống bằng ngôn ngữ tự nhiên.....	29
2. Xác định và mô tả các tác nhân	29
3. Xác nhận và mô tả các use case sử dụng	29
II. PHÂN TÍCH HỆ THỐNG	31
1. Scenario	31

III. THIẾT KẾ HỆ THỐNG	33
1. Mô hình kiến trúc hệ thống	33
CHƯƠNG 3: CÀI ĐẶT CHƯƠNG TRÌNH.....	35
I. CÀI ĐẶT HỆ THỐNG VÀ TRIỂN KHAI HỆ THỐNG.....	35
1. Một số công cụ sử dụng.	35
2. Thư viện hỗ trợ.....	35
3. Cài đặt và triển khai phía Backend.....	35
4. Cài đặt và triển khai phía FE	36
II. KẾT QUẢ CÀI ĐẶT	37
III. KẾT LUẬN	37
1. Kết quả đạt được.....	37
a) Các giao diện chính:	37
PHẦN KẾT LUẬN	40
1. Tóm tắt các nội dung đã thực hiện.....	40
2. Những điểm đạt được và hạn chế của hệ thống.....	40
3. Hướng phát triển trong tương lai.	41
TÀI LIỆU THAM KHẢO.....	42

DANH MỤC CÁC BẢNG BIỂU

Bảng 1: Xác định các tác nhân.....	29
Bảng 2: Xác định và mô tả các use case	30
Bảng 3: Kịch bản Nhận diện cảm xúc theo thời gian thực	31
Bảng 4: Kịch bản Nhận diện cảm xúc từ ảnh tải lên	31
Bảng 5: Kịch bản Nhận diện cảm xúc từ ảnh chụp mới	32
Bảng 6: Kịch bản Điều chỉnh chế độ sáng/tối của giao diện	33

DANH MỤC CÁC HÌNH ẢNH

Hình 1: Kiến trúc mô hình CNN.....	19
Hình 2 :Đồ thị của hàm ReLU	20
Hình 3: Pooling Layer.....	20
Hình 4: Sự khác nhau giữa các tầng trong CNN.....	21
Hình 5: Kiến trúc mô hình VGG19.....	22
Hình 6: Tổng quan use case toàn hệ thống	30
Hình 7: Mô hình kiến trúc hệ thống.....	34
Hình 8: Cấu trúc thư mục phần backend.....	37
Hình 9: Cấu trúc thư mục phần frontend	37
Hình 10: Giao diện chính khi ở chế độ Tối.....	38
Hình 11: Giao diện chính khi ở chế độ sáng.....	38
Hình 12: Giao diện đã tải ảnh lên và hiện kết quả cảm xúc.....	39

DANH MỤC CÁC TỪ VIẾT TẮT

Từ viết tắt	Ý nghĩa
CSS	Cascading Style Sheet
HTML	HyperText Markup Language
AI	Artificial Intelligence

API	Application Programming Interface
DOM	Document Object Model
UI	User Interface
HTTP	HyperText Transfer Protocol
GPU	Graphics Processing Unit
TPU	Tensor Processing Unit
CPU	Central Processing Unit
CNN	Convolutional Neural Network
ReLU	Rectified Linear Unit
VGG	Visual Geometry Group
PCA	Principal Component Analysis
LDA	Linear Discriminant Analysis
SVM	Support Vector Machine
LinearSVC	Linear Support Vector Classification
CK	Cohn-Kanade
CK+	Cohn-Kanade Extended
IMM	IMM Face Database
FER-2013	Facial Expression Recognition 2013
JAFPE	Japanese Female Facial Expression
VGG19	Visual Geometry Group 19
ResNet50	Residual Network 50
SGD	Stochastic Gradient Descent
REST API	Representational State Transfer Application Programming Interface

CHƯƠNG 1: TỔNG QUAN HỆ THỐNG VÀ CÁC CÔNG NGHỆ SỬ DỤNG

I. TỔNG QUAN HỆ THỐNG

1. Mục tiêu và ý nghĩa

Hệ thống nhận diện cảm xúc từ khuôn mặt là một ứng dụng tiên tiến kết hợp giữa công nghệ học sâu và xử lý ảnh để phân tích và nhận diện cảm xúc của con người thông qua hình ảnh hoặc video. Trong bối cảnh công nghệ ngày càng phát triển, việc hiểu và phân tích cảm xúc của con người thông qua các biểu hiện khuôn mặt đã trở thành một lĩnh vực nghiên cứu quan trọng. Hệ thống này được xây dựng trên nền tảng FastAPI, một framework web hiệu suất cao của Python, cho phép triển khai các ứng dụng web API bất đồng bộ với tốc độ nhanh và hiệu quả. Bằng cách sử dụng các thư viện mạnh mẽ như OpenCV và TensorFlow/Keras, kết hợp với mô hình VGG19 qua transfer learning, hệ thống có khả năng xử lý ảnh và phân tích cảm xúc theo thời gian thực với độ chính xác cao.

Mục tiêu chính của dự án là phát triển một hệ thống có khả năng nhận diện chính xác 7 cảm xúc cơ bản của con người (Giận dữ, Ghê tởm, Sợ hãi, Vui vẻ, Buồn bã, Ngạc nhiên và Trung tính) từ hình ảnh khuôn mặt. Hệ thống cung cấp nhiều phương thức đầu vào linh hoạt, bao gồm video stream trực tiếp từ camera, tải lên ảnh từ thiết bị, hoặc chụp ảnh mới thông qua giao diện web. Điều này không chỉ giúp cải thiện trải nghiệm người dùng trong các ứng dụng tương tác mà còn mở ra nhiều cơ hội ứng dụng trong các lĩnh vực như giáo dục, chăm sóc sức khỏe, và dịch vụ khách hàng.

Ý nghĩa của dự án nằm ở việc ứng dụng công nghệ tiên tiến để tạo ra các giải pháp thông minh, góp phần nâng cao chất lượng cuộc sống và tối ưu hóa các quy trình làm việc. Với kiến trúc REST API và khả năng xử lý bất đồng bộ, hệ thống này có thể dễ dàng tích hợp vào các ứng dụng khác để cải thiện giao tiếp giữa con người và máy tính, giúp các hệ thống tự động hiểu và phản hồi theo cách tự nhiên hơn. Giao diện web responsive và hiện đại với tùy chọn theme sáng/tối cũng nâng cao trải nghiệm người dùng trên nhiều thiết bị khác nhau.

2. Phạm vi

Phạm vi của dự án bao gồm việc phát triển một ứng dụng web sử dụng FastAPI có khả năng phát hiện khuôn mặt từ video stream hoặc hình ảnh tĩnh, phân tích và dự đoán 7 loại cảm xúc cơ bản từ khuôn mặt được phát hiện. Hệ thống hiển thị kết quả

nhận diện cảm xúc theo thời gian thực trên giao diện web thông qua streaming response, và cung cấp các API endpoints (/camera/{action}, /video_feed, /detect_emotion) để điều khiển camera, nhận video stream và phân tích ảnh tải lên.

Dự án sử dụng thư viện OpenCV để phát hiện khuôn mặt thông qua **Haar Cascade Classifier** và tích hợp mô hình deep learning **VGG19** đã được transfer learning để nhận diện cảm xúc. Mô hình này được tải từ file trọng số (model_weights.h5) và xử lý các vùng khuôn mặt đã được trích xuất để đưa ra dự đoán. Kết quả được hiển thị trực quan thông qua video có ghi chú và biểu đồ xác suất cảm xúc.

Giao diện người dùng responsive được thiết kế với **HTML5**, **CSS3**, **JavaScript** và **Bootstrap 5**, hỗ trợ đa dạng phương thức đầu vào: camera trực tiếp, tải lên ảnh từ thiết bị, hoặc chụp ảnh mới thông qua WebRTC. Dự án cũng tối ưu hóa hiệu suất thông qua xử lý bất đồng bộ (async/await) và cơ chế xử lý lỗi toàn diện để đảm bảo hoạt động mượt mà và ổn định trên nhiều thiết bị khác nhau.

3. Bài toán cần giải quyết

3.1. Phát hiện khuôn mặt:

Xác định vị trí khuôn mặt trong hình ảnh hoặc video là bước đầu tiên và quan trọng nhất trong quá trình nhận diện cảm xúc. Điều này đòi hỏi việc sử dụng các kỹ thuật xử lý ảnh tiên tiến để phát hiện khuôn mặt một cách chính xác và nhanh chóng.

3.2. Nhận diện cảm xúc:

Sau khi phát hiện khuôn mặt, hệ thống cần phân loại cảm xúc từ khuôn mặt đã phát hiện. Đây là một bài toán phức tạp đòi hỏi việc sử dụng các mô hình học sâu để phân tích các đặc điểm khuôn mặt và dự đoán cảm xúc tương ứng.

3.3. Tích hợp hệ thống:

Xây dựng giao diện web và API để hiển thị và cung cấp kết quả nhận diện là một phần quan trọng của dự án. Điều này bao gồm việc thiết kế giao diện người dùng thân thiện và dễ sử dụng, cũng như phát triển các API endpoints để tích hợp với các hệ thống khác.

3.4. Tối ưu hóa hiệu suất:

Đảm bảo hệ thống hoạt động hiệu quả và chính xác trong thời gian thực là một thách thức lớn. Điều này đòi hỏi việc tối ưu hóa mã nguồn và cấu trúc hệ thống để giảm thiểu thời gian xử lý và cải thiện độ chính xác của mô hình.

4. Phương hướng giải quyết bài toán

4.1. Sử dụng OpenCV và Haar Cascade:

Để phát hiện khuôn mặt trong hình ảnh và video, hệ thống sử dụng thư viện OpenCV và kỹ thuật Haar Cascade. Đây là một phương pháp phổ biến và hiệu quả để phát hiện khuôn mặt, cho phép hệ thống xác định vị trí khuôn mặt một cách nhanh chóng và chính xác.

4.2. Huấn luyện mô hình học sâu:

Sử dụng TensorFlow/Keras để xây dựng và huấn luyện mô hình mạng nơ-ron tích chập cho việc nhận diện cảm xúc. Mô hình này được huấn luyện trên một tập dữ liệu lớn các hình ảnh khuôn mặt với các nhãn cảm xúc tương ứng, cho phép hệ thống học cách phân loại cảm xúc từ các đặc điểm khuôn mặt.

4.3. Phát triển ứng dụng web với FastAPI:

Tạo giao diện người dùng và API endpoints để tương tác với hệ thống. FastAPI là một framework web nhẹ và linh hoạt, cho phép triển khai các ứng dụng web một cách nhanh chóng và hiệu quả. Giao diện người dùng được thiết kế để hiển thị kết quả nhận diện cảm xúc theo thời gian thực, cung cấp trải nghiệm người dùng tốt nhất.

4.4. Tối ưu hóa và kiểm thử:

Thực hiện các bước tối ưu hóa và kiểm thử để đảm bảo độ chính xác và hiệu suất của hệ thống. Điều này bao gồm việc tối ưu hóa mã nguồn, cải thiện cấu trúc hệ thống, và thực hiện các bài kiểm thử để đánh giá độ chính xác và hiệu suất của mô hình nhận diện cảm xúc.

