

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG KHOA CÔNG NGHỆ THÔNG TIN I**

----------

**BÁO CÁO BÀI TẬP**

**Môn: Hệ cơ sở dữ liệu đa phương tiện**

|  |  |  |
| --- | --- | --- |
| *Nhóm học phần:* | INT1418 |  |
| *Báo cáo:* | Xây dựng hệ cơ sở dữ liệu nhận dạng chữ viết tay |  |
| *Nhóm thực hiện:* | Nhóm 10 |  |
| *Sinh viên thực hiện:* | Nguyễn Thị Minh Châu | B18DCCN075 |
|  | Nguyễn Tiến Dũng | B18DCCN099 |
|  | Đỗ Đăng Mạnh | B18DCCN394 |
|  |
|  |
|  |
|  |

***Hà Nội, tháng 06 năm 2022***

**Mục lục**

Contents

[LỜI CẢM ƠN 4](#_Toc106653926)

[Chương 1 5](#_Toc106653927)

[Giới thiệu 5](#_Toc106653928)

[1. Giới thiệu về nhận dạng ảnh 6](#_Toc106653929)

[2. Kỹ thuật nhận dạng ảnh – So sánh mẫu dựa trên đặc trưng được trích chọn 6](#_Toc106653930)

[Chương 2 10](#_Toc106653931)

[Tổng quan về bài toán nhận dạng chữ viết tay 10](#_Toc106653932)

[1. Chữ viết tay 11](#_Toc106653933)

[2. Bài toán nhận dạng chữ viết tay 11](#_Toc106653934)

[Chương 3 13](#_Toc106653935)

[Xây dựng hệ thống nhận dạng chữ viết tay 13](#_Toc106653936)

[ Mô tả hệ thống 14](#_Toc106653937)

[1. Sơ đồ khối hệ thống nhận dạng ảnh chữ cái viết tay 14](#_Toc106653938)

[2. Xây dựng hệ thống nhận dạng ảnh chữ cái viết tay 15](#_Toc106653939)

[2.1. Bước 1: Trích rút đặc trưng cục bộ bất biến của ảnh sử dụng thuật toán SIFT(Scale- Invariant Feature Transform) 15](#_Toc106653940)

[2.2. Phân cụm sử dụng thuật toán Kmean 26](#_Toc106653941)

[2.3. Tạo vector mô tả ảnh duy nhất ( Bag of Features phát triển dựa trên Bag of Word) 27](#_Toc106653942)

[2.4. So sánh đối chiếu ảnh 28](#_Toc106653943)

[3. Cách lưu trữ thuộc tính nhận dạng ảnh 29](#_Toc106653944)

[Chương 4 30](#_Toc106653945)

[Triển khai hệ thống và đánh giá kết quả 30](#_Toc106653946)

[1. Triển khai cài đặt hệ thống 31](#_Toc106653947)

[1.1. Xậy dựng bộ dữ liệu 31](#_Toc106653948)

[1.2. Triển khai (mô tả một số hàm cơ bản bằng code) 32](#_Toc106653949)

[1.3. Kết quả thử nghiệm 40](#_Toc106653950)

[2. Đánh giá kết quả 42](#_Toc106653951)

[Chương 5 45](#_Toc106653952)

[Kết luận 45](#_Toc106653953)

[1. Nhận xét 46](#_Toc106653954)

[2. Thu hoạch đạt được 46](#_Toc106653955)

[TÀI LIỆU THAM KHẢO 47](#_Toc106653956)

# LỜI CẢM ƠN

Đầu tiên, chúng em xin gửi lời cảm ơn đến Học viện Công nghệ Bưu chính Viễn thông đã đưa bộ môn **“*Hệ cơ sở dữ liệu đa phương tiện”*** vào trong chương trình giảng dạy để chúng em có điều kiện được học tập, nghiên cứu và tìm hiểu.

Đặc biệt, chúng em xin gửi lời cảm ơn chân thành và sâu sắc đến giảng viên bộ môn Thầy ***Nguyễn Đình Hóa*** đã tận tình dạy dỗ, rèn luyện và truyền đạt cho chúng em những kiến thức quý báu trong suốt kỳ học vừa qua. Trong thời gian được tham dự lớp học của Thầy, chúng em đã được tiếp thu nhiều kiến thức, những kỹ năng cần thiết, học tập được tinh thần làm việc hiệu quả và nghiêm túc. Đây thực sự là điều cần thiết cho quá trình học tập và công tác của chúng em sau này.

Chúng em đã cố gắng hoàn thành bài báo cáo một cách tốt nhất, tuy nhiên do kiến thức còn hạn chế, không có nhiều kinh nghiệm nên bài báo cáo của chúng em không tránh khỏi những sai sót. Chúng em rất mong nhận được sự chỉ bảo, góp ý của Thầy để có thể bổ sung hoàn thiện bài báo cáo, nâng cao kiến thức của bản thân nhằm giúp ích cho công việc trong tương lai.

**Chúng em xin chân thành cảm ơn!**

*Hà Nội, ngày 20 tháng 06 năm 2022*

**Nhóm sinh viên thực hiện**

# Chương 1

# Giới thiệu

Trong chương 1, báo cáo sẽ trình bày một cách tổng quan nhất về các vấn đề trong xử lý và nhận dạng ảnh được sử dụng trong hệ thống qua các phần:

* Giới thiệu về nhận dạng ảnh
* Giới thiệu về kỹ thuật nhận dạng ảnh

1. **Giới thiệu về nhận dạng ảnh**

Xử lí ảnh số (*DIP - Digital Image Processing*) nói chung và nhận dạng đối tượng (*Object Recognition*) nói riêng, đã và đang được ứng dụng rất nhiều. Nó trải rộng trên mọi mặt trong cuộc sống từ xử lí các bản in ấn, báo, bìa tạp chí đến vấn đề về năng lực nhìn trong máy học, đôi khi nó rất gần gũi trong cuộc sống hàng ngày mà nhiều người không nhận ra như chức năng nhận dạng khuôn mặt, đối tượng chuyển động trong các máy ảnh, máy quay phim hay đơn giản là chức năng camera của điện thoại đi động cũng được tích hợp các công cụ xử lí ảnh và nhận dạng đối tượng. Đề cập sâu hơn vào vấn đề nhận dạng đối tượng là các bài toán cụ thể như nhận dạng dấu vân tay, nhận dạng khuôn mặt, nhận dạng chữ viết, nhận dạng dộng vật. Các bài toán này được ứng dụng rất nhiều trong cuộc sống. Từ những ứng dụng rộng rãi như vậy mà tại các tổ chức, các trung tâm nghiên cứu của các trường đại học trên thế giới, các đề tài về xử lí ảnh luôn được đầu tư nhiều về vật chất lẫn thời gian.

