**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

KHOA CÔNG NGHỆ THÔNG TIN

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

A picture containing screenshot, text, graphics, logo

Description automatically generated

**BÁO CÁO THỰC NGHIỆM HỌC PHẦN**

**XỬ LÝ ẢNH SỐ VÀ THỊ GIÁC MÁY TÍNH**

**CHỦ ĐỀ**

**TÌM HIỂU, CÀI ĐẶT, ỨNG DỤNG**

**CỦA PHƯƠNG PHÁP TRÍCH CHỌN ĐẶC TRƯNG LBP**

|  |  |
| --- | --- |
| Ngô Thế Tài | 2020601029 |
| Nguyễn Văn Vũ | 2020601029 |
| Đỗ Trung Hiếu | 2020601029 |
| Trần Thị Khánh Linh | 2020601029 |

GVHD: ThS. Lê Thị Thuỷ

Sinh viên:

Lớp: 20222IT6072002 – Khoá 15

**Hà Nội - Năm 2023**

**MỤC LỤC**

[CHƯƠNG 1. TÌM HIỂU PHƯƠNG PHÁP TRÍCH CHỌN ĐẶC TRƯNG LOCAL BINARY PATTERN 5](#_Toc137113859)

[**1.1.** **Giới thiệu** 5](#_Toc137113860)

[**1.2.** **Local binary pattern** 6](#_Toc137113861)

[**1.3.** **Local binary pattern mở rộng** 8](#_Toc137113862)

[**1.4.** **Local binary pattern với phép xoay** 9](#_Toc137113863)

[**1.5.** **Mẫu thống nhất (Uniform pattern)** 10](#_Toc137113864)

[**1.6.** **Tìm hiểu hệ màu trong xử lý ảnh** 12](#_Toc137113865)

[**1.6.1.** **Hệ màu RGB** 12](#_Toc137113866)

[**1.6.2.** **Hệ màu HSV** 12](#_Toc137113867)

[**1.6.3.** **Ảnh xám và ảnh nhị phân** 13](#_Toc137113868)

[CHƯƠNG 2. CHƯƠNG TRÌNH 14](#_Toc137113869)

[**2.1.** **Lớp BaseImage đọc hình ảnh** 14](#_Toc137113870)

[**2.2.** **Lớp LBP** 16](#_Toc137113871)

[**2.3.** **Lớp FaceRecognition** 20](#_Toc137113872)

[**2.4.** **Ứng dụng Local binary pattern vào tìm kiếm hình ảnh tương đồng** 23](#_Toc137113873)

[**2.5.** **Kết quả thực nghiệm** 27](#_Toc137113874)

[**2.6.** **Kết luận** 27](#_Toc137113875)

[TÀI LIỆU THAM KHẢO 28](#_Toc137113876)

MỤC LỤC HÌNH ẢNH

[Hình 1. Dãy nhị phân và giá trị mới theo chiều kim đồng hồ từ vị trí trái trên cùng 7](#_Toc136903252)

[Hình 2. Ví dụ về cách lấy mẫu 9](#_Toc136903253)

[Hình 3. Xoay điểm ảnh theo chiều kim đồng hồ 10](#_Toc136903254)

[Hình 4. Một số ví dụ về mẫu đồng nhất và không phải mẫu đồng nhất 11](#_Toc136903255)

[Hình 5. Hệ màu RGB 12](#_Toc136903256)

[Hình 6. Phân bố kênh màu H trong HSV 13](#_Toc136903257)

[Hình 7. Khởi tạo lớp BaseImage đọc hình ảnh 14](#_Toc136903258)

[Hình 8. Hàm đọc ảnh và trả về danh sách ảnh, đường dẫn 14](#_Toc136903259)

[Hình 9. Lưu lại kết quả tại hàm \_\_init\_\_ sau khi thực hiện đọc ảnh 14](#_Toc136903260)

[Hình 10. Ví dụ về đường dẫn ảnh 15](#_Toc136903261)

[Hình 11. Ví dụ khởi tạo hàm BaseImage 15](#_Toc136903262)

[Hình 12. Các hàm get, set để thực hiện gán và lấy giá trị 16](#_Toc136903263)

[Hình 13. Khởi tạo lớp LBP 17](#_Toc136903264)

[Hình 14. Tạo hàm ánh xạ từ số nhị phân sang khoảng giá trị từ 1 đến 58 17](#_Toc136903265)

[Hình 15. Hàm chuyển ảnh xám thành LBP 18](#_Toc136903266)

[Hình 16. Hàm tính Histogram của một hình ảnh 18](#_Toc136903267)

[Hình 17. Hàm trích xuất đặc trưng 19](#_Toc136903268)

[Hình 18. Câu lệnh thực hiện các chức năng của LBP 19](#_Toc136903269)

[Hình 19. Lớp FaceRecognition - Nhận diện khuôn mặt 20](#_Toc136903270)