5. Các nghiên cứu liên quan

Trong bài báo [3], Raghav Puri và các cộng sự đã giới thiệu một hệ thống phát hiện cảm xúc của người dùng dựa trên biểu cảm khuôn mặt, sử dụng kết hợp các kỹ thuật xử lý ảnh và học máy để phân tích hình ảnh. Bằng cách triển khai quy trình bao gồm thu nhận hình ảnh (từ camera hoặc file có sẵn), phát hiện khuôn mặt, chuyển đổi ảnh sang thang độ xám, sau đó áp dụng Phân tích Thành phần Chính (Principal Component Analysis) và so sánh với cơ sở dữ liệu huấn luyện (như Cohn-Kanade), mô hình được đề xuất đã đạt được độ chính xác khoảng 70% trong việc nhận diện tám loại cảm xúc cơ bản (Trung tính, Vui, Giận, Ghê tởm, Ngạc nhiên, Sợ hãi, Buồn, Khinh miệt). Nghiên cứu cũng chỉ ra rằng độ chính xác tối ưu của phương pháp này

có thể lên đến gần 83%, cho thấy tính khả thi và hiệu quả của việc ứng dụng Python, OpenCV và NumPy trong lĩnh vực nhận dạng cảm xúc tự động.

Trong bài báo [4], Rajesh và cộng sự trình bày một hệ thống nhận dạng khuôn mặt và phát hiện cảm xúc theo thời gian thực. Đối với nhận dạng khuôn mặt, hệ thống sử dụng thuật toán Viola-Jones (thông qua Haar-cascades trong OpenCV) để phát hiện khuôn mặt, sau đó áp dụng thuật toán Fisherface, kết hợp Phân tích Thành phần Chính (PCA) để giảm chiều và Phân tích Biệt thức Tuyến tính (LDA) để phân loại và nhận diện người dùng. Trong việc phát hiện cảm xúc, sau khi phát hiện khuôn mặt bằng Haar-cascades, các đặc trưng chi tiết trên khuôn mặt (như mắt, mũi, môi, đường viền) được trích xuất bằng thư viện dlib. PCA tiếp tục được dùng để giảm chiều dữ liệu từ các đặc trưng này, và cuối cùng, bộ phân loại Support Vector Machine (SVM), cụ thể là LinearSVC với chiến lược "one-vs-all", được sử dụng để phân loại các cảm xúc như vui, buồn, giận dữ, sợ hãi, ghê tởm và ngạc nhiên dựa trên các đặc trưng đã huấn luyện. Hệ thống được triển khai bằng OpenCV và Python, với thử nghiệm trên các cơ sở dữ liệu như CK, CK+ và IMM, cho thấy khả năng nhận dạng và phân loại cảm xúc với thời gian xử lý tương đối thấp.

Trong bài báo [5], Saravanan và cộng sự phân loại khuôn mặt thành 7 cảm xúc cơ bản, sử dụng bộ dữ liệu FER-2013. Nhóm tác giả đã thử nghiệm nhiều mô hình và đề xuất một Mạng Nơ-ron Tích chập (CNN) gồm sáu lớp tích chập, hai lớp gộp cực đại và hai lớp kết nối đầy đủ. Trong quá trình thử nghiệm trực tiếp, Haar Cascades được dùng để phát hiện khuôn mặt, ảnh sau đó được xử lý để phù hợp với đầu vào của mô hình. Sau khi tinh chỉnh siêu tham số, mô hình này đạt độ chính xác 0.60 với trình tối ưu hóa Adam. Nghiên cứu chỉ ra rằng các cảm xúc như hạnh phúc và ngạc nhiên dễ nhận diện hơn, trong khi cảm xúc "ghê tởm" gặp khó khăn do thiếu dữ liệu mẫu.

Trong bài báo [6], Gaurav Meena và các cộng sự đề xuất một phương pháp phân tích cảm xúc từ hình ảnh, tập trung vào việc cải thiện hiệu suất phân loại bằng cách sử dụng mạng nơ-ron tích chập sâu VGG19 và các đặc trưng sâu. Nghiên cứu này sử dụng kỹ thuật học chuyển tiếp với kiến trúc VGG19, điều chỉnh lớp phân loại cuối cùng, để phát hiện và phân loại các cảm xúc khác nhau trên các bộ dữ liệu CK+, FER2013, và JAFFE. Quá trình tiền xử lý bao gồm việc thay đổi kích thước ảnh thành 124×124 pixel và lật ảnh theo chiều ngang. Mô hình được huấn luyện qua 100 epoch với trình tối ưu hóa Adam và hàm mất mát là categorical cross-entropy. Kết quả thực nghiệm cho thấy phương pháp này đạt được độ chính xác cao, lên đến 99% đối với bộ dữ liệu CK+, 0.65% cho FER2013 và 0.93% cho JAFFE. Nghiên cứu kết luận rằng phương pháp này có thể được sử dụng để nghiên cứu thói quen cảm xúc và trạng thái tâm lý của một người từ biểu cảm khuôn mặt.

Trong bài báo [7], Milan Tripathi tập trung vào việc cải thiện độ chính xác nhận dạng cảm xúc khuôn mặt trên bộ dữ liệu FER2013, vốn có độ chính xác thấp hơn so với các bộ dữ liệu khác như CK+ hay JAFFE. Nghiên cứu này sử dụng ba mô hình học sâu dựa trên AlexNet, VGG19 và ResNet50, trong đó mô hình dựa trên VGG19 được chọn để phân tích sâu hơn do cho kết quả ban đầu tốt nhất. Mô hình VGG19 này sau đó được huấn luyện với năm trình tối ưu hóa khác nhau (ADAM, AdaDelta, RMSProb, AdaGrad, SGD) trên bộ dữ liệu FER2013 đã qua tiền xử lý, bao gồm chuẩn hóa và tăng cường dữ liệu. Mô hình hoạt động tốt nhất, sử dụng trình tối ưu hóa AdaDelta hoặc AdaGrad, đã đạt được độ chính xác ấn tượng là 91.89504% trên tập kiểm tra của FER2013. Kết quả này được cho là cải thiện ít nhất 17% so với các mô hình tiên tiến trước đó trên cùng bộ dữ liệu.

II. GIỚI THIỆU CÁC CÔNG NGHỆ SỬ DỤNG TRONG HỆ THỐNG

1. Backend:

a) Fast API

FastAPI là một framework Python hiện đại, được thiết kế để xây dựng các API web nhanh chóng và hiệu quả. Nó được phát triển dựa trên nền tảng của Starlette (để xử lý HTTP) và Pydantic (để kiểm tra và xác thực dữ liệu). FastAPI nổi tiếng với hiệu suất cao, khả năng tự động tạo tài liệu API và cú pháp đơn giản, dễ sử dụng.

Các đặc điểm chính của FastAPI:

- Hiệu suất cao :
 - FastAPI được xây dựng trên ASGI (Asynchronous Server Gateway Interface), cho phép xử lý các yêu cầu không đồng bộ (asynchronous) một cách hiệu quả
 - Hiệu suất của FastAPI ngang bằng với các framework như Node.js hoặc Go, giúp nó trở thành lựa chọn lý tưởng cho các ứng dụng cần tốc độ cao
- Hỗ trợ lập trình bất đồng bộ (async/await) : FastAPI hỗ trợ lập trình bất đồng bộ (async/await), giúp backend có thể xử lý nhiều yêu cầu cùng lúc như nhận luồng video từ frontend, gửi dữ liệu đến mô hình AI để phân tích cảm xúc và trả kết quả về frontend trong thời gian thực mà không bị tắc nghẽn. Điều này đặc biệt quan trọng khi hệ thống cần đáp ứng yêu cầu thời gian thực.

- Tự động tạo tài liệu API : FastAPI tự động tạo tài liệu API theo chuẩn OpenAPI và Swagger UI , giúp người dùng dễ dàng kiểm tra và tương tác với API mà không cần viết thêm mã
- Kiểm tra và xác thực dữ liệu với Pydantic : FastAPI tích hợp chặt chẽ với Pydantic, một thư viện mạnh mẽ để xác thực và kiểm tra dữ liệu đầu vào. Điều này giúp đảm bảo rằng dữ liệu gửi đến API luôn đúng định dạng và tuân thủ quy tắc
- Cú pháp đơn giản và dễ học : FastAPI có cú pháp dễ hiểu và trực quan, giúp các nhà phát triển nhanh chóng xây dựng API mà không cần phải học quá nhiều khái niệm phức tạp
- Hỗ trợ mạnh mẽ cho các tính năng hiện đại :
Hỗ trợ WebSocket cho giao tiếp thời gian thực: FastAPI cung cấp hỗ trợ đầy đủ cho WebSocket , một giao thức cho phép giao tiếp hai chiều giữa client và server. Với WebSocket, backend có thể liên tục gửi dữ liệu cảm xúc đã phân tích về frontend mà không cần client phải gửi yêu cầu lặp lại. Ví dụ: Client gửi luồng video qua WebSocket, backend nhận luồng video, phân tích cảm xúc và trả kết quả về client qua cùng một kết nối WebSocket.
- Dễ dàng tích hợp với các mô hình AI/ML: FastAPI có thể dễ dàng tích hợp với các thư viện AI/ML phổ biến như TensorFlow, PyTorch hoặc OpenCV để thực hiện nhận diện cảm xúc khuôn mặt

b) OpenCV (cv2) - Thư viện xử lý ảnh và computer vision:

OpenCV là tên viết tắt của open source computer vision library – một thư viện nguồn mở cho máy tính. Hay nói OpenCV là kho lưu trữ các mã nguồn mở được dùng để xử lý hình ảnh, phát triển các ứng dụng đồ họa trong thời gian thực. Lý do chọn OpenCV.

- Xử lý hình ảnh và video hiệu quả:
 - OpenCV được tối ưu hóa cho xử lý hình ảnh và video thời gian thực với hiệu suất cao
 - Có thể xử lý hàng loạt frame từ camera một cách nhanh chóng và tiết kiệm tài nguyên
 - Hỗ trợ nhiều định dạng hình ảnh và video phổ biến
- Tích hợp camera dễ dàng
 - Cung cấp API đơn giản để truy cập camera (cv2.VideoCapture())

- Hỗ trợ đa nền tảng (Windows, macOS, Linux) mà không cần thay đổi mã nguồn
 - Cho phép xử lý lỗi camera một cách linh hoạt (thử các camera khác nhau khi camera chính không hoạt động)
- Phát hiện khuôn mặt tích hợp sẵn:
- Cung cấp bộ phân loại Haar Cascade để phát hiện khuôn mặt một cách chính xác và nhanh chóng
 - Hỗ trợ nhiều bộ phân loại được huấn luyện sẵn (mặt, mắt, miệng...)
 - Cung cấp hàm detectMultiScale() cho phép điều chỉnh các tham số phát hiện khuôn mặt
- Xử lý ảnh tiền xử lý cho Deep Learning
- Cung cấp các hàm chuyển đổi không gian màu (từ BGR sang Grayscale) cần thiết cho tiền xử lý
 - Cho phép cắt vùng quan tâm (ROI) một cách dễ dàng để trích xuất khuôn mặt
 - Hỗ trợ nhiều thao tác tiền xử lý như thay đổi kích thước, chuẩn hóa, lọc nhiễu
- Hiển thị kết quả trực quan
- Cung cấp các hàm vẽ đơn giản (rectangle, putText) để hiển thị kết quả nhận diện
 - Cho phép tạo và hiển thị hình ảnh thông báo khi có lỗi (ví dụ: "Camera Error")
- Tương thích với numpy cho Deep Learning
- Làm việc tốt với mảng numpy, định dạng được sử dụng bởi các framework deep learning như TensorFlow
 - Cho phép chuyển đổi dễ dàng giữa mảng numpy và hình ảnh
- Mã nguồn mở và cộng đồng lớn
- Thư viện miễn phí, mã nguồn mở với nhiều tài liệu và hướng dẫn
 - Cộng đồng lớn giúp giải quyết vấn đề khi gặp khó khăn
 - Được cập nhật thường xuyên với các tính năng mới

Trong ứng dụng nhận diện cảm xúc này, OpenCV là thư viện không thể thiếu vì nó xử lý toàn bộ quy trình từ việc thu thập dữ liệu từ camera, phát hiện khuôn mặt, tiền xử lý ảnh cho mô hình deep learning, đến hiển thị kết quả nhận diện cho người dùng.

c) NumPy - Thư viện tính toán số học và xử lý mảng

NumPy (Numerical Python) là một thư viện mã nguồn mở giúp Python xử lý: Mảng nhiều chiều (arrays/matrices), tính toán đại số tuyến tính, các phép toán vector, ma trận hiệu suất cao.

