Mục lục

[**I.Giới thiệu** 2](#_Toc72106176)

[**A. Đề bài** 2](#_Toc72106177)

[**B. Tổng quan quá trình thực hiện nghiên cứu** 2](#_Toc72106178)

[**II.Hệ thống nhận dạng ảnh chim** 3](#_Toc72106179)

[**A. Sơ đồ hệ thống nhận dạng ảnh chim** 3](#_Toc72106180)

[1.Giai đoạn training lưu các thuộc tính, đặc trưng của từng loài chim 3](#_Toc72106181)

[2.Giai đoạn nhận dạng ảnh 6](#_Toc72106182)

[**III.Chương trình nhận dạng** 7](#_Toc72106183)

[**A.** **Giai đoạn training** 7](#_Toc72106184)

[1.Đặc trưng Hu Monents 7](#_Toc72106185)

[2.Đăc trưng Haralick Texture 8](#_Toc72106186)

[3.Đặc trưng Color Histogram 8](#_Toc72106187)

[4.Quá trình training 9](#_Toc72106188)

[5.Lưu trữ các đặc trưng 10](#_Toc72106189)

[**B. Giai đoạn nhận dạng ảnh** 10](#_Toc72106190)

[1.Lấy global\_features và labels đã lưu 10](#_Toc72106191)

[**2.** Đọc ảnh test và tạo feature vector 11](#_Toc72106192)

[**3**.Thêm vào test\_features 11](#_Toc72106193)

[4.Thực hiện test 12](#_Toc72106194)

[**C.Kết quả chạy chương trình demo** 12](#_Toc72106195)

[1.Chạy training dữ liệu 12](#_Toc72106196)

[2.Chạy test ảnh 13](#_Toc72106197)

# **I.Giới thiệu**

## **A. Đề bài**

Xây dựng hệ CSDL nhận dạng ảnh chim.

1.Hãy sưu tầm ít nhất 100 bức ảnh về ít nhất 10 loài chim khác nhau (10 nhãn), các bức ảnh đều có cùng kích thước.

2.Hãy tìm hiểu các kỹ thuật xử lý và phân loại ảnh chim đang hiện hành.

3.Xây dựng hệ thống nhận dạng ảnh chim với đầu vào là một ảnh mới của một loài chim đã có và chưa có trong phần 1, đầu ra là kết quả nhận dạng nhãn của loài chim trong ảnh đầu vào.

a. Trình bày sơ đồ khối của hệ thống và quy trình thực hiện yêu cầu của đề bài.

b. Trình bày các thuộc tính được sử dụng để nhận dạng nhãn về loài chim trong hệ thống, cùng các kỹ thuật để trích rút các thuộc tính đó.

c.Trình bày cách lưu trữ các thuộc tính ảnh chim và cách nhận dạng ảnh chim dựa trên các thuộc tính đó.

4.Demo hệ thống và đánh giá kết quả đã đạt được.

## **B. Tổng quan quá trình thực hiện nghiên cứu**

Các phương pháp nhận dạng ảnh chim đều dựa vào việc đối sánh giữa các đặc điểm đặc trưng trên con chim. Có nhiều phương pháp đối sánh khác nhau. Trong lí thuyết nhận dạng nói chung và nhận dạng ảnh nói riêng, có ba cách tiếp cận khác nhau:

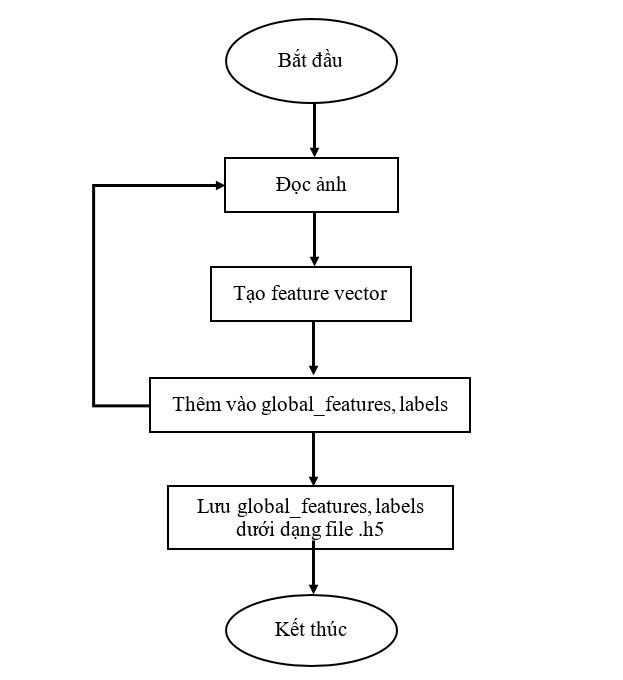
* Nhân dạng ảnh dựa vào phân hoạc không gian
* Nhận dạng dựa vào cấu trúc
* Nhận dạng dựa vào kĩ thuật mạng noron

