**TRƯỜNG ĐẠI HỌC HÀNG HẢI VIỆT NAM**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**BÁO CÁO BÀI TẬP LỚN**

**HỌC PHẦN: THỊ GIÁC MÁY TÍNH**

***Đề tài:* *Nghiên cứu và cài đặt chương trình sử dụng HOG/LBP feature để phát hiện mặt người trong video***

***GVHD: GV.Hồ Thị Hương Thơm***

***Sinh viên thực hiện: Đinh Như Khang – 89768***

***Phạm Quang Long - 86847***

***Hải Phòng, ngày tháng 12 năm 2023***

**LỜI MỞ ĐẦU**

Trong thời đại công nghệ ngày nay, việc nhận dạng ảnh khuôn mặt không chỉ là một chủ đề nghiên cứu quan trọng mà còn là một ứng dụng thực tiễn có ảnh hưởng sâu rộng đối với xã hội và công nghệ. Nhận dạng khuôn mặt không chỉ giúp chúng ta bảo vệ thông tin cá nhân mà còn mở ra nhiều khả năng trong các lĩnh vực như an ninh, y tế, giáo dục và công nghiệp.

Trong ngữ cảnh đó, chúng em chú trọng vào nghiên cứu và phát triển một hệ thống nhận dạng ảnh khuôn mặt chính xác và hiệu quả trong cả ảnh và video. Dự án này không chỉ đặt ra thách thức về việc xử lý dữ liệu ảnh có độ biến động lớn về ánh sáng, góc chụp và chất lượng, mà còn đặt ra câu hỏi về việc kết hợp giữa công nghệ mới như học sâu (deep learning) và các phương pháp truyền thống để đạt được hiệu suất tốt nhất.

Mục tiêu chính của dự án này là nghiên cứu, phát triển và đánh giá các phương pháp và thuật toán để nhận dạng ảnh khuôn mặt trong các điều kiện đa dạng. Chúng em hy vọng rằng kết quả của dự án sẽ không chỉ là một bước tiến quan trọng trong lĩnh vực nhận dạng khuôn mặt mà còn là cơ sở để xây dựng các ứng dụng thực tiễn, góp phần vào sự phát triển của công nghệ thông tin và xã hội hiện đại.

Chúng em xin chân thành cảm ơn sự hỗ trợ và động viên từ giáo viên hướng dẫn cũng như sự đồng lòng tin và nỗ lực không ngừng thành viên nhóm trong. Hy vọng bài tập lớn sẽ đem lại giá trị và kiến thức hữu ích cho mai sau này

Content

[Chương 1 GIỚI THIỆU ĐỀ TÀI 5](#_Toc150179348)

[1.1 Lý do chọn đề tài 5](#_Toc150179349)

[1.2 Bài toán nhận diện khuôn mặt 6](#_Toc150179350)

[1.3 Mục đích – Ngữ cảnh áp dụng: 6](#_Toc150179351)

[1.4 Phạm vi bài toán 7](#_Toc150179352)

[1.4.1 Local Binary Patterns 7](#_Toc150179353)

[1.4.2 Histogram of Oriented Gradient 8](#_Toc150179354)

[1.4.3 Support Vector Machine 9](#_Toc150179355)

[Chương 2 KỸ THUẬT ÁP DỤNG BÀI TOÁN NHẬN DIỆN KHUÔN MẶT 10](#_Toc150179356)

[2.1 Mô hình tổng quát 10](#_Toc150179357)

[2.2 Chi tiết thuật toán 11](#_Toc150179358)

[2.2.1 Phát hiện khuôn mặt 11](#_Toc150179359)

[2.2.2 Tiền xử lý 11](#_Toc150179360)

[2.2.3 Trích xuất đặc trưng 11](#_Toc150179361)

[2.2.4 Phân lớp 14](#_Toc150179362)

[2.2.5 Tập dữ liệu 14](#_Toc150179363)

[Chương 3 CÀI ĐẶT VÀ ĐÁNH GIÁ KẾT QUẢ HỆ THỐNG 15](#_Toc150179364)

[3.1 Ngôn ngữ lập trình , thư viện sử dụng , nền tảng lập trình 15](#_Toc150179365)

[3.1.1 Ngôn ngữ lập trình Python 15](#_Toc150179366)

[3.1.2 Thư viện scikit-learn 16](#_Toc150179367)

[3.1.3 Thư Viện OpenCV 16](#_Toc150179368)

[3.1.4 Nền tảng lập trình – Anaconda: 17](#_Toc150179369)

[3.2 Mã nguồn chương trình 17](#_Toc150179370)

[3.3 Nhận xét kết quả 27](#_Toc150179371)

[3.3.1 Dữ liệu: 27](#_Toc150179372)

[3.3.2 Hình ảnh nhận diện 27](#_Toc150179373)

[3.4 Nhận xét về hệ thống 27](#_Toc150179374)

[Chương 4 ĐÁNH GIÁ 28](#_Toc150179375)

[4.1 Quy trình 28](#_Toc150179376)

[4.2 Ưu điểm và nhược điểm 28](#_Toc150179377)

[4.3 Hướng phát triển 29](#_Toc150179378)

[*4.4* Tổng kết 29](#_Toc150179379)

# GIỚI THIỆU ĐỀ TÀI

## Lý do chọn đề tài

Trong những năm gần đây, Trí tuệ nhân tạo (Artificial Intellegence) là một trong những lĩnh vực được rất nhiều các nhà khoa học quan tâm. Mục tiêu của trí tuệ nhân tạo là tạo ra các máy tính có khả năng học hỏi từ dữ liệu, nắm bắt các mô hình từ kinh nghiệm, và thực hiện các nhiệm vụ thông minh mà trước đây chỉ có con người mới có khả năng thực hiện được.

Trong đó có 2 lĩnh vực chính của trí tuệ nhân tạo bao gồm:

**Học máy (Machine Learning):** Là một phần quan trọng của trí tuệ nhân tạo, học máy giúp máy tính học hỏi từ dữ liệu và cải thiện hiệu suất theo thời gian mà không cần được lập trình cụ thể.

**Thị giác máy tính (Computer Vision):** Là lĩnh vực nghiên cứu về cách máy tính nhận diện, hiểu và giải quyết các vấn đề liên quan đến hình ảnh và video. Mục tiêu của thị giác máy tính là làm cho máy tính có khả năng "nhìn" và "hiểu" thế giới xung quanh chúng giống như con người.

Các ứng dụng của **Thị giác máy tính** rất đa dạng và ngày càng trở nên phổ biến trong cuộc sống hàng ngày:

**Nhận diện vật thể (Object Recognition)**: Hệ thống thị giác máy tính có thể nhận diện và phân loại các đối tượng trong hình ảnh hoặc video, chẳng hạn như nhận diện người, ô tô, động vật, đồ vật, và nhiều đối tượng khác.

**Nhận diện khuôn mặt (Face Recognition)**: Công nghệ này cho phép máy tính nhận diện và xác định danh tính của người dùng dựa trên hình ảnh khuôn mặt, được sử dụng rộng rãi trong an ninh, quản lý giấy tờ và các ứng dụng xác thực.

**Nhận diện chuyển động (Motion Detection)**: Hệ thống thị giác máy tính có thể nhận diện và theo dõi chuyển động trong video, thích hợp cho các ứng dụng an ninh và giám sát.