Tuy nhiên dù nhu cầu rất cần thiết phục vụ cho cuộc sống nhưng hiện nay ở Việt Nam lĩnh vực này vẫn còn rất mới và có ít những ứng dụng được triển khai trong thực tế. Dù là tầm quan trọng của việc xử lý và nhận dạng ảnh là như vậy nhưng hiện nay vấn đề này chưa nhận được nhiều kết quả như mong đợi và cần đầu tư nhiều hơn nữa. Trong vấn đề cụ thể nhận dạng đối tượng thì ngày nay hướng nghiên cứu phổ biến trên thế giới là việc sử dụng các điểm bất biến (Invarian Feature) trong ảnh làm đặc trưng (Keypoint) để nhận dạng. Tiêu biểu nhất trong các thuật toán đối sánh sử dụng keypoint dạng này là thuật toán SIFT (Scale-Invarian Feature Transform, David Lowe 1999 và 2004), SIFT có thể coi là thuật toán tiền đề cho các ứng dụng cũng như giải thuật khác về biến đổi đặc trưng bất biến trong ảnh.

Vì tầm quan trọng trọng của lĩnh vực này nên nhóm mình hôm nay sẽ đưa ra những nghiên cứu về đề tài nhận dạng ảnh. Những kĩ thuật dùng để nhận dạng ảnh, những thuộc tính có thể dùng để nhận dạng ảnh, cũng như áp dụng những thuộc tính, kĩ thuât đó để nhận dạng ảnh động vật.

1. **Kỹ thuật nhận dạng ảnh – So sánh mẫu dựa trên đặc trưng được trích chọn**

Trong kĩ thuật này ta cần 1 tập các mẫu đã được lưu để làm cơ sở so sánh. Khi các ảnh được đưa vào lưu trong hệ thống, cũng sẽ lưu dưới dạng các đặc trưng được trích chọn như ma trận màu sắc, các đường biên, hình dạng ảnh.Vậy người ta trích xuất các đặc trưng bằng cách nào:

Màu sắc của ảnh: đây là đặc trưng dễ nhất để dùng so sánh 2 ảnh khác nhau, khi ảnh được đưa vào hệ thống lưu. Ảnh sẽ được lưu dưới dạng ma trận ảnh.

* + - Nếu ảnh được lưu dưới dạng nhị phân thì giá trị điểm ảnh chỉ có 2 giá trị: 1 hoăc 0 ( ảnh nhị phân).
    - Nếu ảnh được lưu dưới dạng đen trắng thì ảnh chi có 2 màu đen trắng với mức xám khác nhau ở mỗi điểm ảnh.
    - Nếu ảnh được lưu theo hệ màu RGB thì các điểm ảnh sẽ được lưu dưới 3 ma trận. mỗi ma trận biểu thị các giá trị màu sắc tương ứng Red, Green, Blue của điểm ảnh.

Sau đó để so sánh 2 ảnh người ta sẽ so sánh 2 ma trận ảnh: có thể dùng khoảng cách Euclide

Khoảng cách Euclide được tính bằng công thức:

Hoặc:

Khi khoảng cách Euclide bằng 0, là hai ảnh bằng nhau tuyệt đối. Thông thường thì ít có trường hợp như vậy.

Hoặc sử dụng Độ đo Jensen-Shannon divergence (JSD) : Độ đo Jensen- Shannon divergence sử dụng lược độ màu RGB để tính toán độ tương đồng về màu sắc giữa 2 ảnh :

Trong đó : H và H’ là 2 biểu đồ màu được so sánh, Hm là giá trị điểm ảnh thứ m của biểu đồ H.

Các đường biên: các đường biên là tập hợp các điểm biên, điểm biên là điểm ảnh mà tại đó có sự thay đổi đột ngột về màu sắc của điểm ảnh. Khi lưu vào máy tính hệ thống sẽ tìm các điểm biên đó. Và tập hợp lại, tạo ra đường biên.

* **Kỹ thuật phát hiện biên**

Chủ yếu dựa trên đạo hàm, nếu đạo hàm bậc nhất ta có phương pháp Gradient, đạo hàm bậc 2 có phương pháp Laplace. Trong hệ thống chúng em sử dụng phương pháp Gradient.

Gradient là một vector có các thành phần biểu thị tốc độ thay đổi giá trị của điểm ảnh:

Trong đó dx, dy là khoảng cách giữa 2 điểm lân cận theo hướng x và y tương ứng.

Trong những hình ảnh rời rạc, ta có thể xem xét dx, dy về số lượng điểm ảnh giữa hai điểm. Thực tế ta chọn dy = dx = 1 (khoảng cách điểm ảnh) là điểm mà tại đó tọa độ điểm ảnh là (i, j), do đó:

Do tính chất phức tạp trong tính toán khi áp dụng phương pháp Gradient trong xử lý ảnh, người ta sử dụng kỹ thuật Gradient dùng cặp mặt nạ H1, H2 trực giao (theo 2 hướng vuông góc). Nếu định nghĩa g1, g2 là Gradient theo hai hướng x, y tướng ứng thì biên độ g(m,n) tại điểm (m,n) được tính:

Để giảm độ phức tạp tính toán, A0 được tính gần đúng như sau:

Việc xấp xỉ đạo hàm bậc nhất theo các hướng x và y được thực hiện thông qua 2 mặt nạ nhân chập tương ứng sẽ cho ta các kỹ thuật phát hiện biên khác nhau (Roberts, Sobel, Prewitt,..).

