

## NIÊN LUẬN NGÀNH KHOA HỌC MÁY TÍNH

**Đề tài**

**CHỨNG THỰC BẰNG GƯƠNG MẶT KẾT NỐI VÀO CÁC HỆ THỐNG ELEARNING**

## Sinh viên thực hiện: Phan Duy Anh MSSV: B1812252

**Khóa: 44**



**NIÊN LUẬN NGÀNH  
 KHOA HỌC MÁY TÍNH**

**Đề tài**

**CHỨNG THỰC BẰNG GƯƠNG MẶT KẾT NỐI VÀO CÁC HỆ THỐNG ELEARNING**

## Giảng viên hướng dẫn: Sinh viên thực hiện:

**TS. Lưu Tiến Đạo Phan Duy Anh**

**MSSV: B1812252**

**Khóa: 44**

**NHẬN XÉT CỦA GIẢNG VIÊN**

### -----------------------------------------------------------------------------------------------------

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

**-----------------------------------------------------------------------------------------------------**

Kính thưa!

# LỜI CẢM ƠN

Lời nói đầu tiên, em xin gửi lời cảm ơn chân thành và sâu sắc nhất dành cho các Thầy Cô của trường Đại Học Cần Thơ, đặc biệt là các Thầy Cô khoa Công Nghệ Thông Tin và Truyền Thông của Trường Đại Học Cần Thơ đã giảng dạy cho em những môn học chuyên ngành để làm tiền đề giúp em hoàn thành bài báo cáo đồ án niên luận ngành này.

Để báo cáo này đạt kết quả tốt đẹp em đã nhận được sự góp ý từ các anh chị và bạn bè cùng khoa. Em gửi lời cảm ơn chân thành đến tất cả các anh chị bạn bè đã giúp đỡ em trong quá trình học tập và hoàn thành bài báo cáo này.

Và em cũng xin cám ơn thầy Lưu Tiến Đạo đã hướng dẫn cho em những kiến thức cần thiết, giúp em ứng dụng hoàn thành bài báo cáo. Tuy nhiên, trong suốt quá trình thực hiện đề tài em còn nhiều thiếu sót, hạn chế. Vì thế em mong thầy cô xem xét bỏ qua và đóng góp ý kiến để em có thể hoàn thiện bản thân, cũng như tích lũy thêm nhiều kinh nghiệm cho những đồ án, đề tài sau này được tốt hơn.

Em xin chân thành cảm ơn! Trân trọng!

Cần Thơ, ngày tháng năm 2021 Người viết

# MỤC LỤC

[GIỚI THIỆU 6](#_Toc103886000)

[1. Đặt vấn đề 6](#_Toc103886001)

[2. Lịch sử giải quyết vấn đề 6](#_Toc103886002)

[3. Mục tiêu đề tài 7](#_Toc103886003)

[4. Đối tượng và phạm vi nghiên cứu 7](#_Toc103886004)

[5. Phương pháp nghiên cứu 7](#_Toc103886005)

[6. Kết quả đạt được 8](#_Toc103886006)

[7. Bố cục niên luận 8](#_Toc103886007)

[PHẦN NỘI DUNG 9](#_Toc103886008)

[CHƯƠNG 1: MÔ TẢ BÀI TOÁN 9](#_Toc103886009)

[1. Mô tả chi tiết bài toán 9](#_Toc103886010)

[2. Cơ sở lý thuyết 9](#_Toc103886011)

[2.1. KNN 9](#_Toc103886012)

[2.2. OpenCV 11](#_Toc103886013)

[2.3. Autofaiss 12](#_Toc103886014)

[2.4. MTCNN 14](#_Toc103886015)

[2.4.1. The proposal network (P-Net) 15](#_Toc103886016)

[2.4.2. The Refine Network (R-Net) 16](#_Toc103886017)

[2.5. Spring 18](#_Toc103886018)

[2.6. Flask 19](#_Toc103886019)

[CHƯƠNG 2. THIẾT KẾ VÀ CÀI ĐẶT 20](#_Toc103886020)

[1. Quy trình phát hiện và nhận dạng gương mặt 20](#_Toc103886021)

[2. Mô hình phát hiện gương mặt 20](#_Toc103886022)

[2.1. Mô hình MTCNN 20](#_Toc103886023)

[2.2. Load mô hình MTCNN vào biến 21](#_Toc103886024)

[2.3. Chuẩn bị ảnh đầu vào 21](#_Toc103886025)

[2.4. Xử lý đầu ra 21](#_Toc103886026)

[2.5. Kết quả 21](#_Toc103886027)

[*3.* Tạo KNN index cho tập ảnh dữ liệu 21](#_Toc103886028)

[Kết quả 22](#_Toc103886029)

[4. Mô hình nhận dạng 23](#_Toc103886030)

[5. Dự đoán 24](#_Toc103886031)

[6. Xây dựng website 25](#_Toc103886032)

[6.1. Thiết lập Flask 25](#_Toc103886033)

[6.2. Trang chủ 25](#_Toc103886034)

[6.3. Trang xử lý 25](#_Toc103886035)

[6.4. Các hàm xử lý 26](#_Toc103886036)

[6.5. Kết quả đầu ra 26](#_Toc103886037)

[7. Giao diện hệ thống 26](#_Toc103886038)

[CHƯƠNG 3. KIỂM THỬ VÀ ĐÁNH GIÁ 27](#_Toc103886039)

[1. Môi trường kiểm thử 27](#_Toc103886040)

[2. Kiểm thử 27](#_Toc103886041)

[Mô hình phát hiện gương mặt: 27](#_Toc103886042)

[3. Đánh giá 27](#_Toc103886043)

[3.1. Phát hiện gương mặt 27](#_Toc103886044)

[PHẦN KẾT LUẬN 28](#_Toc103886045)

[1. Kết quả đạt được 28](#_Toc103886046)

[2. Ưu và nhược điểm 28](#_Toc103886047)

[2.1. Ưu điểm 28](#_Toc103886048)

[2.2. Nhược điểm 28](#_Toc103886049)

[3. Hướng phát triển 28](#_Toc103886050)

[TÀI LIỆU THAM KHẢO 29](#_Toc103886051)

## DANH MỤC HÌNH

[Hình 1. Mô hình KNN 12](#_Toc103882796)

[Hình 2. Quantize Vectors 14](#_Toc103882797)

[Hình 3. Kiến trúc của Quantize Vectors 15](#_Toc103882798)

[Hình 4. Trích xuất đặc trưng MTCNN 16](#_Toc103882799)

[Hình 5. Mô hình MTCNN 17](#_Toc103882800)

[Hình 6. P-Net Layer 18](#_Toc103882801)

[Hình 7. R-Net Layer 19](#_Toc103882802)

[Hình 8. O-Net Layer 20](#_Toc103882803)

[Hình 9. Quy trình phát hiện và nhận dạng gương mặt 22](#_Toc103882804)

[Hình 10. Kết quả phát hiện gương mặt 23](#_Toc103882805)

[Hình 11. Kết quả KNN index 24](#_Toc103882806)

[Hình 12. Thông tin file infos.json 25](#_Toc103882807)

[Hình 13. Xây dựng mô hình nhận dạng 25](#_Toc103882808)

[Hình 14. Kết quả mô hình nhận dạng 26](#_Toc103882809)

[Hình 15. Sơ đồ hệ thống 27](#_Toc103882810)

[Hình 16. Giao diện website 28](#_Toc103882811)

## ABSTRACT

E-learning is a new teaching and learning method based on a system connected to the Internet. This platform allows teachers and students to communicate, interact and exchange documents and lesson plans with each other without meeting face-to-face. Along with the development of society, many e-learning systems appear more and more diverse in form. Usually e-learning systems only authenticate users with login accounts and information theft is easy to happen if users reveal information. This is the cause leading to the birth of facial recognition systems connected to elearning systems.

There are many different face recognition methods available. Each method has certain advantages and limitations. This topic presents face recognition method by applying machine learning method.