[Hình 20. Trích xuất đặc trưng cho ảnh có khuôn mặt 21](#_Toc136903271)

[Hình 21. Trích xuất đặc trưng cho ảnh không có khuôn mặt 21](#_Toc136903272)

[Hình 22. Chương trình lưu đặc trưng vào file JSON 22](#_Toc136903273)

[Hình 23. Thuật toán K - nearest neighbors và so sánh nhãn 23](#_Toc136903274)

[Hình 24. Quá trình huấn luyện 24](#_Toc136903275)

[Hình 25. Quá trình thử nghiệm 25](#_Toc136903276)

[Hình 26. Cấu trúc thư mục trong dự án 26](#_Toc136903277)

[Hình 27. Ví dụ về tệp cấu hình huấn luyện 26](#_Toc136903278)

[Hình 28. Ví dụ về chương trình trong file main.py 27](#_Toc136903279)

# CHƯƠNG 1. TÌM HIỂU PHƯƠNG PHÁP TRÍCH CHỌN ĐẶC TRƯNG LOCAL BINARY PATTERN

* 1. **Giới thiệu**

Phân tích cấu trúc hai chiều có nhiều ứng dụng tiềm năng, ví dụ ứng dụng cảm biến từ xa, kiểm tra bề mặt, phân tích hình ảnh trong ngành Y, … Nhưng có một số ví dụ hạn chế về việc triển khai các kết cấu đặc trưng của đối tượng. Một vấn đề xảy ra đó là kết cấu trong thế giới thực thường không đồng nhất, do sự thay đổi về hướng, tỉ lệ hoặc hình thức trực quan khác. Sự bất biến của ảnh xám thường rất quan trọng do độ sáng không đồng đều hoặc độ biến thiên của ảnh đó. Ngoài ra độ phức tạp tính toán của hầu hết các thuật toán được cho là quá cao.

Hầu hết các cách tiếp cận để phân loại các kết cấu đều giả định, rằng các mẫu chưa được biết được phân loại giống hệt với các mẫu đào tạo về quy mô không gian, hướng và các thuộc tính trong thang độ xám. Tuy nhiên, các kết cấu trong thế giới thực có thể xảy ra ở độ phân giải và góc quay không gian tuỳ ý và chúng có thể chịu các điều kiện chiếu sáng khác nhau.

Local Binary Patterns (LBP – Mẫu nhị phân cục bộ) là một bộ mô tả nhằm mục đích tóm tắt hiệu quả các cấu trúc cục bộ của hình ảnh. Trong những năm gần đây, sự quan tâm ngày càng tăng trong nhiều lĩnh vực xử lý hình ảnh và thị giác máy tính và đã cho thấy tính hiệu quả của nó trong một số ứng dụng, đặc biệt là phần tích hình ảnh khuôn mặt, bao gồm các nhiệm vụ như phát hiện khuôn mặt, nhận dạng khuôn mặt, phân tích nét mặt, phân loại nhân khẩu học, …

Trong báo cáo này nhóm chúng em trình bày về phương pháp phân tích hình ảnh khuôn mặt dựa trên LBP và ứng dụng vào một số bài toán điển hình của LBP.

* 1. **Local binary pattern**

Local Binary Pattern là một phương pháp rút trích đặc trưng và các đặc trưng sẽ được chọn lọc và thu về một vector đặc trưng. Khi xử lý ảnh, một điều cần thiết đó là một đối tượng bất biến với phép chiếu sáng và phép xoay. Do đó LBP được phát minh để giải quyết hai vấn đề

***Phép chiếu sáng***

Phép chiếu sáng được sử dụng để tăng cường đặc trưng của của các điểm ảnh trong quá trình tính toán.

***Phép xoay***

Tức là khi xoay hình ảnh theo một góc xác định, ví dụ quay hình với góc 90 độ

Vào năm 1996, Ojala đã đề xuất phương pháp LBP ban đầu của LBP chỉ bất biến với phép chiếu sáng. Phiên bản đầu tiên sử dụng một ma trận 3x3 trượt qua từng điểm ảnh, lấy 8 điểm xung quanh và so sánh với điểm trung tâm. Nếu điểm nào lớn hơn hoặc bằng điểm trung tâm thì đặt giá trị mới là 1 ngược lại thành 0. Sau khi tính toán, có thể bắt đầu từ vị trí bất kì tại 8 điểm xung quanh và có thể đi ngược kim đồng hồ và xuôi kim đồng hồ để kết hợp thành mã nhị phân, chuyển chúng thành số thập phân để lưu lại kết quả mới.