* Thuật toán:
  + Với mỗi điểm ảnh I(x,y) tính:
  + Phân ngưỡng:



# Chương 2

# Tổng quan về bài toán nhận dạng chữ viết tay

Trong chương 2, báo cáo sẽ trình bày tổng quan về chữ viết tay và bài toán nhận dạng chữ viết tay qua các phần:

* Chữ viết tay
* Bài toán nhận dạng chữ viết tay và các ứng dụng của bài toán này trong thực tế.

## Chữ viết tay

Theo Wikipedia, chữ viết tay là văn bản được thực hiện với một công cụ viết, chẳng hạn như bút hoặc bút chì và được viết bằng tay. Chữ viết tay của mỗi người là duy nhất và khác nhau, nó có thể được sử dụng để xác minh người viết tài liệu.

Tính duy nhất của chữ viết tay: mỗi người đều có một phong cách viết tay riêng, cho dù đó là chữ viết tay hằng ngày hay chữ ký cá nhân của họ. Ngay cả những cặp song sinh giống hệt nhau có chung ngoại hình và di truyền cũng không có chữ viết tay giống nhau. Môi trường văn hoá và đặc điểm của hình thức viết của ngôn ngữ đầu tiên mà người ta học viết hoà trộn với sự phân phối lực tay khi viết và cách định hình từ ngữ khác nhau là những ảnh hưởng chính đến sự phát triển của phong cách chữ viết tay của cá nhân mỗi người.

Đặc điểm của chữ viết tay bao gồm:

* Hình dạng cụ thể của các chữ cái, ví dụ như độ tròn hay độ rõ nét
* Độ nghiêng của các chữ cái
* Lực mạnh yếu tác dụng lên mặt phẳng viết
* Kích thước của các chữ
* Độ dày, mỏng của các nét chữ.
* …

## Bài toán nhận dạng chữ viết tay

Nhận dạng chữ viết tay (Handwritting recognition – HWR), còn được gọi là nhân dạng văn bản viết tay, là khả năng máy tính nhận ra và giải thích dữ liệu chữ viết tay từ các nguồn như tài liệu giấy, ảnh hay các thiết bị khác hoặc sử dụng chữ viết tay làm đầu vào trực tiếp trên màn hình cảm ứng, các thiết bị khác và sau đó hiểu được đây là văn bản. Cụ thể, đầu vào của bài toán sẽ là hình ảnh của chữ viết tay, khi đó hệ thống sẽ thực hiện lần lượt các nhiệm vụ trích xuất đặc trưng từ ảnh chữ viết tay đó, sau đó đem so sánh với các dữ liệu khác có trong cơ sở dữ liệu và tính toán độ tương đồng từ đó đưa ra được kết quả đầu ra là chữ tương ứng với chữ viết tay đó.

Nhận dạng chữ viết tay là một bài toán phổ biến và đã được nghiên cứu rất rộng rãi, từ đó, có rất nhiều thuật toán ra đời nhằm năng cao hiệu quả cũng như độ chính xác. Mặc dù các thuật toán ngày càng được cải tiến, thế nhưng nhìn chung nhận dạng chữ viết tay vẫn là một bài toán không dễ dàng vì mỗi chữ viết tay của mỗi người khác nhau lại có những đặc điểm riêng.

Việc giải quyết bài toán nhận dạng chữ viết tay giúp cho việc chuyển dần các hệ thống thủ công sang các hệ thống điện tử như là một phần nỗ lực để tiến đến tự động hoá, đặc biệt là ở các thị trường đang phát triển và mới nổi, mang đến cơ hội các hệ thống quét và chuyển đổi tài liệu hiệu quả. Điều này làm phát sinh sự cần thiết của các kỹ thuật nhận dạng chữ số và ký tự viết tay chính xác hơn. Công việc hiện tại của chúng tôi nhắm mục đích tìm kiếm một hệ thống phân loại được cải thiện cho mục đích nhận dạng chữ viết tay.

# Chương 3

# Xây dựng hệ thống nhận dạng chữ viết tay

Trong chương 3, báo cáo sẽ trình bày phương pháp cùng những thông tin liên quan đến việc xây dựng hệ thống nhận dạng chữ viết tay. Quá trình này được thực hiện thông qua 4 bước chính:

* Bước 1: Trích rút các đặc trưng cục bộ bất biến của ảnh sử dụng thuật toán SIFT (Scale- Invariant Feature Transform).
* Bước 2: Phân cụm sử dụng thuật toán Kmean
* Bước 3: Tạo vector mô tả ảnh cục bộ (Bag of Features phát triển dựa trên Bag of Word ).
* Bước 4: So sánh, đối chiếu ảnh (k-Nearest Neighbours)

## Mô tả hệ thống

Hệ cơ sở dữ liệu (CSDL) nhận dạng chữ viết tay gồm(data\_), các bức ảnh đều có cùng kích thước. **Nguồn**: Kaggle :

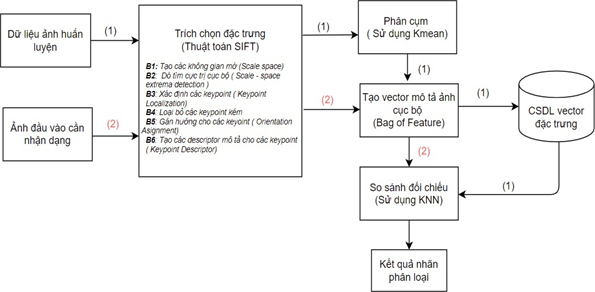
<https://www.kaggle.com/datasets/dhruvildave/english-handwritten-characters-dataset?select=Img>

Ảnh có kích thước 1200 x 900 pixel, gồm 52 folder (26 chữ in hoa, 26 chữ in thường), mỗi folder gồm 50 ảnh có kích thước bằng nhau. Tổng cộng có 50 \* 52 = 2600 ảnh.

