TRƯỜNG ĐẠI HỌC CẦN THƠ

KHOA CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

NIÊN LUẬN NGÀNH

NGÀNH KHOA HỌC MÁY TÍNH

Đề tài

ỨNG DỤNG NHẬN DIỆN KHUÔN MẶT ĐỂ MỞ KHOÁ TỰ ĐỘNG

Sinh viên thực hiện

Đào Công Tính

B1709632, Khóa 43

Tạ Đặng Vĩnh Phúc

B1709618, Khóa 43

Học kỳ I, Năm học: 2020-2021

TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
BỘ MÔN KHOA HỌC MÁY TÍNH

NIÊN LUẬN NGÀNH
NGÀNH KHOA HỌC MÁY TÍNH

Đề tài

ỨNG DỤNG NHẬN DIỆN KHUÔN MẶT ĐỂ MỞ KHOÁ TỰ ĐỘNG

GVHD	**Sinh viên thực hiện**
TS. Trần Nguyễn Minh Thư	Đào Công Tính
TS. Trần Việt Châu	B1709632, Khóa 43
	Tạ Đặng Vĩnh Phúc
	B1709618, Khóa 43

Học kỳ 01, Năm học: 2020-2021

LỜI CẢM ƠN

Lời đầu tiên, chúng em xin chân thành cảm ơn đến Cô TS. Trần Nguyễn Minh Thư và Thầy TS. Trần Việt Châu đã giới thiệu và hướng dẫn chúng em thực hiện đề tài niên luận ngành đầy ý nghĩa và bổ ích. Chúng em có thể hoàn thành đề tài này nhờ sự tận tình dìu dắt, hỗ trợ và tạo điều kiện tốt nhất của Thầy Cô trong suốt quá trình thực hiện đề tài. Qua những góp ý của Thầy Cô đã giúp chúng em hoàn thiện hơn kiến thức về chuyên ngành Khoa học máy tính.

Với lòng biết ơn sâu sắc nhất, chúng em xin cảm ơn Thầy Cô đã nhận xét và góp ý quan trọng nhằm cải thiện nâng cao kiến thức về lĩnh vực máy học nói riêng và ngành Khoa học máy tính nói chung.

Mặc dù, chúng em đã cố gắng nỗ lực để hoàn thành niên luận ngành nhưng không thể tránh khỏi những thiếu sót về mặt hạn chế kiến thức và kinh nghiệm thực tiễn. Chúng em rất mong nhận được những lời nhận xét và đánh giá từ Thầy Cô để chúng em củng cố và hoàn thiện kiến thức.

Kính chúc Thầy Cô nhiều sức khỏe và thành công.

Trân trọng.

Cần Thơ, ngày 11 tháng 12 năm 2020

Người viết

Đào Công Tính

NHẬN XÉT CỦA GIẢNG VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Cần Thơ, ngày 11, tháng 12, năm 2020

Giáo viên hướng dẫn

TS. Trần Nguyễn Minh Thư

MỤC LỤC

DANH MỤC HÌNH

DANH MỤC BẢNG

DANH MỤC TỪ ĐIỂN – VIẾT TẮT

STT	Thuật ngữ	Diễn giải
1	HOG	Histogram of oriented gradient
2	SIFT	Scale Invariant Feature Transform
3	SVM	Support Vector Machine
4	GUI	Graphical User Interface

TÓM TẮT

Trong bối cảnh hội nhập quốc tế hiện nay cùng với sự phát triển mạnh mẽ của cuộc cách mạng công nghiệp 4.0, tin học hóa toàn cầu hứa hẹn sẽ mang lại bước đột phá mới. Những tiến bộ trong lĩnh vực trí tuệ nhân tạo nói chung và máy học nói riêng đã và đang được ứng dụng rộng rãi trong nhiều lĩnh vực khác nhau. Đặc biệt, các nhiệm vụ thị giác máy tính để phát hiện đối tượng trong những năm gần đây đã đạt được nhiều thành tựu đáng kể, thậm chí còn tốt hơn khả năng nhận dạng của con người.

Mục đích của nghiên cứu này là nhận dạng khuôn mặt trong thời gian thực và mở khoá tự động, tức là sử dụng camera và khoá điện được kết nối với máy tính Raspberry. Nhận dạng khuôn mặt sử dụng hai camera và xử lý để tránh trường hợp không phải con người, hay bị tác động bởi môi trường chẳng hạn như độ sáng, góc quay. các phương pháp học máy kết hợp với máy tính Raspberry Pi 3 được sử dụng để nhận diện khuôn mặt một cách tự động đồng thời khoá điện sẽ tự động mở nếu xác thực chính xác (verify). Phương pháp đề xuất được thử nghiệm trên nhiều tình huống khác nhau và dự kiến sẽ áp dụng trong các trường hợp thực tế. Thư viện OpenCV để phát hiện khuôn mặt và sử dụng mô hình học sâu để nhận dạng khuôn mặt. Chúng tôi đánh giá kết quả nhận dạng dựa trên các độ đo khác nhau ACC, Log loss tương ứng là 100% và 0.009.

CHƯƠNG I. GIỚI THIỆU

I. ĐẶT VẤN ĐỀ

Trong những năm gần đây, sự phát triển nhanh chóng của trí tuệ nhân tạo đã tạo động lực để các nhà khoa học nghiên cứu những ứng dụng thông minh đáp ứng nhu cầu của con người trong xã hội hiện đại. Trí tuệ nhân tạo nói chung và xử lý ảnh hay thị giác máy tính nói riêng đã và đang thúc đẩy và phát triển mạnh mẽ với những tiến bộ trong mọi lĩnh vực của xã hội như an ninh, y tế, chính phủ, thương mại điện tử, bán lẻ, giáo dục và nhiều lĩnh vực khác. Một trong những ứng dụng quan trọng trong xử lý hình ảnh là Nhận dạng khuôn mặt ngày càng được quan tâm nhiều hơn. Vì vậy, nhận dạng khuôn mặt đã dần trở thành một lĩnh vực không chỉ quan trọng mà còn là một lĩnh vực phổ biến với nhiều ứng dụng có lợi. Có một loạt các ứng dụng liên quan của phân tích hình ảnh như chẩn đoán bệnh tật, sản xuất phim, tương tác giữa người và máy, an ninh và quốc phòng, v.v. được đặc biệt áp dụng trong việc tham dự.

Hệ thống nhận dạng khuôn mặt là một loại phần mềm sinh trắc học tự động xác định một người nào đó từ một bức hình ảnh kỹ thuật số hoặc một khung hình video. Có nhiều cách để thực hiện điều này, tuy nhiên một trong các phương pháp đó là so sánh các đặc điểm khuôn mặt chọn trước từ hình ảnh và một cơ sở dữ liệu về khuôn mặt. Có hai kỹ thuật để phát hiện khuôn mặt là dựa trên mô hình và dựa trên tính năng. Trong những năm qua, đã có nhiều nghiên cứu về vấn đề nhận dạng khuôn mặt người từ đen trắng, xám, và màu. Nghiên cứu đi từ vấn đề đơn giản là ảnh chỉ có mặt người nhìn thẳng vào máy ảnh và đầu ở tư thế thẳng đứng, đến ảnh màu có nhiều mặt người trong cùng một bức ảnh, khuôn mặt có góc quay nhỏ hoặc một phần được che mờ bằng những hình ảnh nền của ảnh ngoại cảnh nhằm đáp ứng nhu cầu thực sự cần thiết của mọi người. Bài toán xác định khuôn mặt người là sử dụng kỹ thuật máy tính để xác định vị trí và kích thước của khuôn mặt trong ảnh kỹ thuật số. Kỹ thuật này nhận dạng các đặc điểm khuôn mặt và bỏ qua những thứ theo ngữ cảnh như tòa nhà, cây cối, đường xá, v.v. Có nhiều giai đoạn trong hệ thống nhận dạng khuôn mặt bao gồm thu thập hình ảnh, xây dựng cơ sở dữ liệu, phát hiện khuôn mặt, xử lý trước, trích xuất đối tượng và giai đoạn phân loại. Với việc trích xuất khuôn mặt người như đã nói ở trên, nó được camera thu thập từ hình ảnh và xác định các đối tượng trong hệ thống dựa trên hình ảnh khuôn mặt đã được trích xuất.

Ngày nay, xu hướng sử dụng thiết bị nhà thông minh (Smart Home) ngày càng trở nên phổ biến đó là khi một khu vực sinh sống được ứng dụng các thiết bị công nghệ được kết nối với nhau một cách linh hoạt bao gồm: điều hòa, tủ lạnh, lò vi sóng, tivi thông minh, đèn thông minh, bộ sạc thông minh, chuông cửa thông minh,... thông qua các thiết bị smartphone. Các công nghệ nhà thông minh hứa hẹn sẽ làm cho không

gian sống của chúng ta thoải mái, thuận tiện và an toàn hơn. Một trong số các thiết bị an ninh và đáp ứng nhu cầu sinh hoạt của con người đó là thiết bị mở cửa tự động ứng dụng nhận diện gương mặt. Ứng dụng nhận diện khuôn mặt vào môi trường nhà ở dân dụng để nhận diện khuôn mặt để mở cửa mang đến những lợi ít tuyệt vời cho người dùng như không cần sử dụng chìa khóa (dễ thất lạc và hao mòn), giúp phát hiện sự xuất hiện của những người lạ mặt (danh sách đen), lưu trữ video thông minh giúp trích xuất sự kiện đã diễn ra. Ngoài ra, Đài CGTN (phiên bản quốc tế của Đài truyền hình trung ương CCTV của Trung Quốc) tường thuật các nhà khoa học ở thành phố Quảng Châu, tỉnh Quảng Đông ở đông nam Trung Quốc đã phát hiện virus corona chủng mới xuất hiện ở môi trường bên ngoài đó là axit nucleic của virus corona chủng mới (2019-nCoV) trên tay nắm cửa tại nhà của một bệnh nhân nhiễm loại virus mới này. Do đó, trong tình hình dịch bệnh diễn biến phức tạp hiện nay, thiết bị mở cửa tự động bằng camera nhận dạng gương mặt sẽ mang lại lợi ích đáng kể vì người dùng không phải chạm tay hay tiếp xúc với tay nắm cửa nên hạn chế sự lây lan của dịch bệnh.