A picture containing text, screenshot, diagram, font

Description automatically generated

Hình 1. Dãy nhị phân và giá trị mới theo chiều kim đồng hồ từ vị trí trái trên cùng

Trong bức ảnh, lấy ra 9 pixel ứng với ma trận 3x3. Với những giá trị lớn hơn hoặc bằng giá trị trung tâm thì sẽ đặt là 1 ngược lại là 0. Xây dựng Binary Pattern – LBP theo hướng kim đồng hồ. Giá trị mới được tính như sau:

Nói cách khách, với một vị trí (), LBP được định nghĩa là một tập hợp các so sánh nhị phân có thứ tự về cường độ pixel giữa pixel trung tâm và các pixel xung quanh. Giá trị thập phân kết quả thu được từ 8-bit có thể được biểu diễn như sau:

Trong đó tương ứng với giá trị xám của pixel vị trí (), là các giá trị xám của 8 ô xung quanh, hàm s(k) được định nghĩa như sau:

***Ưu điểm***

* Tính toán đơn giản và nhanh chóng
* Khả năng khái quát hoá cao, có thể được áp dụng cho hầu hết các loại hình ảnh, bao gồm ảnh xám và ảnh màu, trích xuất đặc trưng quan trọng của hình ảnh như cạnh, góc, màu sắc, cấu trúc.
* Khả năng chống nhiễu tốt, do phụ thuộc vào các giá trị nhị phân của các điểm lân cận, các nhiễu nhỏ không ảnh hưởng nhiều đến giá trị pixel.
* Độ tin cậy cao, với phiên bản đầu tiên của LBP xét 8 điểm xung quanh thì thường cho kết quả rất tốt trong các nhiệm vụ nhận dạng và phân loại hình ảnh, đặc biệt trong những bộ dữ liệu lớn.

***Nhược điểm***

* Không phù hợp với một số loại hình ảnh cụ thể, ví dụ các hình ảnh có độ phân giải thấp, việc xử dụng LBP phiên bản đầu tiên có thể dẫn đến xự mất mát thông tin hoặc giảm độ chính xác của quá trình trích xuất đặc trưng.
* Biểu diễn cho các đặc trưng thì kích thước quá lớn, đặc trưng kém ổn định
  1. **Local binary pattern mở rộng**

Từ các nhược điểm ở trên Ojala đã đưa ra một phương pháp để cải tiến thuật toán này. Trong một bức hình, tại một điểm ảnh (), thay vì lấy 8 điểm lân cận ở phương pháp cũ, tác giả đã lấy tập hợp P điểm tạo thành hình tròn có bán kính R, và T là một phân phối đại số của P+1 điểm ảnh đang xét. (Tính cả điểm trung tâm).

Trong đó, g\_c và g\_0, g\_1, … ứng với cấp độ xám của điểm ảnh trung tâm và các điểm lân cận. Cách lấy mẫu được ví dụ như hình 2

A picture containing diagram, line

Description automatically generated

Hình 2. Ví dụ về cách lấy mẫu

Khi tăng bán kính R thì mối tương quan giữa các điểm ảnh giảm dần, cho nên thường sẽ thực hiện trên R nhỏ cụ thể là 1 hoặc 2. Thực hiện tính toán trên mỗi điểm ảnh không làm thay đổi khoảng cách giữa các điểm ảnh. Do đó có thể trừ đi một lượng . Thay thế lượng thành hàm s(k) đã được đề cập ở trên sẽ loại bỏ được việc cùng tăng, cùng giảm độ sáng.

Sau đó tính toán các giá trị biểu diễn với là trọng số tương ứng với công thức

***Ưu điểm***

Bất biến với phép chiếu sáng. Có chi phí tính toán thấp, ổn định khi cường độ thay đổi đơn điệu và dễ mở rộng lên không gian nhiều chiều như RGB, HSV.