- Xử lý mảng đa chiều hiệu suất cao (ndarray)
 - Cấu trúc dữ liệu chính của NumPy là ndarray (n-dimensional array)
 - Lưu trữ dữ liệu một cách liên tục trong bộ nhớ, mang lại hiệu suất cao hơn nhiều so với list của Python
- Xử lý dữ liệu nhị phân
 - Chuyển đổi giữa các dạng dữ liệu nhị phân và mảng NumPy
- Tương thích với OpenCV và các thư viện Deep Learning
 - OpenCV trả về và nhận dữ liệu ở dạng mảng NumPy
 - TensorFlow/Keras cũng sử dụng mảng NumPy làm đầu vào/đầu ra
 - Giúp tích hợp liền mạch giữa các thành phần trong ứng dụng nhận diện cảm xúc
- Các phép toán mảng hiệu quả
 - Cho phép thực hiện các phép toán trên toàn bộ mảng mà không cần vòng lặp
 - Hỗ trợ broadcasting cho phép thực hiện phép toán giữa các mảng có kích thước khác nhau

NumPy là thư viện không thể thiếu trong dự án nhận diện cảm xúc của bạn, cung cấp nền tảng vững chắc cho việc xử lý dữ liệu hình ảnh và tích hợp các thành phần khác nhau của hệ thống.

d) PIL (Python Imaging Library) - Thư viện xử lý ảnh

PIL (Python Imaging Library), còn được biết đến qua thư viện Pillow (fork hiện đại của PIL), là một thư viện xử lý ảnh mạnh mẽ cho Python. Thư viện này cung cấp nhiều chức năng xử lý ảnh và hỗ trợ đa dạng các định dạng ảnh.

- Mở và lưu nhiều định dạng ảnh

- Hỗ trợ các định dạng phổ biến như JPEG, PNG, GIF, BMP, TIFF, và nhiều định dạng khác
- Chuyển đổi giữa PIL và các định dạng dữ liệu khác
 - Chuyển đổi giữa đối tượng Image của PIL và mảng NumPy
 - Tạo đối tượng Image từ dữ liệu nhị phân hoặc byte stream
- Thao tác cơ bản trên ảnh
 - Thay đổi kích thước, cắt, xoay, lật ảnh
 - Điều chỉnh độ sáng, độ tương phản, màu sắc
 - Áp dụng các bộ lọc (filter) và hiệu ứng
- Truy cập và thao tác trên từng pixel
 - Đọc và sửa đổi giá trị pixel
- Kết quả của phép tích chập là một feature map, thể hiện mức độ phù hợp của bộ lọc với các vùng khác nhau trên hình ảnh
 - Công thức tích chập: hỗ trợ nhiều chế độ màu (RGB, RGBA, Grayscale, v.v.)

Trong ứng dụng nhận diện cảm xúc này, PIL đóng vai trò hỗ trợ OpenCV trong việc xử lý ảnh đầu vào từ người dùng, đặc biệt khi cần xử lý các định dạng ảnh đa dạng hoặc các thao tác xử lý đơn giản trước khi chuyển sang phát hiện khuôn mặt và nhận diện cảm xúc.

e) Tensorflow Keras

TensorFlow là một thư viện mã nguồn mở được phát triển bởi Google, chuyên dùng để xây dựng và huấn luyện các mô hình học máy (Machine Learning) và học sâu (Deep Learning). TensorFlow cung cấp một nền tảng mạnh mẽ để xử lý dữ liệu số, tính toán tensor (ma trận nhiều chiều), và thực hiện các phép toán phức tạp trên GPU/TPU.

Đặc điểm nổi bật của TensorFlow:

- Khả năng mở rộng : Nhận diện cảm xúc khuôn mặt đòi hỏi xử lý một lượng lớn dữ liệu hình ảnh và tính toán phức tạp. TensorFlow hỗ trợ tối ưu hóa hiệu suất bằng cách tận dụng GPU/TPU, giúp tăng tốc độ huấn luyện và suy luận (inference)
- Hệ sinh thái phong phú : TensorFlow bao gồm nhiều công cụ hỗ trợ như TensorFlow Lite (cho thiết bị di động), TensorFlow.js (cho trình duyệt), và TensorFlow Extended (TFX) cho sản xuất

- Tính linh hoạt : Hỗ trợ xây dựng từ các mô hình đơn giản đến phức tạp, bao gồm mạng neural convolutional (CNN), recurrent neural network (RNN), và transformer
- Khả năng phân tán : TensorFlow hỗ trợ tính toán phân tán, giúp tăng tốc độ huấn luyện mô hình trên nhiều máy chủ

Keras là một API cấp cao được tích hợp vào TensorFlow, cho phép xây dựng và huấn luyện mô hình học sâu một cách dễ dàng và nhanh chóng. Keras được thiết kế với mục tiêu chính là đơn giản hóa quá trình phát triển mô hình học sâu, giúp người dùng tập trung vào việc giải quyết bài toán thay vì lo lắng về các chi tiết kỹ thuật.

Đặc điểm nổi bật của Keras:

- Tích hợp với TensorFlow : Keras được tích hợp sẵn trong TensorFlow, tận dụng toàn bộ sức mạnh của TensorFlow
- Hỗ trợ đa nền tảng : Keras có thể chạy trên cả CPU và GPU mà không cần thay đổi mã nguồn
- Cú pháp đơn giản và trực quan : Keras sử dụng cú pháp dễ hiểu, giúp người mới bắt đầu dễ dàng xây dựng mô hình

f) Unicorn

- Là một máy chủ ASGI (Asynchronous Server Gateway Interface) được viết bằng Python, hỗ trợ chạy các ứng dụng web bất đồng bộ. Đây là một lựa chọn phổ biến để triển khai các ứng dụng FastAPI nhờ vào hiệu suất cao và khả năng xử lý nhiều yêu cầu đồng thời.
- Tóm tắt về Unicorn:
 - ASGI Server: Unicorn là một máy chủ ASGI, cho phép các ứng dụng web Python chạy bất đồng bộ, giúp tối ưu hóa thời gian phản hồi và hiệu suất.
 - Hỗ trợ async Python: Unicorn hỗ trợ các tính năng bất đồng bộ của Python như `async` và `await`, giúp xử lý nhiều yêu cầu đồng thời mà không cần chờ đợi.
 - Hiệu suất cao: Được thiết kế để hoạt động nhanh chóng và hiệu quả, Unicorn có thể xử lý hàng ngàn yêu cầu mỗi giây, làm cho nó trở thành một lựa chọn lý tưởng cho các ứng dụng web có lưu lượng cao.

2. Frontend

a) HTML:

Ngôn ngữ đánh dấu được sử dụng để tạo cấu trúc cơ bản của các trang web.

Cho phép xây dựng giao diện người dùng bằng cách định nghĩa các thành phần như văn bản, hình ảnh, liên kết, và biểu mẫu.

b) CSS:

Ngôn ngữ định kiểu giúp tạo giao diện đẹp mắt và trực quan.

Sử dụng để tùy chỉnh màu sắc, bố cục, và hiệu ứng động trên trang web, đảm bảo trải nghiệm người dùng tốt hơn.

c) JavaScript:

Ngôn ngữ lập trình kịch bản được sử dụng để thêm logic động vào giao diện người dùng.

d) Cơ chế hoạt động:

Chạy trực tiếp trên trình duyệt web thông qua môi trường JavaScript Engine.

Thực thi theo mô hình **event-driven, non-blocking I/O** thông qua **event loop**, đảm bảo hiệu suất cao.

Làm việc với **DOM (Document Object Model)** để cập nhật nội dung trang web mà không cần tải lại toàn bộ trang.

e) Bootstrap:

Một framework front-end mã nguồn mở dùng để thiết kế giao diện web nhanh chóng.

- Cung cấp sẵn các thành phần HTML, CSS, JavaScript như nút, form, modal...
- Hỗ trợ responsive (tương thích nhiều kích thước màn hình).
- Dễ tích hợp vào bất kỳ dự án web nào.

f) Font Awesome

Là một thư viện biểu tượng (icon) mã nguồn mở phổ biến, dùng để thêm các biểu tượng vector đẹp mắt và dễ tùy chỉnh vào giao diện người dùng, đặc biệt phù hợp khi kết hợp với các framework như Bootstrap hoặc các ứng dụng FastAPI.

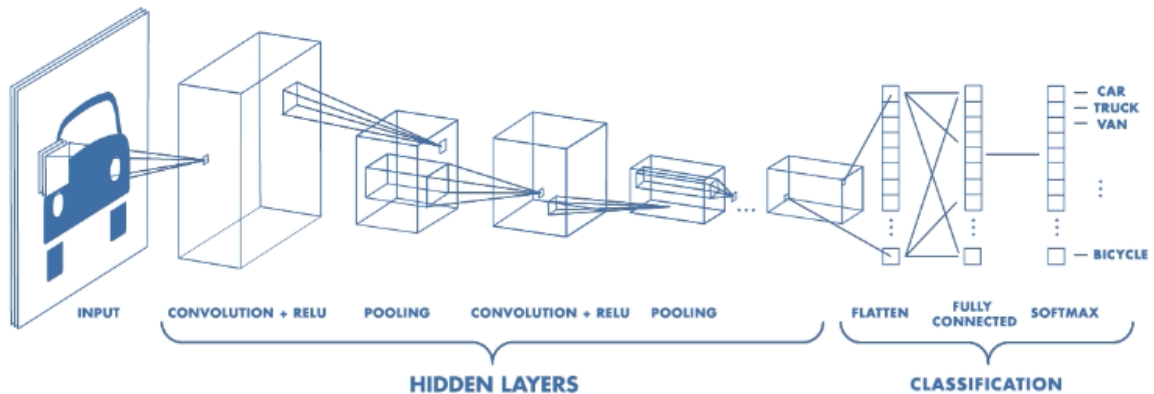
3. Machine Learning/Deep Learning:

3.1. Mô hình CNN (Convolutional Neural Network)

Convolutional Neural Network (CNN) là một loại mạng nơ-ron nhân tạo (Artificial Neural Network) được thiết kế đặc biệt để xử lý dữ liệu có cấu trúc dạng lưới, chẳng hạn như hình ảnh. CNN đã trở thành công cụ mạnh mẽ

trong các lĩnh vực như thị giác máy tính (Computer Vision), nhận diện đối tượng, phân loại hình ảnh, phát hiện bất thường và nhiều ứng dụng khác.

Khác với các mạng nơ-ron truyền thống (Fully Connected Neural Network - FCNN), CNN tận dụng cấu trúc không gian của dữ liệu đầu vào, giúp giảm thiểu số lượng tham số cần huấn luyện và tăng hiệu quả trong việc xử lý dữ liệu đa chiều.