The system using face detection model by MTCNN has been studied before. With the image processing methods of the OpenCV library to write, read the face images and build the search model using KNN with Autofaiss create and adjust the quantized KNN indexes to search a efficiently similarity with low memory usage.

.

## TÓM TẮT

E-learning là phương pháp giảng dạy và học tập mới được thực hiện dựa trên một hệ thống có kết nối mạng Internet. Nền tảng này cho phép giáo viên và học sinh giao tiếp, tương tác trao đổi tài liệu, giáo án với nhau mà không cần gặp mặt trực tiếp. Cùng với sự phát triển của xã hội, nhiều hệ thống e-learning ngày càng xuất hiện nhiều và đa dạng về hình thức. Thông thường các hệ thống e-learning chỉ xác thực người dùng bằng tài khoản đăng nhập và việc đánh cắp thông tin rất dễ xảy ra nếu người dùng để lộ thông tin. Đây là nguyên nhân dẫn đến sự ra đời của hệ thống nhận dạng gương mặt kết nối vào các hệ thống elearning.

Hiện có nhiều phương pháp nhận dạng gương mặt khác nhau. Mỗi phương pháp có ưu điểm và hạn chế nhất định. Đề tài này trình bày phương pháp nhận dạng gương mặt bằng cách áp dụng phương pháp máy học.

Hệ thống sử dụng mô hình phát hiện gương mặt bằng MTCNN đã được nghiên cứu trước đó. Với các phương pháp xử lý ảnh của thư viện OpenCV để ghi, đọc các hình ảnh gương mặt và xây dựng mô hình tìm kiếm bằng KNN với Autofaiss tạo và điều chỉnh các index (chỉ mục) KNN được quantize để tìm kiếm một cách hiệu quả độ tương đồng với mức sử dụng bộ nhớ thấp.

## GIỚI THIỆU

### Đặt vấn đề

Xã hội ngày càng phát triển thì công nghệ cũng ngày càng phát triển theo từ các cuộc cách mạng công nghiệp đầu tiên đến bây giờ là cuộc cách mạng công nghiệp 4.0. Là cuộc cách mạng mà cốt lõi là công nghệ thông tin với các thiết bị thông minh như robot hay xe tự lái để tăng năng suất lao động và phục vụ cho các nhu cầu đời sống con người. Cùng với sự phát triển của khoa học công nghệ, trí tuệ nhân tạo được xem là nhân tố định hình tương lai. Với sự phát triển nhanh chóng trong những năm gần đây, trí tuệ nhân tạo đã và đang đi vào thực tiễn của cuộc sống. Hàng loạt các công nghệ mới sử dụng trí tuệ nhân tạo đã ra đời giúp ích cho con người trong nhiều lĩnh vực như an ninh, y học, giáo dục, bảo mật,…

Trong thời hiện đại, việc tiếp nhận và trao đổi thông tin với nhau là rất cần thiết. Con người có thể trao đổi thông tin trực tiếp với nhau bằng lời nói, cử chỉ, ngoài ra còn có thể trao đổi gián tiếp với nhau thông qua hình ảnh, văn bản,…

Khi loài người chưa có chữ viết, con người đã biết dùng lối vẽ làm phương tiện thông tin. Trong các hang động cổ xưa có nhiều bức tranh động vật được khắc lên vách đá, họ thông báo cho nhau những điều cần biết .Từ tranh chuyển sang chữ viết là một quá trình trừu tượng hoá, sau dần người ta lược bỏ các chi tiết cụ thể, phức tạp, dùng các đường nét đơn giản làm kí hiệu ghi lại ngôn ngữ, mở rộng thông tin cho con người.

Hình ảnh xuất hiện ở khắp mọi nơi trong môi trường xung quanh của chúng ta như cuộc sống, trường học, văn phòng,… Hầu hết tất cả mọi người đều có thể tiếp cận hình ảnh một cách dễ dàng bằng việc nhìn bằng mắt thường để xác định nội dung hình ảnh truyền đạt. Tuy nhiên việc tiếp cận hình ảnh vẫn còn khó khăn với những hệ thống máy tính.

Với mục tiêu tạo ra sản phẩm hỗ trợ con người trong việc xác thực gương mặt trong các hệ thống e-learning. Hệ thống xây dựng dựa trên nền tảng máy học để phù hợp với thời đại công nghệ.

Hệ thống sẽ mang đến lợi ích giúp người dùng có thể dễ dàng nhận biết được gương mặt cần xác thực là ai bằng các thư viện xử lý ảnh của thị giác máy tính. Cùng việc xây dựng mô hình với học máy sẽ làm tăng độ chính xác về việc nhận diện gương mặt qua các hình ảnh.

### Lịch sử giải quyết vấn đề

Ở Việt nam hiện nay, ứng dụng máy học vào các ứng dụng cũng đã rất phổ biến nhưng áp dụng nhận dạng gương mặt tích hợp vào các hệ thống cũng chưa nhiều, đặc biệt là nhận dạng gương mặt trong hệ thống e-learning.

Hệ thống nhận dạng khuôn mặt là một ứng dụng máy tính tự động xác định hoặc

nhận dạng một người nào đó từ một bức hình ảnh kỹ thuật số hoặc một khung hình từ một nguồn video. Một trong những cách để thực hiện điều này là so sánh các đặc điểm khuôn mặt chọn trước từ hình ảnh và một cơ sở dữ liệu về khuôn mặt.

Hệ thống này thường được sử dụng trong các hệ thống an ninh và có thể được so sánh với các dạng sinh trắc học khác như các hệ thống nhận dạng vân tay hay tròng mắt.

Những người tiên phong đầu tiên về công nghệ nhận diện khuôn mặt là Woody Bledsoe, Helen Chan Wolf và Charles Bisson. Năm 1964 và 1965, Bledsoe cùng với Wolf và Bisson bắt đầu sử dụng máy tính để nhận diện khuôn mặt con người. Do kinh phí của dự án được bắt nguồn từ một cơ quan tình báo giấu tên, phần lớn công việc đều không được công bố. Tuy nhiên, sau này người ta tiết lộ rằng hầu hết các công việc đều liên quan đến việc đánh dấu bằng tay các điểm khác nhau trên khuôn mặt như tâm mắt, miệng, v.v. Khoảng cách giữa các điểm mốc cũng được tự động tính toán và so sánh giữa các hình ảnh khác nhau để xác định danh tính.

Cơ quan Dự án Nghiên cứu Tiên tiến Quốc phòng (DAPRA) và Viện Tiêu chuẩn và Công nghệ Quốc gia (NIST) đã triển khai chương trình Công nghệ Nhận diện Khuôn mặt (FERET) vào đầu những năm 1990 để khuyến khích thị trường của công nghệ này. Dự án liên quan đến việc tạo ra một cơ sở dữ liệu về hình ảnh khuôn mặt. Trong bộ thử nghiệm, có tới 2.413 ảnh tĩnh đại diện cho 856 người. Chương trình này là bước ngoặt lớn cho công nghệ nhận diện khuôn mặt và những ngành nghề liên quan tới nó.

### Mục tiêu đề tài

Xây dựng hệ thống thu nhận hình ảnh đầu vào từ việc upload file hoặc camera là ảnh có chứa gương mặt, sau đó phát hiện được gương mặt và so sánh độ tương đồng của nó với các gương mặt có trong cơ sở dữ liệu.

### Đối tượng và phạm vi nghiên cứu

*Đối tượng nghiên cứu*: Đề tài nghiên cứu về phương pháp phát hiện gương mặt bằng mô hình MTCNN được nghiên cứu trước đó. Tiền xử lý ảnh đầu vào bằng thư viện OpenCV của thị giác máy tính và nhận dạng gương mặt bằng kỹ thuật máy học với giải thuật KNN (K-Nearest Neighbors).

*Phạm vi nghiên cứu*: Nghiên cứu cách thu nhận hình ảnh từ upload file và từ camera, phát hiện khu vực chứa các gương mặt có trong hình ảnh, xử lý ảnh để tìm các đặc trưng của ảnh, phân lớp hình ảnh để đưa ra dự đoán về nhận dạng chính xác.