II. LỊCH SỬ GIẢI QUYẾT VẤN ĐỀ

Trong nhiều phương pháp hiện đại, học sâu như một cách tiếp cận để trích xuất mô tả phân cấp của dữ liệu trong bối cảnh nhận dạng khuôn mặt. Học sâu là một khuôn khổ quan trọng trong học máy liên quan đến một loạt các thuật toán giải quyết các vấn đề khác nhau bao gồm hình ảnh, văn bản và giọng nói để đạt được kết quả cao. Song song giữa việc phát triển học sâu và mạng nơ-ron tích hợp (CNN), độ chính xác và giảm thời gian nhận dạng khuôn mặt. Trong suốt hai thập kỷ qua, nhận dạng khuôn mặt đã được chú ý đáng kể. Các nhà nghiên cứu đã đề xuất nhiều thuật toán nhận dạng khuôn mặt như Eigenfaces [1], Fisherfaces [2] (PCA [3] + LDA [4]), phân tích thành phần độc lập, phân tích đặc điểm cục bộ, đối sánh đồ thị chùm đàn hồi (EBGM) [5].

Như đã đề cập ở trên, nhận dạng khuôn mặt là một chủ đề nghiên cứu quan trọng. Bên cạnh đó, việc giải quyết vấn đề bối cảnh phức tạp bằng cách sử dụng các thuật toán tốt nhất xử lý được tính toán cao để xử lý thời gian thực. Cách tiếp cận nhận dạng khuôn mặt đã cung cấp rất nhiều khả năng quan sát trong nhiều ứng dụng phân tích hình ảnh. Hiện nay, một loạt các dự án cạnh tranh về đổi mới sinh trắc học đang diễn ra, đặc biệt là nhận dạng khuôn mặt. Các công ty lớn nhất trên thế giới như Google, Apple, Facebook, Amazon và Microsoft phát hiện nhanh chóng để triển khai phân tích thông minh luồng video trong điều kiện thực tế.

Năm 2018, Cục Khoa học và Công nghệ An ninh Nội địa Hoa Kỳ đã trình bày các kết quả tiềm năng của hệ thống nhận dạng khuôn mặt tốt nhất. Bên cạnh đó, Đại

học Hồng Kông Trung Quốc đã nhận xét thành công khi xác định bằng Mô hình biến tiềm ẩn quá trình Gaussian phân biệt (DGPLVM) có tên GaussianFace [6] rằng con số cho điểm tốt hơn so với con người là một tiến bộ lớn. Hơn nữa, Facebook đã phát triển một chương trình nổi tiếng, đó là DeepFace [7], có thể xác định xem hai khuôn mặt có thuộc cùng một cá nhân hay không. Mặt khác, Google đã đi tốt hơn với FaceNet [8] được sử dụng tập dữ liệu Khuôn mặt được gắn nhãn trong vùng hoang dã (LFW), FaceNet đạt được độ chính xác kỷ lục là 99,63%. OpenFace [9] được phát triển bởi Mountain View đã chứng tỏ tầm quan trọng trong bối cảnh sinh trắc học.

Tỷ lệ lỗi nhận dạng khuôn mặt đã giảm gấp ba lần trong hai mươi năm qua [10]. khi nhận dạng khuôn mặt chính diện trong ảnh tĩnh được chụp trong môi trường được kiểm soát (hạn chế) không nhất quán. Nhiều nhà cung cấp triển khai các hệ thống phức tạp để ứng dụng kiểm soát biên giới và nhận dạng sinh trắc học thông minh [11]. Nhận dạng khuôn mặt bao gồm hai bước, trong bước đầu tiên, khuôn mặt được phát hiện trong hình ảnh và sau đó những khuôn mặt được phát hiện này được so sánh với cơ sở dữ liệu để xác minh. Một số phương pháp đã được đề xuất như thuật toán Ada Boost, thuật toán Float Boost, thuật toán S-Ada Boost Hỗ trợ máy véc-tơ (SVM) và bộ phân loại Bayes. Hiệu quả của thuật toán nhận dạng khuôn mặt có thể được tăng lên với thuật toán nhận diện khuôn mặt nhanh. Trong tất cả các phương pháp trên. Phương pháp Tăng tốc tính năng mạnh mẽ (SURF) là hiệu quả nhất. Các tác giả trong hệ thống đã sử dụng thuật toán này để phát hiện khuôn mặt trong hình ảnh phòng làm việc. Các tác giả trong đã đề xuất một phương pháp cho hệ thống điểm danh của học sinh trong lớp học sử dụng kỹ thuật nhận dạng khuôn mặt bằng cách kết hợp Biến đổi Wavelet rời rạc (DWT) và Biến đổi Cosin rời rạc (DCT). Các thuật toán này được sử dụng để trích xuất các đặc điểm của khuôn mặt của một học sinh, sau đó áp dụng Hàm cơ bản xuyên tâm (RBF) để phân loại các đối tượng trên khuôn mặt. Hệ thống này đạt tỷ lệ chính xác là 82%.

III. MỤC TIÊU ĐỀ TÀI

Mục tiêu đề tài là xây dựng Hệ thống khóa cửa nhận dạng khuôn mặt nhằm giải quyết vấn đề an ninh của ngôi nhà. Hệ thống được phát triển bằng việc ứng dụng máy tính Raspberry-pi 3 và các thuật toán nhận dạng từ thư viện OpenCV dùng để ràng buộc khi khuôn mặt được nhận dạng chính xác thì khóa cửa được mở và ngược lại. Ngoài ra, chủ sở hữu ngôi nhà có thể kiểm tra từng người truy cập trong bảng điều khiển từ xa nên tránh được việc camera nhận dạng bằng ảnh sẽ không hoạt động. Đối với nhận dạng khuôn mặt, một hình ảnh sẽ được chụp bởi camera pi và được xử lý trước bởi Raspberry pi như chuyển đổi, thay đổi kích thước và cắt xén. Sau đó, nhận diện và phát hiện khuôn mặt được thực hiện. Sau khi bộ phân loại nhận dạng khuôn

mặt dựa trên thư viện hình ảnh được lưu trữ trước, hình ảnh sẽ được gửi đến bảng điều khiển từ xa để chờ quyết định của chủ sở hữu ngôi nhà.

IV. ĐỐI TƯỢNG VÀ PHẠM VI NGHIÊN CỨU

1. Đối tượng nghiên cứu

- Ngôn ngữ lập trình Python và các thư viện hỗ trợ khác
- Các thuật toán học sâu/ máy học dành cho nhận diện khuôn mặt
- Công cụ Raspberry Pi 3
- Camera chuyên dụng

2. Phạm vi nghiên cứu

- Ứng dụng trong nhà dân dụng đảm bảo an ninh
- Hệ thống tự động mở khóa cửa bằng nhận diện khuôn mặt

V. PHƯƠNG PHÁP NGHIÊN CỨU

- Tài liệu và bài báo khoa học về nhận diện gương mặt
- Hướng dẫn sử dụng và cài đặt công cụ Raspberry Pi 3

VI. NHỮNG ĐÓNG GÓP CHÍNH CỦA ĐỀ TÀI

- Lấy mẫu trực tiếp thời gian thực để có thể huấn luyện cho nhận dạng thành viên mới.
- Sử dụng mô hình SVM cho huấn luyện và kỹ thuật chống giả danh (thông qua hình ảnh).
- Nhận dạng khuôn mặt thời gian thực cho ngôi nhà thông minh.
- Sử dụng công cụ Raspberry cài đặt cho hộ gia đình sẽ tiết kiệm chi phí so với các camera giám sát trên thị trường.