Hình 1: Kiến trúc mô hình CNN

Cấu trúc cơ bản của CNN: Một mô hình CNN điển hình bao gồm các lớp sau:

- Lớp Convolutional (Conv Layer)
 - Đây là lớp cốt lõi của CNN, thực hiện phép tích chập (convolution) giữa ma trận đầu vào (input) và bộ lọc (filter/kernel)
 - Mỗi bộ lọc sẽ trượt qua toàn bộ hình ảnh đầu vào để phát hiện các đặc trưng cục bộ, chẳng hạn như cạnh, góc hoặc kết cấu
 - Kết quả của phép tích chập là một feature map, thể hiện mức độ phù hợp của bộ lọc với các vùng khác nhau trên hình ảnh
 - Công thức tích chập:

$$Y[i, j] = \sum_m \sum_n X[i + m, j + n] \cdot K[m, n]$$

Trong đó:

X: Ma trận đầu vào biểu diễn hình ảnh.

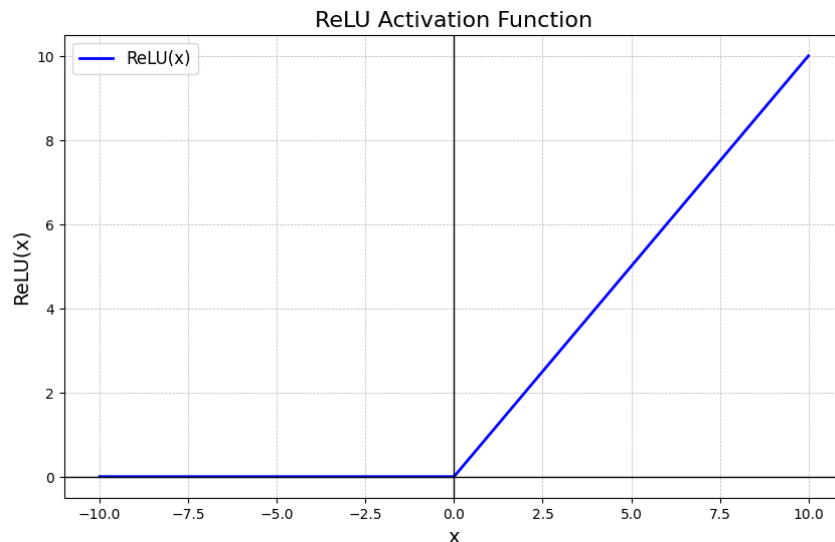
K: Ma trận kernel (bộ lọc).

$Y[i, j]$: Giá trị đầu ra tại vị trí (i,j) sau khi thực hiện phép tích chập.

- Lớp Activation (Activation Layer)

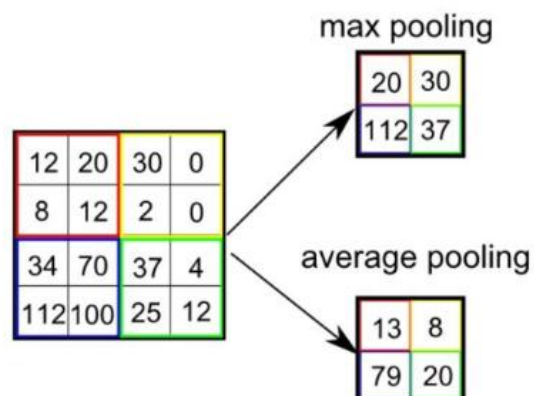
- Sau khi tính toán tích chập, một hàm kích hoạt (activation function) như ReLU (Rectified Linear Unit) được áp dụng để thêm tính phi tuyến vào mô hình

$$ReLU(x) = \max(0, x)$$



Hình 2 : Đồ thị của hàm ReLU

- Hàm ReLU thường được sử dụng vì nó giúp tăng tốc độ hội tụ và giảm hiện tượng "vanishing gradient".
- Lớp Pooling (Pooling Layer)
 - Lớp pooling thực hiện việc giảm kích thước của feature map bằng cách lấy giá trị lớn nhất (Max Pooling) hoặc giá trị trung bình (Average Pooling) trên các vùng nhỏ của feature map



Hình 3: Pooling Layer

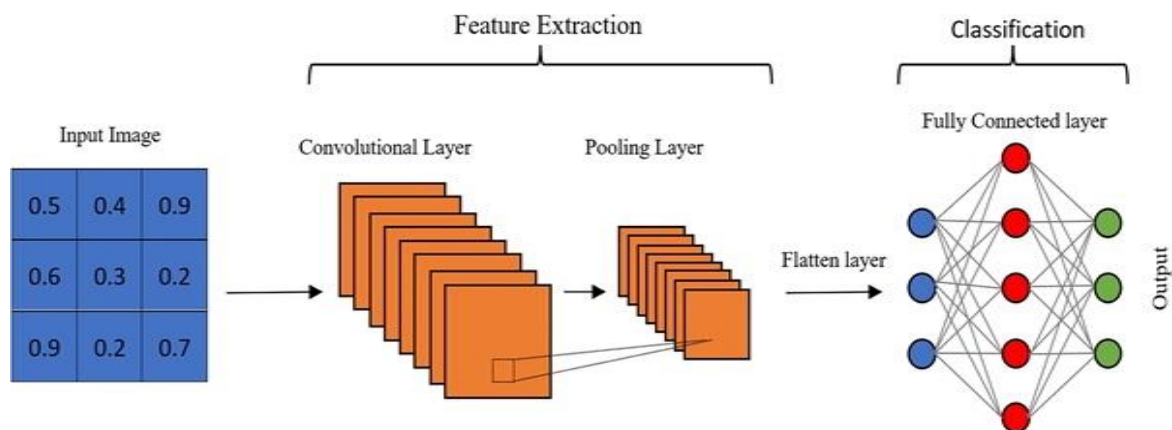
- Mục đích của pooling là giảm số lượng tham số, tránh overfitting và tăng tính bất biến đối với các phép biến đổi nhỏ (như dịch chuyển hoặc xoay)
- Lớp Fully Connected (FC Layer)
 - Sau khi dữ liệu đã được xử lý qua các lớp convolutional và pooling, nó sẽ được "làm phẳng" (flatten) thành một vector và truyền vào các lớp fully connected.
 - Các lớp fully connected thực hiện nhiệm vụ phân loại dựa trên các đặc trưng đã được trích xuất từ các lớp trước đó.
 - Sử dụng hàm kích hoạt softmax cho các bài toán phân loại đa lớp:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^n e^{z_j}}$$

Trong đó:

z_i : Đầu ra của nơ-ron tại lớp fully connected.

n : Số lớp phân loại đầu ra.



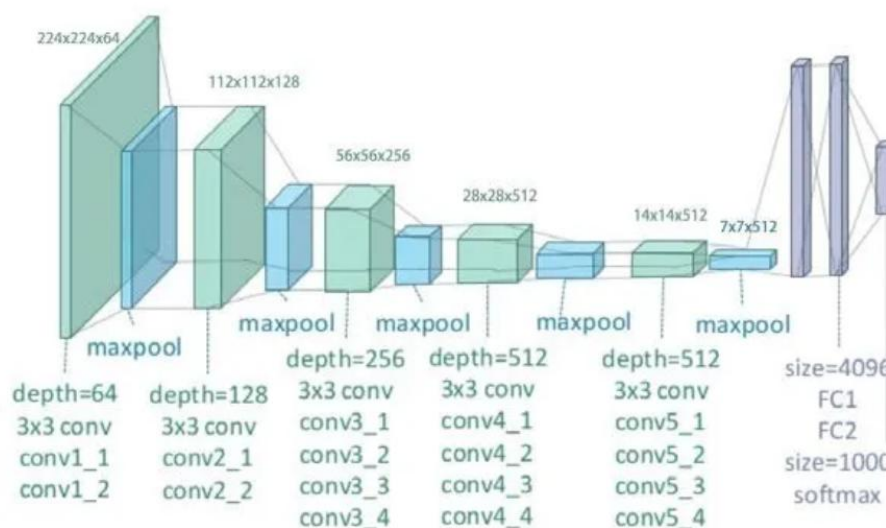
Hình 4: Sự khác nhau giữa các tầng trong CNN

- Lớp Output
 - Lớp cuối cùng của CNN thường là một lớp fully connected với hàm softmax (cho bài toán phân loại) hoặc hàm hồi quy (cho bài toán dự đoán giá trị liên tục).
 - Hàm softmax sẽ chuyển đổi các giá trị đầu ra thành xác suất cho từng lớp

3.2. Một số mạng CNN phổ biến:

- LeNet-5: là mô hình CNN đầu tiên do Yann LeCun phát triển cho nhận diện chữ số viết tay trên MNIST. Mô hình gồm các lớp Convolution, sigmoid, Average Pooling và Fully Connected giúp học đặc trưng cục bộ và toàn cục. Nhờ cơ chế chia sẻ tham số, LeNet-5 giảm số lượng tham số đáng kể. Dù hiệu quả, mô hình chưa vượt trội so với SVM do hạn chế kỹ thuật lúc bấy giờ.
- Alexnet: là mô hình CNN chiến thắng ILSVRC 2012, giảm mạnh lỗi phân loại từ 25.8% xuống 16.4%. Mô hình mở rộng LeNet với 5 lớp CNN, dùng ReLU thay sigmoid, Max Pooling thay Average Pooling và tận dụng GPU để huấn luyện. Nó còn áp dụng Dropout, Data Augmentation và LRN giúp tăng hiệu quả và tránh overfitting.
- ZFNet: là mô hình cải tiến từ AlexNet, ZFNet sử dụng kỹ thuật DeconvNet để trực quan hóa đặc trưng học được, giúp hiểu và điều chỉnh mô hình. Các cải tiến gồm: giảm kernel size từ 11×11 xuống 7×7 và giảm stride từ 4 xuống 2, giúp học được đặc trưng rõ hơn, giảm hiện tượng aliasing.
- VGG: VGG đánh dấu bước tiến lớn sau AlexNet và ZFNet nhờ thiết kế đơn giản, có hệ thống và dễ áp dụng. Thay vì kernel lớn, VGG chỉ dùng kernel 3×3 , stride 1, padding 1 để giữ nguyên kích thước ảnh, kết hợp nhiều lớp liên tiếp để tăng độ phức tạp phi tuyến. Mỗi khối convolution được theo sau bởi Max Pooling 2×2 , giảm dần kích thước ảnh. Số filter tăng dần theo lũy thừa 2 ($64 \rightarrow 128 \rightarrow 256 \rightarrow 512$).

Trong dự án này chúng em sẽ sử dụng mô hình VGG19, một trong những mô hình CNN nổi tiếng và được sử dụng rộng rãi trong lĩnh vực thị giác máy tính, đặc biệt là trong các bài toán phân loại ảnh.



Hình 5: Kiến trúc mô hình VGG19

3.3. Giới thiệu về Mô hình VGG19

- VGG19 là một mô hình học sâu được gọi là kiến trúc Mạng nơ-ron tích chập (CNN). Mô hình VGG19 có 19 lớp và bao gồm 16 lớp tích chập và 3 lớp lớp được kết nối đầy đủ. Các lớp tích chập trích xuất các đặc điểm từ hình ảnh đầu vào bằng bộ lọc.
- Kiến trúc của VGG19: Gồm 19 tầng có trọng số (bao gồm cả tầng tích chập và tầng kết nối đầy đủ)
 - 16 tầng tích chập (convolutional layers) : Dùng bộ lọc (filter) kích thước 3×3 , bước dịch (stride) = 1, thường đi kèm với hàm kích hoạt ReLU.
 - 5 tầng max-pooling : Kích thước cửa sổ pooling là 2×2 , stride = 2.
 - 3 tầng fully connected (FC) : Các tầng FC này có số nơ-ron lần lượt là 4096, 4096 và 1000 (đầu ra cuối cùng).
- Ban đầu, VGG19 được huấn luyện trên tập dữ liệu ImageNet để phân loại hình ảnh thành hàng nghìn lớp khác nhau. Các trọng số đã được huấn luyện trước (pre-trained weights) của VGG19 thường được sử dụng trong kỹ thuật học chuyển giao (transfer learning).