Hai cách tiếp cận đầu là cách tiếp cận kinh điển .Các đối tượng ảnh quan sát và thu nhận được trải qua giai đoạn tiền xử lí ảnh nhằm tang cường chất lượng, làm nổi các chi tiết ,tiếp theo là trích rút chọn và biểu diễn các đặc trưng ,cuối cùng mới là nhận dạng ảnh .Cách tiếp cận thứ ba hoàn toàn khác .Nó dựa vào cơ chế đoán nhận, lưu trữ và phân biệt đối tượng thu mô phỏng theo hệ thần kinh con người.Do cơ chế đặc biệt, các đối tượng thu nhận bởi thị giác người không cần qua giai đoạn cải thiện mà chuyển sang giai đoạn tổng hợp, đối sánh với các mẫu đã lưu trữ để nhận dạng.

# **II.Hệ thống nhận dạng ảnh chim**

## **A. Sơ đồ hệ thống nhận dạng ảnh chim**

### 1.Giai đoạn training lưu các thuộc tính, đặc trưng của từng loài chim



Sau khi đọc ảnh đầu vào là tập dữ liệu training ảnh chim đã được phân theo nhãn thì ta cần xây dựng các bộ mô tả hình ảnh để trích xuất ra đặc trưng quan trọng của dữ liệu. Đề tài là nhận dạng ảnh chim, nên nhóm nghiên cứu đã xác định nhận dạng ảnh chim cần dựa vào 3 đăc điểm của chim để phân biệt chúng. Đó là màu sắc chủ đạo, hình dạng đối tượng trong bức ảnh, kết cấu hoa văn của các loài chim.

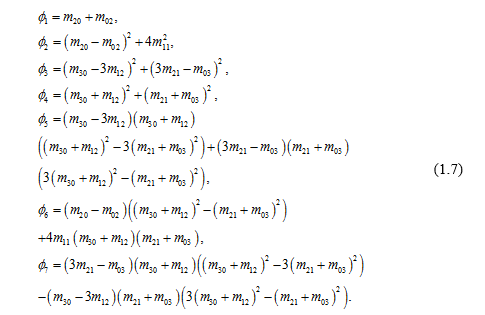
1. Color histogram

Color Histogram (CH) là một dạng đặc trưng toàn cục biểu diễn phân phối của các màu trên ảnh. CH thống kê số lượng các pixel có giá trị nằm trong một khoảng màu nhất định cho trước. CH có thể được tính trên các ảnh dạng RGB hoặc HSV, thông dụng là HSV. Cách tính là với mỗi kênh màu nhất định, chia kênh màu này thành “*n* bin”. Sau đó, thống kê số lượng pixel trên ảnh có giá trị màu thuộc về “*n* bin” này. Cuối cùng, nối các bin của các kênh màu lại với nhau để tạo thành đặc trưng. Lưu ý là các màu khác nhau số lượng bin màu không nhất thiết phải giống nhau. Ví dụ: Tính CH của ảnh HSV thành: 14 bin cho H, 4 bin cho S, 4 bin cho V. Thông thường người ta chia ảnh ra thành các “*sub region”* để tăng số chiều đặc trưng.

1. Hu Monents

Hu Monents là một Image Descriptor sử dụng các phép thống kê để mô tả hình dạng của một đối tượng có trong bức ảnh nhị phân hoặc edged-image. Hu Moments Image Descriptor trả về một Feature Vector gồm 7 giá trị. Feature Vector này sẽ được so sánh với nhau để xác định sự tương đồng giữa hai vật thể.