**Xử lý hình ảnh y tế (Medical Image Analysis)**: Thị giác máy tính được sử dụng trong y học để phân loại và phân tích hình ảnh y tế, giúp bác sĩ chẩn đoán các bệnh lý và tìm kiếm điều trị.

**Xử lý hình ảnh trong ô tô tự lái (Autonomous Vehicles)**: Hệ thống thị giác máy tính giúp các xe tự lái nhận biết và phản ứng đối với các vật cản, biển báo giao thông và điều kiện đường, đóng vai trò quan trọng trong công nghệ xe tự lái.

**Thị giác máy tính trong công nghệ sản xuất (Industrial Vision):** Được sử dụng để kiểm tra chất lượng sản phẩm, đánh giá sản xuất, và tối ưu hóa quy trình sản xuất trong các nhà máy và xưởng sản xuất.

Các công nghệ và thuật toán trong lĩnh vực thị giác máy tính ngày càng phát triển nhanh chóng, giúp nâng cao hiệu suất và chính xác của các ứng dụng trong nhiều lĩnh vực khác nhau.

## Bài toán nhận diện khuôn mặt

Bài toán nhận diện khuôn mặt (Face Recognition) là một trong những ứng dụng quan trọng của thị giác máy tính. Nó liên quan đến việc sử dụng công nghệ để nhận diện và xác định danh tính của một người dựa trên hình ảnh khuôn mặt của họ. Bài toán này đã có ứng dụng rộng rãi trong nhiều lĩnh vực, bao gồm an ninh, quản lý giấy tờ, giám sát, và cả trong các ứng dụng di động và web

Mặc dù, nhận dạng khuôn mặt không còn là một đề tài quá mới mẻ nhưng vẫn là một thách thức mới đòi hỏi được khám phá. Ngoài ra để giải quyết bài toán ta phải giải quyết các vấn đề khác của bài toán như: ánh sáng, độ mờ, đỗ nhiễu, độ phân giải của ảnh, góc ảnh…

Ngày nay có rất nhiều thuật toán và phương pháp nhận diện ảnh được phát triển và sử dụng rộng rãi trong nhiều ứng dụng khác nhau. Ví dụ như : **Deep Learning và Convolutional Neural Networks (CNNs), Transfer Learning, YOLO V8**. Tuy nhiên, ở trong bài tập lớn này chúng em sẽ sử dụng 2 phương pháp chính và được coi là hiệu quả không kém so với các thuật toán khác đó là Local Binary Patterns (viết tắt là LBP) và Histogram of Oriented Gradient (viết tắt là HOG). LBP là thuật toán đơn giản và hiệu quả do không bị biến đổi độ sáng về mặt hình ảnh ,còn HOG là thuật toán phân loại ảnh và nhận diện vật thể nhờ tập trung vào các canh và hình dạng của ảnh đưa ra

## Mục đích – Ngữ cảnh áp dụng:

* Mục đích:
* Tìm hiểu mô hình bài toán và các bước thực hiện hệ thống nhận diện trên khuôn mặt người
* Tìm hiểu các thuật toán, hướng phát triển bài toán được đưa ra trong mô hìnhd
* Nghiên cứu, tìm hiểu sử dụng phương pháp trích chọn đặc trưng LBP và thuật toán HOG trong bài toán nhận diện khuôn mặt
* Ngữ cảnh áp dụng:

Đối với bài toán nhận dạng khuôn mặt cần phải có điều kiện đưa ra để máy nhận diện:

* Ảnh trực diện, góc mặt ngưởi hướng vào camera chính xác
* Không bị che khuất bởi bất kỳ các vật thể nào
* Ánh sáng cung cấp đủ
* Khuôn mặt nhận diện không được xa quá hoặc gần quá

## Phạm vi bài toán

Phát hiện khuôn mặt của một hoặc nhiều người qua video hoặc webcam máy tính

### Local Binary Patterns

LBP là một toán tử kernel 3×3, nó tổng quát hóa cấu trúc không gian cục bộ của một ảnh. Ojala và các đồng nghiệp đã giới thiệu phương pháp LBP và chỉ ra khả năng phân tách cao của chúng cho sự phân lớp vân. Bởi vì khả năng phân tách và chi phí tính toán thấp, LBP trở nên rất phổ biến trong nhận dạng mẫu LBP đã được áp dụng cho phát hiện khuôn mặt, nhận dạng khuôn mặt, xác thực khuôn mặt, truy vấn ảnh

Ưu Điểm:

* LBP dễ hiểu và dễ triển khai.
* Yêu cầu ít tài nguyên tính toán so với các phương pháp phức tạp hơn.
* LBP rất hiệu quả trong việc mô tả các đặc trưng cục bộ của hình ảnh, giúp trong việc nhận dạng vật thể.
* Do LBP chỉ tập trung vào các khu vực cục bộ của hình ảnh, nó có khả năng chống đối phó tốt với nhiễu.
* Có thể kết hợp LBP với các phương pháp rút trích đặc trưng khác để tăng cường hiệu suất của hệ thống nhận dạng.
* So với các phương pháp sâu hơn, LBP yêu cầu ít dữ liệu huấn luyện.

Nhược Điểm: Trong các trường hợp hình ảnh phức tạp và có nhiều biến thể, LBP có thể không đủ mạnh mẽ để rút trích được các đặc trưng quan trọng.

### Histogram of Oriented Gradient

HOG (Histogram of Oriented Gradients) là một phương pháp phổ biến được sử dụng để rút trích đặc trưng trong xử lý ảnh và nhận dạng vật thể. Dưới đây là ưu điểm và nhược điểm của phương pháp HOG:

Ưu Điểm:

* HOG có khả năng tốt trong việc mô tả các đặc trưng cục bộ của hình ảnh, đặc biệt là các cạnh và các đặc trưng hình học.
* HOG có thể chống đối phó tốt với các biến đổi ánh sáng, giúp tăng tính ổn định của thuật toán trong các điều kiện ánh sáng khác nhau.
* HOG có khả năng chịu được một số biến đổi đơn giản như quay, dịch chuyển, và co giãn, giúp tăng khả năng nhận diện trong các tình huống thực tế.
* Các đặc trưng HOG có thể được tổng hợp lại thành các vector đặc trưng để sử dụng trong các thuật toán máy học, như máy học phổ biến.
* HOG được sử dụng rộng rãi trong các ứng dụng nhận dạng vật thể, như nhận diện khuôn mặt và nhận diện chữ viết.

Nhược Điểm:

* HOG có thể không hiệu quả nếu vị trí hoặc tư thế của đối tượng thay đổi đáng kể.
* Trong trường hợp các biến đổi đối với đối tượng lớn hoặc các biến đổi không đều, HOG có thể không còn hiệu quả.
* So với các phương pháp đơn giản hơn như LBP, HOG có độ phức tạp cao trong việc rút trích đặc trưng, đặc biệt là trong các hình ảnh có độ phân giải cao.
* HOG đòi hỏi một số lượng dữ liệu huấn luyện lớn để học được các trọng số tốt, đặc biệt là trong các tác vụ nhận diện phức tạp.
* HOG có thể không hiệu quả nếu đối tượng được quan sát từ các góc nhìn khác nhau và không có sự chuẩn hóa góc nhìn.

### Support Vector Machine

SVM là viết tắt của "Support Vector Machine" (Máy Vector Hỗ Trợ) là một thuật toán học máy được sử dụng chủ yếu cho các nhiệm vụ phân loại và hồi quy. Thuật toán này thuộc về loại thuật toán học máy được gọi là "thuật toán học máy dựa trên margin".