**Bài toán**

* **Input**: Ảnh mới của một chữ cái đã có và chưa có trong CSDL.
* **Output**: Nhãn của chữ cái trong ảnh đầu vào.

## Sơ đồ khối hệ thống nhận dạng ảnh chữ cái viết tay



## Xây dựng hệ thống nhận dạng ảnh chữ cái viết tay

Hệ thống nhận dạng chữ viết tay được xây dựng theo 4 bước chính:

**Bước 1**: Trích rút các đặc trưng cục bộ bất biến của ảnh sử dụng thuật toán SIFT (Scale- Invariant Feature Transform)

**Bước 2**: Phân cụm sử dụng thuật toán Kmean

**Bước 3**: Tạo vector mô tả ảnh cục bộ (Bag of Features phát triển dựa trên Bag of Word )

**Bước 4**: So sánh, đối chiếu ảnh (k-Nearest Neighbours)

### 2.1. Bước 1: Trích rút đặc trưng cục bộ bất biến của ảnh sử dụng thuật toán SIFT(Scale- Invariant Feature Transform)

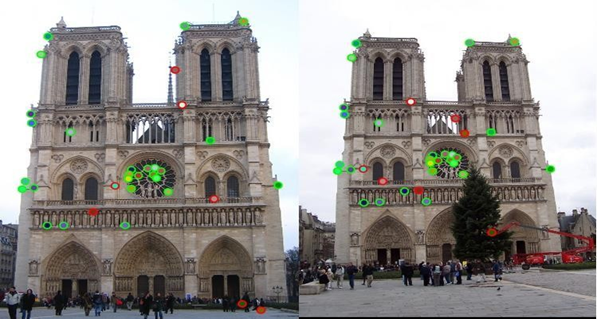
Trong bài này sử dụng thuật toán SIFT để trích rút đặc trưng cho ảnh. SIFT (viết tắt của Scale-Invariant Feature Transform) là thuật toán phát hiện các đặc trưng cục bộ bất biến của ảnh trong Computer Vision.

SIFT được sử dụng với các đặc điểm nổi trội:

* Nội dung ảnh được chuyển đổi thành các đặc trưng bất biến ngay cả khi:
* Dịch chuyển ảnh (image translation)
* Xoay ảnh (rotation)
* Thay đổi kích thước (scale)
* Ngoài ra các đặc trưng còn bất biến một phần ( partially invariant ) khi:
* Thay đổi độ sáng (illumination change)
* Biến đổi affine, hay phép chiếu (affine or 3D projection)

Ví dụ: Hai ảnh ban đầu có cùng nội dung chính nhưng góc chụp khác nhau tại hai thời điểm khác nhau:

Sau khi trích xuất đặc trưng SIFT → Ta được nhiều đặc trưng cục bộ bất biến tương đồng giữa 2 ảnh:



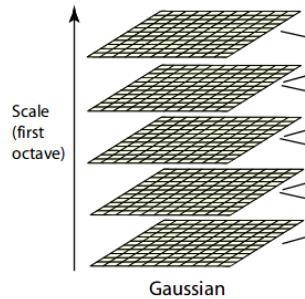
**2.1.1 Phương pháp tiếp cận - Các bước giải thuật SIFT**

Giải thuật SIFT thực hiện qua các bước:

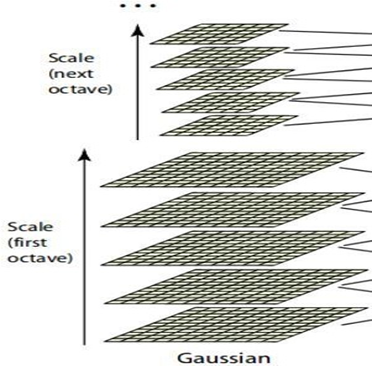
* Bước 1: Xây dựng không gian mờ (Scale-space)
* Bước 2: Dò tìm các cực trị cục bộ (Scale-space extrema detection)
* Bước 3: Xác định các keypoint (Keypoint Localization)
* Bước 4: Loại bỏ các keypoint kém
* Bước 5: Gán hướng cho các keypoint (Orientation Assignment)
* Bước 6: Tạo các descriptor mô tả cho các keypoint (Keypoint Descriptor)
  + 1. **Chi tiết các bước thực hiện**

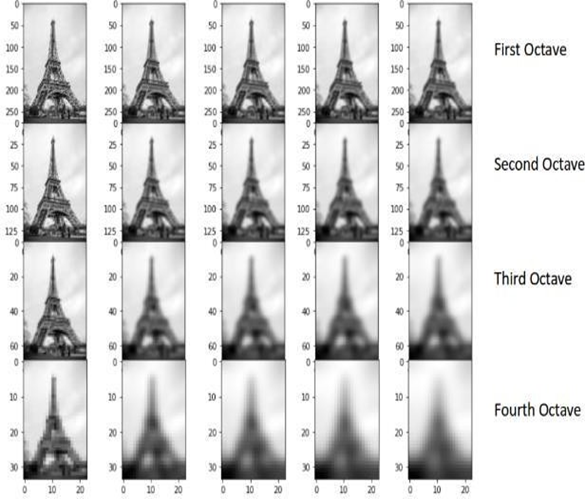
#### Bước 1: Xây dựng không gian mờ: Đảm bảo tính bất biến theo tỉ lệ

* Ý tưởng: Nếu muốn nhìn vào cái cây trong bức ảnh -> làm mờ các chi tiết lá, cành nhỏ… một cách có chủ đích -> Sử dụng Gaussian Blur. Kỹ thuật Gausian Blur dùng để: làm mờ ảnh, làm nổi bật các chi tiết của ảnh.
* Để tạo ra một không gian mờ, ta sẽ lấy ảnh ban đầu và tạo ra các hình ảnh với mức độ mờ dần (Khi làm mờ thì ảnh sẽ mất đi các chi tiết nhỏ vd: như râu,tóc,…).