VII. BẢNG PHÂN CÔNG CHI TIẾT

STT	Tuần	Nội dung công việc	Người thực hiện
1	01 - 02	- Xác định bài toán - Thiết kế kiến trúc tổng thể hệ thống	Đào Công Tính Tạ Đặng Vĩnh Phúc
2	02 - 03	- Thu thập dữ liệu	Đào Công Tính
3	04 - 06	- Khảo sát và chạy thực nghiệm các mô hình máy học	Đào Công Tính Tạ Đặng Vĩnh Phúc
		- Khảo sát hệ thống	Đào Công Tính Tạ Đặng Vĩnh Phúc
4	07 - 10	- Xây dựng mô hình máy học	Đào Công Tính
		- Kiểm thử mô hình	Tạ Đặng Vĩnh Phúc
5	11 - 15	- Cài đặt hệ thống trên Raspberry Pi	Tạ Đặng Vĩnh Phúc
6	16 - 17	- Kiểm thử và đánh giá	Đào Công Tính Tạ Đặng Vĩnh Phúc
7	17 - 18	- Tổng hợp kết quả thực nghiệm	Đào Công Tính
8	18 - 19	- Viết báo cáo - Báo cáo	Đào Công Tính Tạ Đặng Vĩnh Phúc

VIII. BỐ CỤC CỦA QUYỂN NIÊN LUẬN

Bố cục gồm có 6 chương:

CHƯƠNG I. Giới thiệu

CHƯƠNG II. Mô tả bài toán

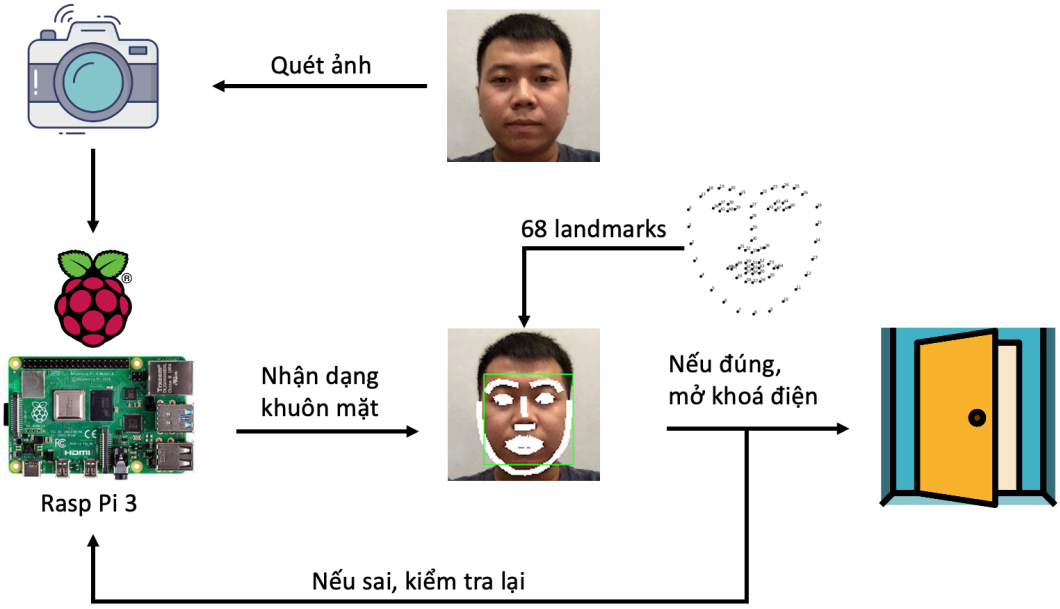
CHƯƠNG III. Thiết kế và cài đặt hệ thống

CHƯƠNG IV. Kiểm thử và đánh giá

CHƯƠNG V. Kết luận

CHƯƠNG II. MÔ TẢ BÀI TOÁN

I. MÔ HÌNH TỔNG QUAN



Hình 1. Lưu đồ hoạt động của hệ thống

Hình 1 mô tả sơ đồ hoạt động của hệ thống khóa cửa nhận dạng khuôn mặt. Theo sơ đồ trên, hệ thống được chia ra bốn công việc chính:

- **Quét ảnh:** Người dùng đứng trước camera đã được cố định khung cố kích thước 400×400 pixels. Khi này, hệ thống sẽ bắt mỗi 30 mili giây trên 1 khung để có thể xử lý và nhận dạng ở bước tiếp theo.
- **Nhận dạng khuôn mặt:**
  + Đầu vào: mô hình *"face_recognition_model_location"* trong thư viện *dlib* [12] để huấn luyện mô hình này nhằm tạo ra 128 đặc trưng
  + Đầu ra: véc-tơ 128 đặc trưng
- **Huấn luyện mô hình:** để có thể nhận dạng cần phải có mô hình chính xác. Ở đây chúng tôi đề xuất giải thuật SVM để huấn luyện.
- **Truyền tín hiệu mở khoá:** kết quả được trả về ở bước nhận dạng, nếu là thành viên trong gia đình sẽ truyền tín hiệu để mở khoá điện ngược lại sẽ không mở khoá.

II. MÔ TẢ TẬP DỮ LIỆU

Ở đây chúng tôi thực hiện lấy mẫu ở một hộ gia đình với 4 thành viên. Dữ liệu được sử dụng để làm mẫu là các véc-tơ 128 đặc trưng đã được mã hoá. Trung bình mỗi người sẽ có 360 mẫu trên tổng 1.440 mẫu (Bảng 1).

Bảng 1. Mô tả dữ liệu

Huấn luyện	Kiểm thử	Tổng
1.008	432	1.440

Đối với tập dữ liệu huấn luyện được thực hiện trên máy tính cá nhân, lấy dữ liệu từ camera trước với chất lượng là 30 khung trên giây (FPS). Mỗi người sẽ thực hiện lấy mẫu trong vòng một phút (thực hiện quay đầu trái, phải, lên, xuống để lấy được tất cả các góc của khuôn mặt). Tổng cộng có 1.800 mẫu tuy nhiên ở đây chỉ lấy cách nhau 5 ảnh nên tất cả thu được 360 mẫu trên một người. Thực hiện việc này cho 4 thành viên trong gia đình.

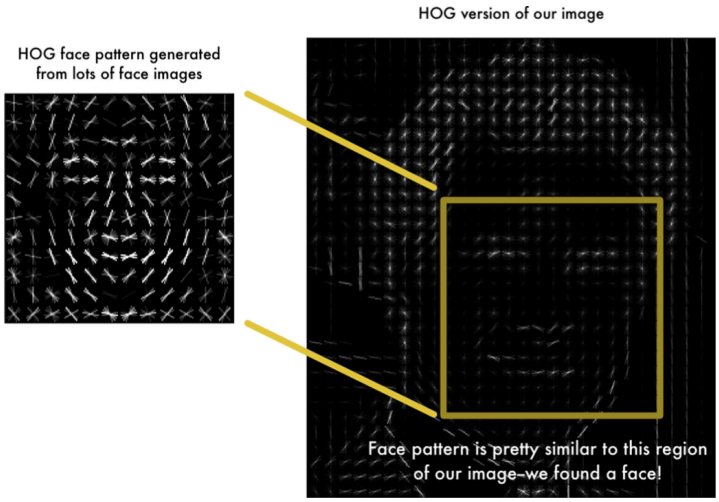
III. PHƯƠNG PHÁP THỰC HIỆN

1. Các thư viện hỗ trợ

Hệ thống sử dụng ngôn ngữ lập trình Python phiên bản 3.6.8 [13] và sự hỗ trợ mạnh mẽ của thư viện OpenCV [14] để có thể đọc vào một video hình ảnh và thao tác, xử lý ảnh đầu vào để có thể chuyển qua bước nhận dạng khuôn mặt. Ngoài ra, việc sử dụng thư viện *face_recognition* [15] để nhận dạng, mã hoá và tìm vị trí các điểm đặc trưng trên khuôn mặt và thư viện *sklearn* [16] để huấn luyện qua các mô hình máy học khác nhau với nghi thức k-fold (k = 10).

2. Nhận diện khuôn mặt bằng HOG

Bước đầu tiên, việc sử dụng đặc trưng Histogram of Oriented Gradients (HOG) để phát hiện con người, song song đó nó đã đạt được thành công lớn trong lĩnh vực thị giác máy tính. Với nhận dạng đối tượng nói chung và nhận dạng khuôn mặt nói riêng, HOG [17] gần đây đã minh họa để trở thành một bộ mô tả hiệu quả. HOG là các bộ mô tả hình ảnh bất biến đối với phép quay 2D vốn thường thấy trong nhiều vấn đề khác nhau trong thị giác máy tính nhằm mục đích phát hiện các đối tượng [18]. Chúng tôi đã nghiên cứu sức mạnh đại diện của các tính năng HOG để nhận dạng khuôn mặt và để xây dựng các bộ mô tả HOG mạnh mẽ. Một ví dụ về cấu trúc HOG của khuôn mặt được thể hiện dưới đây trong Hình 2.

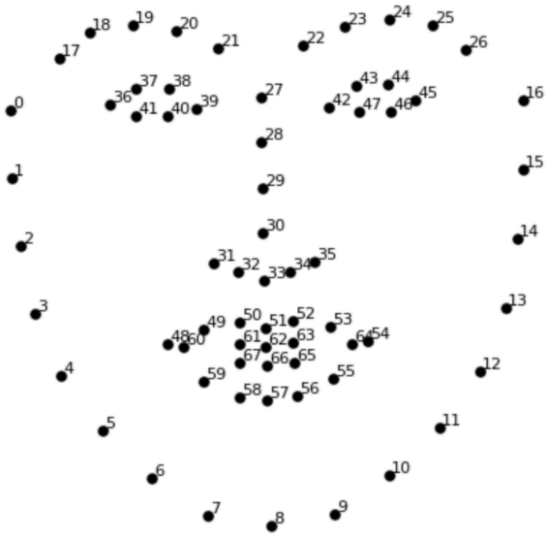


Hình 2. Tổng quan về cách tìm khuôn mặt trong ảnh bằng HOG

Tính năng phát hiện đặc trưng khuôn mặt bằng cách lấy mẫu đồng nhất các tính năng HOG nhằm loại bỏ dư thừa trong dữ liệu, cải thiện hiệu quả tính toán và tránh bị overfitting. Chúng tôi đề xuất sử dụng giảm chiều trong biểu diễn HOG. Kết quả sử dụng các tính năng HOG được trích xuất từ các kích thước bản vá hình ảnh khác nhau cải thiện đáng kể việc chọn một kích thước bản vá tốt nhất. Thứ nhất, hình ảnh được chia thành các ô nhỏ được kết nối cho mỗi tỷ lệ và việc xoay cũng có thể được thực thi bằng cách trích xuất các bộ mô tả chỉ từ các điểm chính trong không gian tỷ lệ của hình ảnh sau khi chuẩn hóa xoay. Các bước liên quan là Biến đổi tính năng bất biến quy mô (SIFT) để đạt được bất biến quy mô, gán định hướng để tìm hướng gradient chi phối và trích xuất bộ mô tả.