### Phương pháp nghiên cứu

*Lý thuyết*: Tìm hiểu về lĩnh vực xử lý ảnh OpenCV, phương pháp phát hiện gương mặt bằng MTCNN và phương pháp tìm kiếm gương mặt tương đồng với mô hình KNN.

*Thực hành*: Sử dụng ngôn ngữ lập trình Python, các thư viện hỗ trợ xử lý ảnh, tập

dữ liệu liên quan đến hình ảnh, các thư viện xây dựng mô hình như OpenCV, MTCNN, KNN, Keras và một số thư viện hỗ trợ khác.

### Kết quả đạt được

Áp dụng mô hình tìm kiếm gương mặt KNN được tối ưu quá trong quá trình truy vấn bằng cách tạo index từ Autofaiss. Xây dựng và phát hiện gương mặt bằng MTCNN, kết hợp với thư viện OpenCV để xác định và đọc hình ảnh. Tích hợp mô hình vào website để tạo ra hệ thống.

### Bố cục niên luận

Phần giới thiệu

* + Giới thiệu tổng quát về đề tài

Phần nội dung

* + Chương 1: Mô tả bài toán và cơ sở lý thuyết
  + Chương 2: Thiết kế và cài đặt giải thuật, trình bày các bước xây dựng hệ thống
  + Chương 3: Kiểm thử và đánh giá Phần kết luận
  + Trình bày kết quả đạt được, những hạn chế và hướng phát triển trong tương lai.

# PHẦN NỘI DUNG

## CHƯƠNG 1: MÔ TẢ BÀI TOÁN

### Mô tả chi tiết bài toán

Hiện nay, việc ứng dụng các công nghệ thông minh đang rất phổ biến, đặc biệt là trong lĩnh vực thị giác máy tính. Đối với con người bình thường thì việc tiếp nhận các thông tin từ hình ảnh bằng cách nhìn trực tiếp là điều khá dễ dàng. Nhưng đối với máy tính thì đó là việc khó khăn. Để đồng thời tạo ra hệ thống phù hợp với thời đại công nghệ và mang lại lợi ích cho người dùng thì việc xây dựng hệ thống nhận dạng gương mặt trong hệ thống là rất cần thiết.

Bằng cách áp dụng mô hình KNN được tối ưu hóa về việc truy vấn tìm kiếm bằng Autofaiss để tạo index và cùng với OpenCV để đọc từng hình ảnh cũng như từng khung hình được thu nhận từ việc upload file hoặc camera. Sau đó hệ thống áp dụng các kỹ thuật xử lý ảnh đầu vào như MTCNN để phát hiện vị trí gương mặt nhằm rút trích đặc trưng trên từng ảnh, nâng cao được độ chính xác khi dự đoán hình ảnh xuất hiện.

Xây dựng một trang web để người dùng có thể thực hiện một cách trực tiếp bằng Flask framework kết hợp với Spring framework của ngôn ngữ lập trình Java. Website cho phép người dùng upload một hình ảnh có chứa gương mặt và đầu ra sẽ là thông tin từ người dùng đó như: tên, thông tin chi tiết,...

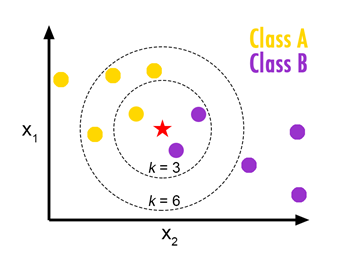
### Cơ sở lý thuyết

### KNN

K-nearest neighbor là một trong những thuật toán supervised-learning đơn giản nhất (mà hiệu quả trong một vài trường hợp) trong Machine Learning. Khi training, thuật toán này không học một điều gì từ dữ liệu training (đây cũng là lý do thuật toán này được xếp vào loại lazy learning), mọi tính toán được thực hiện khi nó cần dự đoán kết quả của dữ liệu mới. K-nearest neighbor có thể áp dụng được vào cả hai loại của bài toán Supervised learning là Classification và Regression. KNN còn được gọi là một thuật toán Instance-based hay Memory-based learning.

KNN (K láng giềng): Một thuật toán đơn giản bao gồm việc tìm kiếm k vectơ tương đồng gần nhất với một vectơ truy vấn dựa trên điểm số được đưa ra bởi một hàm tính độ tương đồng.

.



Hình 1. Mô hình KNN

Hình 1 mô tả sơ đồ của mô hình KNN. Trong KNN, việc tính toán khoảng cách giữa hai mẫu có thể là Euclidian, Manhattan, Minkowski, Chebyshev … Giá trị của k là số quan sát gần nhất với điểm cần dự báo khi lựa chọn giá trị của k.

Thuật toán KNN phụ thuộc nhiều vào việc chọn k, khi k lớn sẽ luôn xác định được lớp cho mẫu dữ liệu mới, nhưng khi k nhỏ thì có thể bị nhiễu. Ví dụ như hình dưới đây, khi k = 3 mẫu mới là hình sao đỏ, trong 3 điểm gần nó có 2 điểm màu tím nên nó sẽ thuộc lớp B màu tím. Khi k = 6, nó lại gần với 4 điểm màu vàng nên nó sẽ thuộc lớp màu vàng A.

**Ưu điểm:**

* Thuật toán đơn giản, dễ dàng sử dụng và cài đặt.
* Dự đoán kết quả của dữ liệu mới khá dễ dàng.
* Độ phức tạp tính toán thấp.
* Có thể dùng cho cả hai loại học có giám sát là phân loại và hồi quy.
* Không có giả định về phân bố dữ liệu

**Nhược điểm:**

* Khi bị nhiễu và k nhỏ thì kết quả đưa ra không chính xác.
* Chi phí tính toán cao và yêu cầu bộ nhớ lớn.
* Cần thời gian lưu training set tăng lên nhiều sẽ mất nhiều thời gian tính toán.
* Thời gian dự đoán có thể chậm do phải tính toán khoảng cách với tất cả các đối tượng trong tập dữ liệu.
* Đôi khi cần chuyển đổi kiểu dữ liệu thành các yếu tố định tính.

### OpenCV

Dự án OpenCV được bắt đầu từ Intel năm 1999 bởi Gary Bradsky[5]. OpenCV viết tắt cho Open Source Computer Vision Library. OpenCV là thư viện mã nguồn mở hàng đầu cho Thị giác máy tính và Học máy, và hiện có thêm tính năng tăng tốc GPU cho các hoạt động theo thời gian thực (real-time).

OpenCV được phát hành theo giấy phép BSD, do đó nó miễn phí cho cả học tập và sử dụng với mục đích thương mại. Nó có trên các giao diện C++, C, Python và Java và hỗ trợ Windows, Linux, Mac OS, iOS và Android. OpenCV được thiết kế để hỗ trợ hiệu quả về tính toán và chuyên dùng cho các ứng dụng real-time (thời gian thực). Nếu được viết trên C/C++ tối ưu, thư viện này có thể tận dụng được bộ xử lý đa lỗi (multi-core processing).

**Ứng dụng: OpenCV được sử dụng cho nhiều mục đích và ứng dụng khác nhau**

* Hình ảnh street view
* Kiểm tra và giám sát tự động
* Robot và xe hơi tự lái
* Phân tích hình ảnh y học
* Tìm kiếm và phục hồi hình ảnh/video
* Phim – cấu trúc 3D từ chuyển động
* Nghệ thuật sắp đặt tương tác **Tính năng và các module:**
* Xử lý và hiển thị hình ảnh/ video/ I/O(core, imgproc, highui)
* Phát hiện các vật thể (object, features2d, nonfree)
* Geometry-based monocular hoặc stereo computer vision (calib3d)
* Computational photography (photo, video, superres)
* Machine learning & clustering (ml, flann)
* CUDA acceleration (gpu)

### Autofaiss

* + 1. **Tổng quan**

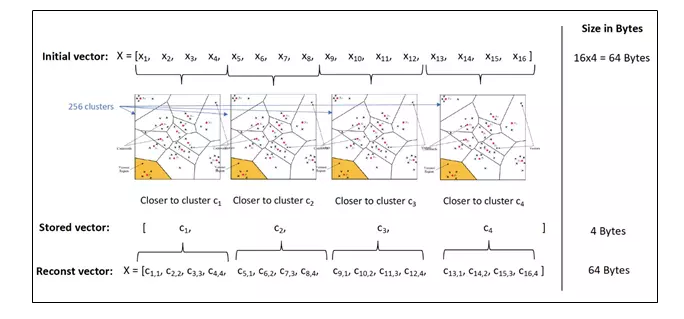
Autofaiss tạo và điều chỉnh các index (chỉ mục) KNN được quantize để tìm kiếm một cách hiệu quả độ tương đồng với mức sử dụng bộ nhớ thấp.