Công thức tính 7 Moment:

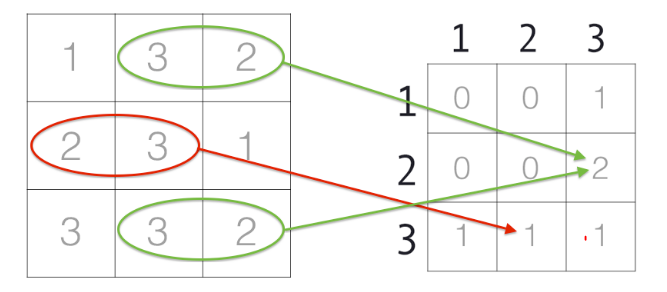


Các Moment được tính toán dựa trên các central moment. Các central moment là bất biến, bất kể đốm màu ở đâu trong bức ảnh mà hình dạng giống nhau thì các moment sẽ giống nhau.

1. Haralick Texture

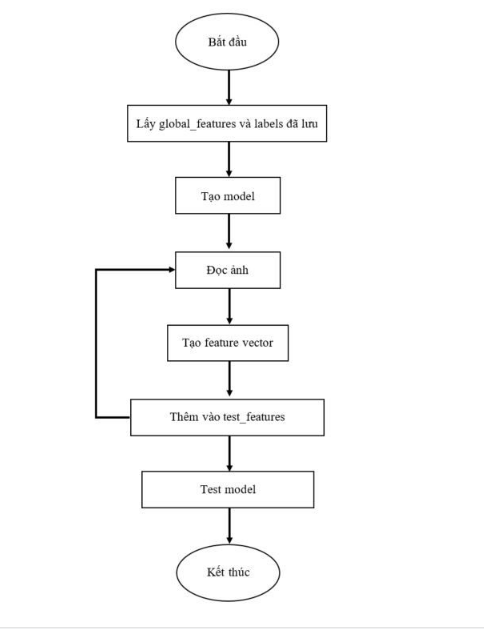
Haralick Texture Features được dùng để mô tả “texture” (kết cấu) và “pattern” (hoa văn) của một bức ảnh/đối tượng, bao gồm vẻ bề ngoài (appearance), sự nhất quán (consistency) và cảm giác về bề mặt (“feeling of surface”) có trong bức ảnh.

Haralick Texture Features được tính toán bằng cách sử dụng ***Gray-Level Co-occurrence Matrix*** (*GLCM*) [1]. Ma trận này đặc trưng cho kết cấu của bức ảnh bằng cách ghi lại tần suất xuất hiện trong bức ảnh của các cặp điểm ảnh liền kề với các giá trị cụ thể.



Có 4 ma trận GLCM được sử dụng để tính toán, từ mỗi ma trận sẽ cho ra 13 giá trị của Haralick Texture Featues (tức là feature shape = (4, 13)), sau đó tìm ra 13 giá trị trung bình từ list này. Các giá trị này sẽ được dùng để mô tả sự tương phản (contrast), tương quan (correlation), sự khác biệt (dissimilarity), entropy, tính đồng nhất (homogeneity) và các đặc tính thống kê khác.

### 2.Giai đoạn nhận dạng ảnh



Cũng giống như giai đoạn training dữ liệu, giai đoạn 2 nhận dạng ảnh ta cũng cần phân tích các đặc trưng của bức ảnh test rồi sau đó so sánh để nhận dạng được nhãn

Trong giai đoạn 2 nhận diện ảnh nhóm có sử nghiên cứu và lựa chọn sử dụng thuật toán Random Forest để gán nhãn cho ảnh

1. Giới thiệu về thuật toán Random Forest

Random Forest là một thuật toán học có giám sát. Như tên gọi của nó, Rừng ngẫu nhiên sử dụng các cây (tree) để làm nền tảng.

Random Forest là một tập hợp của các Decision Tree, mà mỗi cây được chọn theo một thuật toán dựa vào ngẫu nhiên.

Random là ngẫu nhiên, Forest là rừng, nên ở thuật toán Random Forest mình sẽ xây dựng nhiều cây quyết định bằng thuật toán Decision Tree, tuy nhiên mỗi cây quyết định sẽ khác nhau (có yếu tố random). Sau đó kết quả dự đoán được tổng hợp từ các cây quyết định.