3. Mã hoá khuôn mặt với thư viện dlib

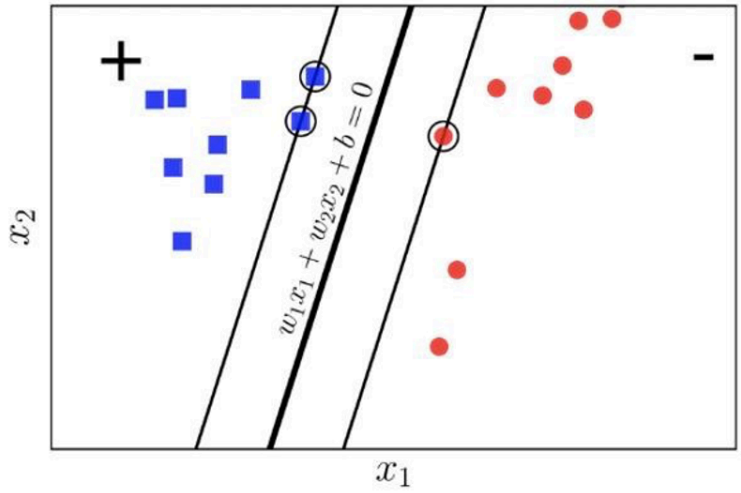
Để mã hóa một khuôn mặt, trước tiên chúng tôi phát hiện từng khuôn mặt trong ảnh. Sau đó, chúng tôi sẽ huấn luyện một thuật toán học máy để có thể tìm ra 68 điểm đặc trưng (được gọi là điểm landmarks) tồn tại trên mọi khuôn mặt như đỉnh cằm, mép ngoài của mỗi mắt, mép trong của mỗi lông mày, đường thẳng của mũi và môi (như trong Hình 3). Chúng tôi sử dụng mô hình *"face_recognition_model_location"* trong *dlib* [12] để đào tạo nó nhằm tạo ra 128 đặc trưng cho mỗi mã hóa khuôn mặt 128 chiều cho mỗi khuôn mặt trong hình ảnh.



Hình 3. Nhận diện khuôn mặt dựa trên 68 điểm Landmarks

4. Máy học véc-tơ hỗ trợ (Support Vector Machine – SVM)

Support Vector Machine (SVM) là một thuật toán thuộc học có giám sát dùng để phân chia dữ liệu (classification) thành các nhóm riêng biệt (như Hình 4) [19] SVM cố gắng tìm ra một "siêu phẳng" có khả năng phân chia bộ dữ liệu ra làm hai lớp (lớp âm và lớp dương) một cách tốt nhất:

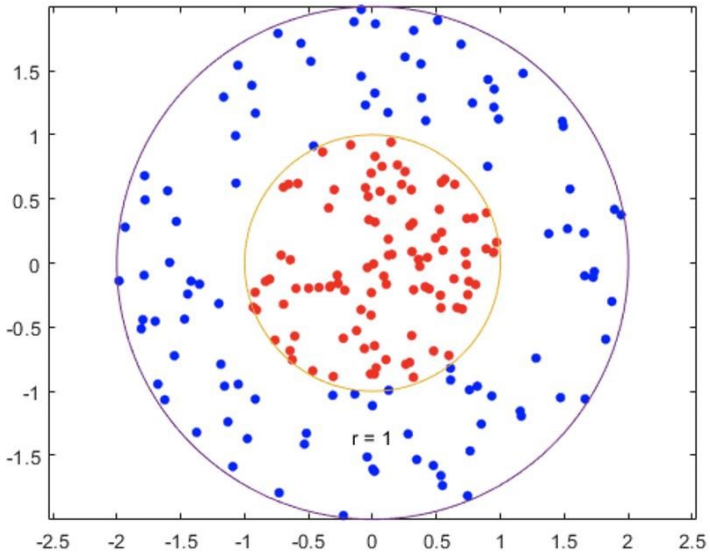


Hình 4. Mô hình SVM cho bài toán phân lớp [20]

Chúng tôi chọn giải thuật này các ưu điểm sau:

- **Xử lý trong không gian số chiều cao:** SVM đã được áp dụng với dữ liệu về gen, vốn có số chiều rất lớn. Chúng tôi áp dụng cho nghiên cứu gắn nhãn khuôn mặt của mình và kết quả thực tế cho thấy mô hình này rất phù hợp cho nhận diện khuôn mặt của chúng tôi khi đầu vào cần sử dụng một véc-tơ với số chiều lớn.

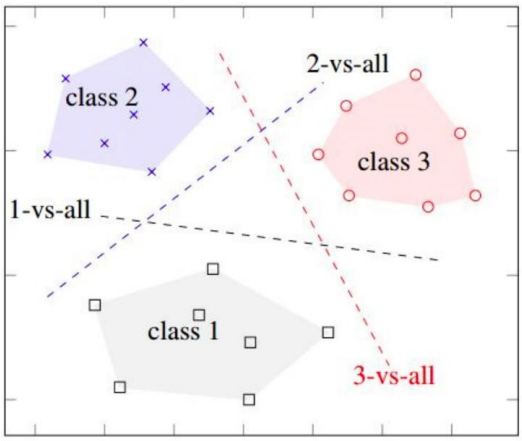
- **Tiết kiệm bộ nhớ:** Do chỉ có một tập hợp con của các điểm được sử dụng trong quá trình huấn luyện và ra quyết định thực tế cho các điểm dữ liệu mới nên chỉ có những điểm cần thiết mới được lưu trữ trong bộ nhớ khi ra quyết định.
- **Tính linh hoạt:** Phân lớp thường là phi tuyến tính thể hiện ở Hình 5 [21] Khả năng áp dụng Kernel mới cho phép linh động giữa các phương pháp tuyến tính và phi tuyến tính từ đó khiến cho hiệu suất phân loại lớn hơn, thay vì chỉ sử dụng siêu phẳng thuần tuý.



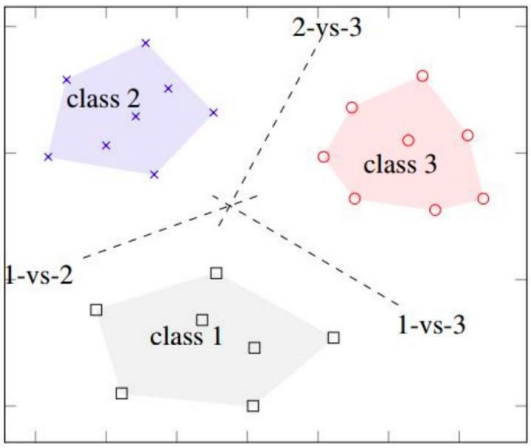
Hình 5. SVM với hàm nhân phi tuyến

Trong thực tế, chúng tôi cần phân lớp tới 40 học viên cho một lớp, SVM cũng có thể giải quyết bài toán lớn hơn 2 lớp bằng một trong 2 cách sau:

- **Cách thứ nhất (Hình 6: 1-vs-all) [22] :** Sử dụng n mô hình SVM, với n là số nhãn đầu vào, mỗi mô hình thứ i coi lớp i là lớp dương, các lớp còn lại đều được coi là lớp âm. Sau khi huấn luyện, từ n mô hình, nhãn cuối cùng sẽ được quyết định bằng cách: Điểm (score) của nhãn nào cao nhất thì kết quả chính là nhãn đó
- **Cách thứ hai (Hình 7: 1-vs-1) [23] :** Với mỗi 2 lớp phân biệt i và j bất kỳ ta huấn luyện một mô hình. Như vậy, chúng ta cần huấn luyện $n(n-1)/2$ mô hình SVM, với n là số nhãn đầu vào. Nhãn dự đoán sẽ được quyết định nhờ tổng điểm "tin cậy" cao nhất sau khi đưa vào $n(n-1)/2$ mô hình để dự đoán, tổng điểm tin cậy nhãn nào cao nhất thì kết quả của dự đoán sẽ là nhãn đó.