Trong hệ thống của chúng em, mô hình VGG19 được áp dụng để nhận diện cảm xúc khuôn mặt trên tập dữ liệu **FER2013**. Tập dữ liệu này chứa 35,887 ảnh xám khuôn mặt, thể hiện 7 loại cảm xúc khác nhau: giận dữ (anger), ghê tởm (disgust), sợ hãi (fear), vui vẻ (happiness), trung tính (neutral), buồn bã (sad), và ngạc nhiên (surprise).

- Hệ thống sử dụng kỹ thuật **học chuyển giao** bằng cách tận dụng các trọng số đã được huấn luyện trước của VGG19 trên ImageNet. Điều này giúp mô hình có khả năng trích xuất các đặc trưng cơ bản từ ảnh một cách hiệu quả mà không cần huấn luyện lại từ đầu trên một lượng lớn dữ liệu.
- Vì VGG19 ban đầu được thiết kế cho ImageNet, lớp kết nối đầy đủ cuối cùng của nó cần được điều chỉnh để phù hợp với bài toán nhận diện 7 loại cảm xúc của FER2013:
 - Lớp đầu vào (input_shape) được giữ nguyên là (48, 48, 3) để phù hợp với kích thước ảnh đầu vào sau tiền xử lý (ảnh xám được chuyển thành RGB)
 - Phần thân của VGG19 (các lớp tích chập) được giữ lại để trích xuất đặc trưng, với include_top = False để loại bỏ các lớp kết nối đầy đủ ban đầu
 - Đầu ra từ VGG19 (base_model.layers[-2].output) được đưa qua lớp GlobalAveragePooling2D
 - Tiếp theo, một lớp Dense mới (out_layer) với 7 đầu ra (softmax) được thêm vào để phân loại 7 cảm xúc

- Dữ liệu FER2013 sẽ được tiền xử lý và tăng cường sau đó đưa vào mô hình VGG19, với kiến thức nền tảng từ ImageNet, học cách nhận diện các đặc trưng cảm xúc cụ thể trên khuôn mặt từ tập dữ liệu FER2013
- Sau khi huấn luyện, các trọng số (weights) tối ưu của mô hình được lưu vào file `model_weights.h5`. File này sau đó được sử dụng để tải lại các trọng số đã học, cho phép mô hình thực hiện dự đoán cảm xúc trên ảnh mới mà không cần huấn luyện lại từ đầu.

3.4. Train mô hình

3.4.1. Bộ dữ liệu

- Trong dự án này chúng em sử dụng bộ dataset FER2013, một bộ dataset phổ biến với 35,887 grayscale ảnh khuôn mặt có kích thước 48x48 pixels. Bộ data gồm 7 loại: Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral.



Những hình ảnh dữ liệu này thì đã được lưu trữ dưới dạng file csv

Hàng đầu tiên sẽ là tên 3 cột: emotion, pixels, usage. Còn lại 35,887 row sẽ lưu thông tin của từng ảnh với các chỉ số sau: (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral).

Link download dataset: <https://www.kaggle.com/deadskull7/fer2013>

3.4.2. Tiền xử lý dữ liệu

Quá trình tiền xử lý dữ liệu đóng vai trò then chốt trong việc chuẩn bị dữ liệu thô cho mô hình học sâu, đảm bảo tính nhất quán và tối ưu hóa hiệu suất của mô hình. Đối với tập dữ liệu FER2013 chúng em đã thực hiện các bước sau:

- Tải và khám phá dữ liệu: Dữ liệu được tải từ tệp `fer2013.csv` vào một cấu trúc DataFrame của Pandas. Tập dữ liệu ban đầu bao gồm 35,887 mẫu, mỗi mẫu chứa thông tin về cảm xúc (emotion), chuỗi pixel của hình ảnh (pixels), và mục đích sử dụng (Usage - training, validation, testing). Mỗi hình ảnh có kích thước 48x48 pixel và ở định dạng thang độ xám.
- Chuyển đổi pixel thành ảnh: Chuyển chuỗi pixel thành mảng NumPy 48x48, với kiểu dữ liệu float32

- Chuyển đổi không gian màu: các hình ảnh thang độ xám (48x48) từ tập FER2013 đã được chuyển đổi sang định dạng RGB (48x48x3) để đảm bảo tính tương thích của dữ liệu đầu vào với kiến trúc của mô hình VGG19.
- Mã hóa nhãn cảm xúc: Nhãn cảm xúc ban đầu (số nguyên từ 0-6) được chuyển đổi sang định dạng one-hot encoding. LabelEncoder (sklearn.preprocessing) được dùng để ánh xạ nhãn số nguyên sang dãy 0 đến (số lớp - 1), sau đó to_categorical (keras.utils) tạo vector one-hot. Mỗi nhãn sau mã hóa là một vector nhị phân, với giá trị 1 tại vị trí lớp tương ứng
- Phân chia dữ liệu: Toàn bộ dữ liệu hình ảnh (đã chuyển đổi sang RGB) và nhãn cảm xúc (đã mã hóa one-hot) được phân chia thành hai tập: tập huấn luyện (training set) và tập xác thực (validation set). Tỷ lệ phân chia là 90% cho tập huấn luyện và 10% cho tập xác thực.
- Chuẩn hóa dữ liệu hình ảnh: tất cả các giá trị pixel được chia cho 255.0. Bước này đưa các giá trị pixel về khoảng [0, 1], giúp cải thiện sự ổn định và tốc độ hội tụ của mô hình mạng nơ-ron trong quá trình huấn luyện.
- Tăng cường dữ liệu (Data Augmentation): Để tăng tính đa dạng của dữ liệu huấn luyện và giảm thiểu nguy cơ overfitting, kỹ thuật tăng cường dữ liệu đã được áp dụng cho tập huấn luyện (X_train) thông qua ImageDataGenerator của Keras. Các phép biến đổi ngẫu nhiên bao gồm: xoay ảnh, dịch chuyển chiều rộng, dịch chuyển chiều cao, biến dạng cắt, thu phóng, lật ngang

3.4.3. Huấn luyện mô hình

- Mô hình nhận dạng cảm xúc được xây dựng bằng cách sử dụng kiến trúc VGG19 được huấn luyện trước trên tập dữ liệu ImageNet để tận dụng khả năng học đặc trưng mạnh mẽ của mô hình. Một số bước chính trong việc thiết lập mô hình:
 - VGG19 làm xương sống:
 - Sử dụng VGG19 với tham số weights = 'imagenet' để tải các trọng số đã được huấn luyện trước và include_top=False để loại bỏ các tầng phân loại mặc định của ImageNet.
 - Kích thước đầu vào được đặt là (48, 48, 3) để phù hợp với dữ liệu đã tiền xử lý.
 - Tầng phía trên (Head Layers):
 - Thêm một tầng GlobalAveragePooling2D để chuyển đổi đầu ra đa chiều của VGG19 thành một vector đặc trưng 1D.
 - Thêm một tầng Dense với số lượng đơn vị bằng số lớp cảm xúc (7 lớp) và hàm kích hoạt softmax làm tầng đầu ra để thực hiện phân loại.

- Cấu hình mô hình :
 - Bộ tối ưu hóa : Optimizer
 - Hàm mất mát (Loss function): categorical_crossentropy
 - Chỉ số đánh giá (Metrics): accuracy (độ chính xác)

Dữ liệu đầu vào

- **Train Generator và Validation Data:**

- ImageDataGenerator được cấu hình với các phép tăng cường dữ liệu (xoay, dịch chuyển, cắt, thu phóng, lật ngang) và được áp dụng cho tập huấn luyện (X_train, y_train) thông qua phương thức .flow(). Điều này cung cấp dữ liệu huấn luyện đã được chuẩn hóa và tăng cường, được chia thành các lô nhỏ (batches).
- Dữ liệu xác thực (X_valid, y_valid) được sử dụng trực tiếp mà không qua tăng cường dữ liệu để đánh giá hiệu suất thực của mô hình trên dữ liệu chưa từng thấy.
- Kích thước lô (batch_size) được đặt là 32.

Các tham số huấn luyện

- **Callbacks:**
 - **EarlyStopping:** Theo dõi chỉ số val_accuracy. Quá trình huấn luyện sẽ dừng sớm nếu không có cải thiện đáng kể (min_delta = 0.00005) sau 11 epoch (patience = 11), và trọng số tốt nhất sẽ được khôi phục.
- **Epochs:** Tối đa 25 epoch (có thể dừng sớm hơn do EarlyStopping).
- **Steps per epoch:** Được tính bằng len(X_train) / batch_size để đảm bảo mô hình duyệt qua toàn bộ tập huấn luyện trong mỗi epoch.

3.4.4. Đánh giá mô hình

Quá trình huấn luyện được thực hiện qua 25 epoch. Độ chính xác trên tập huấn luyện tăng dần, đạt khoảng 76.87% ở epoch cuối cùng. Độ chính xác trên tập xác thực cũng cho thấy sự cải thiện, đạt giá trị cao nhất khoảng 68.74%

4. Công cụ phát triển:

a) Kaggle (Kernels/Notebooks) - Môi trường Phát triển và Huấn luyện Model

- Kaggle, nổi tiếng với các cuộc thi khoa học dữ liệu, cũng cung cấp một nền tảng miễn phí dựa trên đám mây tương tự như Google Colab, được gọi là **Kaggle Kernels** (hay **Kaggle Notebooks**). Nền tảng này cho phép viết và

chạy mã (chủ yếu là Python và R) trực tiếp trên trình duyệt. Notebooks được tích hợp sẵn các thư viện khoa học dữ liệu phổ biến và cung cấp tài nguyên tính toán miễn phí (bao gồm cả GPU).

b) VS Code - IDE phát triển

- Là một trình soạn thảo mã nguồn (source-code editor) miễn phí, mạnh mẽ, và đa nền tảng do Microsoft phát triển. Ra mắt vào năm 2015, VS Code đã trở thành một trong những IDE (Integrated Development Environment) phổ biến nhất nhờ tính linh hoạt, hỗ trợ nhiều ngôn ngữ lập trình, và hệ sinh thái mở rộng phong phú. VS Code đặc biệt được yêu thích trong cộng đồng lập trình viên Python, web development, và các lĩnh vực như khoa học dữ liệu, học máy, khi làm việc với các thư viện như NumPy hay OpenCV.
- Thư mục **.vscode** là một thư mục cấu hình đặc biệt được tạo trong dự án khi sử dụng VS Code để lưu trữ các thiết lập và tùy chỉnh cụ thể cho dự án đó.

c) Git - Quản lý phiên bản

- Git là một hệ thống quản lý phiên bản phân tán (distributed version control system) được thiết kế để theo dõi các thay đổi trong mã nguồn và hỗ trợ làm việc nhóm hiệu quả. Được phát triển bởi Linus Torvalds vào năm 2005, Git hiện là công cụ quản lý phiên bản phổ biến nhất, được sử dụng rộng rãi trong phát triển phần mềm, khoa học dữ liệu, và các dự án liên quan đến mã nguồn, bao gồm cả các dự án sử dụng Python với thư viện như NumPy và OpenCV.
- Thư mục **.git** là một thư mục ẩn được tạo trong thư mục gốc của dự án khi khởi tạo một kho lưu trữ Git (repository). Nó chứa toàn bộ dữ liệu liên quan đến lịch sử, cấu hình, và trạng thái của kho lưu trữ Git.