* SIFT tạo ra các không gian mờ ở mức tiếp theo. Ta lấy ban đầu → tạo ra các hình ảnh mờ dần. Sau đó, thay đổi kích thước ảnh bằng một nửa kích thước ảnh ban đầu → Tạo ra các ảnh mờ dần lần nữa. Và lặp lại quá trình trên ta được các không gian tỉ lệ chứa các ảnh ở các tỉ lệ khác nhau được làm mờ với các mức độ khác nhau.





* Các hình ảnh trong cùng một mức (Octave) có cùng kích thước.
* Hình ảnh bên trên, ta có 4 mức (octave). Tại mỗi mức có 5 ảnh, các hình ảnh được hình thành bằng việc tăng “scale” ( tăng mức độ mờ).

***Kỹ thuật chi tiết***

* ***Octaves and Scales:***

Số lượng các mức (Octave) và số lần làm mờ trong một mức (scale) phụ thuộc vào kích thước ảnh ban đầu. Ta phải quyết định có bao nhiêu octave và scale. Tuy nhiên, người tạo ra giải thuật SIFT gợi ý rằng 4 octave và 5 blur level là lý tưởng cho thuật toán.

* Octave đầu tiên:

Nếu ảnh gốc được gấp đôi kích thước và được khử răng cưa(bằng cách làm mờ nó) thì thuật toán sẽ tạo ra nhiều keypoint hơn gấp 4 lần. Và càng nhiều keypoint thì càng tốt

* Làm mờ:

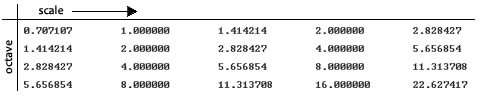
Ta “làm mờ” ảnh bằng cách: thực hiện tích chập của toán tử gaussian và hình ảnh áp dụng cho từng pixel:

Trong đó:

* L: hình ảnh sau khi được làm mờ
* G: toán tử Gaussian
* I: hình ảnh gốc
* (x, y): vị trí tọa độ các điểm ảnh
* σ : tham số mờ (mức độ làm mờ)
* : toán tử tích chập
* **Toán tử Gaussian:**

Lượng mờ:

Giả sử lượng mờ của một hình ảnh là σ. Sau đó, lượng mờ của hình ảnh ở scale tiếp theo sẽ là: k\*σ (k là hằng số mà ta chọn).

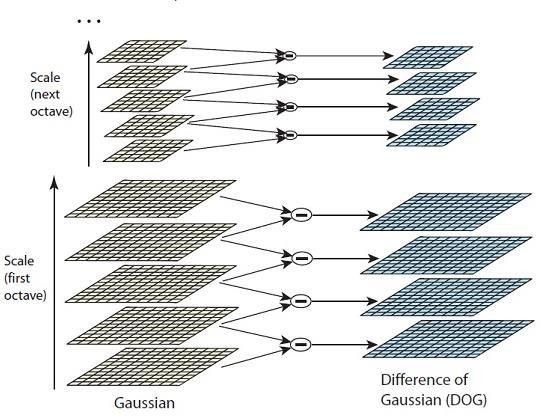
Ví dụ với k= √2 :

→ **Kết luận**: Trong bước đầu tiên của SIFT ta đã:

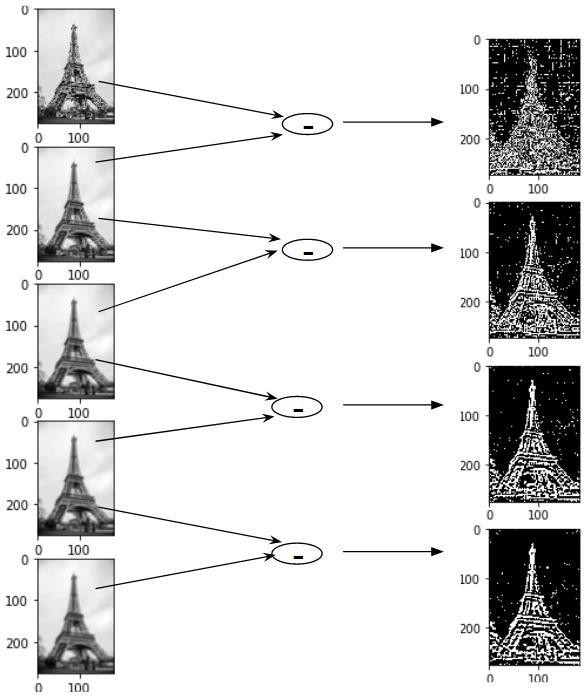
* Sinh ra nhiều mức (octave): kích thước hình ảnh trong mỗi mức sau bằng ½ kích thước hình ảnh của mức trước đó.
* Trong một mức, hình ảnh được làm mờ dần bằng toán tử Gaussian Blur.

#### Bước 2: Dò tìm các cực trị cục bộ (Scale-space extrema detection): Different Of Gausian (DoG)

* + Ý tưởng: Từ những tập hình ảnh được làm mờ → Ta sinh ra một tập các hình ảnh khác (sử dụng DoG) → mục đích là để tìm ra các điểm cực trị cục bộ trong ảnh. Kĩ thuật DoG làm nổi bật lên các cạnh và góc trên một hình ảnh → Các cạnh và góc này rất có ích để tìm ra các điểm cực trị cục bộ hay các keypoint của ảnh.
  + Thực hiện:
  + Sử dụng các không gian mờ ở ***Bước 1***. Ta tính sự khác nhau giữa 2 hình ảnh với mức độ mờ liên tiếp trong cùng một mức (octave).