Để tìm kiếm các vector tương đồng nếu so sánh vector truy vấn với toàn bộ tất cả các vector thì chi phí tìm kiếm sẽ vô cùng lớn về thời gian chạy, về không gian bộ nhớ. KNN ước lượng tìm kiếm láng giềng gần nhất để giảm độ phức tạp của thuật toán. Tìm kiếm xấp xỉ láng giềng gần nhất trong một thời gian thích hợp, thì lượng dữ liệu được lưu trữ trong RAM có thể bị cấm và không tương thích với các ứng dụng có thể mở rộng với hàng trăm nghìn vector. Và để giải quyết vấn đề này, Faiss triển khai quantize dựa trên các thuật toán, là các loại thuật toán mới có thể làm giảm đáng kể bộ nhớ RAM đã sử dụng trong khi vẫn cân bằng tốt recall score và thời gian truy vấn.



Hình 2. Quantize Vectors

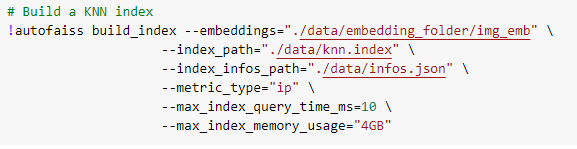
Về mặt kỹ thuật, Vector Quantize (VQ) ban đầu là một kỹ thuật được sử dụng để nén dữ liệu. Nó hoạt động bằng cách chia một tập hợp lớn các điểm (vectơ) thành các nhóm có cùng số điểm gần nhất với chúng. Mỗi nhóm được biểu diễn bằng điểm trung tâm của nó.



Hình 3. Kiến trúc của Quantize Vectors

Quantize: chia không gian vector ban đầu thành nhiều không gian con (sub space) nhỏ hơn. Sau đó, đối với mỗi không gian con này, chúng ta sẽ tìm thấy 256 tâm cụm sẽ đại diện tốt cho dữ liệu bên trong. Mỗi phần của vector con (sub vector) trong không gian con sẽ được liên kết với số tương ứng với cụm gần nhất. Do đó, thay vì lưu trữ một floating-point sub-vector, ta chỉ cần lưu trữ một số nguyên và thủ thuật này làm giảm rất nhiều dung lượng RAM cần thiết để lưu trữ các vector.

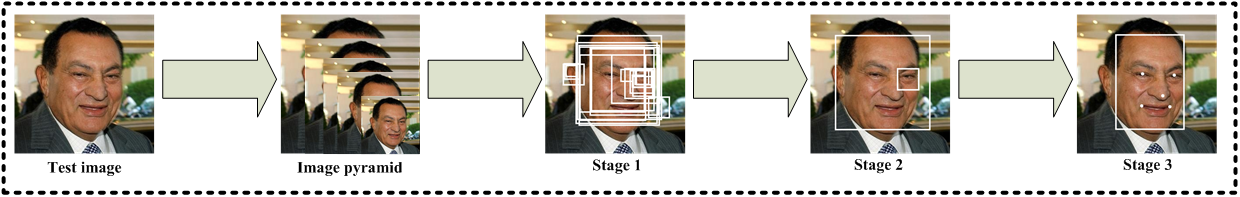
Có hàng trăm tổ hợp thuật toán có thể được sử dụng trong Faiss và mỗi tổ hợp có thể yêu cầu tối đa 6 hyperparameters để xác định cách xây dựng index. Ngoài ra, có những hyperparameters khác cần được thiết lập để tối ưu hóa sự cân bằng giữa recall và thời gian tìm kiếm trên index. Đây là một điều phức tạp và khó khăn. Các nghiên cứu đã được thực hiện để điều chỉnh các thuật toán tạo lập index, mục tiêu tìm ra sự cân bằng tốt nhất giữa tốc độ truy vấn, kích thước index, recall và thời gian xây dựng index. Và Autofaiss được tạo ra, một thư viện tạo lập index, nhằm xây dựng các index tối ưu với tốc độ truy vấn và RAM. Lệnh và parameters điều chỉnh autofaiss:



* embeddings path: đường dẫn đến thư mục img\_emb có chứa file img\_emb\_0.npy gồm các ảnh database được embedding dưới dạng numpy
* index\_path: đường dẫn đến thư mục để lưu ra file knn.index.
* index\_infos\_path: đường dẫn đến thư mục để lưu ra file infos.json chứa thông tin của knn.index.
* metric\_type: độ đo tương đồng có thể là "ip" cho inner product, "l2" cho euclidian distance
* max\_index\_query\_time\_ms: thời gian truy vấn tối đa (ms)
* max\_index\_memory\_usage: kích thước tối đa của file index đã tạo.

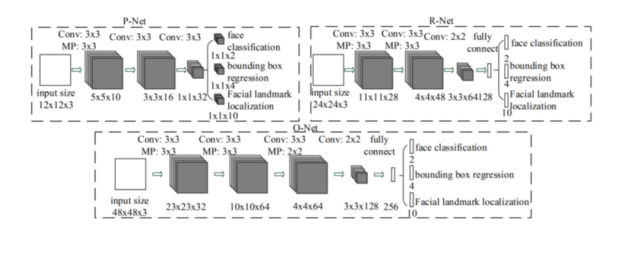
### MTCNN

MTCNN là viết tắt của Multi-task Cascaded Convolutional Networks. Nó là bao gồm 3 mạng CNN xếp chồng và đồng thời hoạt động khi detect khuôn mặt. Mỗi mạng có cấu trúc khác nhau và đảm nhiệm vai trò khác nhau trong task. Đầu ra của MTCNN là vị trí khuôn mặt và các điểm trên mặt như: mắt, mũi, miệng…



Hình 4. Trích xuất đặc trưng MTCNN

MTCNN hoạt động theo 3 bước, mỗi bước có một mạng neural riêng lần lượt là: P-Net, R-Net và O-net. Với mỗi bức ảnh đầu vào, nó sẽ tạo ra nhiều bản sao của hình ảnh đó với các kích thước khác nhau.



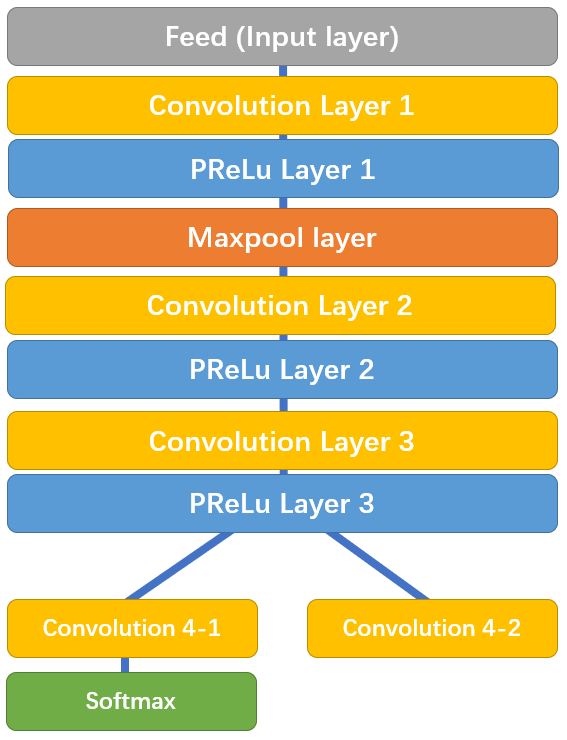
Hình 5. Mô hình MTCNN

### The proposal network (P-Net)

Ở bước đầu tiên sẽ sử dụng mạng FCN (fully convolutional network). Mạng FCN khác mạng CNN ở chỗ mạng FCN không sử dụng lớp Dense layer. P-Net được sử dụng để có được các windows tiềm năng và bounding box regression vectors của chúng (tọa độ).