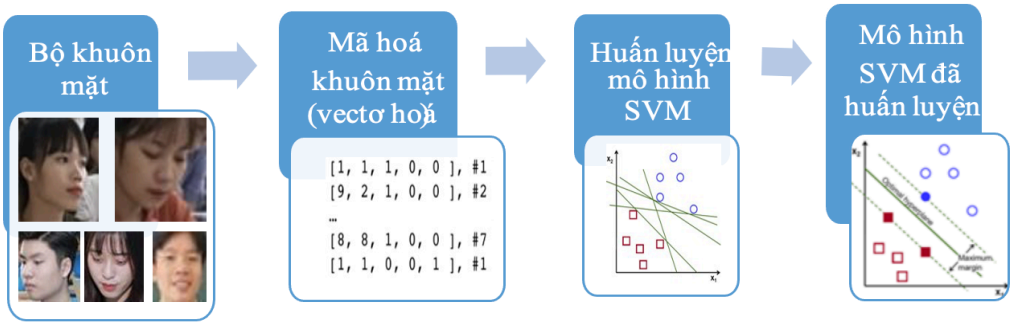


Hình 6. Giải quyết vấn đề SVM với bài toán đa lớp: 1-vs-all



Hình 7. Giải quyết vấn đề SVM với bài toán đa lớp: 1-vs-1

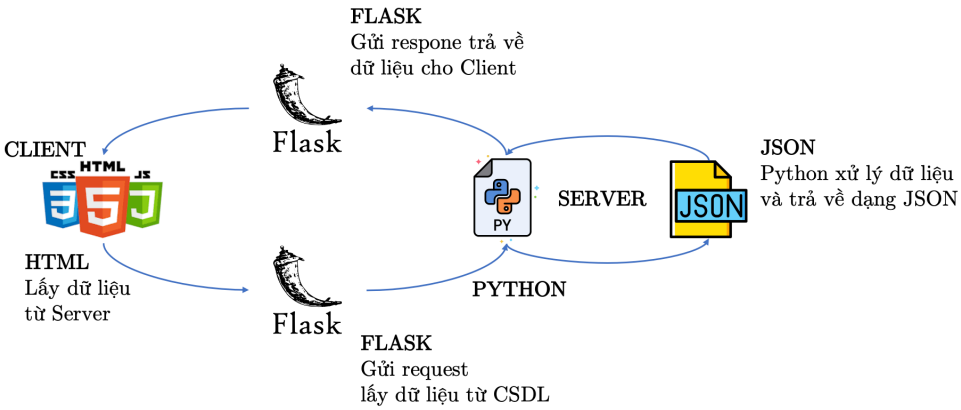
Cách thứ hai được đề xuất cho các mô hình như SVM, cộng với việc được hỗ trợ mặc định từ thư viện nổi tiếng scikit-learn cho SVM, chúng tôi quyết định sử dụng cách thứ hai này cho nghiên cứu của mình. Cũng trong bài toán nhận diện này, đầu vào của chúng tôi là một bộ các véc-tơ 128 chiều và tên khuôn mặt mà véc-tơ thể hiện, với số nhãn chính là số học viên trong lớp và một lớp "không nhận diện được". Sau huấn luyện, chúng tôi thu được một mô hình có khả năng nhận diện các thành viên bằng cách gắn tên cho mỗi gương mặt (như minh họa Hình 8).



Hình 8. Mô hình nhận diện khuôn mặt bằng thuật toán SVM

5. Flask Framework

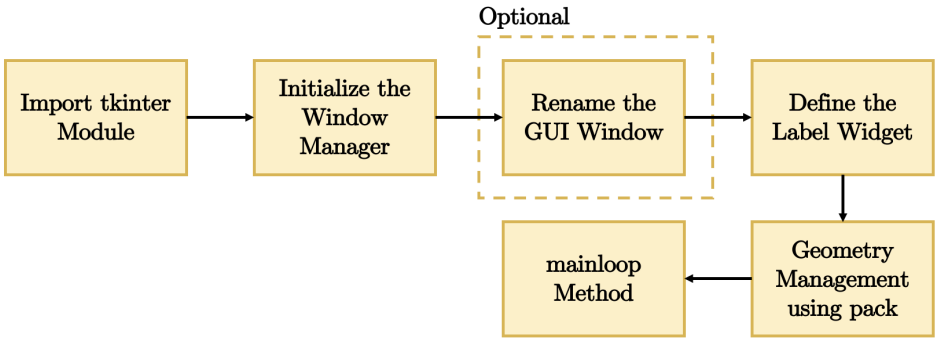
Flask [24] là một micro web framework được viết bởi Python. Web framework là một khuôn khổ phần mềm được thiết kế để hỗ trợ phát triển các ứng dụng web bao gồm các dịch vụ web, tài nguyên web và các API web, cung cấp các chức năng cần thiết để tạo ra một ứng dụng website. Flask được phân loại là micro framework vì nó không yêu cầu các công cụ hoặc thư viện cụ thể, cung cấp một lõi chức năng đơn giản nhưng có thể mở rộng. Đề tài sử dụng Flask như một framework chính để phát triển web service khai thác mô hình đề xuất đã huấn luyện. Sơ đồ hoạt động của Flask được thể hiện tại Hình 9.



Hình 9. Sơ đồ hoạt động của Flask

6. Tkinter

Tkinter [25] là mô-đun Python được tích hợp sẵn dùng để tạo ra các ứng dụng giao diện đồ hoạ người dùng (GUI). Tkinter được phát triển bằng TCL (Tool Command Language) và đa nền tảng, với sự hỗ trợ cho Linux, MacOS và MS Windows. Tkinter là một trong những mô-đun được sử dụng phổ biến để tạo ra các ứng dụng GUI bằng Python vì tính đơn giản và dễ làm việc. Ngoài ra, Tkinter còn cung cấp giao diện hướng đối tượng cho bộ công cụ Tk GUI. Hình 10 dưới đây thể hiện luồng để hiển thị một GUI cơ bản.



Hình 10. Sơ đồ hiển thị GUI cơ bản

CHƯƠNG III. THIẾT KẾ VÀ CÀI ĐẶT

I. CẤU HÌNH CÔNG CỤ THỰC NGHIỆM

- Sử dụng ngôn ngữ Python phiên bản 3.6.8
- Máy tính hệ điều hành MacOS Catalina 10.15 với cấu hình gồm bộ xử lý Intel(R) Core(TM) i7-7920HQ CPU @ 3.10 GHz với 8 cores. Bộ nhớ là LPDDR3 với dung lượng RAM là 16GB. Card màn hình là Intel HD Graphics 630, Radeon Pro 555 (1).
- Máy tính Raspberry Pi 3 model B+ với hệ điều hành Raspbian 5.4.51-v7+, cấu hình CPU 64 bit quad-core bộ vi xử lý ARM Cortex A53, tốc độ 1.2GHz. Bộ nhớ RAM 1GB. Tích hợp Wireless chuẩn 802.11n và Bluetooth 4.1 (2).
- 1 Camera Logitech

II. CÁC BƯỚC THỰC HIỆN

1. Thu thập dữ liệu

Dữ liệu được thu thập được thu thập trực tiếp. Người dùng sẽ phải đứng trước camera vào xoay khuôn mặt 360 độ để có thể bắt được mọi khía cạnh. Tuy nhiên, để làm tăng độ chính xác, cần phải lấy mẫu với độ nhiễu nhất định (chẳng hạn như đeo kính, đội mũ, ánh sáng, …). Camera được định khung cố định với kích thước 400×400 pixels. Ở đây chúng tôi thực hiện lấy mẫu ở một hộ gia đình với 4 thành viên. Mỗi thành viên sẽ thực hiện việc lấy mẫu trong 1 phút với chất lượng camera là 30 khung trên giây (FPS). Tuy nhiên việc lấy khung liên tục sẽ không có ý nghĩa vì 2 khung liên tiếp sẽ không có sự khác biệt to lớn. Cho nên ta thực hiện lấy mỗi 5 khung sẽ được 1 mẫu huấn luyện. Đây chỉ mới là dữ liệu hình ảnh thô 400×400 pixels.

Tiếp theo chúng tôi nhận dạng khuôn mặt qua từng tấm ảnh và ảnh khuôn mặt nhận dạng sẽ được giảm về tỷ lệ 128×128 pixels. Mã hoá từng khuôn mặt qua 68 điểm đặc trưng (landmarks) thành véc-tơ 128 đặc trưng cho mỗi khuôn mặt. Sử dụng đặc trưng HOG để phát hiện khuôn mặt kết hợp với thư viện *dlib* mã hoá thành véc-tơ 128 đặc trưng.

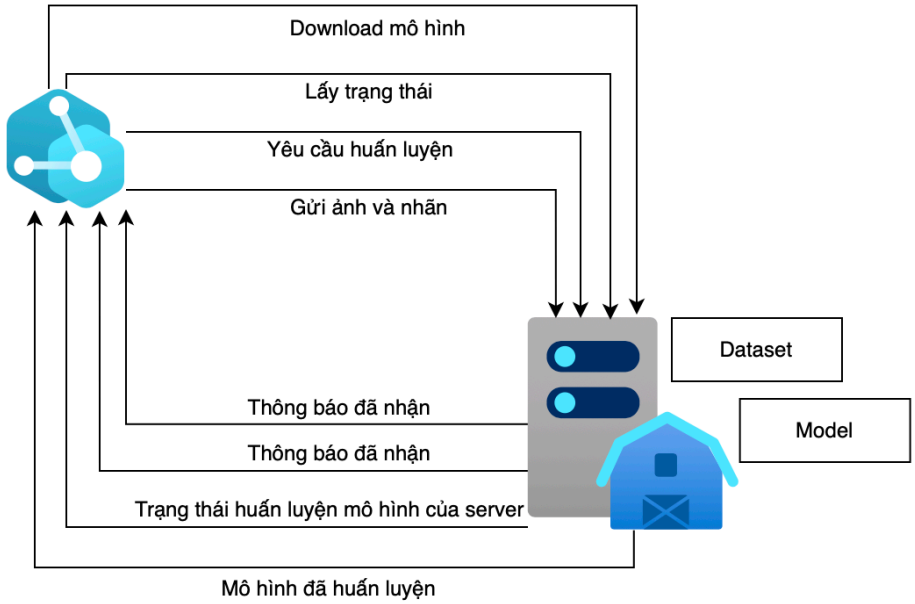
2. Nghiên cứu mô hình hiệu quả

Thử nghiệm qua các mô hình phân lớp có sẵn của thư viện scikit-learn, nhận xét thấy mô hình SVM với hàm nhân RBF cho kết quả tốt nhất với độ chính xác Accuracy 100% và log loss chỉ 0,009. Việc huấn luyện mô hình SVM sử dụng hàm nhân RBF với các siêu tham số như hằng số dung hoà (trade-off) C = 100.000, grama = 0,01. Dữ liệu được chia theo nghi thức Hold-out.