Ý tưởng cơ bản của SVM là tìm một siêu phẳng (hyperplane) trong không gian đa chiều sao cho nó tách biệt tốt nhất giữa các điểm dữ liệu thuộc các lớp khác nhau. Trong trường hợp phân loại nhị phân, SVM cố gắng tìm ra siêu phẳng chia không gian thành hai phần sao cho các điểm dữ liệu thuộc hai lớp khác nhau nằm ở hai bên của siêu phẳng đó.

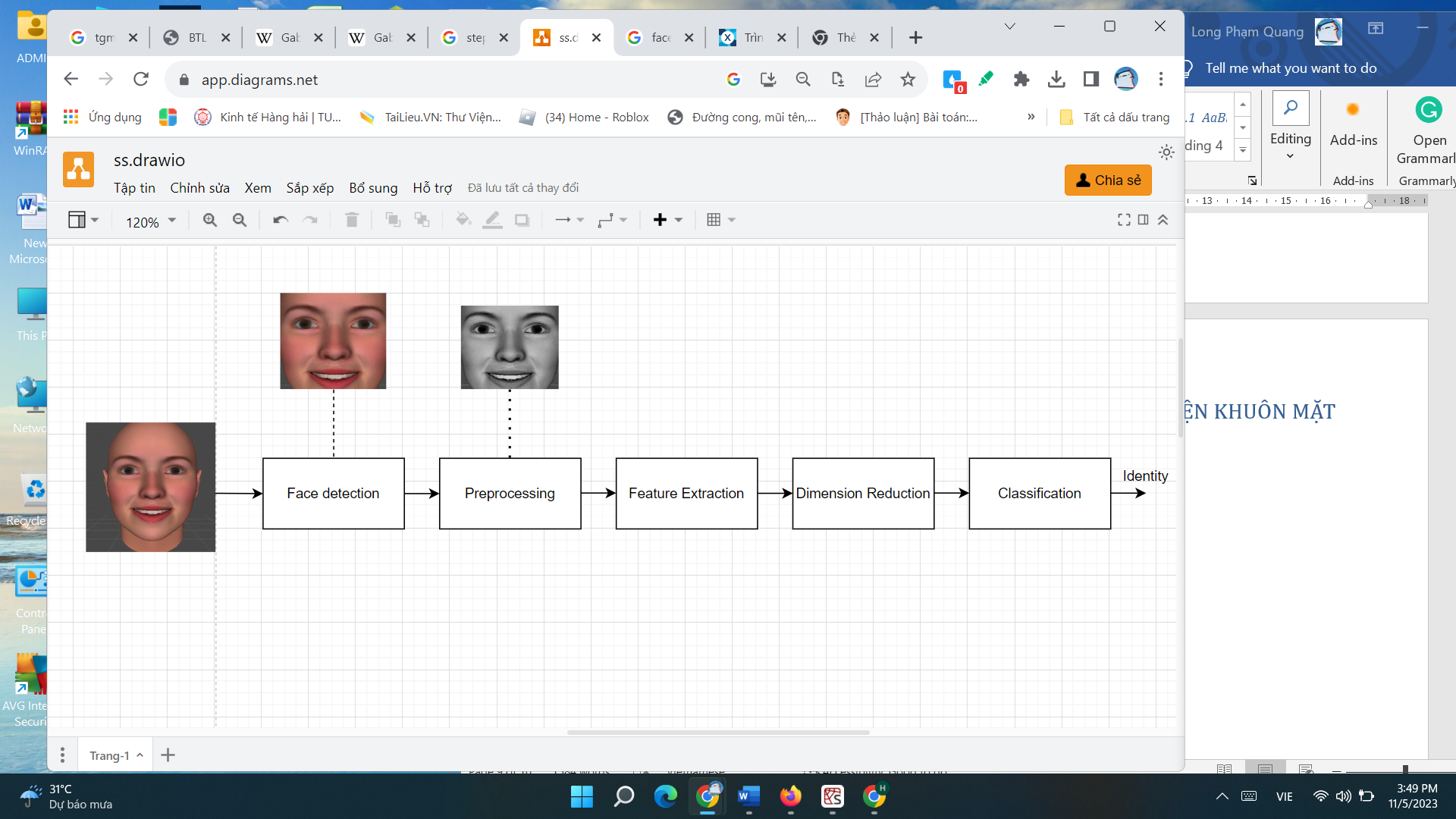
Siêu phẳng được chọn sao cho khoảng cách (margin) giữa siêu phẳng và các điểm dữ liệu gần nhất của hai lớp là lớn nhất. Các điểm dữ liệu gần nhất này được gọi là các vector hỗ trợ (support vectors), và chúng quyết định vị trí và hình dạng của siêu phẳng.

Nếu dữ liệu không thể được phân chia tuyến tính (không thể chia thành hai lớp bằng một siêu phẳng), SVM sử dụng một kỹ thuật được gọi là "kernel trick" để chuyển dữ liệu từ không gian ban đầu sang một không gian chiều cao hơn, nơi mà nó có thể được phân chia tuyến tính. Các kernel phổ biến được sử dụng bao gồm kernel tuyến tính, kernel đa thức, và kernel Gaussian (RBF - Radial Basis Function).

SVM được sử dụng rộng rãi trong các ứng dụng như nhận diện ký tự, nhận diện chữ viết tay, nhận diện vật thể trong hình ảnh, dự đoán chuỗi thời gian, và nhiều ứng dụng khác trong lĩnh vực học máy và thị giác máy tính

# KỸ THUẬT ÁP DỤNG BÀI TOÁN NHẬN DIỆN KHUÔN MẶT

## Mô hình tổng quát



* Module phát hiện khuôn mặt: Nhằm mục đích phát hiện vùng ảnh (image region) chứa ảnh mặt cần nhận dạng, module này còn thực hiện cắt lấy phần ảnh (face cropping) chứa đúng khuôn mặt để nhận dạng
* Module tiền xử lý (preprocessing): Thực hiện việc chuẩn hóa ánh sáng nhằm làm giảm ảnh hưởng của yếu tố ánh sáng (ví dụ như bằng cách cân bằng histogram, đổi sang ảnh xám, cân bằng độ sáng của ảnh màu, .... .... )
* Module trích chọn đặc trưng (feature extraction): ở bước này các thuật toán có nhiệm vụ lấy các đặc trưng của ảnh như LBP, HOG, SIFT, .... được áp dụng lên mọi ảnh đã được qua bước tiền xử lý
* Module giảm chiều vector (dimension reduction): ở bước này sẽ nhạn nhiệm vụ giảm số chiều của không gian vector đặc điểm nhận được từ bước trước nhằm loại bỏ các thông tin thừa , tăng tính phân biệt giữa các bức ảnh của đối tượng những khác nhau
* Module phân lớp (Classification): nhằm tìm ra định danh hoặc đưa ra quyết định về bức ảnh cần nhận dạng, ở đây thuật toán KNN (K nearest neighbors) hoặc SVM (support vector machine) được áp dụng vào hệ thống
* Ngoài ra, các dữ liệu ảnh đầu vào sẽ là ảnh màu hoặc video

## Chi tiết thuật toán

### Tiền xử lý

Lọc nhiễu ảnh và cân bằng Histogram, chuản hóa ánh sáng cho phù hợp với điều kiện phát hiện ảnh.

### Trích xuất đặc trưng

Thuật toán HOG (Histogram of Oriented Gradients) và thuật toán LBP (Local Binary Pattern) đều là các phương pháp được sử dụng để rút trích đặc trưng từ hình ảnh trong lĩnh vực xử lý ảnh và nhận dạng vật thể.