***Nhược điểm***

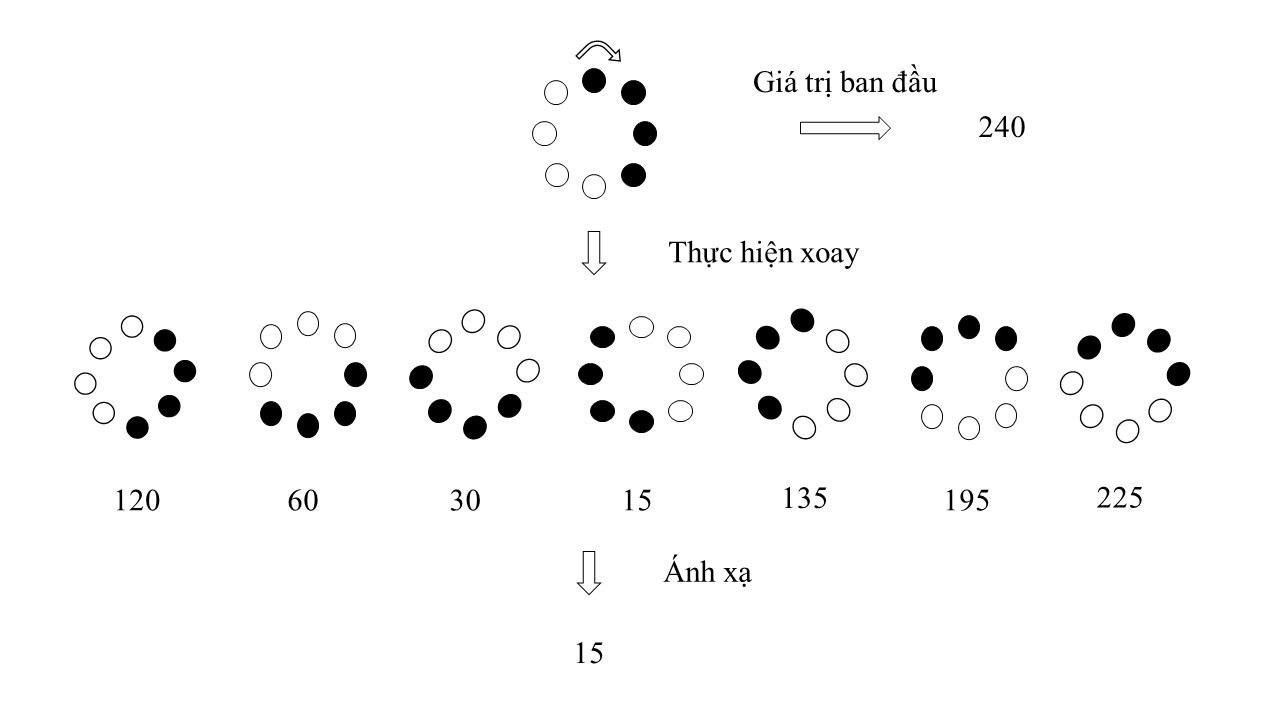
Nếu số điểm lân cận được xét quá lớn, giá trị đặc trưng cho kết cấu sẽ tăng quá cao. Không rõ ràng vì hai biểu diễn cục bộ giống nhau có thể nằm ở hai đối tượng khác nhau

* 1. **Local binary pattern với phép xoay**

Ngoài việc bất biến với phép chiếu sáng như đã được đề cập ở phần trước thì phương pháp này cần phải bất biến với phép xuay. Kí hiệu là

Với là viết tắt của rotation invatiant (Phép quay không đổi, hoặc bất biến với phép quay)

Để bất biến với phép xoay, gọi là ảnh được xoay một góc . Sau khi xoay tại vị trí sẽ được chuyển đổi, ánh xạ sang .



Hình 3. Xoay điểm ảnh theo chiều kim đồng hồ

Sau khi được các kết quả trên, lấy kết quả nhỏ nhất đại diện cho tất cả phép quay. Do đó công thức bất biến của phép chiếu sáng và phép xoay là:

Trong đó là phép dịch vòng tròn số nhị phân lần. Phương pháp này kèm theo bất biến với phép xoay nên có nhiều ứng dụng hơn với Local binary pattern mở rộng.

* 1. **Mẫu thống nhất (Uniform pattern)**

Một bộ LBP được gọi là thống nhất nếu mẫu nhị phân hình tròn theo chiều kim đồng hồ có chứa nhiều nhất 2 lần chuyển đổi bit từ 0 sang 1 và ngược lại.

A picture containing text, screenshot, number, font

Description automatically generated

Hình 4. Một số ví dụ về mẫu đồng nhất và không phải mẫu đồng nhất

Mỗi mẫu trong số này có một bin – (Đại diện cho khoảng dữ liệu, hoặc mức xám trong biểu đồ Histogram). Các mẫu còn lại với hơn 2 lần chuyển đổi bit sẽ được tính vào một bin riêng.

Dưới đây là công thức để kiểm tra mẫu nhị phân đó co phải là mẫu nhị phân nhỏ hơn hoặc bằng 2 lần chuyển đổi bit hay không

Khi thì mẫu nhị phân đó là UNIFORM. Các mẫu còn lại được tính chung giá trị chuyển thành 0. Như vậy công thức cuối cùng của UNIFORM PATTERN là:

Trong đó f(x) là ánh xạ thành các giá trị từ 1 đến 58. Như vậy với việc biểu diễn như này thì sẽ có 59 mẫu, giảm từ 255 mẫu. Việc sử dụng UNIFORM PATTERN (Mẫu nhị phân thống nhất) giúp giảm số lượng đặc trưng và cải thiện tính ổn định của các đặc trưng được tạo ra trong quá trình nhận dạng. Phương pháp này chỉ có tối đa 2 lần chuyển đổi bit nên ít bị ảnh hưởng bởi nhiễu và thay đổi độ sáng. Do đó việc sử dụng UNIFORM PATTERN có thể giúp tăng độ chính xác của quá trình nhận dạng, giảm thiểu các sai sót không mong muốn.

* 1. **Tìm hiểu hệ màu trong xử lý ảnh**

Không gian màu (Colour Space) được hiểu là các mô hình toán để miêu tả màu sắc. Mỗi không gian màu đều có một tác dụng và ứng dụng trong các bài toán khác nhau.