1. Thuật toán **Random Forest làm việc như thế nào**

Random Forest hoạt động bằng cách đánh giá nhiều Cây quyết định ngẫu nhiên, và lấy ra kết quả được đánh giá tốt nhất trong số kết quả trả về

Hoạt động của **Random Forest**

**Tạo ra một Random Forest**

1. *Chọn ngẫu nhiên****“k”****features từ tập****“m”****features.*

*Để ý****k << m***

1. *từ tập****“k”****features, tính toán ra node****“d”****là tốt nhất cho Node phân loại.*
2. *Chia các node con theo node tốt nhất vừa tìm được*
3. *Lặp lại bước****1-3****cho đến khi đạt đến****k****node*
4. *Lặp lại bước****1-4****để tạo ra****“n”****cây*

Để biểu diễn dự đoán sử dụng Random Forest đã huấn luyệ, ta sử dụng các bước bên dưới:

1. *Lấy các****test features****và sử dụng các Cây quyết định đã tạo ra để dự đoán kết quả, lưu nó vào một danh sách.*
2. *TÍnh toán số lượng vote trên toàn bộ Forest cho từng kết quả*
3. *Lấy kết quả có số lượng vote lớn nhất làm kết quả cuối cho mô hình*

c.Tại sao lựa chọn thuật toán **Random Forest**

Trong thuật toán Decision Tree, khi xây dựng cây quyết định nếu để độ sâu tùy ý thì cây sẽ phân loại đúng hết các dữ liệu trong tập training dẫn đến mô hình có thể dự đoán tệ trên tập validation/test, khi đó mô hình bị overfitting, hay nói cách khác là mô hình có phương sai cao

Thuật toán Random Forest gồm nhiều cây quyết định, mỗi cây quyết định đều có những yếu tố ngẫu nhiên:

1. Lấy ngẫu nhiên dữ liệu để xây dựng cây quyết định.
2. Lấy ngẫu nhiên các thuộc tính để xây dựng cây quyết định.

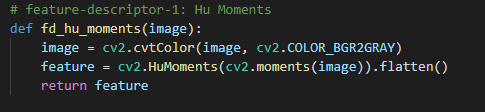
Do mỗi cây quyết định trong thuật toán Random Forest không dùng tất cả dữ liệu training, cũng như không dùng tất cả các thuộc tính của dữ liệu để xây dựng cây nên mỗi cây có thể sẽ dự đoán không tốt, khi đó mỗi mô hình cây quyết định không bị overfitting mà có thế bị thiếu dữ liệu, hay nói cách khác là mô hình có tính thiên vị cao. Tuy nhiên, kết quả cuối cùng của thuật toán Random Forest lại tổng hợp từ nhiều cây quyết định, thế nên thông tin từ các cây sẽ bổ sung thông tin cho nhau, dẫn đến mô hình có độ thiên vị thấp và phương sai thấp, hay mô hình có kết quả dự đoán tốt.

# **III.Chương trình nhận dạng**

## **Giai đoạn training**

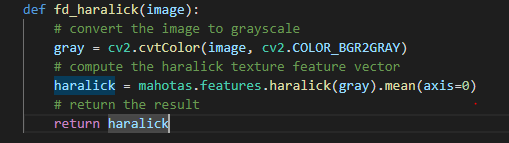
Đề tài là nhận dạng ảnh chim, nên nhóm nghiên cứu đã xác định nhận dạng ảnh chim cần dựa vào 3 đăc điểm của chim để phân biệt chúng. Đó là màu sắc chủ đạo ,hình dạng đối tượng trong bức ảnh ,kết cấu hoa văn của các loài chim.

### 1.Đặc trưng Hu Monents



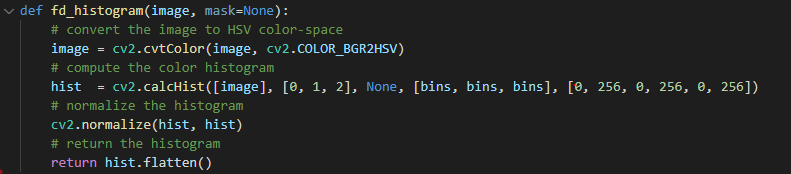
Để trích xuất các tính năng Hu Moments từ hình ảnh, chúng tôi sử dụng hàm cv2.HuMoments () do OpenCV cung cấp. Đối số cho hàm này là các khoảnh khắc của hình ảnh cv2.moments () được làm phẳng. Nó có nghĩa là chúng tôi tính toán các khoảnh khắc của hình ảnh và chuyển đổi nó thành một vector bằng cách sử dụng flatten (). Trước khi làm điều đó, chúng tôi chuyển đổi hình ảnh màu của chúng tôi thành một hình ảnh thang độ xám khi những khoảnh khắc mong đợi hình ảnh có thang độ xám.