1. Histogram of Oriented Gradients (HOG):

HOG là một phương pháp rút trích đặc trưng từ hình ảnh dựa trên việc tính toán hướng và độ lớn của các gradient (đạo hàm) trong các ô (cell) nhỏ của hình ảnh. HOG chủ yếu được sử dụng trong các ứng dụng nhận diện vật thể.

**Các bước tính đặc trưng HOG**

* **Chuẩn bị ảnh đầu vào:**

**Chuyển ảnh màu thành ảnh grayscale (nếu ảnh không phải là grayscale).**

**Điều chỉnh kích thước ảnh thành kích thước cố định nếu cần thiết để đơn giản hóa quá trình tính toán.**

* **Tính toán Gradient:**

**Tính đạo hàm theo x và y bằng cách sử dụng kernel gradient như Sobel để tìm gradient theo hướng x và y.**

**Tính toán magnitude và hướng của gradient**

* **Chia ảnh thành các ô (cells):**

**Chia ảnh thành các ô nhỏ không chồng lấn. Một ô thường có kích thước là 8x8 pixels.**

* **Tính toán histogram của hướng gradient trong từng ô**

**Chia mỗi ô thành các điểm (pixels).**

**Đối với mỗi điểm trong mỗi ô, thêm hướng gradient vào một trong số các bin của histogram dựa trên hướng gradient của điểm đó. Thường thì có 9 bins để bao quát 180 độ.**

**Các giá trị trong histogram thể hiện phân phối của các hướng gradient trong ô đó.**

* **Chuẩn hóa các block:**

**Ghép các ô thành các khối (blocks) bằng cách kết hợp các ô liền kề. Một khối thường chứa 2x2 ô.**

**Chuẩn hóa histogram của các khối để giảm ảnh hưởng của ánh sáng và cải thiện invariance đối với sự thay đổi về độ sáng.**

* **Kết hợp các vector đặc trưng HOG:**

**Kết hợp các vector đặc trưng HOG từ các khối thành một vector đặc trưng duy nhất để biểu diễn toàn bộ ảnh.**

1. Local Binary Pattern (LBP):

LBP là một phương pháp rút trích đặc trưng cục bộ từ hình ảnh bằng cách so sánh giá trị của các điểm lân cận với điểm trung tâm trong các ô nhỏ của hình ảnh. LBP được sử dụng chủ yếu trong việc nhận diện khuôn mặt và các ứng dụng khác trong xử lý ảnh.

Các bước tính toán thuật toán LBP trích chọn đặc trưng:

* **Chia ảnh thành các ô (cells):**

Chia ảnh thành các ô nhỏ không chồng lấn. Một ô thường có kích thước là 3x3 hoặc 5x5 pixels.

* **Tính toán LBP cho từng ô**:

Với mỗi ô, chọn một điểm trung tâm và so sánh giá trị pixel của các điểm xung quanh với điểm trung tâm.

Nếu giá trị pixel của điểm xung quanh lớn hơn hoặc bằng giá trị của điểm trung tâm, gán giá trị 1, ngược lại gán giá trị 0.

Sắp xếp giá trị nhị phân thu được thành một chuỗi 8-bit (vì mỗi ô có 8 điểm xung quanh).

* **Chuyển đổi giá trị nhị phân thành giá trị thập phân**:

Chuyển chuỗi nhị phân thành giá trị thập phân để tạo ra một số nguyên duy nhất để đại diện cho mẫu LBP của mỗi ô.

* **Kết hợp các giá trị LBP**:

Kết hợp các giá trị LBP của các ô thành một vector hoặc ma trận để biểu diễn toàn bộ ảnh.

1. Kết hợp cả 2 HoG + LBP:

Kết hợp 2 đặc trưng lại thành một để nhận diện ảnh

### Phân lớp

Sử dụng thuật toán SVM (Support Vector Machine) để phân lớp cho vật thể (khuôn mặt) – kiểm tra liệu khuôn mặt có trong ảnh hoặc video đưa vào hay là không

### Tập dữ liệu

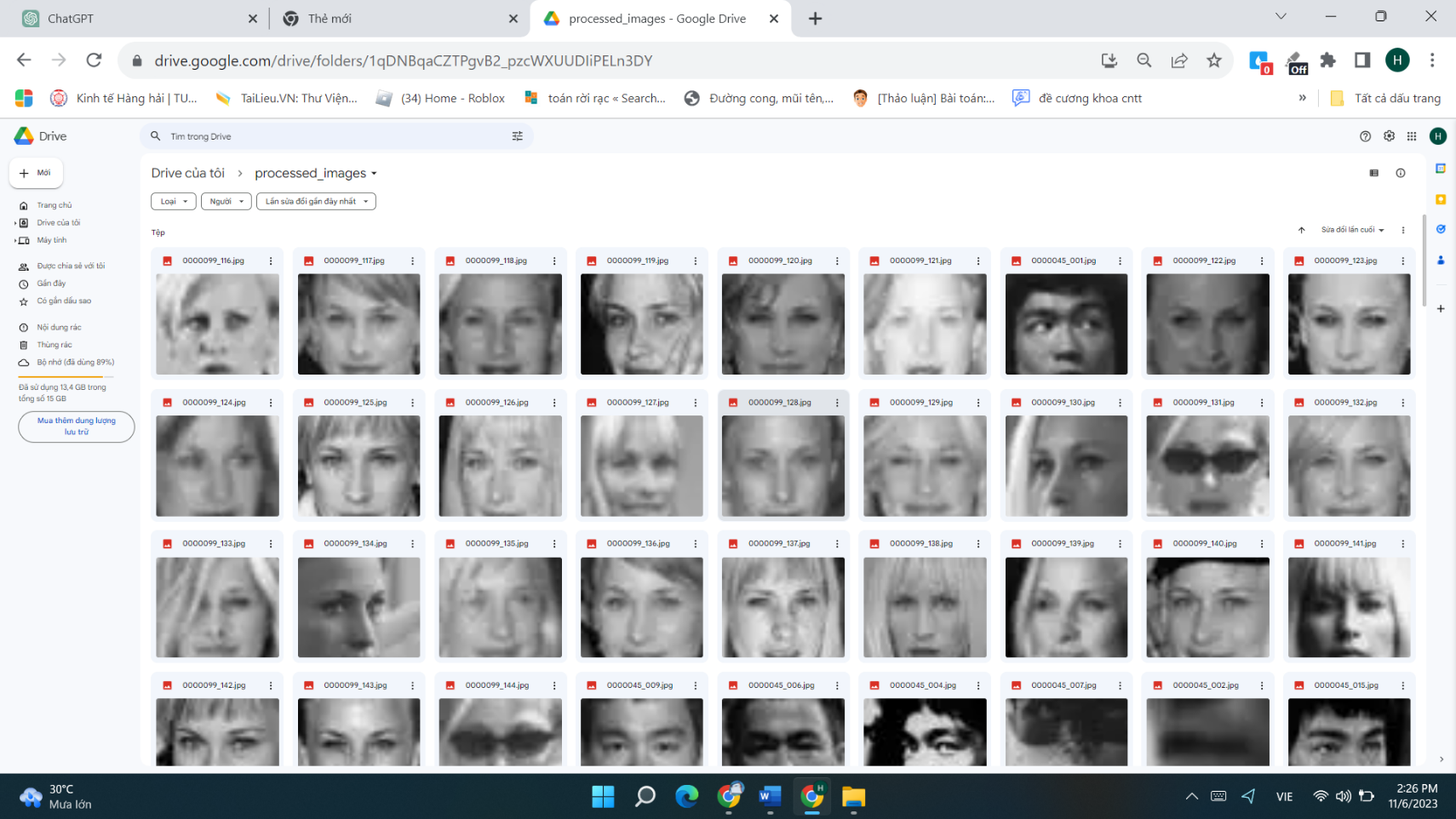
Dữ liệu để huấn luyện gồm có 2 tập:

* Tập dữ liệu huấn luyện gồm có các khuôn mặt đã được định nghĩa

Dữ liệu ảnh được lấy từ tệp CASIA\_WEBFACE + CSDL ảnh fetch\_lfw\_people

của sklearn

* Tập dữ liệu nhận dạng gồm có các khuôn mặt và không có khuôn mặt



# CÀI ĐẶT VÀ ĐÁNH GIÁ KẾT QUẢ HỆ THỐNG

## Ngôn ngữ lập trình , thư viện sử dụng , nền tảng lập trình

### Ngôn ngữ lập trình Python

Python là một ngôn ngữ lập trình thông dịch, đa năng, dễ đọc và dễ viết. Nó được tạo ra để tối đa hóa sự đơn giản và đọc được của mã nguồn, giúp người lập trình tập trung vào giải quyết vấn đề thay vì lo lắng về cú pháp phức tạp. Python được sử dụng rộng rãi trong nhiều lĩnh vực như phát triển web, trí tuệ nhân tạo, khoa học dữ liệu, thị giác máy tính và nhiều ứng dụng khác.

**Ưu Điểm của Python**:

Dễ Đọc và Viết: Python có cú pháp rõ ràng, dễ đọc và viết, giúp người lập trình tập trung vào logic của chương trình hơn là cú pháp phức tạp.

Đa Năng: Python được sử dụng trong nhiều lĩnh vực khác nhau, bao gồm phát triển web, trí tuệ nhân tạo, khoa học dữ liệu, tự động hóa, game, và nhiều ứng dụng khác.

Thư Viện Phong Phú: Python có một cộng đồng lớn và nhiều thư viện mạnh mẽ được xây dựng để hỗ trợ nhiều nhiệm vụ khác nhau, bao gồm thị giác máy tính (OpenCV), học máy (Scikit-learn), và đồ họa (Matplotlib).

Thực Thi Tự Động: Python là một ngôn ngữ thông dịch, điều này có nghĩa là bạn có thể thực thi các dòng mã một cách trực tiếp mà không cần biên dịch.

Hỗ Trợ Đa Nhiệm: Python hỗ trợ lập trình đa luồng và đa tiến trình, giúp xử lý đa nhiệm và tận dụng tối đa tài nguyên hệ thống.

**Ưu Điểm của Python trong Thị Giác Máy Tính (Computer Vision)**:

Thư Viện Mạnh Mẽ: Python có các thư viện như OpenCV (Open Source Computer Vision Library) cung cấp các công cụ mạnh mẽ cho việc xử lý hình ảnh, nhận diện vật thể, theo dõi đối tượng và nhiều nhiệm vụ thị giác máy tính khác.

Dễ Kết Hợp: Python dễ kết hợp với các công cụ và thư viện khác, giúp việc tích hợp các hệ thống thị giác máy tính vào các ứng dụng và dự án trở nên thuận lợi.

Cộng Đồng Lớn: Cộng đồng lớn của Python và sự hỗ trợ từ cộng đồng này giúp giải quyết vấn đề và tìm kiếm giải pháp cho các thách thức trong lĩnh vực thị giác máy tính.

Như vậy, Python không chỉ là một ngôn ngữ lập trình dễ học và sử dụng, mà còn là một công cụ mạnh mẽ trong lĩnh vực thị giác máy tính và nhiều lĩnh vực khác nữa.

### Thư viện scikit-learn

scikit-learn, thường được gọi là sklearn, là một thư viện mã nguồn mở phổ biến trong ngôn ngữ lập trình Python được thiết kế cho các nhiệm vụ liên quan đến học máy (machine learning), khai phá dữ liệu (data mining) và phân tích dữ liệu (data analysis). Nó cung cấp các công cụ đơn giản và hiệu quả để xây dựng và đánh giá các mô hình học máy.

Dưới đây là một số chức năng chính của scikit-learn:

Thuật toán Học Máy: Sklearn cung cấp nhiều thuật toán học máy như hồi quy tuyến tính, hồi quy logistic, máy vector hỗ trợ (SVM), cây quyết định, và nhiều thuật toán khác cho các nhiệm vụ phân loại, dự đoán và gom cụm.

Tiền xử lý Dữ liệu: Nó chứa các công cụ để tiền xử lý dữ liệu, bao gồm chuẩn hóa, mã hóa biến hạng mục, xử lý giữ liệu thiếu, và trích xuất đặc trưng (feature extraction).

Đánh Giá Mô Hình: Sklearn cung cấp các hàm để đánh giá hiệu suất của các mô hình học máy thông qua các phép đo như độ chính xác, độ đo F1, và roc\_auc\_score.

Lựa Chọn Mô Hình và Tối Ưu Hóa Tham Số: Sklearn cung cấp các công cụ để lựa chọn mô hình tốt nhất và tối ưu hóa các tham số của mô hình thông qua kỹ thuật như lưới tìm kiếm (Grid Search) và tìm kiếm ngẫu nhiên (Random Search).

### Thư Viện OpenCV

OpenCV (Open Source Computer Vision Library) là một thư viện mã nguồn mở chuyên về xử lý hình ảnh và thị giác máy tính. Nó được thiết kế để cung cấp các công cụ và thuật toán mạnh mẽ cho việc xử lý, phân tích và nhận diện hình ảnh và video. Dưới đây là một số chức năng chính và tính năng của OpenCV:

Xử Lý Hình Ảnh: OpenCV cung cấp các hàm và phương pháp để thực hiện các thao tác xử lý hình ảnh như chuyển đổi màu sắc, làm mịn hình ảnh, điều chỉnh độ tương phản và độ sáng, cắt ảnh, xoay ảnh và nhiều thao tác khác.

Nhận Diện Đối Tượng và Khuôn Mặt: OpenCV cung cấp các thuật toán để nhận diện đối tượng trong hình ảnh và video, bao gồm cả nhận diện khuôn mặt, mắt và các vật thể khác.

Xử Lý Video: Thư viện này cho phép bạn thực hiện các thao tác xử lý trên video như trích xuất khung hình, ghi video, và thậm chí cả theo dõi và phân loại đối tượng trong video.

Xử Lý Thị Giác Máy Tính (Computer Vision): OpenCV bao gồm các thuật toán và công cụ cho việc xử lý thị giác máy tính như tính toán biểu đồ hướng gradient (HOG), Local Binary Pattern (LBP), và rất nhiều thuật toán nhận diện đối tượng khác.

Xử Lý Thị Giác Máy Tính Tiên Tiến: OpenCV hỗ trợ các tính năng tiên tiến như phân loại sử dụng mạng nơ-ron (deep learning) thông qua module dnn.