***Các hệ màu cơ bản***

* + 1. **Hệ màu RGB**

A picture containing colorfulness, circle, graphics, creativity

Description automatically generated

Hình 5. Hệ màu RGB

RGB là không gian màu phổ biến dùng trong máy tính, máy ảnh, điện thoại, và nhiều thiết bị kĩ thuật số khác nhau. Không gian này khá gần với cách mắt người tổng hợp màu sắc. Nguyên lý cơ bản sử dụng 3 màu sắc cơ bản R(red – đỏ), G(green – xanh lục), B(blue – xanh lam) để biểu diễn tất cả màu xắc.

Hệ màu RGB sử dụng 8bit để biểu diễn cho mỗi kênh màu, tức là giá trị R, G, B nằm trong khoảng 0 đến 255. Bộ 3 số này biểu diễn cho từng điểm ảnh, mỗi số biểu diễn cho cường độ của một màu. Với mô hình này thì tối đa có thể tạo ra 255x255x255=16581375 màu. Với thư viện matplotlib, Pillow của Python sẽ đọc ảnh đầu vào là RGB. Tuy nhiên với Opencv thì sẽ đọc ảnh ở dạng BGR.

* + 1. **Hệ màu HSV**

Không gian màu HSV là một cách để mô tả màu sắc dựa trên 3 số liệu:

* H: (Hue) vùng màu

Hue là phần màu của mô hình màu và được biểu thị dưới dạng một số từ 0 đến 360 độ:

A picture containing text, screenshot, font, number

Description automatically generated

Hình 6. Phân bố kênh màu H trong HSV

* S: (Saturation) Độ bão hoà màu

Độ bão hoà là lượng màu xám trong màu, từ 0 đến 100%. Một hiệu ứng mờ nhạt có thể có được từ việc giảm độ bão hoà về 0 để thể hiện nhiều màu xám hơn.

* V: (Value) Độ sáng hoặc giá trị

Giá trị kết hợp với độ bão hoà và mô tả sáng hoặc cường độ màu sắc, từ 0 đến 100%. Trong đó 0% là hoàn toàn đen, 100% là sáng nhất.

* + 1. **Ảnh xám và ảnh nhị phân**

Ảnh xám (GrayScale) đơn giản là một hình ảnh trong đó các màu sắc là các sắc thái của màu xám. Ảnh xám cung cấp ít thông tin hơn cho mỗi pixel trong khi các hệ màu khác thường là 3 kênh màu thì ảnh xám chỉ có 1.

Ảnh nhị phân (Binary) là kết quả của việc giảm thông tin của ảnh xám về 2 giá trị 0 hoặc 255. Tuy thông tin bị giảm khá nhiều nhưng ảnh nhị phân lại thể hiện được rất rõ các yếu tố về góc cạnh và hình dạng của vật thể, điều này có lợi trong những bài toán như lọc nhiễu, nhận dạng vật thể, …

# CHƯƠNG 2. CHƯƠNG TRÌNH

* 1. **Lớp BaseImage đọc hình ảnh**

A picture containing text, screenshot, font, line

Description automatically generated

Hình 7. Khởi tạo lớp BaseImage đọc hình ảnh

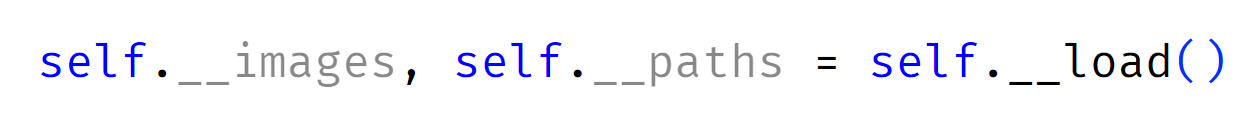
Lớp BaseImage khởi tạo gồm 3 tham số đầu vào:

* root: đường dẫn đến thư mục chứa các ảnh, mặc định là “data/”
* types: kiểu đuôi file có thể là “.png”, “.jpeg”, nếu đọc hết thì sẽ là “\*”
* size: kích cỡ của toàn bộ ảnh khi đọc vào mặc định là (128, 128)

A picture containing text, screenshot, font, line

Description automatically generated

Hình 8. Hàm đọc ảnh và trả về danh sách ảnh, đường dẫn



Hình 9. Lưu lại kết quả tại hàm \_\_init\_\_ sau khi thực hiện đọc ảnh

Hàm \_\_load thực hiện đọc ảnh từ đường dẫn ảnh đã được khởi tạo ở trên. Sử dụng thư viện glob để liệt kê tất cả các file có trong đường dẫn self.\_\_root, có đuôi file là self.\_\_types. Thực hiện đọc ảnh bằng thư viện opencv-python và chỉnh lại kích thước về kích thước đã được định nghĩa sẵn ở trên. Cuối cùng hàm trả về 2 giá trị đó là \_\_images và \_\_paths lần lượt là 1 danh sách ảnh được đọc ở hệ màu BGR và 1 danh sách đường dẫn của mỗi ảnh.