### 2.Đăc trưng Haralick Texture



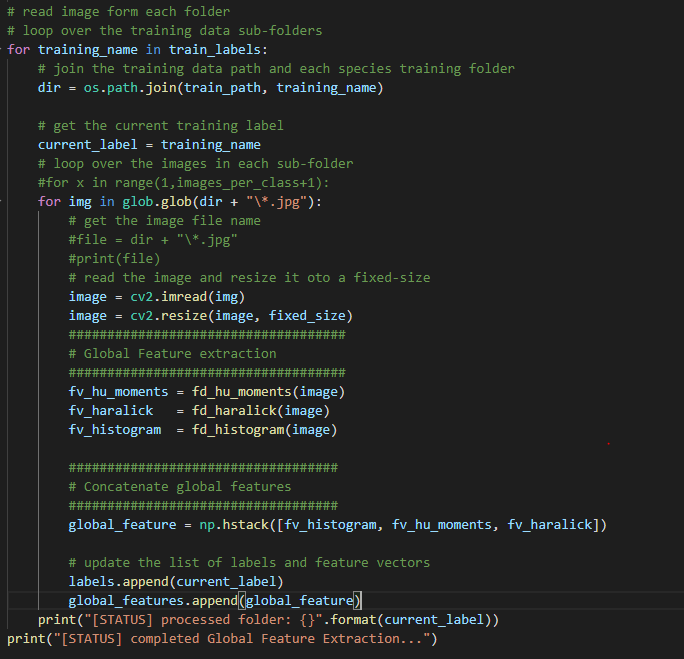
Để trích xuất các tính năng của Haralick Texture từ hình ảnh, chúng tôi sử dụng thư viện mahotas. Hàm chúng ta sẽ sử dụng là mahotas.features.haralick () . Trước khi làm điều đó, chúng tôi chuyển đổi hình ảnh màu của chúng tôi thành một hình ảnh thang độ xám vì bộ mô tả tính năng haralick mong muốn hình ảnh có thang độ xám

### 3.Đặc trưng Color Histogram



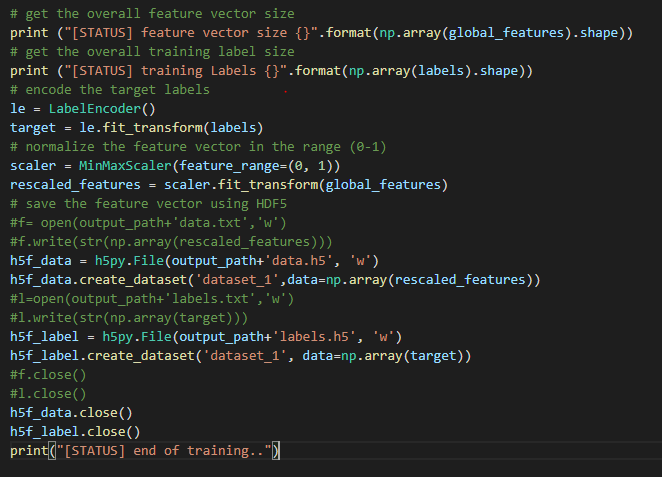
Để trích xuất các tính năng Biểu đồ màu từ hình ảnh, chúng tôi sử dụng hàm cv2.calcHist () do [OpenCV](https://docs.opencv.org/3.2.0/d1/db7/tutorial_py_histogram_begins.html) cung cấp. Các đối số mà nó mong đợi là hình ảnh, các kênh, mặt nạ, histSize (thùng) và phạm vi cho mỗi kênh [thường là 0-256). Sau đó, chúng tôi chuẩn hóa biểu đồ bằng cách sử dụng hàm normalize () của OpenCV và trả về phiên bản phẳng của ma trận chuẩn hóa này bằng cách sử dụng flatten ().

### 4.Quá trình training



Đối với mỗi tên nhãn đào tạo, chúng tôi lặp qua thư mục tương ứng để lấy tất cả các hình ảnh bên trong nó. Đối với mỗi hình ảnh mà chúng tôi lặp lại, trước tiên chúng tôi thay đổi kích thước hình ảnh thành một kích thước cố định. Sau đó, chúng tôi trích xuất ba tính năng toàn cục và nối ba tính năng này bằng cách sử dụng hàm np.hstack () của NumPy . Chúng tôi theo dõi đối tượng địa lý bằng nhãn của nó bằng cách sử dụng hai danh sách mà chúng tôi đã tạo ở trên - nhãn và global\_features.