Hỗ Trợ Đa Nền Tảng: OpenCV được thiết kế để hoạt động trên nhiều nền tảng như Windows, Linux, macOS và cả các thiết bị di động

OpenCV là một công cụ mạnh mẽ và phổ biến trong lĩnh vực thị giác máy tính và xử lý hình ảnh, được sử dụng rộng rãi trong nhiều ứng dụng như nhận diện khuôn mặt, xe hơi tự lái, kiểm tra chất lượng sản phẩm, và nhiều lĩnh vực khác

### Nền tảng lập trình – Anaconda:

Anaconda là nền tảng mã nguồn mở về Khoa học dữ liệu (Data Science) trên Python thông dụng nhất hiện nay. Với hơn 6 triệu người dùng, Anaconda là cách nhanh nhất và dễ nhất để học Khoa học dữ liệu với Python hoặc R trên Windows, Linux và Mac OS X

## Mã nguồn chương trình

File train.py

import matplotlib.pyplot as plt

import seaborn as sns; sns.set()

import numpy as np

import cv2

from skimage import data, color, feature

import skimage.data

from sklearn.pipeline import make\_pipeline

from sklearn.preprocessing import StandardScaler

from sklearn.svm import SVC

import joblib

from skimage import data, transform

from sklearn.feature\_extraction.image import PatchExtractor

from itertools import chain

from sklearn.naive\_bayes import GaussianNB

from sklearn.model\_selection import cross\_val\_score

from sklearn.svm import LinearSVC

from sklearn.model\_selection import GridSearchCV

dataset\_folder = "/content/drive/MyDrive/haascades/"

faces1 = fetch\_lfw\_people()

faces2 = fetch\_olivetti\_faces()

patches1 = faces1.images

patches2 = faces2.images

desired\_shape = (45, 45)

positive\_patches1\_resized = [resize(image, desired\_shape) for image in patches1]

positive\_patches2\_resized = [resize(image, desired\_shape) for image in patches2]

positive\_patches = np.concatenate([positive\_patches1\_resized, positive\_patches2\_resized], axis=0)

path\_imgidx = "/content/drive/MyDrive/zip/CASIA-WebFace\_crop.zip"

imgzip = zipfile.ZipFile(path\_imgidx)

inflist = imgzip.infolist()

image\_file = []

cropped\_img = []

crop = []

X\_train = []

npoint = 8

rad = 1.0

def detect(img, idx):

faces = face\_cascade.detectMultiScale(img, scaleFactor=1.1, minNeighbors=5, minSize=(45, 45))

try:

x, y, w, h = faces[0]

img = img[y:y+h, x:x+w]

img = cv2.resize(img, (45,45))

except:

img = None

return img

for f in inflist:

ifile = imgzip.open(f)

img = Image.open(ifile)

img = color.rgb2gray(img)

img = (img \* 255).astype(np.uint8)

image\_file.append(img)

for i, img in enumerate(image\_file):

img = detect(img, i)

if img is not None:

cropped\_img.append(img)

for i in cropped\_img:

img = cv2.cvtColor(i, cv2.COLOR\_BGR2GRAY)

crop.append(img)

positive\_patches = np.concatenate([positive\_patches, cropped\_img], axis=0)

imgs\_to\_use = ['camera', 'text', 'coins', 'moon',

'page', 'clock', 'immunohistochemistry',

'chelsea', 'coffee', 'hubble\_deep\_field']

images = [getattr(data, name)()

for name in imgs\_to\_use]

for i, data in enumerate(images):

if len(data.shape) > 2:

images[i] = color.rgb2gray(data)

def extract\_patches(img, N, scale=1.0, patch\_size=positive\_patches[0].shape):

extracted\_patch\_size = tuple((scale \* np.array(patch\_size)).astype(int))

extractor = PatchExtractor(patch\_size=extracted\_patch\_size,

max\_patches=N, random\_state=0)

patches = extractor.transform(img[np.newaxis])

if scale != 1:

patches = np.array([transform.resize(patch, patch\_size)

for patch in patches])

return patches

negative\_patches = np.vstack([extract\_patches(im, 1000, scale)

for im in images for scale in [0.5, 1.0, 2.0]])

for im in chain(positive\_patches, negative\_patches):

# LBP

lbp\_feature = feature.local\_binary\_pattern(im, npoint, rad)

lbp\_feature = np.histogram(lbp\_feature.ravel(), bins=np.arange(0, npoint + 3), range=(0, npoint + 2))

lbp\_feature = lbp\_feature[0]

# HOG

hog\_feature = feature.hog(im)

concatenated\_feature = np.concatenate((hog\_feature, lbp\_feature))

X\_train.append(concatenated\_feature)

X\_train = np.array(X\_train)

y\_train = np.zeros(X\_train.shape[0])

y\_train[:positive\_patches.shape[0]] = 1

cross\_val\_score(GaussianNB(), X\_train, y\_train)

grid = GridSearchCV(LinearSVC(), {'C': [1.0, 2.0, 4.0, 8.0]})

model = grid.best\_estimator\_

model = make\_pipeline(StandardScaler(), SVC(C=1, gamma='auto', probability=True))

model.fit(X\_train, y\_train)

joblib.dump(model, "best\_model.pkl")

File dang.py

# -\*- coding: utf-8 -\*-

"""

Created on Thu Sep 28 06:42:28 2023

@author: ADMIN

"""

import cv2

import os

import sys

import time

import numpy as np

import joblib

import matplotlib.pyplot as plt

import shutil

import glob

from skimage import data, color, feature, transform

import skimage.data

from joblib import Parallel, delayed

import imutils

cv2.namedWindow('frame', cv2.WINDOW\_NORMAL)

# Mặt

cap = cv2.VideoCapture(0)

# cap = cv2.VideoCapture("D:/tai xuong/20210628\_Face\_Detection\_with\_Dlib\_using\_HOG\_and\_Linear\_SVM/20210628\_Face\_Detection\_with\_Dlib\_using\_HOG\_and\_Linear\_SVM/input/video1.mp4")

dataset\_folder = "D:/tai xuong/"

model = joblib.load(dataset\_folder + "best\_model (2) .pkl")

fps = 0

frame\_counter = 0

start\_time = time.time()

def kiemtra(test\_image):

test\_image = cv2.cvtColor(test\_image, cv2.COLOR\_BGR2GRAY)

indices, patches = zip(\*sliding\_window(test\_image))

npoint = 8

rad = 1.0

patches\_hog = []

for im in patches:

# áp dụng LBP

lbp\_feature = feature.local\_binary\_pattern(im, npoint, rad)

lbp\_feature = np.histogram(lbp\_feature.ravel(), bins=np.arange(0, npoint + 3), range=(0, npoint + 2))[0]

# áp dụng HOG

hog\_feature = feature.hog(im)

# kết hợp vào LBP và HOG vector đặc trưng

concatenated\_feature = np.concatenate((hog\_feature, lbp\_feature))

# dự đoán + tập hợp các đặc trưng lại vào 1 danh sách list patches\_hog

# result = model.predict\_proba([concatenated\_feature])

patches\_hog.append(concatenated\_feature)

labels = model.predict(patches\_hog)

indices = np.array(indices)

for i, j in indices[labels == 1]:

k = cv2.rectangle(img, (j, i), (j + 45, i + 45), (0, 255, 0), 2)

# cv2.putText(img, '{}: {:.2f}%'.format("face", (result[0][0]) \* 100 - 1), (j, i - 5),

# cv2.FONT\_HERSHEY\_COMPLEX, 0.25, (0, 0, 255), 1)

// Cửa sổ trượt

def sliding\_window(img, patch\_size=(45,45),

istep=15, jstep=15, scale=1):