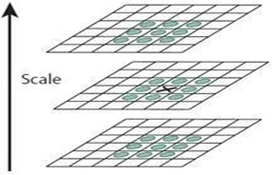
* Ví dụ:



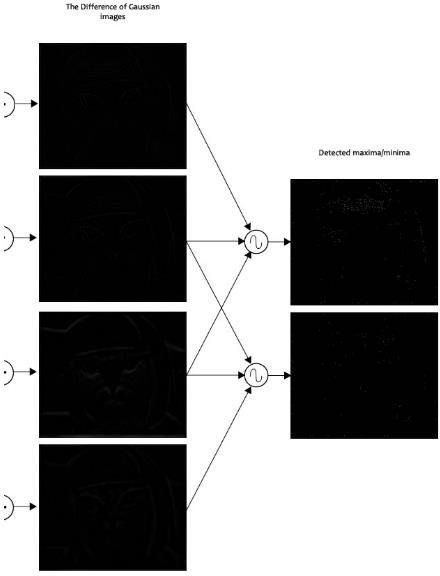
* + Trong hình ảnh trên, ta thực hiện phép trừ 2 hình ảnh liên tiếp với nhau trong cùng một mức (octave) . Quá trình trên sẽ thực hiện tương tự cho tất cả các mức → Ta sẽ sinh ra được các hình ảnh DoG ở các kích thước khác nhau.

#### Bước 3: Xác định các keypoint (Keypoint Localization)

* Ta sẽ tìm các keypoint tiềm năng bằng việc xác định vị trí các cực đại cực đại, cực tiểu trên các ảnh DoG.
* Ta sẽ lặp qua từng pixel trong bức ảnh và kiểm tra với tất cả các láng giềng của nó. Việc kiểm tra thực hiện trong hình ảnh hiện tại cũng như hình ảnh bên trên và bên dưới nó trong cùng một octave.



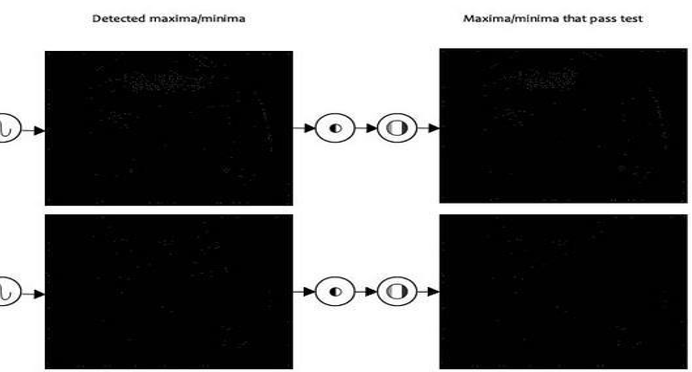
Ví dụ: X là pixel được xét. O là các láng giềng của X → Có 8 láng giềng

* O trong hình ảnh chứa X và 18 láng giềng O trong 2 hình ảnh ở 2 scale bên trên và dưới. **X được đánh dấu là keypoint nếu X là lớn nhất hoặc nhỏ nhất trong 26 láng giềng của nó. (Các keypoint sẽ không được phát hiện ở scale thấp nhất và cao nhất trong một octave)**.
* Ví dụ:
* 
* Ví dụ trên ta chỉ minh họa việc thực hiện trong một mức (octave). Việc thực hiện sẽ làm tương tự trong tất cả các octave khác.

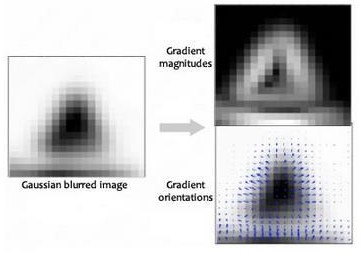
#### Bước 4: Loại bỏ các keypoint kém

* Trong các keypoint tiềm năng được sinh ra từ bước trước, có một số nằm trên cạnh, hoặc là chúng không có đủ độ tương phản. Trong cả 2 trường hợp, chúng không phải là những keypoint hữu ích. Ta cần loại bỏ chúng.
* Với các đặc trưng có độ tương phản kém, chúng ta đơn giản chỉ cần kiểm tra cường độ của chúng.
* Với các đặc trưng nằm trên cạnh, ta sẽ sử dụng cách tiếp cận tương tự Harris Corner Detector để loại bỏ những đặc trưng này.
* Loại bỏ những đặc trưng tương phản thấp: Nếu cường độ tại pixel hiện tại (pixel đã được kiểm tra là cực trị) trong hình ảnh DoG nhỏ hơn một giá trị nhất định. nó sẽ bị loại bỏ.
* Loại bỏ cạnh: Ý tưởng là tính 2 gradient tại điểm keypoint (cả 2 vuông góc với nhau). Dựa trên vùng ảnh xung quanh keypoint có 3 khả năng. Vùng ảnh xung quanh điểm chính có thể là:
* Một vùng phẳng (A flat region): Trong trường hợp này 2 gradient sẽ đều nhỏ
* Một cạnh (An edge): Một gradient lớn(vuông góc với cạnh) và một gradient nhỏ(song song với cạnh)
* Một góc: (A corner): Cả 2 gradient đều lớn.

Trong các trường hợp trên, ta chỉ lấy những keypoint tiềm năng nằm trên góc. Nếu cả hai gradient đủ lớn, ta sẽ coi nó như keypoint chính thức. Ngược lại thì loại bỏ. Ví dụ:

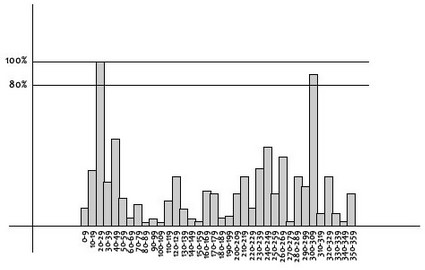


#### Bước 5: Gán hướng cho các keypoint (Orientation Assignment)

* Ý tưởng: Thu thập các hướng và độ lớn gradient xung quanh mỗi điểm keypoint. Sau đó tìm ra các hướng nổi trội nhất trong vùng đó. Và ta sẽ gán các hướng này cho keypoint.
* Những tính toán sau này sẽ tương quan với hướng chính của keypoint. Điều này sẽ đảm bảo tính bất biến khi xoay.
* Kích thước của “vùng thu thập hướng” xung quanh keypoint phụ thuộc vào “scale” của nó (mức độ mờ của ảnh trong octave). Scale càng lớn thì vùng thu thập càng lớn.
* Triển khai chi tiết:
  + Độ lớn và hướng gradient được tính toán dựa trên công thức sau:
* Độ lớn và hướng gradient được tính cho tất cả các pixel xung quanh keypoint. Từ đó ta xây dựng lên một lược đồ (histogram).
* Trong histogram này, 360 độ về hướng sẽ được chia thành 36 bin (mỗi bin 10 độ).
* Giả sử hướng gradient tại một điểm (trong vùng thu thập hướng của keypoint) là 18.75 độ → Nó thuộc bin: [10 → 20).
* Và ta sẽ thêm vào bin này một lượng tỉ lệ thuận với độ lớn của gradient tại điểm đó.