3. Huấn luyện mô hình trên máy chủ

Để có thể đưa nghiên cứu đến gần với người sử dụng, ứng dụng sử dụng mô hình cần có khả năng cập nhật khuôn mặt của người sở hữu mới, dẫn đến sự cần thiết của việc "huấn luyện lại" mô hình. Tuy nhiên, Raspberry Pi Model 3B+ với bộ nhớ RAM tổng cộng khoảng 1GB. Khi khởi động, bộ nhớ chiếm dụng bởi hệ điều hành nên tầng ứng dụng chỉ sử dụng được khoảng 120MB, gây ra vấn đề về cạn kiệt bộ nhớ, mất thời gian khi huấn luyện lại mô hình.

Để giải quyết vấn đề trên, chúng tôi sử dụng một máy chủ (server) chịu trách nhiệm huấn luyện mô hình. Server sẽ là một RESTful API server, sử dụng giao thức HTTP dùng để nhận dữ liệu và gửi mô hình đã huấn luyện về Raspberry Pi (Hình 11). API server được viết dựa trên Flask framework phiên bản 1.1.2. Trong nghiên cứu này, giải pháp được triển khai đơn giản, chúng tôi chưa cung cấp cách cài đặt middleware làm nhiệm vụ phân quyền hay cài đặt các kỹ thuật nén tệp khi truyền gửi mà chỉ tập trung vào chứng minh khả năng ứng dụng của giải pháp đã đề xuất.



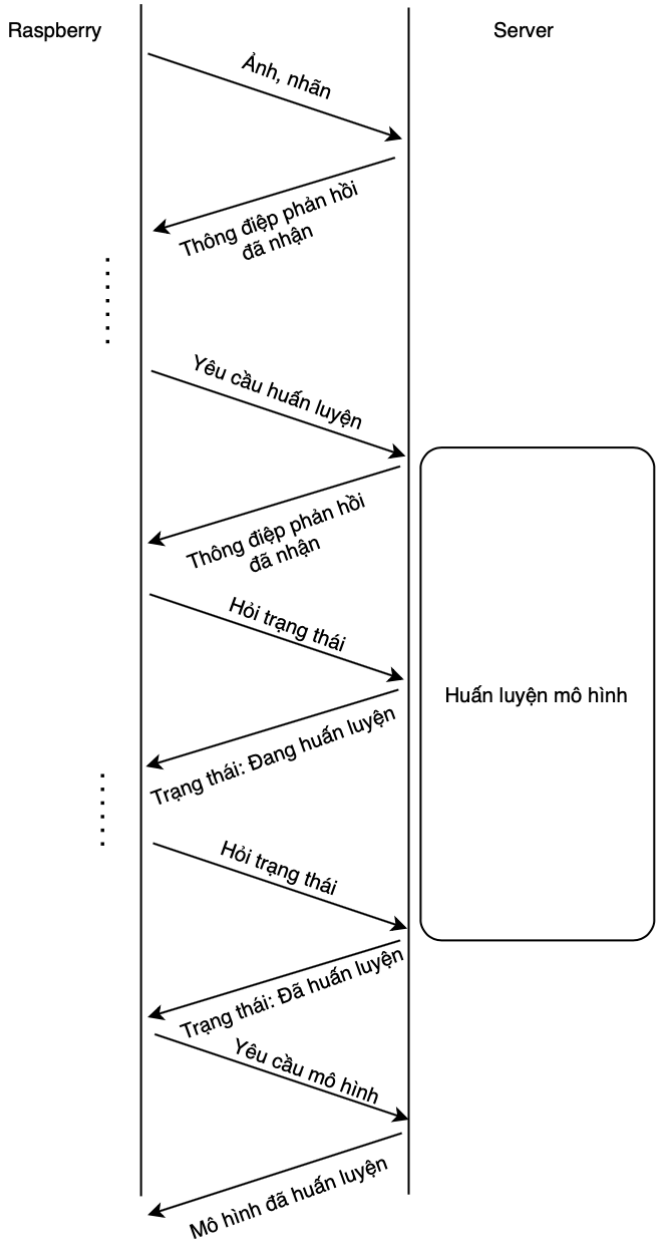
Hình 11. Mô hình Client - Server giải quyết vấn đề "huấn luyện lại"

Bên dưới là mô tả ngắn gọn quy trình thực hiện "huấn luyện lại":

1. Client gửi ảnh và nhãn đến Server, Server sẽ phản hồi thông điệp mỗi khi nhận được.
2. Sau khi gửi đi tất cả các ảnh, Client sẽ gửi thông điệp yêu cầu Server huấn luyện mô hình dựa trên các dự liệu trên Server. Dữ liệu dùng để huấn luyện là tất cả các dữ liệu từ trước đến hiện tại mà Raspberry đã gửi lên.

3. Server huấn luyện sau khi nhận được thông điệp yêu cầu ở Bước 2. Trong lúc này, Raspberry sẽ kiểm tra quá trình huấn luyện trên Server thông qua thông điệp hỏi (polling), nếu Server trả về quá trình huấn luyện hoàn tất, Raspberry sẽ chuyển sang hành vi ở bước 4, ngược lại quá trình hỏi sẽ tiếp tục.
4. Raspberry tải về mô hình mới và thay thế mô hình hiện tại.

Các bước được thể hiện trực quan như hình bên dưới:



Hình 12. Mô tả quá trình gửi-nhận từ Client đến Server và ngược lại

Bảng 2. Mô tả API Server cung cấp

Phương thức	**HTTP request**	**Mô tả**	**Kết quả trả về**
upload	POST /upload	Tải ảnh và nhãn của ảnh lên server Các trường yêu cầu: Form-data: *image*: ảnh encoding base64, *label*: nhãn của ảnh	Kết quả nếu thành công: Mã HTTP Response: 200, nội dung: { 'result': 1 } Kết quả nếu thất bại: Mã HTTP Response: 400, nội dung { 'result': 0, 'err': <Nội dung lỗi> }
train	POST /train	Yêu cầu server huấn luyện bộ dữ liệu đã tải lên	Kết quả nếu thành công: Mã HTTP Response: 200, nội dung: { 'result': 1 } Kết quả nếu thất bại: Mã HTTP Response: 400, nội dung { 'result': 0, err': <Nội dung lỗi> }
status	GET /status	Truy vấn trạng thái huấn luyện trên server.	Kết quả trả về: Nếu trạng thái huấn luyện là training (đang huấn luyện): { 'result': 1, 'status': 'training' } Nếu trạng thái huấn luyện là trained (đã huấn luyện): { 'result': 1, 'status': 'trained', 'model_name': <tên_model>} Đối với có lỗi xảy ra trong quá trình huấn luyện: { 'result': 1, 'status': 'failed' }
model	GET /model/ <model _name>	Yêu cầu tải về model đã được huấn luyện Yêu cầu trường model_name trên request	Kết quả nếu thành công: Server trả về model với tên chỉ ra trong <model_name> Kết quả nếu thất bại: Mã HTTP Response: 400, nội dung { 'result': 0, err': <Nội dung lỗi> }

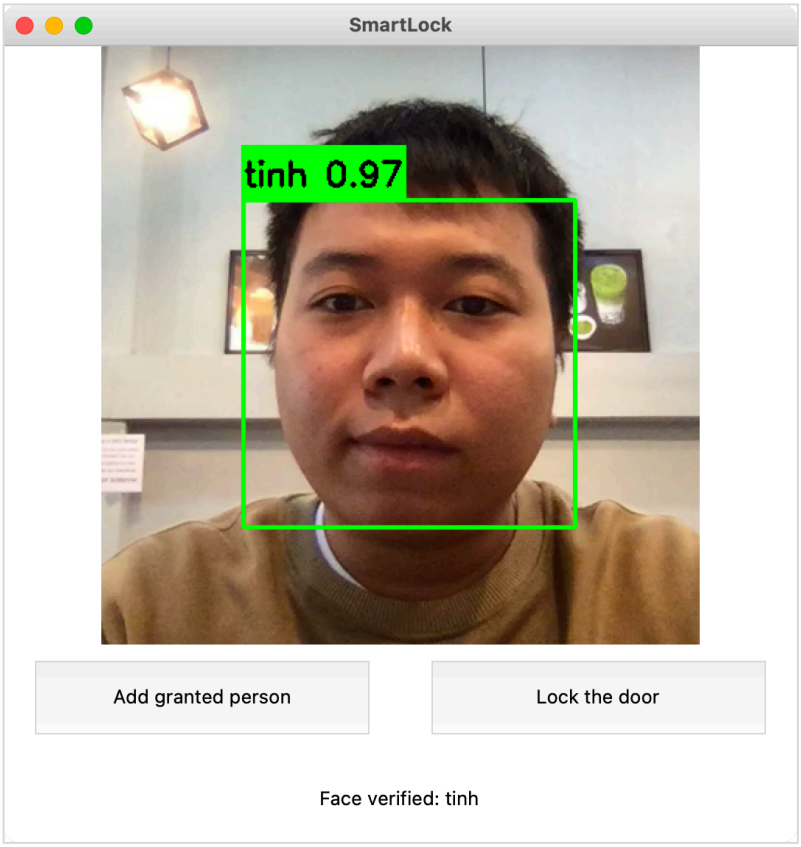
4. Xây dựng Ứng dụng giao diện đồ hoạ trên Raspberry Pi 3+

4.1. Xây dựng ứng dụng

Tương tự như việc thực nghiệm trên máy tính các nhân (1), trên máy tính Raspberry (2) trước hết cài đặt các thư viện cần thiết như: OpenCV, face_recognition, numpy,… Tuy nhiên, khi cài đặt OpenCV thông qua lệnh *pip install opencv-python* không hoạt động bình thường trên Raspberry Pi. Có thể gặp phải *"undefined symbol: __atomic_fetch_add8"* cho lỗi *libatomic* khi *import cv2*. Vì vậy, chúng tôi phải cài phiên bản cụ thể của OpenCV là *opencv-contrib-python==4.1.0.**25***. Sau khi đã hoàn tất việc cài đặt các gói thư viện cần thiết, chúng tôi kéo project từ kho cá nhân trên github: *https://github.com/vinhphuctadang/SmartLock.*

Để điều khiển máy tính từ xa, ứng dụng AnyDesk [26] được cài đặt thêm trên Raspberry. Ứng dụng hỗ trợ truy cập vào màn hình Raspberry mọi lúc mọi nơi, tương tự IP Webcam.