### 5.Lưu trữ các đặc trưng

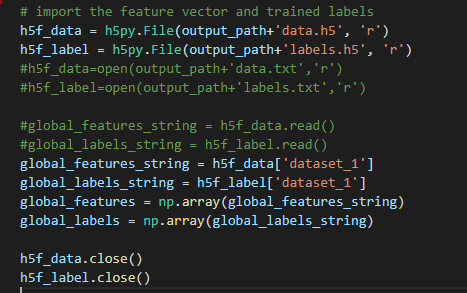


Sau khi có được các đặc trưng và nối chúng lại, chúng ta cần lưu dữ liệu này cục bộ. Trước khi lưu dữ liệu này, chúng tôi sử dụng một thứ gọi là LabelEncoder () để mã hóa các nhãn của chúng tôi ở định dạng thích hợp. Điều này là để đảm bảo rằng các nhãn được biểu diễn dưới dạng số duy nhất. Vì chúng tôi đã sử dụng các tính năng toàn cầu khác nhau, nên một tính năng có thể chiếm ưu thế so với tính năng khác về giá trị của nó. Trong các tình huống như vậy, tốt hơn là nên chuẩn hóa mọi thứ trong một phạm vi (giả sử 0-1). Do đó, chúng tôi chuẩn hóa các tính năng bằng cách sử dụng hàm MinMaxScaler () của scikit-learning. Sau khi thực hiện hai bước này, chúng tôi sử dụng h5py để lưu cục bộ các tính năng và nhãn của chúng tôi ở định dạng tệp .h5

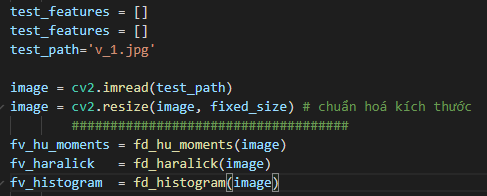
## **B. Giai đoạn nhận dạng ảnh**

### 1.Lấy global\_features và labels đã lưu





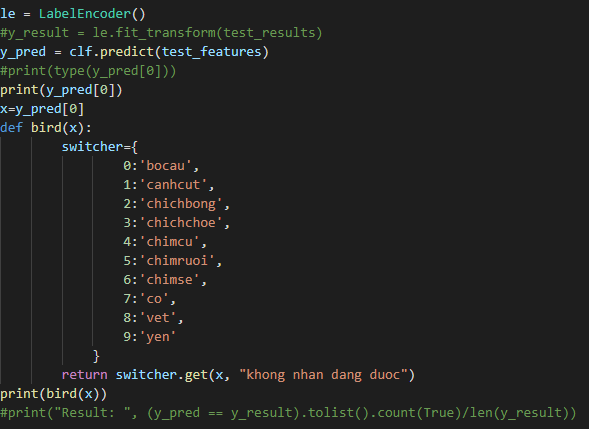
### **2.** Đọc ảnh test và tạo feature vector



### **3**.Thêm vào test\_features

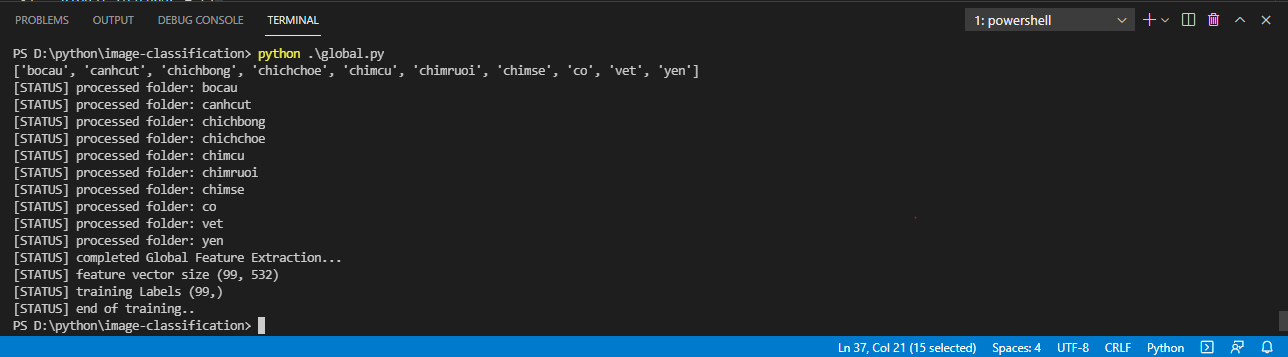


### 4.Thực hiện test



## **C.Kết quả chạy chương trình demo**

### 1.Chạy training dữ liệu



### 2.Chạy test ảnh