Bounding box regression là kỹ thuật để dự đoán vị trí của bounding box khi chúng ta cần phát hiện đối tượng (ở đây là khuôn mặt). Sau khi có được tọa độ của bounding boxes một vài tinh chỉnh được thực hiện để loại bỏ một số bounding boxes overlap với nhau. Đầu ra của bước này là tất cả bounding boxes sau khi đã thực hiện sàng lọc.

Tại P-Net, thuật toán sử dụng 1 kernel 12x12 chạy qua mỗi bức hình để tìm kiếm khuôn mặt.



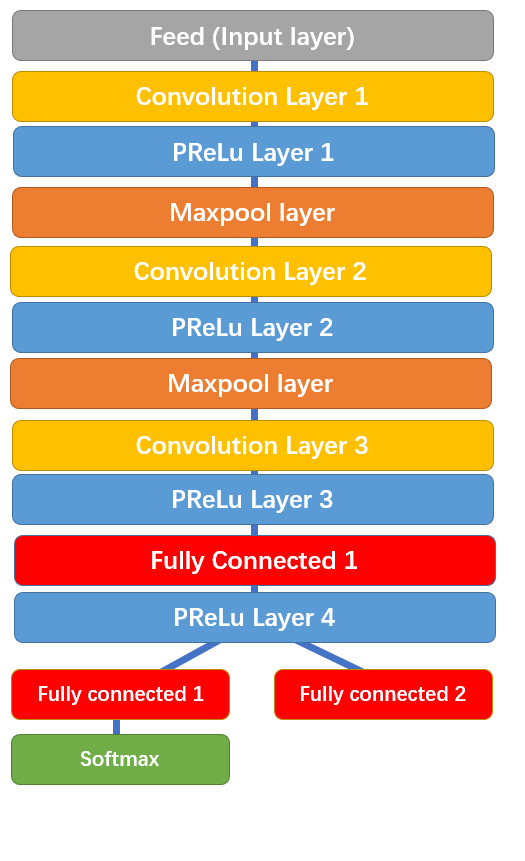
Hình 6. P-Net Layer

Sau lớp convolution thứ 3, mạng chia thành 2 lớp. Convolution 4-1 đưa ra xác suất của một khuôn mặt nằm trong mỗi bounding boxes, và Convolution 4-2 cung cấp tọa độ của các bounding boxes.

### The Refine Network (R-Net)

Tất cả bounding boxes từ P-Net được đưa vào R-Net. R-Net là mạng CNN và giảm số lượng bounding boxes xuống, tinh chỉnh lại tọa độ, có áp dụng Non-max suppression.

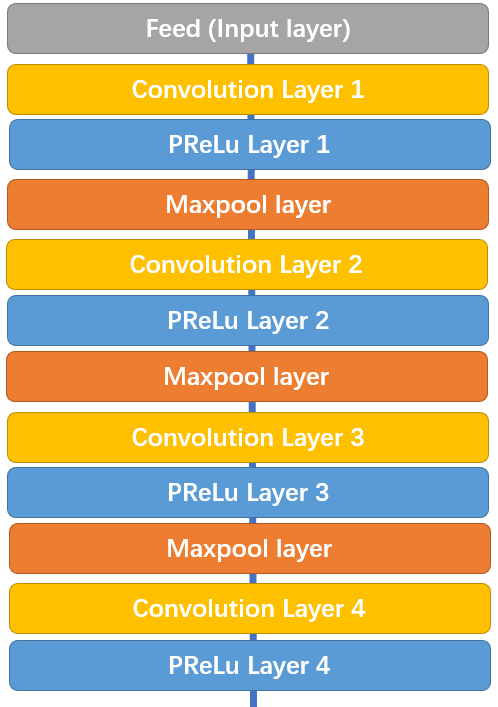
R-Net có cấu trúc tương tự vói P-Net. Tuy nhiên sử dụng nhiều layer hơn.



Hình 7. R-Net Layer

Tại đây, network sẽ sử dụng các bounding boxes được cung cấp từ P-Net và tinh chỉnh lại tọa độ. Tương tự R-Net chia ra làm 2 layers ở bước cuối,cung cấp 2 đầu ra đó là tọa độ mới của các bounding boxes, cùng độ tin cậy của nó.

Đầu ra của R-Net được sử dụng làm đầu vào của O-Net. Trong O-Net có đưa ra vị trí của facial landmarks (2 mắt, mũi, 2 vị trí của miệng). Như hình trên có thể thấy gần cuối chúng ta có 3 nhánh: xác suất xuất hiện khuôn mặt, tọa độ bounding box, tọa độ của facial landmarks (mỗi vị trí có tọa độ x và y, ảnh hưởng đến số lượng units trong layer).



Hình 8. O-Net Layer

O-Net lấy các bounding boxes từ R-Net làm đầu vào và đánh dấu các tọa độ của các mốc trên khuôn mặt. Ở bước này, thuật toán đưa ra 3 kết quả đầu ra khác nhau bao gồm: xác suất của khuôn mặt nằm trong bounding box, tọa độ của bounding box và tọa độ của các mốc trên khuôn mặt (vị trí mắt, mũi, miệng)

### Spring

Spring là framework phát triển ứng dụng phổ biến nhất dành cho Java Enterprise. Ban đầu nó được viết bởi Rod Johnson và lần đầu tiên được phát hành theo giấy phép Apache 2.0 vào tháng 6 năm 2003. Spring có kích thướng nhẹ, phiên bản cơ bản của Spring framework có kích thước khoảng 2MB.

Spring framework là một Java Platform mã nguồn mở, một giải pháp gọn nhẹ dành cho Java Enterprise. Với Spring Framework các nhà phát triển có thể tạo ra các mã có hiệu suất cao, dễ kiểm thử và có thể sử dụng lại được.

Các tính năng core của Spring Framework có thể được sử dụng trong việc phát triển bất kỳ ứng dụng Java nào. Bên cạnh đó, phần mở rộng được sử dụng để xây dựng các ứng dụng web trên nền tảng Java EE. Mục tiêu của Spring Framework là làm cho việc phát triển ứng dụng J2EE dễ dàng hơn và thúc đẩy việc lập trình tốt hơn bằng mô hình POJO-based.

### Flask

Flask là một web frameworks, nó thuộc loại micro-framework được xây dựng bằng ngôn ngữ lập trình Python. Flask cho phép xây dựng các ứng dụng web từ đơn giản đến phức tạp, nó có thể xây dựng các API nhỏ, ứng dụng web như các trang web, blog, wiki hoặc website dựa theo thời gian hay thậm chí là một trang web thương mại. Flask cung cấp công cụ các thư viện và công nghệ hỗ trợ làm những việc trên.

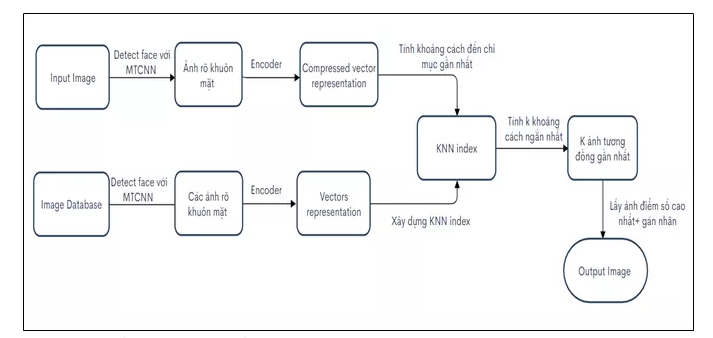
Flask là một micro-framework có nghĩa Flask là một môi trường độc lập, ít sử dụng các thư viện khác bên ngoài. Do vậy, Flask có ưu điểm là nhẹ, có rất ít lỗi cho ít bị phụ thuộc cũng như dễ dàng phát hiện và xử lý các lỗi bảo mật.