Để đưa mô hình đã huấn luyện đến với người sử dụng, việc xây dựng giao diện đồ hoạ là cần thiết. Giao diện được xây dựng sử dụng tkinter (phiên bản kết hợp với thư viện opencv-python (phiên bản như mô tả trên), joblib (phiên bản 0.17) và thư viện chuẩn request trên Python 3. Giao diện được thiết kế như sau:



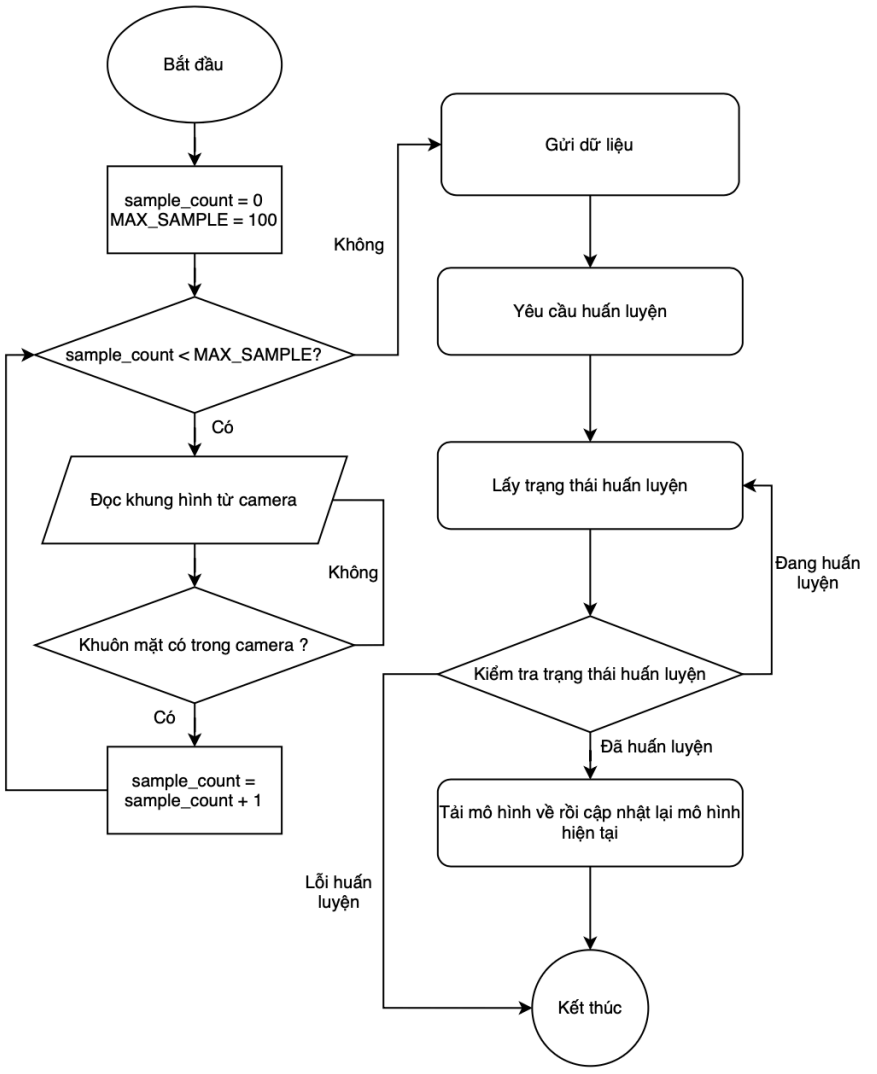
Hình 13. Giao diện ứng dụng

Ứng dụng sử dụng joblib để nạp mô hình đã huấn luyện. Tương tác với các route trên server (như đã đề cập ở *mục 3*) nhờ hỗ trợ của thư viện *request*. Các bước thiết kế giao diện khá đơn giản, phần chính nằm ở khung nhìn thể hiện camera. Ảnh đọc từ camera sử dụng hàm *read* của đối tượng *cv2.VideoCapture*.

Sau khi xây dựng giao diện hoàn tất, chúng tôi cài đặt cho ứng dụng của mình khả năng điều khiển khoá điện và ràng buộc về hành vi cho người dùng như sau:

1. Nếu ứng dụng mới được sử dụng lần đầu, người sử dụng có thể thêm người tin cậy (*Add granted person*), ngược lại, người sử dụng phải xác nhận họ đã là người tin cậy chưa trước khi *thêm một người tin cậy mới* hoặc *mở khoá.*
2. Sau khi mở khoá, người mở khoá phải tự đóng lại khoá của mình, hành vi tiếp theo của khoá điện tương tự như bước 1.

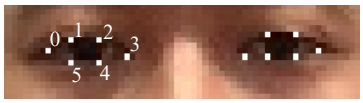
Lưu đồ hành vi khi nhấn vào nút “Add granted person” được thể hiện như Hình 14:



Hình 14. Lưu đồ giải thuật thêm người tin cậy vào ứng dụng

4.2. Giải quyết vấn đề mạo danh

Trong quá trình xây dựng ứng, chúng tôi phát hiện ra vấn đề mạo danh khi xác thực khuôn mặt. Kẻ mạo danh có thể vượt qua hệ thống nhận diện bằng ảnh khuôn mặt của chủ nhà (ảnh giấy hoặc ảnh trên bất kỳ thiết bị di động nào). Vì vậy, để giải quyết vấn đề này, chúng tôi thực hiện giải pháp yêu cầu người dùng thực hiện việc chớp mắt ít nhất một lần nhằm để xác thực con người. Sau đó, hệ thống mới bắt đầu thực hiện việc xác thực khuôn mặt.



Hình 15. 5 điểm landmarks của mỗi mắt

Như đề cập trên Hình 3, với 68 landmarks chúng tôi lấy ra 10 điểm của mắt với 5 điểm cho mỗi mắt (mắt trái và mắt phải) và được đánh số như Hình 15. Tiếp theo, thực hiện việc tính khoảng cách Euclid như sau:

$$A = dist.euclidean(e_1, e_5)$$

$$B = dist.euclidean(e_2, e_4)$$

$$C = dist.euclidean(e_0, e_3)$$

Sau khi có được các khoảng cách cần thiết, chúng tôi tính toán tỷ lệ co giãn của mắt bởi công thức:

$$R = \frac{A + B}{2 * C}$$

Nếu $R \leq 0.2$ thì kết luận rằng mắt đang nhắm. Thực hiện việc tính toán trên cho cả 2 mắt.

5. Truyền phát tín hiệu mở khoá điện

Khi triển khai trên Raspberry Pi, chúng tôi thử nghiệm trên một đèn LED gắn trên bảng mạch có điều khiển. Sử dụng thư viện GPIOs [27] .

CHƯƠNG IV. KIỂM THỬ VÀ ĐÁNH GIÁ

I. NGHI THỨC

Trong bài viết này, chúng tôi thử nghiệm qua các mô hình phân lớp có sẵn của thư viện scikit-learn. Cụ thể là mô hình k láng giềng gần nhất (kNN), Bayes thơ ngây, Cây quyết định, Bagging và AdamBoosting của cây quyết định, Rừng ngẫu nhiên và Máy học vector hỗ trợ (SVM). Đối với mô hình SVM, chúng tôi sẽ thử nghiệm qua các hàm nhân như hàm tuyến tính, đa thức, RBF và Sigmoid. Kết quả được đánh giá bằng nghi thức kiểm tra chéo k-fold với k=10, sử dụng độ đo Accuracy và hàm mất mát log hay hàm cross-entropy.

II. ĐÁNH GIÁ KẾT QUẢ

Trong bài báo cáo, sử dụng 2 độ đo để đánh giá hiệu quả huấn luyện của mô hình trên là Accuracy (ACC) và Log Loss.

Accuracy là tỷ lệ giữa số lượng dự đoán chính xác với tổng số lượng dự đoán. Giả sử ta có ma trận sau:

		Thực tế	
		Bệnh	Không bệnh
Dự đoán	Bệnh	TP	FP
	Không bệnh	FN	TN

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

- TP (True Positive): Dự đoán đúng người có bệnh
- FN (False Negative): Dự đoán sai người có bệnh thành không bệnh
- TN (True Negative): Dự đoán đúng người không có bệnh
- FP (False Positve): Dự đoán sai người không bệnh thành có bệnh
- ACC để đánh giá tốt khi dữ liệu các lớp là cân bằng.