Ni, Nj = (int(scale \* s) for s in patch\_size)

for i in range(0, img.shape[0] - Ni, istep):

for j in range(0, img.shape[1] - Ni, jstep):

patch = img[i:i + Ni, j:j + Nj]

if scale != 1:

patch = transform.resize(patch, patch\_size)

yield (i, j), patch

while True:

ret, img = cap.read()

frame\_counter += 1

fps = (frame\_counter / (time.time() - start\_time))

if img is not None:

img = cv2.resize(img, (155, 155))

kiemtra(img)

img = imutils.resize(img, width=410)

cv2.putText(img, 'FPS: {:.2f}'.format(fps), (20, 20), cv2.FONT\_HERSHEY\_COMPLEX, 0.7, (0, 0, 255),2)

cv2.imshow('Capture Faces', img)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

"""

cap.release()

cv2.destroyAllWindows()

img = cv2.imread("D:/tai xuong/images.jpg")

img = cv2.resize(img, (150, 150))

kiemtra(img)

cv2.imshow('Capture Faces', img)

if cv2.waitKey(1) & 0xFF == ord('q'):

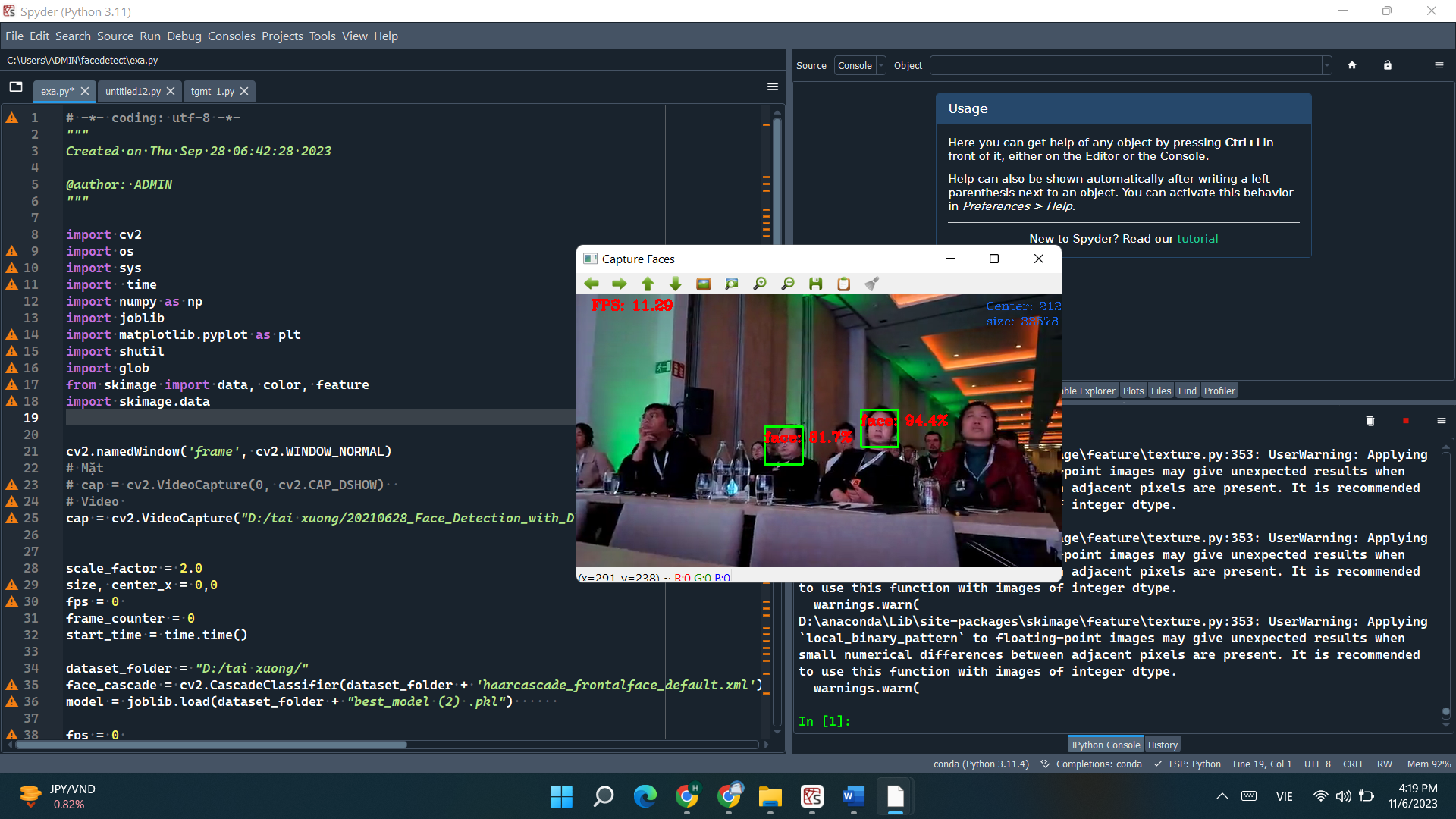
cv2.destroyAllWindows()

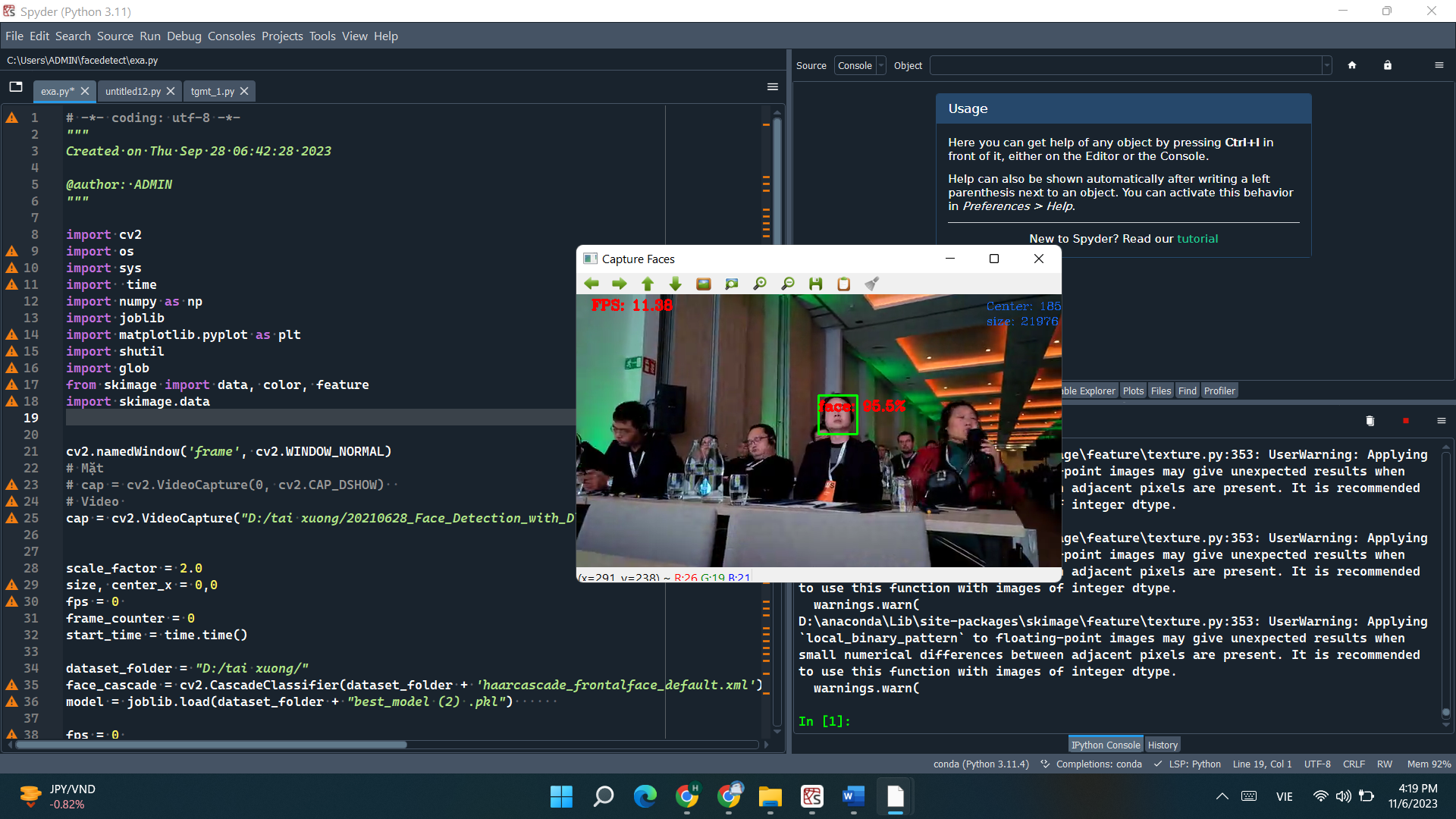
"""