=> Ta sẽ làm tương tự với tất cả các pixel xung quanh keypoint. Khi đó ta sẽ có một lược đồ đầy đủ và sẽ tồn tại một bin có giá trị lớn nhất trong 36 bin.

* Ví dụ:



- Trong lược đồ trên, bin có giá trị lớn nhất là bin [20,30)

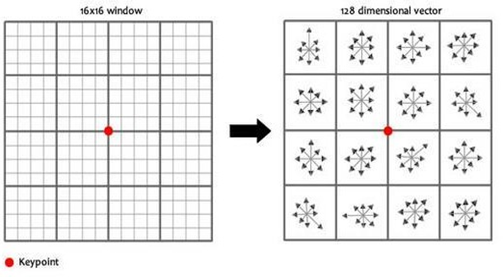
=> keypoint được gán hướng là 3 (Tương ứng với bin thứ 3)

- Ngoài ra bất kì bin nào có giá trị lớn hơn 80% giá trị của bin lớn nhất ta cũng sẽ tạo một keypoint mới cùng vị trí với keypoint cũ và hướng tương ứng với hướng của bin đó.

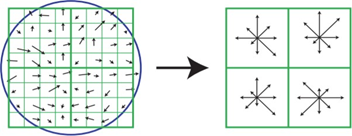
=> Việc gán hướng như trên có thể chia một keypoint thành nhiều keypoint với các hướng khác nhau.

#### Bước 6: Tạo các descriptor mô tả cho các keypoint (Keypoint Descriptor)

* Ý tưởng: Ta muốn sinh ra một mô tả duy nhất cho mỗi keypoint. Mỗi keypoint sẽ được định danh duy nhất bởi mô tả này.
* Cách thực hiện:
  + Xác định cửa sổ 16x16 xung quanh keypoint. Cửa sổ 16x16 này chia thành 16 cửa sổ 4x4 nhỏ hơn.



* Với mỗi cửa sổ 4x4. Độ lớn và hướng gradient được tính toán cho mỗi pixel trong cửa sổ



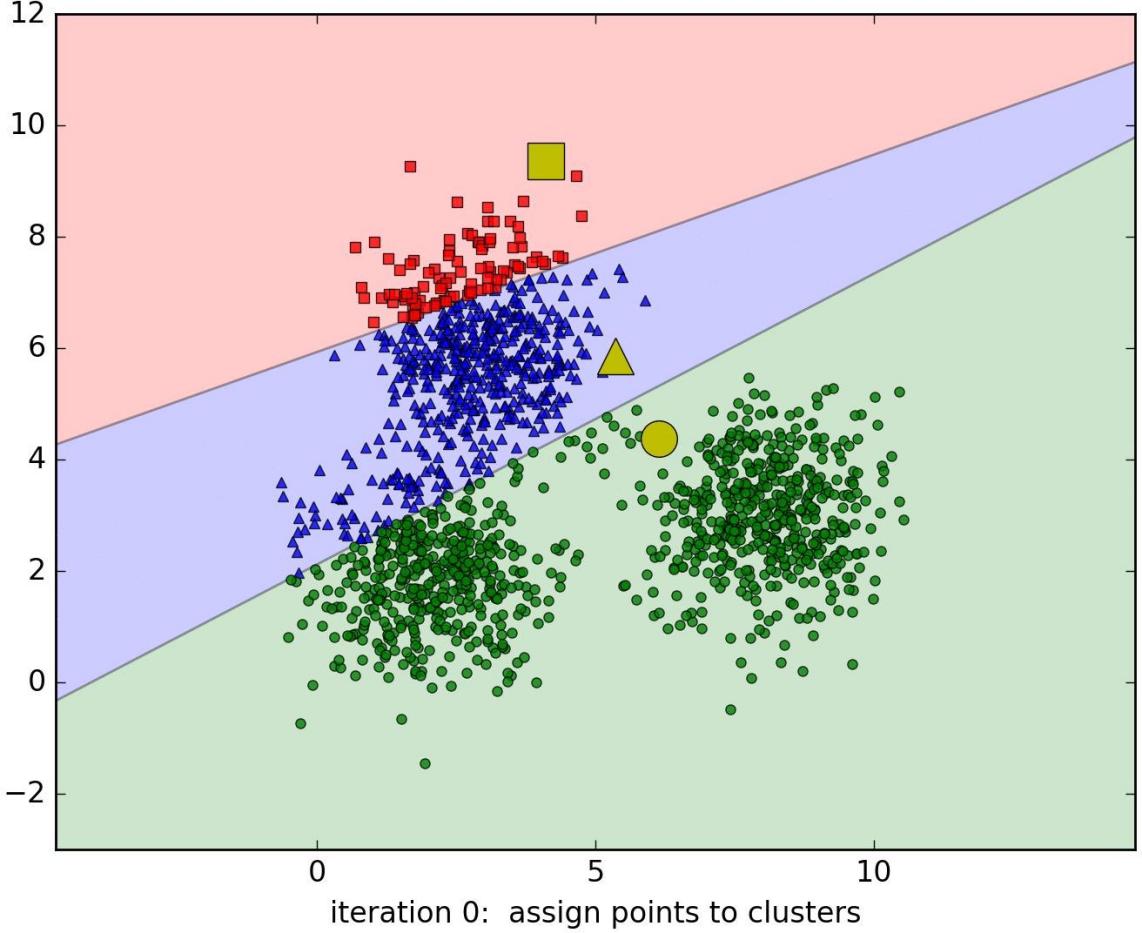
* Ta tạo một histogram gồm 8 bin và phân các hướng trên vào các bin tương ứng. (360 độ về hướng sẽ được chia làm 8 khoảng 0→45, 45→90,…tương ứng với các bin. Các hướng trong khoảng [0,45) sẽ nằm ở bin thứ nhất, [45, 90) nằm ở bin thứ hai,….)
* Độ lớn được thêm vào mỗi bin phụ thuộc và độ lớn gradient, và khoảng cách tới keypoint (điều này được thực hiện bằng hàm trọng số gaussian).