## CHƯƠNG 2. THIẾT KẾ VÀ CÀI ĐẶT

### 1. Quy trình phát hiện và nhận dạng gương mặt

Quy trình phát hiện và nhận dạng hình ảnh qua các bước chính được mô tả ở hình

10. Ở đây, sau khi nhận vào ảnh từ người dùng, hệ thống sẽ phát hiện vị trí gương mặt có trong ảnh bằng giải thuật MTCNN. Sau đó hình ảnh được encoder nén thành 1 vector 128 chiều. KNN index sẽ đi tính k khoảng cách ngắn nhất từ vector input và cuối cùng sẽ lấy ảnh có điểm số cao nhất và gán nhãn.



Hình 9. Quy trình phát hiện và nhận dạng gương mặt

Sau đây là mô tả chi tiết cho các bước trên.

### Mô hình phát hiện gương mặt

Giai đoạn này sử dụng mô hình phát hiện gương mặt đã được nghiên cứu trước đó. Các bước thực hiện để sử dụng mô hình được mô tả như sau:

### Mô hình MTCNN

Mô hình MTCNN được import

“from mtcnn import MTCNN”.

### Load mô hình MTCNN vào biến

Sử dụng hàm **MTCNN()** để load model vào bộ nhớ thông qua tên biến. Model này đã được viết sẵn những hàm cần thiết để phát hiện gương mặt (detect face), tìm kiếm gương mặt (search face) và hàm trả về kết quả.

### Chuẩn bị ảnh đầu vào

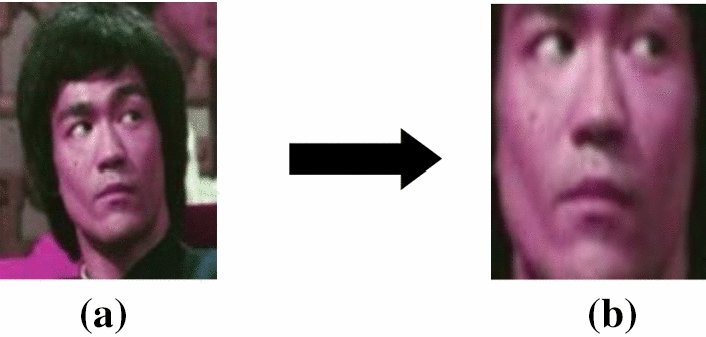
Ảnh đầu vào có phần mở rộng là “png”, “jpeg”, để cung cấp hình ảnh vào mạng bằng hàm **detect\_faces(image).**

* Đối số đầu tiên là hình ảnh đã được convert sang array thông qua hàm **np.asarray(image)**

### Xử lý đầu ra

Kết quả đầu sau khi đi qua mạng MTCNN là trả về một mảng chứa tất cả các vị trí của gương mặt có trong ảnh. Mỗi phần tử trong mảng chứa 4 tham số là (x, y, width, height) đại diện cho một bounding boxes, hình hộp chữ nhật chỉ chứa gương mặt. Cuối cùng ta sẽ cắt hình lại theo đúng vị trí của gương mặt thông qua hàm **crop** của thư viện **Image.**

### Kết quả



Hình 10. Kết quả phát hiện gương mặt

Kết quả ở hình 11 là sau khi sử dụng mô hình *MTCNN* phát hiện được vị trí gương mặt và cắt lại hình ảnh ban đầu theo sát vị trí của gương mặt.

### Tạo KNN index cho tập ảnh dữ liệu

Với phần tạo KNN index này sẽ tiến hành thực hiện trên Google Colab. Đầu tiên là nén tập ảnh dữ liệu và tải lên Google Drive, sau đó kết nối Google Colab và Google Drive.

Sau đó, tạo Image Embedding, để embedding cho hình ảnh AutoFaiss sử dụng Clip Retrieval của Clip OpenAI. Install Clip Retrieval và AutoFaiss và thực hiện embedding, chọn thư mục input là tập ảnh đầu vào, và nơi lưu thư mục embedding\_folder:

“!pip install clip-retrieval autofaiss”

“!clip-retrieval inference --input\_dataset ./data/train/ --output\_folder ./data/embedding\_folder”

Sau khi thực hiện xong, kết quả thu được thư mục embedding\_folder gồm 2 thư mục nhỏ chứa các file:

* img\_emb/img\_emb\_0.npy: chứa hình ảnh nhúng dưới dạng numpy
* metadata/metadata\_0.parquet: chứa đường dẫn hình ảnh

Cuối cùng là xây dựng chỉ mục KNN với AutoFaiss cho tập ảnh, lựa chọn đường dẫn và các thông số phù hợp:

!autofaiss build\_index --embeddings="./data/embedding\_folder/img\_emb" \

--index\_path="./data/knn.index" \

--index\_infos\_path="./data/infos.json" \

--metric\_type="ip" \

--max\_index\_query\_time\_ms=10 \

--max\_index\_memory\_usage="4GB"

--embeddings: là đường dẫn tới file chứa những hình ảnh đã nhúng.

--index-path: đường dẫn nơi sẽ chứa kết quả đầu ra là KNN index.

--index\_infos\_path: nơi sẽ chứa thông tin cấu hình chi tiết.

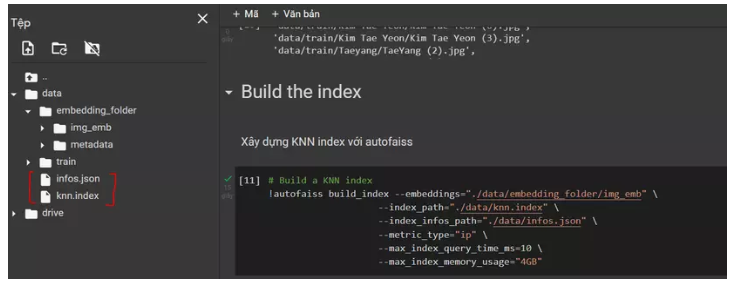
--metric\_type: loại số liệu theo mặc định là “ip”

--max\_index\_query\_time\_ms: thời gian tối đa cho phép truy vấn.

--max\_index\_memory\_usage: bộ nhớ tối đa cho phép sử dụng.

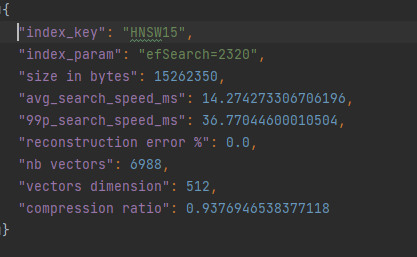
### Kết quả

Ta sẽ được 2 file knn.index và infos.json trong thư mục data.



Hình 11. Kết quả KNN index

Cấu trúc file infos.json chứa thông tin cấu hình và kết quả



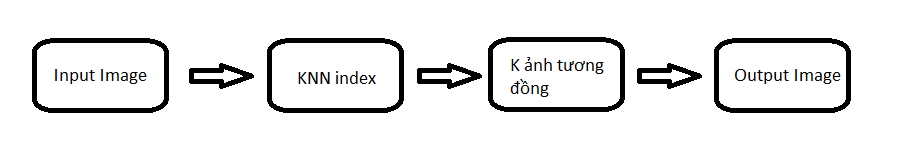
Hình 12. Thông tin file infos.json

Quan sát thông tin file thấy được rằng:

**“vectors dimension”:** 512, đại diện cho số chiều của một hình ảnh.

**“nb vectors”:** 6988, có tất cả 6988 hình ảnh trong tập dữ liệu, mỗi hình ảnh đã được nén lại thành một vector 512 chiều.