***Ví dụ cụ thể***

A picture containing text, font, line, diagram

Description automatically generated

Hình 10. Ví dụ về đường dẫn ảnh

Ví dụ trên trong thư mục data có 3 loại ảnh là png, bmp và jpg. Khởi tạo hàm BaseImage như sau

A picture containing text, font, line, screenshot

Description automatically generated

Hình 11. Ví dụ khởi tạo hàm BaseImage

Lớp BaseImage nhận vào thư mục root là “data/”, gồm 3 kiểu đuôi file là png, bmp và jpg. Cuối cùng là kích cỡ ảnh (224, 224). Sau đó, hàm \_\_load được gọi và kích thước của 2 giá trị trong hình 7 như sau

* self.\_\_images: kích cỡ là (3, 224, 224, 3). Do có 3 ảnh và kích thước là 224x224 với 3 kênh màu B, G, R
* self.\_\_paths: Kích cỡ là (3, ) do có 3 ảnh và với mỗi giá trị là đường dẫn của mỗi ảnh: ['data/obj.png' 'data/obj1.bmp' 'data/test.jpg']

A screenshot of a computer code

Description automatically generated with low confidence

Hình 12. Các hàm get, set để thực hiện gán và lấy giá trị

* 1. **Lớp LBP**

**A screen shot of a computer code

Description automatically generated with low confidence**

Hình 13. Khởi tạo lớp LBP

Lớp LBP kế thừa từ lớp BaseImage

* root: đường dẫn ảnh, mặc định “data/”
* types: đuôi file có thể là “\*.png”, … Mặc định là “\*”
* size: kích cỡ của ảnh khi đọc vào
* is\_json: có thể đọc ảnh từ file json, nếu đọc ảnh từ file json thì giá trị này bằng True và root để rỗng. ngược lại là False
* json\_data: đường dẫn file json chứa dữ liệu ảnh
* json\_path: đường dẫn file chứa các đường dẫn ảnh
* save: Nếu giá trị bằng True thì hàm sẽ thực hiện lưu lại giá trị đặc trưng của ảnh vào file json

**A picture containing text, screenshot, font, number

Description automatically generated**

Hình 14. Tạo hàm ánh xạ từ số nhị phân sang khoảng giá trị từ 1 đến 58

Hàm \_\_genBIT trên tạo một ánh xạ kiểm tra tất cả các số nhị phân nằm trong khoảng 0 đến 255. Kiểm tra xem số nhị phân đó có phải là Uniform pattern hay không. Nếu đúng thì sẽ lưu lại vào một dictionary (từ điển) để có thể dễ dàng truy cập khi sử dụng hàm LBP.

A picture containing text, screenshot, font, number

Description automatically generated

Hình 15. Hàm chuyển ảnh xám thành LBP

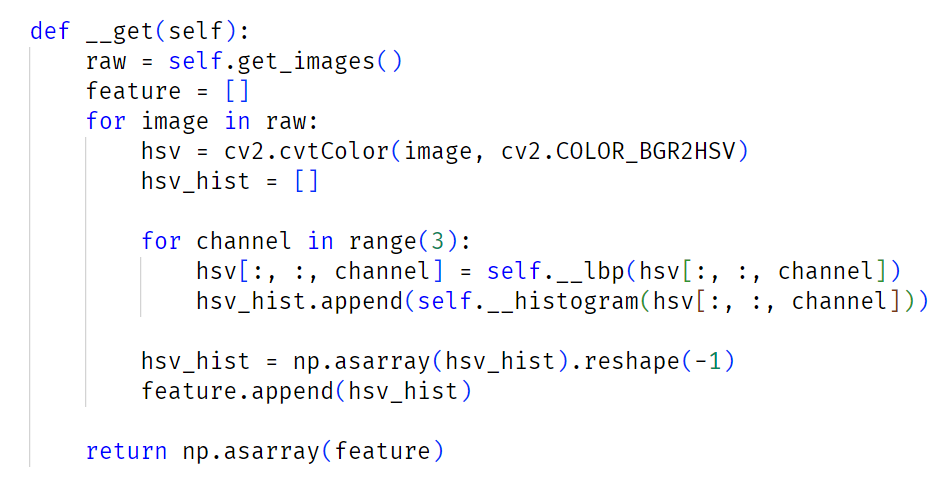
Hàm LBP trên sử dụng phương pháp Uniform pattern để chuyển đổi một ảnh xám thành LBP gồm 59 giá trị khác nhau.