Log Loss hay còn gọi là Logistic loss hoặc Cross-entropy loss. Đây là hàm mất mát được sử dụng trong hồi quy logistic (đa thức) và các phần mở rộng của nó, chẳng hạn như mạng nơ-ron, được định nghĩa là khả năng log âm của một mô hình logistic trả về xác suất **y_pred** cho dữ liệu huấn luyện **y_true** của nó. Log loss chỉ được xác

định cho hai hoặc nhiều nhãn. Đối với một mẫu đơn có nhãn đúng $y \in \{0, 1\}$ và ước tính xác suất $p = \Pr(y = 1)$, log loss được tính như sau [28] :

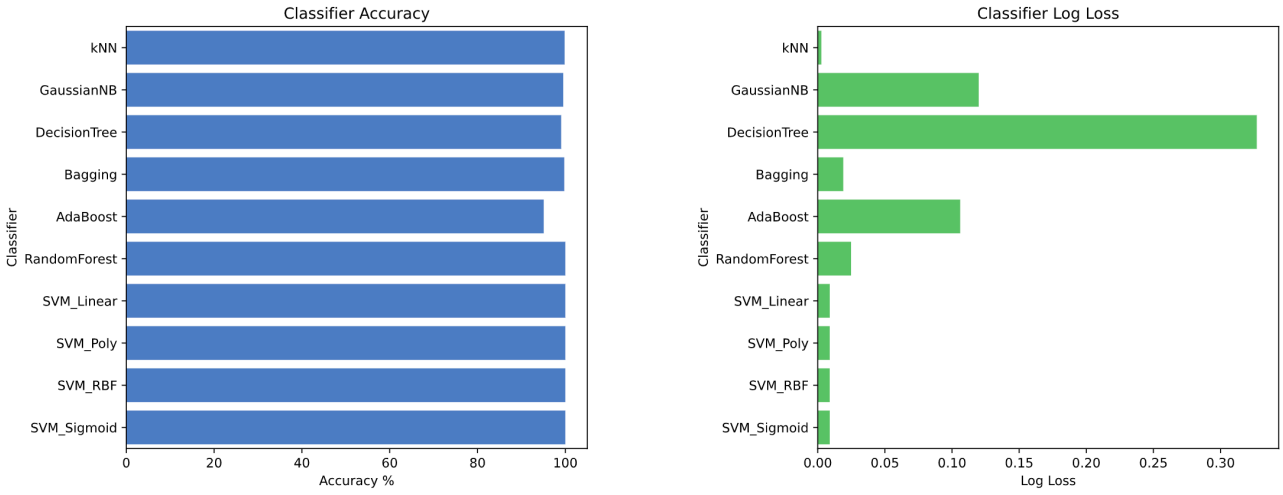
$$L_{log}(y, p) = -y(\log(p) + (1 - y)\log(1 - p))$$

Hàm log được sử dụng là hàm logarit tự nhiên (cơ số e).

Kết quả được trình bày ở Bảng 3 và Hình 16:

Bảng 3. Kết quả thực nghiệm theo nghi thức k-fold (k=10)

	Độ chính xác (%)	**Giá trị mất mát (Log loss)**
K láng giềng gần nhất	99.84251968	0.002686761
Bayes thơ ngây	99.52630921	0.119893884
Cây quyết định	99.05324334	0.326998164
Bagging của cây quyết định	99.76377952	0.019007621
AdamBoost của cây quyết định	95.07936507	0.106089599
Rừng ngẫu nhiên	100.0000000	0.024827676
SVM với hàm tuyến tính	100.0000000	0.008948390
SVM với hàm đa thức	100.0000000	0.008926670
SVM với hàm RBF	**100.0000000**	**0.008914790**
SVM với hàm Sigmoid	100.0000000	0.008945375



Hình 16. So sánh độ chính xác và độ lỗi của các mô hình

CHƯƠNG V. KẾT LUẬN

I. KẾT QUẢ

1. Kết quả đạt được

- Kết quả phân lớp đạt hiệu quả cao với mô hình SVM
- Xử lý các trường hợp nhiễu và giả danh
- Truyền phát tín hiệu mở khoá điện thông qua máy tính Raspberry
- Triển khai thực tế trong hộ gia đình

2. Khó khăn

- Cấu hình máy tính Raspberry tương đối thấp nên trong quá trình triển khai ứng dụng lên máy tính Raspberry đòi hỏi phải tối ưu rất nhiều

II. HƯỚNG PHÁT TRIỂN

- Triển khai mô hình cửa mở tự động vào các tổ chức, công ty,… Đặc biệt là ngay trong trường đại học, hệ thống sẽ kiểm tra sinh viên có thuộc lớp học phần hay không, tích hợp điểm danh tự động và hơn hết là tránh trường hợp điểm danh hộ.
- Hiện tại hệ thống triển khai trong khu vực nội bộ (local) vì thế cần cài đặt máy chủ có thể điều khiển từ xa để tiện lợi hơn cho việc kiểm soát ở nơi đặt camera kèm theo truyền phát tín hiệu loa để cảnh báo nếu không được phép truy cập hoặc xác thực không hợp lệ.

TÀI LIỆU THAM KHẢO

[1] Turk, M., Pentland, A. Eigenfaces for recognition, Journal of Cognitive Neuroscience, vol.3, no. 1, pp. 71-86, 1991.

[2] Belhumeur, P.N., Hespanha, J.P., Kriegman, D.J. Eigenfaces against Fisherfaces: recognition using class specific linear projection, IEEE Trans. Patt. Anal. Mach. Intell., 1997, 19, (7), pp. 711–720. https://doi.org/10.1109/34.598228.

[3] Bartlett, M., Movellan, J., Sejnowski, T. Face recognition by independent component analysis, IEEE Trans. Neural Netw., 2002, 13, (6), pp. 1450–1464. https://doi.org/10.1109/TNN.2002.804287.

[4] Penev, A., Atick, J. Local feature analysis: a general statistical theory for object representation, Netw. Comput. Neural Syst., 1996, 7, (3), pp. 477–500 6.

[5] Wiskott, L., Fellous, J., Kruger, N., Malsburg, C. Face recognition by elastic bunch graph matching, IEEE Trans. Patt. Anal. Mach. Intell., 1997, 19, (7), pp. 775–779. https://doi.org/10.1109/ICIP.1997.647401.

[6] Chaochao Lu, Xiaoou Tang. Surpassing Human-Level Face Verification Performance on LFW with GaussianFace. AAAI Publications, Twenty-Ninth AAAI Conference on Artificial Intelligence.

[7] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, Lior Wolf. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. Published in: 2014 IEEE Conference on Computer Vision and Pattern Recognition. https://doi.org/10.1109/CVPR.2014.220.

[8] Schroff, Florian; Kalenichenko, Dmitry; Philbin, James (2015). FaceNet: A unified embedding for face recognition and clustering. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 815–823. https://doi.org/10.1109/CVPR.2015.7298682.

[9] Baltrusaitis, Tadas; Robinson, Peter; Morency, Louis-Philippe (2016). OpenFace: An open source facial behavior analysis toolkit. 2016 IEEE Winter Conference on Applications of Computer Vision (WACV), 1–10. https://doi.org/10.1109/WACV.2016.7477553.

[10] P. J. P. et al (2011). An introduction to the good, the bad, & the ugly face recognition challenge problem. IEEE. https://doi.org/10.1109/FG.2011.5771424.

[11] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, Lior Wolf. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. Published

in: 2014 IEEE Conference on Computer Vision and Pattern Recognition. https://doi.org/10.1109/CVPR.2014.220.

[12] Davis E. King. (2009). Dlib-ml: A Machine Learning Toolkit. Journal of Machine Learning Research 10, pp. 1755-1758

[13] Van Rossum, G. (2020). The Python Library Reference, release 3.8.2. Python Software Foundation.

[14] Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.

[15] Adam Geitgey, https://github.com/ageitgey/face_recognition

[16] Buitinck et al. (2013). API design for machine learning software: experiences from the scikit-learn project.

[17] O. Déniz, G. Bueno, J. Salido, F. De la Torre. Face recognition using Histograms of Oriented Gradients. https://doi.org/10.1016/j.patrec.2011.01.004.

[18] D´eniz, O., Bueno, G., Salido, J., De la Torre, F.: Face recognition using Histograms of Oriented Gradients (2011). https://doi.org/10.1016/j.patrec.2011.01.004

[19] Wikipedia: https://en.wikipedia.org/wiki/Support_vector_machine.

[20] machinelearningcoban.com, Available in: https://machinelearningcoban.com/2017/04/09/smv/

[21] Support Vector Machines for Binary Classification. https://www.mathworks.com/help/stats/support-vector-machines-for-binaryclassification.html.

[22] V. Vapnik. (1995): The Nature of Statistical Learning Theory, Springer-Verlag

[23] U. Krebel. (1999): Pairwise classification and support vector machines. Advances in Kernel Methods: Support Vector Learning, pp. 255-268

[24] Grinberg, M. (2018). Flask web development: developing web applications with python. "O'reilly Media, Inc."

[25] Lundh, F. (1999). An introduction to tkinter. URL: www.Pythonware. Com/Library/Tkinter/Introduction/Index. Htm.

[26] AnyDesk. 2016a. Homepage. Retrieved from http://anydesk.com/remote-desktop.

[27] Warren Gay. 2014. Raspberry Pi Hardware Reference (1st. ed.). Apress, USA.

[28] C.M. Bishop (2006). Pattern Recognition and Machine Learning. Springer, p. 209.

[29] Duda, Richard O., Peter E. Hart, and David G. Stork. (2012). Pattern classification. John Wiley & Sons

[30] Pham, Nguyen-Khang & Nguyen, Minh & Do, Thanh-Nghi. (2017). Điểm danh bằng mặt người với đặc trưng Gist và máy học véc-tơ hỗ trợ. 10.15625/vap.2017.00019.