5. Các thành phần khác:

• Môi trường ảo (virtual environment - venv)

Trong Python **venv** là một công cụ giúp tạo ra các môi trường độc lập để chạy các dự án Python, tránh xung đột giữa các gói (package) hoặc phiên bản Python khác nhau. Mục đích của venv:

- **Cô lập dự án:** Mỗi môi trường ảo có bộ thư viện riêng, không ảnh hưởng đến hệ thống Python toàn cục hoặc các dự án khác.

- **Quản lý phiên bản gói:** Cho phép cài đặt các phiên bản cụ thể của gói (ví dụ: `numpy==1.18` cho dự án A, `numpy==1.21` cho dự án B).
- **Tránh xung đột:** Đảm bảo các dự án không bị lỗi do khác biệt về phụ thuộc.

CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

I. LẤY YÊU CẦU HỆ THỐNG

1. Mô tả hệ thống bằng ngôn ngữ tự nhiên

Hệ thống nhận diện cảm xúc khuôn mặt theo thời gian thực là một ứng dụng web tích hợp giữa trí tuệ nhân tạo, xử lý ảnh và giao diện người dùng hiện đại. Khi người dùng truy cập vào trang web, họ có thể sử dụng camera của thiết bị hoặc tải lên ảnh khuôn mặt để hệ thống tự động phát hiện và phân tích cảm xúc.

2. Xác định và mô tả các tác nhân

Bảng 1: Xác định các tác nhân

STT	Tên Actor	Giải thích
1	Người dùng	Người mở web và tiến hành cấp quyền truy cập camera để nhận diện cảm xúc

a) Chức năng cho Người dùng

- **Nhận diện cảm xúc qua video trực tiếp:** Hệ thống cho phép người dùng bật/tắt camera trực tiếp trên trình duyệt. Khi camera được bật, hình ảnh từ webcam sẽ được truyền liên tục về server. Server sử dụng thư viện OpenCV để phát hiện khuôn mặt trong từng khung hình, sau đó cắt vùng khuôn mặt và đưa vào mô hình deep learning (VGG19 đã huấn luyện) để dự đoán cảm xúc. Kết quả nhận diện sẽ được hiển thị trực tiếp trên video với nhãn cảm xúc
- **Phân tích cảm xúc từ ảnh tĩnh:**
 - + Tải ảnh từ thiết bị
 - + Chụp ảnh mới

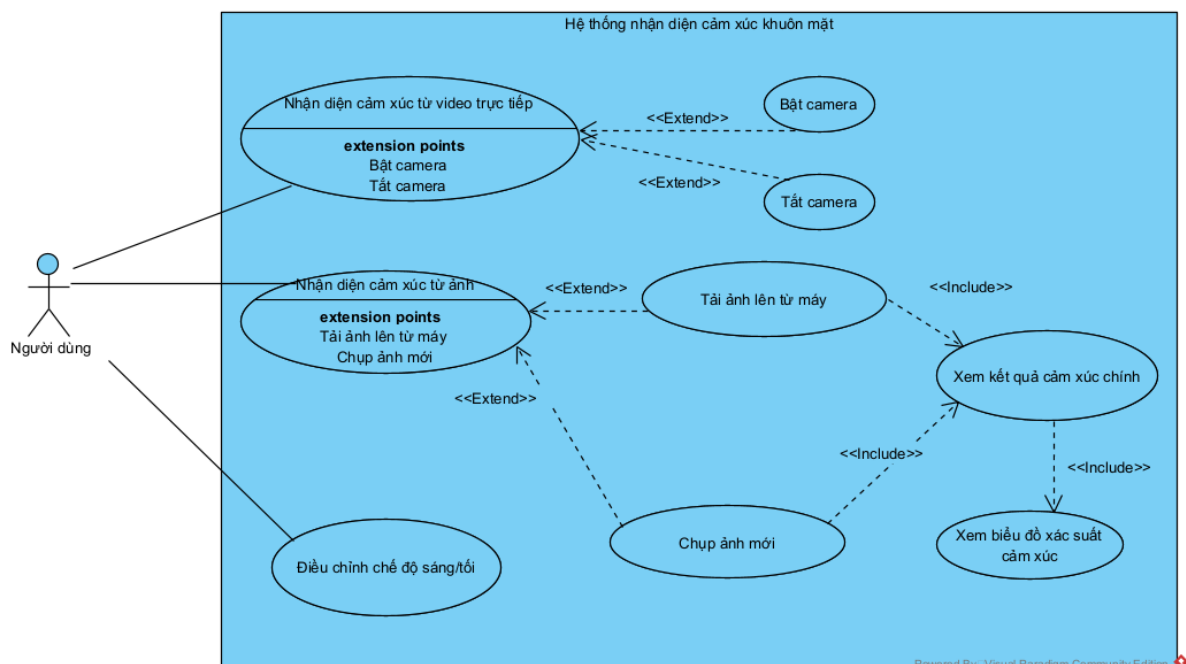
Ảnh này sẽ được gửi lên server, xử lý tương tự như video stream: phát hiện khuôn mặt, phân tích cảm xúc và trả về kết quả. Giao diện web hiển thị ảnh đã chọn, kết quả nhận diện và biểu đồ xác suất cảm xúc. Có thể:
- **Điều chỉnh chế độ sáng/tối của giao diện:** chuyển đổi giữa giao diện sáng/tối linh hoạt với môi trường

3. Xác nhận và mô tả các use case sử dụng

Bảng 2: Xác định và mô tả các use case

STT	Tên usecase	Mô tả
1	Nhận diện cảm xúc theo thời gian thực	Người dùng mở Camera để bắt đầu sử dụng chức năng nhận diện cảm xúc theo thời gian thực
2	Nhận diện cảm xúc từ ảnh	Người dùng có thể tải ảnh lên từ thiết bị hoặc chụp ảnh trực tiếp để phân tích cảm xúc
3	Điều chỉnh chế độ sáng/tối của giao diện	Người dùng có thể đổi giao diện sang màu tối hoặc sáng để dễ nhìn hơn

3.1. Usecase tổng quát của hệ thống (chi tiết các use case con nếu có)



Hình 6: Tổng quan use case toàn hệ thống

Các modul chính :

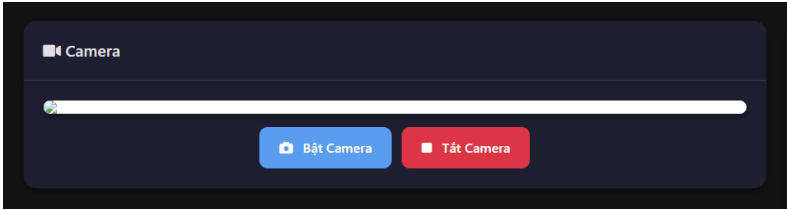
- Nhận diện cảm xúc theo thời gian thực
- Nhận diện cảm xúc từ ảnh:
 - Tải ảnh lên từ máy
 - Chụp ảnh mới
- Điều chỉnh chế độ sáng/tối của giao diện

II. PHÂN TÍCH HỆ THỐNG

1. Scenario

a) Kịch bản chức năng “Nhận diện cảm xúc theo thời gian thực”

Bảng 3: Kịch bản Nhận diện cảm xúc theo thời gian thực

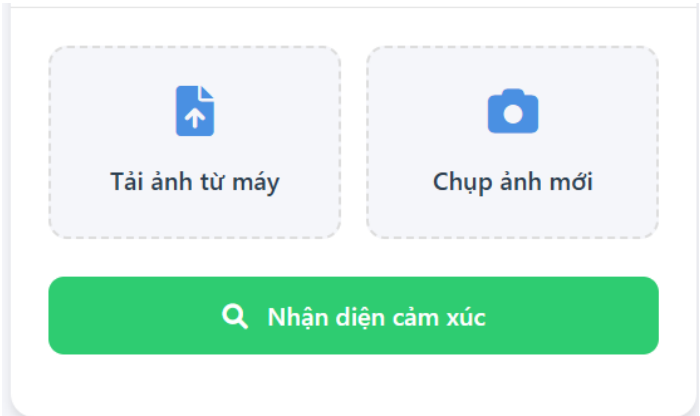
Use case	Nhận diện cảm xúc theo thời gian thực
Actor	Người dùng
Tiền điều kiện	Người dùng đã truy cập vào ứng dụng web. Thiết bị của người dùng có camera. Trình duyệt hỗ trợ truy cập camera và đã cấp quyền truy cập camera.
Hậu điều kiện	Kết quả nhận diện cảm xúc được hiển thị cho người dùng.
Kịch bản chính	<ol style="list-style-type: none">1. Người dùng truy cập vào trang web.2. Hệ thống hiển thị giao diện với 2 nút “Bật camera” và “Tắt camera” 3. Người dùng nhấn nút "Bật Camera"4. Hệ thống kích hoạt camera và hiển thị video stream. Hệ thống liên tục xử lý từng frame video. Phát hiện khuôn mặt trong frame. Trích xuất vùng khuôn mặt. Áp dụng mô hình AI để phân tích cảm xúc. Vẽ hộp giới hạn và nhãn cảm xúc lên video.5. Người dùng quan sát kết quả nhận diện theo thời gian thực trên màn hình camera. Khi kết thúc người dùng ấn nút "Tắt Camera".6. Hệ thống ngừng xử lý video và giải phóng camera
Ngoại lệ	2. Không bật được camera do quyền truy cập hoặc bị lỗi

b) Kịch bản chức năng “Nhận diện cảm xúc từ ảnh”

- Tải ảnh lên từ máy

Bảng 4: Kịch bản Nhận diện cảm xúc từ ảnh tải lên

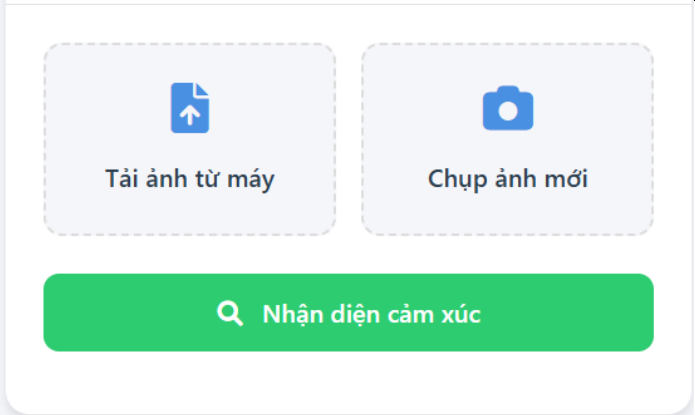
Use case	Nhận diện cảm xúc từ ảnh
----------	--------------------------

Actor	Người dùng
Tiền điều kiện	Người dùng đã truy cập vào ứng dụng web. Người dùng đã có sẵn các ảnh để tải lên
Hậu điều kiện	Kết quả nhận diện cảm xúc được hiển thị cho người dùng.
Kịch bản chính	<ol style="list-style-type: none"> 1. Người dùng truy cập vào trang web 2. Hệ thống hiển thị giao diện với 2 nút “Tải ảnh từ máy” và “Chụp ảnh mới”  <ol style="list-style-type: none"> 3. Người dùng ấn vào nút “Tải ảnh từ máy” 4. Hệ thống hiện giao diện để chọn 1 ảnh từ máy 5. Người dùng chọn 1 ảnh từ máy tính 6. Hệ thống tải ảnh đã chọn lên web 7. Người dùng ấn nút “Nhận diện cảm xúc” 8. Hệ thống hiện kết quả cảm xúc chính của ảnh cho người dùng