A picture containing text, font, screenshot, line

Description automatically generated

Hình 16. Hàm tính Histogram của một hình ảnh

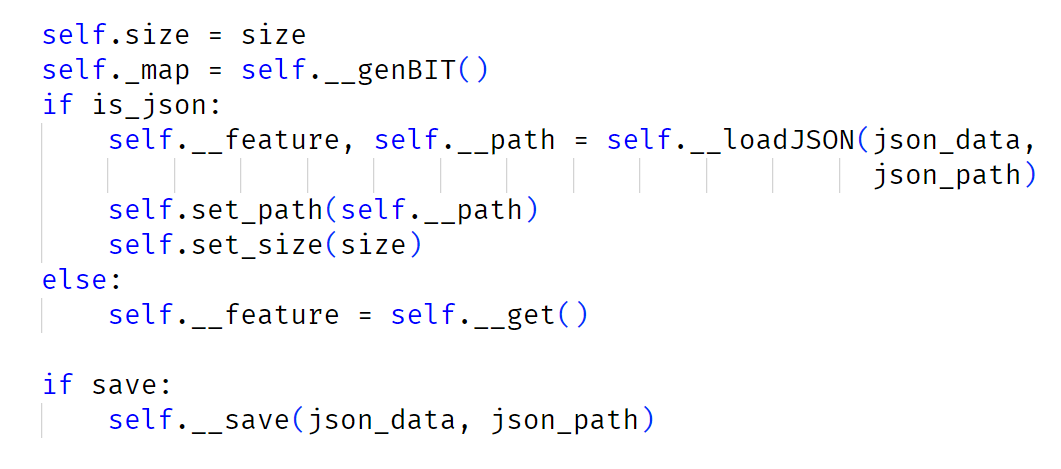
Do tính toán bằng Uniform pattern nên giá trị nằm trong khoảng 0 đến 58. Vì vậy giá trị bins ở hàm np.histogram là 59. Nếu đặt là 256 như các ảnh RGB, BGR, thì sẽ bị thừa. Kết hợp các hàm ở trên để tạo thành một hàm trích xuất đặc trưng hoàn chỉnh như hình sau.



Hình 17. Hàm trích xuất đặc trưng

Hàm trên xử lý ảnh trên hệ màu HSV và chuyển mỗi kênh màu H, S, V sang LBP được và chuyển chúng thành Histogram, kết quả là tạo thành 3 vector có kích cỡ là 59. Nối chúng lại với nhau để đặc trưng cuối cùng là một vector có kích cỡ là 177. Như vậy với ví dụ 3 hình ảnh ở trên thì đặc trưng thu được sau khi hàm \_\_get chạy có kích thước là (3, 177).

Khi được các hàm ở trên, ở hàm \_\_init\_\_ thực hiện các câu lệnh sau câu lệnh super như sau



Hình 18. Câu lệnh thực hiện các chức năng của LBP

Đầu tiên khởi tạo hàm ánh xạ các mẫu nhị phân đã được đề cập ở trên và lưu vào biến self.\_map. Nếu is\_json là True thì sẽ thực hiện đọc các đặc trưng, đường dẫn ảnh từ file json đã được truyền ở biến json\_data và json\_path. Ngược lại đọc dữ liệu bằng hàm self.\_\_get. Cuối cùng biến save nếu bằng True sẽ lưu các đặc trưng vào file json.

* 1. **Hàm huấn luyện**

A picture containing text, screenshot, font, number

Description automatically generated

Hình 19. Lớp FaceRecognition - Nhận diện khuôn mặt

Lớp FaceRecognition nhận các đầu vào

* config, data\_json, path\_json, labels\_json: các file này sẽ lưu các giá trị đặc trưng, nhãn, đường dẫn ảnh, file config sẽ được giải thích ở mục 2.4
* KNN: dùng để tính toán đưa ra nhãn dự đoán.

Hàm huấn luyện cho lớp FaceRecognition, thực hiện huấn luyện cho 2 ảnh có khuôn mặt và không có khuôn mặt

*Viết chương trình cho ảnh có khuôn mặt*

A screenshot of a computer code

Description automatically generated with medium confidence

Hình 20. Trích xuất đặc trưng cho ảnh có khuôn mặt

Thực hiện đọc các thông tin cần thiết như đường dẫn ảnh, số lượng ảnh đầu vào, số lượng ảnh cho phép huấn luyện mỗi lần chạy. Thực hiện xử lý ảnh trên kích cỡ (112, 112). Tương tự với chương trình trích xuất đặc trưng ảnh không có khuôn mặt dưới đây.