### Mô hình nhận dạng



Hình 13. Xây dựng mô hình nhận dạng

Tiến hành download 4 file **metadata\_0.parquet, img\_emb\_0.npy, knn.index** và **infos.json**, tải cả 4 file với mục đích lưu trữ lại kết quả của quá trình build KNN index từ Google Colab, sử dụng đọc 2 file là **metadata\_0.parquet** và **knn.index** chứa chỉ mục KNN ảnh database được tạo bởi AutoFaiss, **image\_list** chứa đường dẫn hình ảnh.

# Read file KNN index

df = pd.read\_parquet(".\data\embedding\_folder\metadata\metadata\_0.parquet")

image\_list = df["image\_path"].tolist()

ind = faiss.read\_index(".\data\knn.index")

Đầu vào là một ảnh có chứa người (có thể 1 hoặc nhiều người), nhận biết những khuôn mặt người đó là ai tương ứng trong cơ sở dữ liệu. Sau khi phát hiện được khuôn mặt đưa khuôn mặt đó vào tìm kiếm trong cơ sở dữ liệu. Đầu tiên là phải **preprocess**, **encode image** và **image embeddings** cho ảnh khuôn mặt truy vấn, sử

dụng model **Clip OpenAI**

# Load the model

device = "cuda" if torch.cuda.is\_available() else "cpu"

model, preprocess = clip.load("ViT-B/32", device=device)

Xây dựng hàm **search\_face()** thực hiện tìm kiếm ảnh khuôn mặt trong database tương đồng với ảnh khuôn mặt truy vấn. Dùng phương thức **search()** của KNN index, tìm ra ảnh tương đồng nhất với ảnh truy vấn. Giá trị D là độ tương đồng của ảnh tìm được với ảnh truy vấn, giá trị I là đường dẫn của ảnh đó trong database. Đặt điều kiện nếu tìm được ảnh kết quả có độ tương đồng lớn hơn 0.8 thì sẽ cho ra kết quả. Trả về kết quả là tên thư mục ảnh (tên người) của ảnh kết quả.

# Search image

def search\_face(image):

image\_tensor = preprocess(image)

image\_features = model.encode\_image(torch.unsqueeze(image\_tensor.to(device), dim=0))

image\_features /= image\_features.norm(dim=-1, keepdim=True)

image\_embeddings = image\_features.cpu().detach().numpy().astype('float32')

D, I = ind.search(image\_embeddings, 1)

if D[0][0] > 0.8:

name = os.path.basename(os.path.dirname(image\_list[I[0][0]]))

print("Name:",os.path.basename(os.path.dirname(image\_list[I[0][0]])))

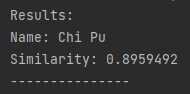
print("path:",image\_list[I[0][0]])

print("Similarity:",D[0][0])

return name

### Dự đoán

Sử dụng hàm search\_face đã xây dựng để dự đoán hình ảnh. Kết quả trả về bao gồm Name (tên người trong ảnh), Similarity (độ tương đồng).

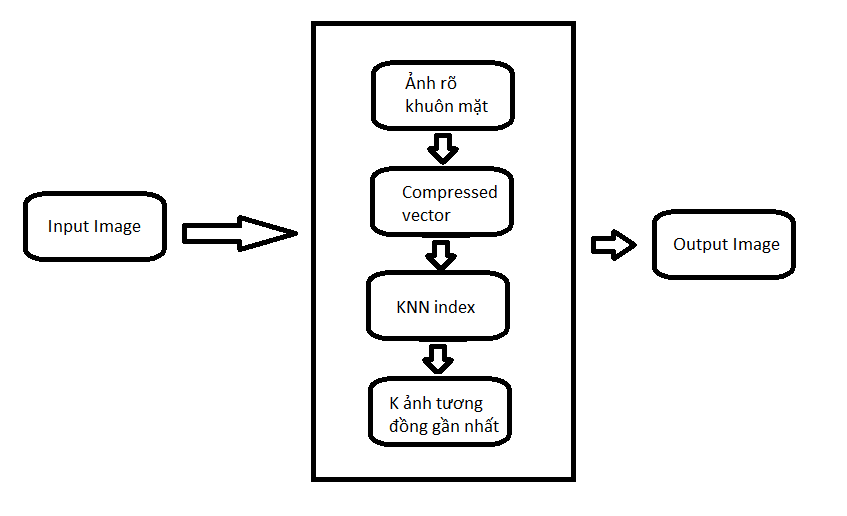


Hình 14. Kết quả mô hình nhận dạng

### Xây dựng website

Sau khi xây dựng được mô hình, giai đoạn cuối là xây dựng hệ thống hoàn chỉnh để tích hợp mô hình này vào.

Các chức năng chính của hệ thống bao gồm: Chọn ảnh đầu vào, tải ảnh để xử lý, và nhận dạng gương mặt có trong ảnh.



Hình 15. Sơ đồ hệ thống

Hình 16 mô tả sơ đồ tổng thể của hệ thống. Đầu vào là ảnh chứa gương mặt do người dùng tải lên. Hệ thống sẽ tiếp nhận hình ảnh và bắt đầu tiến hành xử lý theo tuần tự. Đầu tiên sẽ phát hiện gương mặt có trong hình ảnh. Tiếp đến sẽ xử lý cắt ra tất cả các hình ảnh gương mặt có trong ảnh. Cuối cùng là chuyển tiếp các hình ảnh vào KNN index tính toán khoảng cách giữa hình ảnh đầu vào với hình ảnh có trong cơ sở dữ liệu để tìm ra hình ảnh có độ tương đồng cao nhất. Sau đó xuất ra được tên của đối tượng trong ảnh.

### Thiết lập Flask

Sử dụng trình quản lý gói pip “pip install flask”.

### Trang chủ

Khởi tạo trang chủ “home.html” với chức năng cho phép người dùng nhập đầu vào là một tập tin hình ảnh.

### Trang xử lý

Tạo trang xử lý “form\_find.html”. Trang này sẽ nhận ảnh từ trang chủ và xử lý ảnh với các hàm được thiết lập. Sau khi xử lý xong sẽ trả ra kết quả như yêu cầu.