- Chụp ảnh mới

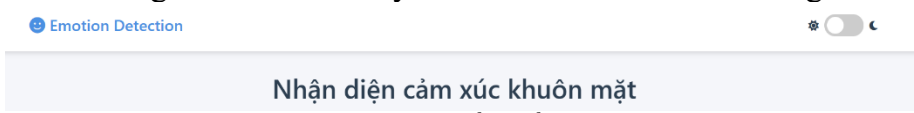
Bảng 5: Kịch bản Nhận diện cảm xúc từ ảnh chụp mới

Use case	Nhận diện cảm xúc từ ảnh
Actor	Người dùng
Tiền điều kiện	Người dùng đã truy cập vào ứng dụng web. Thiết bị của người dùng có camera. Trình duyệt hỗ trợ truy cập camera và đã cấp quyền truy cập camera.
Hậu điều kiện	Kết quả nhận diện cảm xúc được hiển thị cho người dùng.
Kịch bản chính	<ol style="list-style-type: none"> 1. Người dùng truy cập vào trang web 2. Hệ thống hiển thị giao diện với 2 nút “Tải ảnh từ máy” và “Chụp ảnh mới”

	 <p>3. Người dùng ấn vào nút “Chụp ảnh mới”</p> <p>4. Hệ thống hiện giao diện chụp ảnh và nút “Chụp ảnh”</p> <p>5. Người dùng ấn vào nút “Chụp ảnh” và sau đó ấn vào nút “Nhận diện cảm xúc”</p> <p>6. Hệ thống tiến hành phân tích, dự đoán cảm xúc và trả kết quả về cho người dùng</p>
--	---

c) Kịch bản chức năng “Điều chỉnh chế độ sáng/tối của giao diện”

Bảng 6: Kịch bản Điều chỉnh chế độ sáng/tối của giao diện

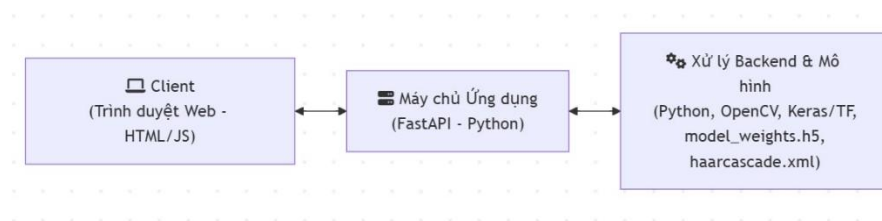
Use case	Điều chỉnh chế độ sáng/tối của giao diện
Actor	Người dùng
Tiền điều kiện	Người dùng đã truy cập vào ứng dụng web.
Hậu điều kiện	Người dùng có thể điều chỉnh chế độ sáng/tối tùy chọn
Kịch bản chính	<p>1. Người dùng truy cập vào trang web</p> <p>2. Hệ thống hiển thị nút chuyển đổi theme trên thanh navigation</p>  <p>3. Người dùng click vào nút chuyển đổi theme</p> <p>4. Hệ thống lưu trữ cài đặt theme và cập nhật kết quả giao diện cho người dùng</p>

III. THIẾT KẾ HỆ THỐNG

1. Mô hình kiến trúc hệ thống

Hệ thống nhận diện cảm xúc khuôn mặt được thiết kế với các thành phần chính sau

- Giao diện Người dùng (Frontend và JavaScript phía client):
 - Là giao diện web chính, được phát triển bằng HTML, CSS và JavaScript, được phục vụ bởi FastAPI.
 - Tương tác trực tiếp với người dùng: cho phép tải ảnh lên, điều khiển (bật/tắt) camera.
 - Hiển thị luồng video từ webcam và các kết quả nhận diện (khuôn mặt được khoanh vùng, nhãn cảm xúc dự đoán).
 - Gửi các yêu cầu HTTP (ví dụ: POST để tải ảnh, POST để điều khiển camera, GET để nhận luồng video) đến Máy chủ Ứng dụng (FastAPI)
- Máy chủ Ứng dụng (Server – FastAPI)
 - Là trung tâm xử lý logic và điều phối các hoạt động của hệ thống, được xây dựng bằng FastAPI (Python).
 - Tiếp nhận các yêu cầu HTTP từ Giao diện Người dùng qua các RESTful API (ví dụ: /detect_emotion cho ảnh tải lên, /video_feed cho luồng video, /camera/{action} để quản lý camera).
 - Thực hiện xử lý logic nghiệp vụ:
 - Đối với ảnh tải lên: gọi module xử lý ảnh để phát hiện khuôn mặt, sau đó gọi module mô hình để dự đoán cảm xúc và trả kết quả về cho frontend.
 - Đối với luồng video: liên tục lấy khung hình từ camera, xử lý từng khung hình để phát hiện khuôn mặt, dự đoán cảm xúc, vẽ kết quả lên khung hình và truyền (stream) về frontend.
 - Quản lý trạng thái và điều khiển thiết bị camera.
- Xử lý Backend và mô hình: bao gồm phát hiện khuôn mặt, tiền xử lý ảnh cho mô hình cảm xúc, khởi tạo và vận hành mô hình Nhận diện cảm xúc và dự đoán cảm xúc



Hình 7: Mô hình kiến trúc hệ thống

CHƯƠNG 3: CÀI ĐẶT CHƯƠNG TRÌNH

I. CÀI ĐẶT HỆ THỐNG VÀ TRIỂN KHAI HỆ THỐNG

1. Một số công cụ sử dụng.

- **Visual Studio Code (VS Code):**

- Trình soạn thảo mã nguồn phổ biến với nhiều tiện ích mở rộng hỗ trợ phát triển dự án.
- Tích hợp Git giúp dễ dàng quản lý và kiểm soát phiên bản mã nguồn.

2. Thư viện hỗ trợ.

- FastAPI: Framework web Python hiện đại, hiệu suất cao, dùng để xây dựng các API endpoint cho hệ thống.
- Uvicorn: ASGI server dùng để chạy ứng dụng FastAPI.
- OpenCV (cv2): Thư viện xử lý ảnh và video, dùng để phát hiện khuôn mặt và xử lý frame từ camera.
- NumPy: Thư viện tính toán số học, xử lý mảng dữ liệu hình ảnh.
- TensorFlow/Keras: Framework học sâu, dùng để xây dựng và sử dụng mô hình nhận diện cảm xúc (VGG19).
- Pillow (PIL): Thư viện xử lý ảnh hỗ trợ đọc, ghi và chuyển đổi định dạng ảnh.
- Base64, io: Hỗ trợ mã hóa/giải mã và xử lý dữ liệu nhị phân.
- Pydantic: Hỗ trợ kiểm tra và xác thực dữ liệu đầu vào cho FastAPI.
- Bootstrap 5: Framework CSS giúp xây dựng giao diện web hiện đại, responsive.
- Chart.js: Thư viện JavaScript để hiển thị biểu đồ xác suất cảm xúc trên giao diện web.
- Font Awesome: Thư viện icon vector cho giao diện người dùng.
- JavaScript (ES6+): Xử lý logic phía client, gọi API, điều khiển giao diện động.

3. Cài đặt và triển khai phía Backend

- **Chuẩn bị môi trường:**

- Bước 1: Khởi tạo thư mục dự án

- Bước 2: Tạo môi trường ảo python venv bằng câu lệnh [python -m venv venv] trên Windows và [source venv/bin/activate] trên Linux/macOS
- Bước 3: Kích hoạt môi trường ảo python venv bằng câu lệnh [venv\Scripts\activate] với window
- Bước 4: Đặt các thư viện cần thiết cho dự án trong file requirements.txt
- Bước 5: Cài đặt các thư viện cần thiết bằng câu lệnh [pip install -r requirements.txt]
- **Triển khai Backend:**
 - Backend của hệ thống được triển khai thông qua ba module chính:
 - app.py - File chính định nghĩa các API endpoint và xử lý các request từ client. Các chức năng chính bao gồm:
 - Khởi tạo ứng dụng FastAPI và mô hình AI
 - Quản lý kết nối camera
 - Cung cấp video stream xử lý theo thời gian thực
 - Xử lý tải lên ảnh và phân tích cảm xúc
 - Phục vụ giao diện web
 - model.py - Mô hình nhận diện cảm xúc, chịu trách nhiệm:
 - Tạo mô hình nhận diện cảm xúc dựa trên VGG19
 - Tải trọng số đã huấn luyện
 - Thực hiện dự đoán cảm xúc từ ảnh khuôn mặt
 - utils.py - Xử lý hình ảnh, cung cấp thông tin về :
 - Hằng số và ánh xạ tên cảm xúc
 - Hàm phát hiện khuôn mặt sử dụng Haar Cascade
 - Hàm vẽ khung và nhãn cảm xúc lên hình ảnh
 - Khởi chạy backend:
 - Để chạy backend sử dụng lệnh [python app.py] hoặc sử dụng Uvicorn trực tiếp: [uvicorn app:app --host 127.0.0.1 --port 5000 --reload]
 - Server sẽ chạy tại địa chỉ http://127.0.0.1:5000 với tài liệu API tự động sinh tại http://127.0.0.1:5000/docs.

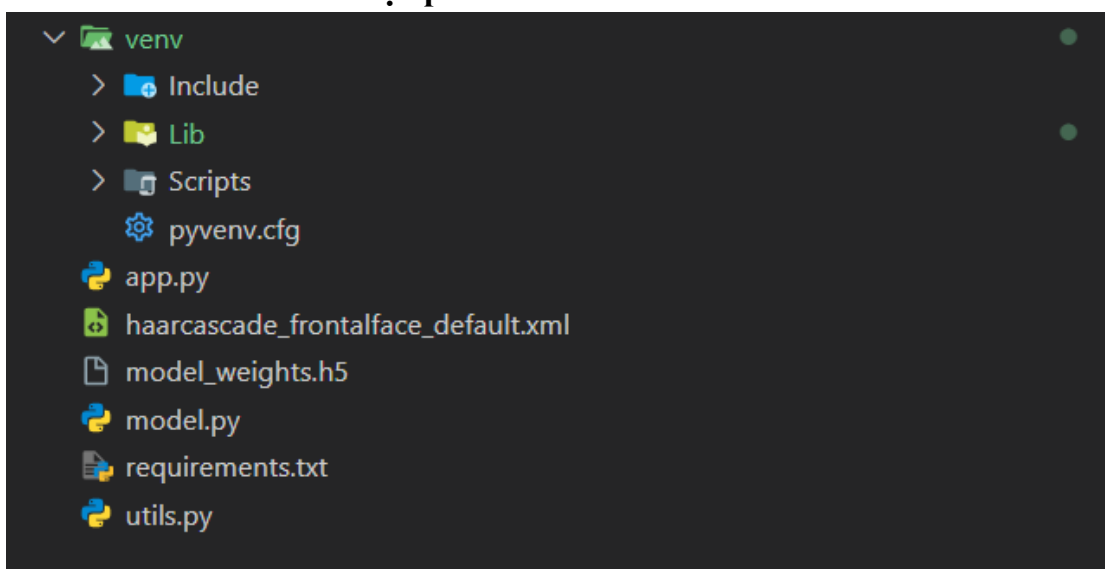
4. Cài đặt và triển khai phía FE

- **Cấu trúc giao diện:** Giao diện được chia thành các thành phần chính:
 - Thanh điều hướng với logo và công tắc chuyển đổi theme
 - Phần hiển thị video stream và điều khiển camera
 - Biểu đồ xác suất cảm xúc
 - Khu vực tải lên ảnh và chụp ảnh mới
 - Phần hiển thị kết quả nhận diện