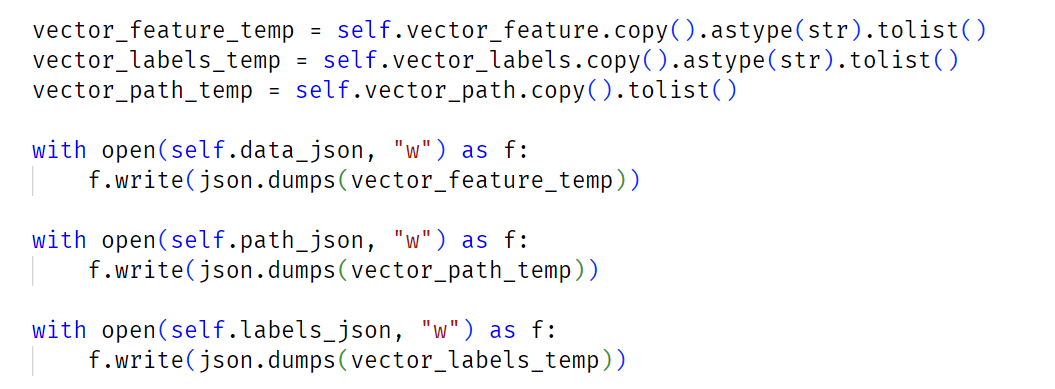
*Viết chương trình cho ảnh không có khuôn mặt*

A picture containing text, screenshot, font

Description automatically generated

Hình 21. Trích xuất đặc trưng cho ảnh không có khuôn mặt

*Viết chương trình lưu các đặc trưng vào file JSON*



Hình 22. Chương trình lưu đặc trưng vào file JSON

Hàm đánh giá mô hình, thực hiện tương tự như hàm huấn luyện ở trên nhưng thay một số chỗ như sau:

* Tham số move truyền vào LBP sẽ chuyển thành False
* Không lưu vector đặc trưng vào file json
* Thực hiện tính toán khoảng cách từ 1 ảnh đến toàn bộ dữ liệu bằng thuật toán K – nearest neighbors. Và so sánh nhãn.

*Chương trình thực hiện so sánh nhãn*

A picture containing text, screenshot, font, number

Description automatically generated

Hình 23. Thuật toán K - nearest neighbors và so sánh nhãn

* 1. **Ứng dụng Local binary pattern vào tìm kiếm hình ảnh tương đồng**

Để ứng dụng nhận diện khuôn mặt, nhóm nghiên cứu đã chia ra làm 2 giai đoạn đó là giai đoạn huấn luyện và giai đoạn thử nghiệm.

* Giai đoạn huấn luyện

Chuẩn bị dữ liệu: Bộ dữ liệu mà nhóm chuẩn bị được lấy từ bộ dữ liệu Coco 2017 – một tập dữ liệu lớn được sử dụng trong lĩnh vực nhận diện đối tượng. Tập dữ liệu này bao gồm 330000 hình ảnh và hơn 2,5 triệu đối tượng được gán nhãn thuộc về 80 đối tượng khác nhau.

Dưới đây là quá trình huấn luyên được sử dụng

A picture containing text, screenshot, diagram

Description automatically generated

Hình 24. Quá trình huấn luyện

Chương trình thực hiện đọc ảnh đầu vào là ảnh chứa khuôn mặt và ảnh không chứa khuôn mặt. Mặc định đầu vào là hệ màu BGR. Chuyển sang hệ màu HSV, áp dụng LBP Uniform pattern cho mỗi kênh màu. Tương tự áp dụng tiếp Histogram cho mỗi kênh màu của ảnh, kết quả được 3 vector histogram tương ứng cho 3 kênh màu HSV. Nối 3 vector thành 1 vector, vì dùng LBP Uniform pattern nên mỗi vector đặc trưng sẽ có 59 giá trị. Vector đặc trưng cuối cùng sẽ có 59x3 = 177 giá trị. Sau khi được vector đặc trưng, lưu lại vào file JSON để sử dụng cho giai đoạn thử nghiệm.

* Giai đoạn thử nghiệm
  1. **Kết quả thực nghiệm**
  2. **Kết luận**

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | Sánchez López, Laura, Local Binary Patterns applied to Face Detection and Recognition, Project/Final Thesis, UPC, Barcelona School of Telecommunications Engineering, Department of Signal Theory and Communications, 2010. |
| [2] | T. Ojala, M. Pietikainen and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, July 2002, pp. 971 - 987. |
| [3] | Di Huang, Caifeng Shan, Mohsen Ardebilian, Yunhong Wang, and Liming Chen, "Local Binary Patterns and Its Application to Facial Image Analysis: A Survey," in *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 6, Nov. 2011, pp. 765-781. |
| [4] | L. V. Đông, "Local Binary Patterns -LBP," 17 04 2022. [Online]. Available: https://www.levandong.com/local-binary-pattern-lbp/. [Accessed 5 6 2023]. |
| [5] | M. Nguyen, "Hiện thực trích đặc trưng Local Binary Patterns (LBP)," 29 08 2020. [Online]. Available: https://minhng.info/tutorials/local-binary-patterns-lbp-opencv.html. [Accessed 05 06 2023]. |
| [6] | "OpenCV Wiki-pages," SourceForge, 2001. [Online]. Available: https://sourceforge.net/projects/opencvlibrary/. |
| [7] | T. Oliphant, "Numpy Wiki pages," Community project, 2006. [Online]. Available: https://wiki.python.org/moin/NumPy. |