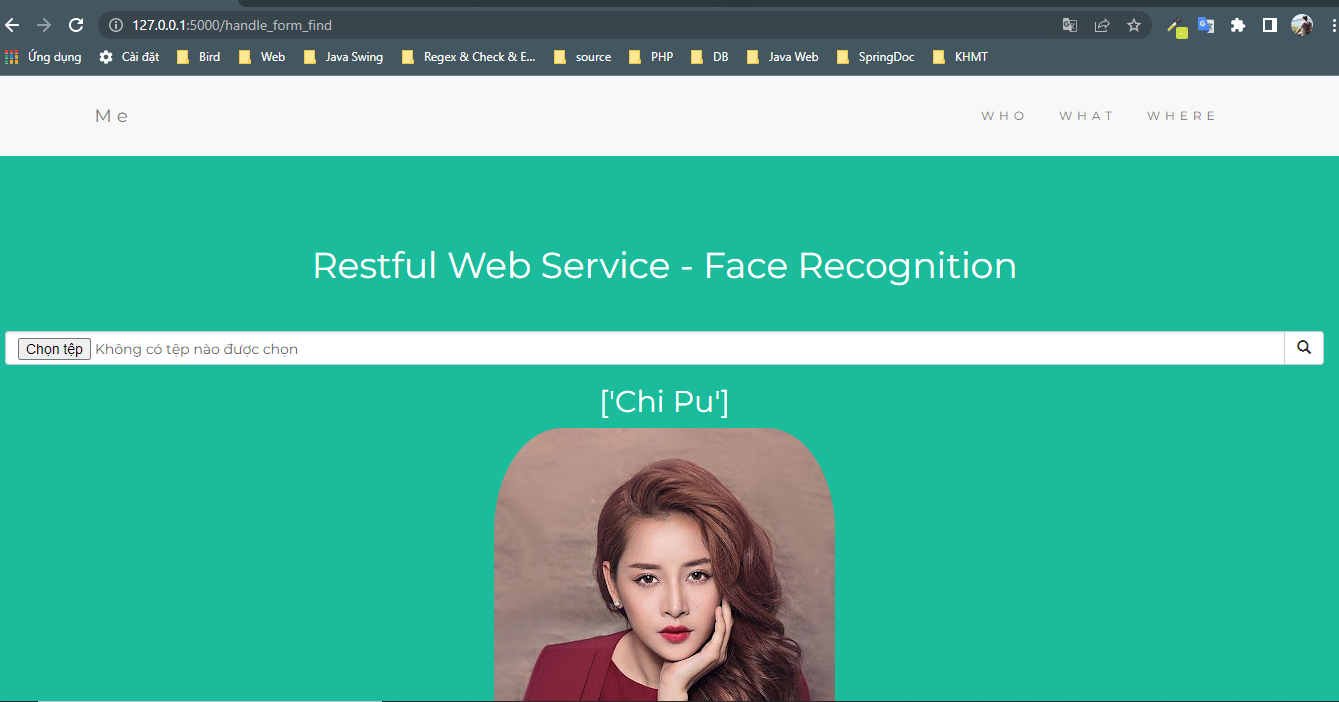
### Các hàm xử lý

* Kiểm tra tệp tin đầu vào: Nếu tệp tin không tải được lên trang chủ, tệp tin rỗng thì sẽ nhận được thông báo. Và tệp tin tải lên thành công sẽ lưu lại với theo tên đường dẫn.
* Hàm phát hiện hình ảnh: Hàm xử lý hình ảnh với mô hình MTCNN để phát hiện tất cả gương mặt có trong hình ảnh. Đọc ảnh đầu vào là đường dẫn tệp tin được lưu ở trên. Sau đó gọi lại hàm phát hiện hình ảnh đã được định nghĩa.
* Hàm xử lý ảnh: Hàm xử lý hình ảnh là bước tiếp theo sử dụng thư viện OpenCV để xử lý ảnh. Lặp lại các bước như đọc đường dẫn, xử lý và lưu lại.
* Hàm thay đổi định dạng: Sử dụng hàm để thay đổi cấu trúc hình dạng của ảnh để phù hợp với mô hình như: convert ảnh sang numpy array.
* Hàm dự đoán: Dự đoán với tất cả các gương mặt phát hiện được. Chuyển về dạng array và sau đó xuất ra màn hình cho người dùng.
* Hàm trả về trang chủ: Sử dụng sự kiện Onclick của JavaScript

### Kết quả đầu ra

Đầu ra của hệ thống là tên của tất cả đối tượng trong hình ảnh kèm theo đó là độ tương đồng của hình ảnh đó với cơ sở dữ liệu.

### Giao diện hệ thống



Hình 16. Giao diện website

* Trang chủ ở hình 17 hiển thị với chức năng cho phép người dùng nhập vào là một tệp tin hình ảnh
* Sau đó người dùng bấm chọn nút **Upload** để đưa ảnh cần xử lý vào hệ thống.

## CHƯƠNG 3. KIỂM THỬ VÀ ĐÁNH GIÁ

### Môi trường kiểm thử

Chương trình được viết và chạy trên cửa sổ dòng lệnh (Command Prompt) Pycharm dưới nền tảng ngôn ngữ lập trình Python. Sau khi hoàn thiện hệ thống được chạy trên trình duyệt web như Chrome, Cốc Cốc, Firefox,… với http://localhost:5000 (cổng 5000).

### Kiểm thử

Quá trình kiểm thử hệ thống theo 2 bước. Kiểm thử mô hình phát hiện gương mặt, kiểm thử hệ thống đầu vào tệp tin hình ảnh và xử lý cho ra ảnh và dự đoán.

### Mô hình phát hiện gương mặt:

Sử dụng OpenCV: Với các bước xử lý ảnh của OpenCV

Sử dụng mô hình phát hiện gương mặt MTCNN: Load mô hình phát hiện gương mặt được nghiên cứu trước đó.

### Đánh giá

### Phát hiện gương mặt

Bảng 1. So sánh kết quả phát hiện gương mặt

|  |  |  |
| --- | --- | --- |
| Mô hình | Số ảnh đầu vào | Kết quả |
| OpenCV | 20 | 6/10 |
| MTCNN | 20 | 18/10 |

Kết quả so sánh của bảng 1 cho thấy:

**OpenCV**: Tách được vị trí của gương mặt chỉ tốt khi ảnh đạt chất lượng cao hoặc đủ điều kiện như ánh sáng và ít bị nhiễu bởi phông nền phía sau. Đối với ảnh phông nền quá phức tạp thì hầu như đưa ra kết quả phát hiện sai.

**MTCNN**: Mô hình có thể phát hiện gương mặt với nhiều hình nền, phông chữ, định hướng, kích thước, màu sắc khác nhau. Mô hình cũng có thể hoạt động khá tốt đối với hình ảnh bị biến dạng. Tuy nhiên vẫn có hạn chế là cắt các khung hình chứa gương mặt bị dư các phần không phải gương mặt.

## PHẦN KẾT LUẬN

### Kết quả đạt được

Đề tài “Chứng thực bằng gương mặt kết nối vào các hệ thống elearning” đã xây dựng thành công và đạt được những kết quả sau:

* Áp dụng mô hình phát hiện gương mặt vào hệ thống thành công. Cắt ra được các gương mặt có trong ảnh.
* Mô hình nhận dạng gương mặt được xây dựng với kỹ thuật máy học.
* Hệ thống xử lý tốt với các hàm phát hiện, tốc độ xử lý tùy thuộc vào độ phức tạp của ảnh đầu vào.
* Xây dựng thành công API theo Restful Web Service nhận vào là tệp hình ảnh và kết quả là chuỗi Json để những hệ thống elearning dễ dàng sử dụng (đường dẫn API: <https://localhost:5000/api/v1>).
* Tích hợp thành công giữa 2 công nghệ xây dựng Web là Flask Python và Spring Framework Java. Sử dụng Spring để kết nối với cơ sở dữ liệu và xây dựng API để trả về dữ liệu. Sử dụng Flask để tạo giao diện cho người dùng đồng thời tạo chức năng chính là nhận dạng gương mặt có trong ảnh.

### Ưu và nhược điểm

### Ưu điểm

* + - Mô hình nhận dạng đơn giản, nhỏ gọn, độ chính xác khá cao.
    - Dễ dàng thay đổi mô hình để cải thiện độ chính xác.
    - Giao diện hiển thị đủ các chức năng tải và xử lý ảnh.

### Nhược điểm

* Xử lý ảnh để nhận dạng gương mặt còn chưa được tốt. Cắt dễ bị thiếu gương mặt nên việc nhận dạng sẽ dẫn đến kết quả sai, và cắt dư nên sẽ nhận ra các kết quả lạ.
* Hệ thống chỉ xử lý từng ảnh một chưa thể xử lý được nhiều ảnh cùng lúc.

### Hướng phát triển

Tăng thêm số lượng tập dữ liệu hình ảnh. Tích hợp vào các hệ thống elearning, mạng xã hội, thương mại điện tử,...

## TÀI LIỆU THAM KHẢO

1. Christopher Pramerdorfer and M. Kampel, *Facial Expression Recognition using Convolutional Neural Networks: State of the Art*, 2016.
2. Kaipeng Zhang et al., "*Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks*", IEEE Signal Processing Letters, vol. 23, no. 99, pp. 1499-1503, 2016.
3. Gates, G.: *The Reduced Nearest Neighbour Rule. IEEE Transactions on Information Theory 18, 431–433 (1972).*
4. Alexander Mordvintsev & Abid K, “*OpenCV-Python Tutorials Documentation”*, 2017.
5. Miguel Grinberg, “*Flask Web Development”,* 2018.
6. Saad ALBAWI, Tareq Abed MOHAMMED, Saad AL-ZAWI, *“Understanding of a Convolutional Neural Network”*, 2017.