## Nhận xét kết quả

* Chạy chậm do chưa có tối ưu thuật toán
* Chỉ nhận diện ở các khuôn mặt ở gần màn hình
* Nhận diện tốt khuôn mặt ở mức gần đến > 90%

## Hình ảnh nhận diện





## Nhận xét về hệ thống

* Các ảnh có góc bị thiếu ánh sáng tỷ lệ giảm dưới 80%
* Các ảnh chưa đủ điều kiện để thực hiện nhận dạng thì độ chính xác khoảng 70%
* Một số khuôn mặt do các yếu tố về tuổi tác, biểu cảm bị ảnh hưởng
* Một số trường hợp không nhận dạng được nếu đeo vật thể trên mặt (kính, mặt nạ khẩu trang), .... hoặc nhận dạng sai chỗ khuôn mặt định vị như ở trên mặt hoặc dưới khuôn mặt

# ĐÁNH GIÁ

## Quy trình

* Mục đích: Nhận dạng mặt người trên ảnh hoặc video
* Phương pháp sử dụng: Thuật toán HOG + LBP
* Ngôn ngữ, thư viên: Python, anaconda, OpenCV, scikit-learn
* Kết quả: Chương trình xác định được khuôn mặt trong môi trường

## Ưu điểm và nhược điểm

Ưu điểm:

* HOG + LBP kết hợp cả hai giúp hệ thống nhận dạng có sự đa dạng trong việc trích xuất đặc trưng từ hình ảnh.
* LBP + HOG kết hợp cả hai giúp hệ thống nhận dạng giảm thiểu ảnh hưởng từ nhiễu.
* Nhận diện nhiều mặt người trong một khung ảnh
* Nhận diện chính xác hơn 98% nếu ở trong môi trường tương đối tốt, phù hợp hoàn cảnh

Nhược điểm:

* Khả năng Nhận Diện Trong Điều Kiện Khó Khăn: Khi áp dụng chỉ HOG và LBP, hệ thống có thể không hiệu quả đối với các ảnh khuôn mặt có các biến đổi lớn về ánh sáng, góc chụp, hoặc nhiễu. Các biến đổi này có thể làm giảm độ chính xác của quá trình nhận diện.
* Độ Chính Xác Thấp Trong Các Trường Hợp Cụ Thể: Trong một số tình huống, đặc biệt là khi khuôn mặt bị che khuất, độ chính xác của HOG và LBP có thể giảm đi đáng kể. Các đặc trưng này không nhất thiết phản ánh các thông tin đặc biệt về các đối tượng khuôn mặt khi một phần của khuôn mặt bị che khuất hoặc ẩn sau các vật thể khác.
* Tính Nhạy Cảm Với Thay Đổi Tập Dữ Liệu: HOG và LBP có thể yêu cầu một lượng lớn dữ liệu huấn luyện để tối ưu hóa hiệu suất của chúng. Khi tập dữ liệu huấn luyện không đủ lớn hoặc không đại diện cho đủ các điều kiện chụp hình thực tế, hệ thống có thể trở nên không ổn định và không chính xác.
* Thời Gian Xử Lý Tăng Lên: Việc tính toán HOG và LBP đòi hỏi một lượng lớn tính toán, đặc biệt là đối với các hình ảnh có độ phân giải cao. Điều này có thể dẫn đến việc tăng thời gian xử lý, làm giảm hiệu suất của hệ thống trong các ứng dụng thời gian thực.
* Trong thực tế, các hệ thống nhận diện khuôn mặt hiện đại thường kết hợp nhiều kỹ thuật và mô hình học sâu để cải thiện độ chính xác và ổn định

## Hướng phát triển

Hoàn thiện phần nhận dạng ở điều kiện MT chưa tốt

Tránh các lỗi phát sinh như bị trễ frame ảnh, video khi nhận diện video

Cải thiện thêm phần nhận dạng các khuôn mặt ở vị trí từ xa

Cải thiện tốc độ FPS + Cải thiện phần nhận dạng ở vị trị khác nhau tùy vào từng điều kiện cự thể của môi trường và không gian xung quang.

**KẾT LUẬN**

Trong bài tập lớn này, nhóm bọn em đã tập trung vào việc nghiên cứu và phát triển một hệ thống nhận dạng ảnh khuôn mặt trong cả ảnh và video. Qua quá trình nghiên cứu, nhóm em đã áp dụng và thử nghiệm các phương pháp và thuật toán để xây dựng một hệ thống nhận dạng chính xác và hiệu quả.

Một trong những thách thức lớn nhất mà chúng em đã gặp phải là xử lý dữ liệu ảnh và video có chất lượng thấp và biến động về ánh sáng, góc chụp, và độ nghiêng.

Trong quá trình thực hiện dự án, chúng em cũng nhận thức được sức mạnh của việc kết hợp giữa học sâu (deep learning) và phương pháp truyền thống. Sự kết hợp này giúp tăng cường khả năng nhận diện khuôn mặt trong các tình huống đa dạng và phức tạp.

Ngoài ra, chúng em cũng đã học được rất nhiều về quản lý dự án, tư duy phân tích và giải quyết vấn đề, cũng như kỹ năng làm việc nhóm và giao tiếp hiệu quả. Những kỹ năng này không chỉ hữu ích trong lĩnh vực nghiên cứu khoa học mà còn trong các lĩnh vực khác của cuộc sống.

Tổng kết lại, bài tập lớn nhận dạng ảnh khuôn mặt đã mang lại cho chúngem những trải nghiệm quý báu và học được nhiều kiến thức mới. Chúng em hy vọng rằng bài tập lớn này không chỉ là kết quả của sự nỗ lực của chúng em mà còn là sự đóng góp nhỏ bé vào lĩnh vực ngày càng phát triển của công nghệ nhận dạng ảnh.

Mỗi ý kiến và sự góp ý của thầy (Cô) sẽ là nguồn động lực và kinh nghiệm cho chúng em để hoàn thiện nốt bài tập lơn này ạ. Chúng em xin chân thành cảm ơn thầy (Cô).