- **Triển khai Frontend:** File index.html trong thư mục templates chứa toàn bộ giao diện người dùng, bao gồm HTML, CSS và JavaScript. Dưới đây là các thành phần chính:

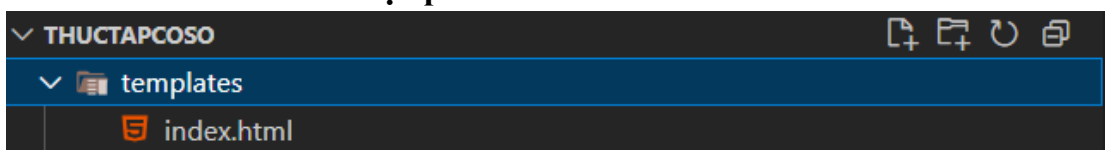
II. KẾT QUẢ CÀI ĐẶT

- **Triển khai Backend:**
Cấu trúc thư mục phần backend



Hình 8: Cấu trúc thư mục phần backend

- **Triển khai Frontend:**
Cấu trúc thư mục phần frontend



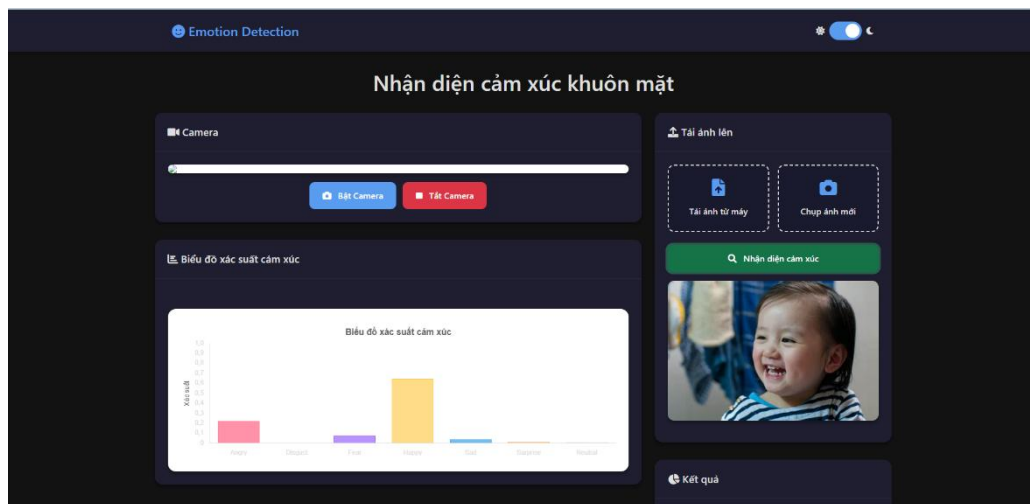
Hình 9: Cấu trúc thư mục phần frontend

III. KẾT LUẬN

1. Kết quả đạt được

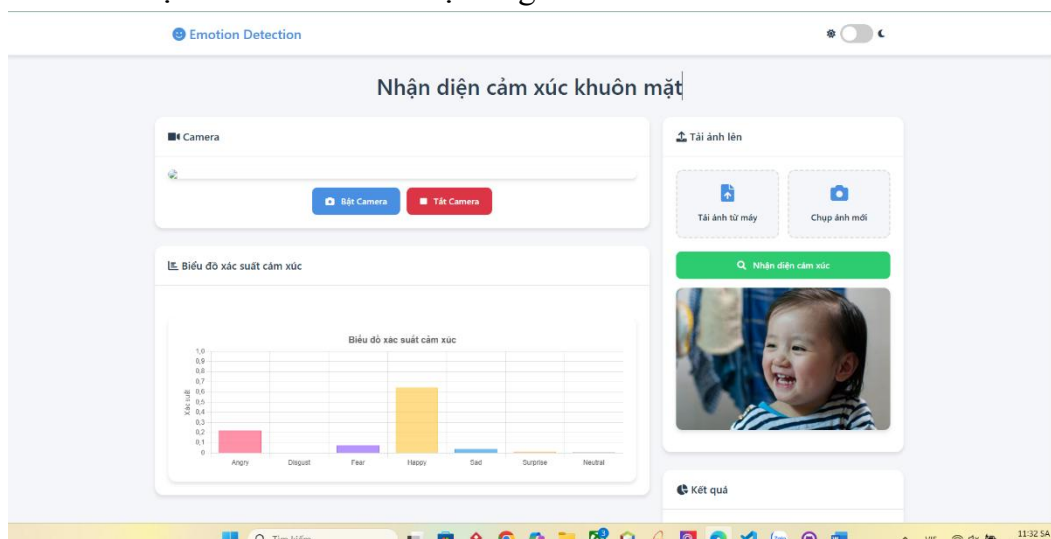
a) Các giao diện chính:

- Giao diện chính khi ở chế độ Tối:



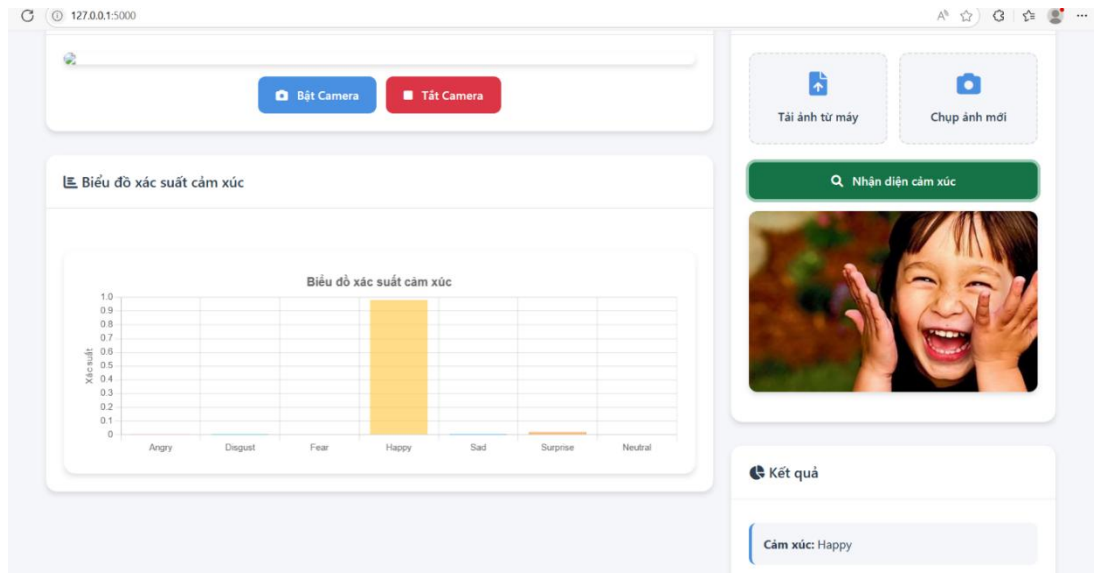
Hình 10: Giao diện chính khi ở chế độ Tối

- Giao diện chính khi ở chế độ Sáng:



Hình 11: Giao diện chính khi ở chế độ sáng

- Giao diện tải ảnh lên và hiện kết quả cảm xúc



Hình 12: Giao diện đã tải ảnh lên và hiện kết quả cảm xúc

CHƯƠNG 4: PHẦN KẾT LUẬN

1. Tóm tắt các nội dung đã thực hiện.

- Website đã hoàn thành được những mục tiêu ban đầu đặt ra
- Hoàn thành những nội dung cơ bản như:
 - Website đã được xây dựng thành công và hoạt động ổn định
 - Hệ thống có thể phát hiện khuôn mặt từ ảnh tĩnh hoặc video stream, phân tích và dự đoán cảm xúc, hiển thị kết quả theo thời gian thực hoặc ảnh tĩnh
 - Huấn luyện và triển khai mô hình nhận diện cảm xúc khuôn mặt: dựa vào mô hình VGG19 có sẵn áp dụng transfer learning để huấn luyện mô hình mới phù hợp với đầu ra của dự án, tích hợp mô hình lên web

2. Những điểm đạt được và hạn chế của hệ thống.

- Điểm đạt được:

- Nhận diện cảm xúc đa phương thức:
 - Người dùng có thể nhận diện cảm xúc qua luồng video trực tiếp từ camera
 - Hỗ trợ tải ảnh tĩnh lên để phân tích cảm xúc
 - Cho phép người dùng chụp ảnh mới trực tiếp từ camera trên giao diện để phân tích
- Giao diện người dùng tương tác và trực quan:
 - Cung cấp giao diện web để người dùng dễ dàng tương tác với các chức năng như bật/tắt camera, tải ảnh, chụp ảnh
 - Hiển thị luồng video trực tiếp và vẽ hộp nhận diện khuôn mặt cùng với nhãn cảm xúc
 - Hiển thị biểu đồ xác suất cho các cảm xúc được nhận diện từ ảnh, giúp người dùng hiểu rõ hơn về kết quả của mô hình

- Hạn chế:

- Các chức năng còn ít và mới chỉ xây dựng ở mức cơ bản
- Giao diện người dùng có thể cần được tối ưu hóa thêm để trở nên thân thiện và dễ sử dụng hơn
- Tốc độ xử lý và phản hồi của hệ thống trong một số trường hợp vẫn chưa được hoàn thiện

3. Hướng phát triển trong tương lai.

- Cải tiến giao diện người dùng. Tiếp tục tối ưu hóa giao diện để trở nên dễ sử dụng hơn, đặc biệt là đối với người dùng không quen với công nghệ. Đảm bảo hệ thống giao diện linh hoạt, dễ dàng tương thích với nhiều thiết bị và nền tảng.
- Cải thiện mô hình nhận diện cảm xúc với độ chính xác và các chỉ số cao hơn
- Tích hợp dự án vào các dự án lớn hơn như Giám sát tâm trạng trong lớp học, phân tích cảm xúc khách hàng,...

TÀI LIỆU THAM KHẢO

- [1] W3schools: <https://www.w3schools.com/>
- [2] WikipediA[Online]:
https://en.wikipedia.org/wiki/Convolutional_neural_network
<https://en.wikipedia.org/wiki/VGGNet>
- [3] Puri, Raghav, et al. "Emotion detection using image processing in python." *arXiv preprint arXiv:2012.00659* (2020).
- [4] Rajesh, K. M., & Naveenkumar, M. (2016, December). A robust method for face recognition and face emotion detection system using support vector machines. In *2016 international conference on electrical, electronics, communication, computer and optimization techniques (ICEECCOT)* (pp. 1-5). IEEE.
- [5] Sarvakar, K., Senkamalavalli, R., Raghavendra, S., Kumar, J. S., Manjunath, R., & Jaiswal, S. (2023). Facial emotion recognition using convolutional neural networks. *Materials Today: Proceedings*, 80, 3560-3564.
- [6] Abbassi, N., Helaly, R., Hajjaji, M. A., & Mtibaa, A. (2020, December). A deep learning facial emotion classification system: a VGGNet-19 based approach. In *2020 20th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)* (pp. 271-276). IEEE.
- [7] Tripathi, M. (2021). Facial emotion recognition using convolutional neural network. *ICTACT Journal on Image and Video Processing*, 12(01).