=> Làm điều này với tất cả 16 pixel trong ô 4x4 → Ta được một histogram 8 bins. Và làm tương tự với 16 cửa sổ 4x4. Ta được 8x16 = 128 số đại diện mô tả cho pixel.

* Từ 128 số này, ta hình thành nên một vector 128 chiều → Chuẩn hóa bằng cách chia mỗi phần tử cho căn của tổng bình phương 128 phần tử. (Để đạt được bất biến khi xoay, ta cần chuẩn hóa vector đặc trưng này dựa vào hướng chính của keypoint ta tìm được ở bước trước. Có thể chuẩn hóa về cùng một hướng là 0 độ). Ta sẽ được một vector 128 chiều là vector đặc trưng mô tả cho keypoint. Mỗi keypoint sẽ được xác định duy nhất bởi vector đặc trưng này.

### 2.2. Phân cụm sử dụng thuật toán Kmean

* Ý tưởng: Ý tưởng của k-mean là tập hợp các điểm ở gần nhau ở trong một khoảng không gian nào đó (không gian này có thể rất nhiều chiều, trong trường hợp của bài toán là 128 chiều, vì SIFT trích xuất được các vector đặc trưng 128 chiều). (Một cách đơn giản nhất, xét một điểm bất kỳ, ta xét xem điểm đó gần với center nào nhất thì nó thuộc về cùng nhóm với center đó.)

Ví dụ:

Giả sử mỗi cluster có một điểm đại diện (center) màu vàng. Và những điểm xung quanh mỗi center thuộc cùng nhóm với center đó. Một cách đơn giản nhất, xét một điểm bất kỳ, ta xét xem điểm đó gần với center nào nhất thì nó thuộc về cùng nhóm với center đó.

* Thuật toán:

**Input**: Dữ liệu X và số lượng cluster cần tìm K

**Output**: Các center M và label vector cho từng điểm dữ liệu Y

* Bước 1. Chọn K điểm bất kỳ làm các center ban đầu
* Bước 2. Phân mỗi điểm dữ liệu vào cluster có center gần nó nhất
* Bước 3. Nếu việc gán dữ liệu vào từng cluster ở Bước 2 không thay đổi so với vòng lặp trước nó thì ta dừng thuật toán.
* Bước 4. Cập nhật center cho từng cluster bằng cách lấy trung bình cộng của tất cả các điểm dữ liệu đã được gán vào cluster đó sau Bước 2.
* Bước 5. Quay lại Bước 2.
* **Hạn chế:**
  + Trọng thuật toán k-mean, cần phải xác định trước được đại lượng K là số lượng clusters.
  + Tùy thuộc vào các center ban đầu mà thuật toán có thể có tốc độ hội tụ rất chậm

### 2.3. Tạo vector mô tả ảnh duy nhất ( Bag of Features phát triển dựa trên Bag of Word)

* Mục tiêu của bước này từ các đặc trưng cục bộ của ảnh, tạo một vector mô tả toàn cục cả bức ảnh, đồng thời có thể so sánh các bức ảnh bằng vector mô tả toàn cục ảnh đó

**Cách thực hiện:**

* + Mỗi ảnh qua SIFT ta thu được m vector đặc trưng cục bộ (số lượng m là khác nhau với từng ảnh)
  + Tạo một mảng k phần tử tương ứng với k cụm => Đại diện cho vector toàn cục mô tả ảnh, khởi tạo giá trị các phần tử bằng 0
  + Phân m vector đặc trưng vào k cụm, mỗi vector được thêm vào cụm i ta tăng giá trị phần tử mảng thứ i lên 1.
  + Chia mỗi phần tử trong mảng cho m => Ta được vector k phần tử với tổng các phần tử bằng 1 => Là vector đại diện cho ảnh (mô tả ảnh)

### 2.4. So sánh đối chiếu ảnh

* Sau các bước trên, mỗi ảnh ta đều có thể cho ra được một vector đặc trưng k chiều mô tả ảnh. Với một ảnh đầu vào, ta có thể so sánh trực tiếp với các ảnh trong cơ sở dữ liệu dựa vào các phép đo khoảng cách để tìm ra ảnh giống nhất.
* Tuy nhiên, ta sẽ làm thêm một bước nữa, thay vì tìm một ảnh giống nhất ta sẽ tìm k (k này khác với k trong kmean ở bước trước) ảnh giống nhất trong cơ sở dữ liệu và dựa vào nhãn đa số của k ảnh này, ta sẽ dự đoán được loại con vật của ảnh đầu vào. Ta sẽ sử dụng giải thuật k-Nearest Neighbours để thực hiện kĩ thuật này:
* Với KNN, label của một điểm dữ liệu mới được suy ra trực tiếp từ K điểm dữ liệu gần nhất trong training set. Label của một test data có thể được quyết định bằng major voting (bầu chọn theo số phiếu) giữa các điểm gần nhất, hoặc nó có thể được suy ra bằng cách đánh trọng số khác nhau cho mỗi trong các điểm gần nhất đó rồi suy ra label.
* KNN là thuật toán đi tìm đầu ra của một điểm dữ liệu mới bằng cách chỉ dựa trên thông tin của K điểm dữ liệu trong training set gần nó nhất (K-láng giềng)
* Cách thực hiện:
  + Bước 1: Ta có D là tập các điểm dữ liệu đã được gắn nhãn và A là dữ liệu chưa được phân loại.
  + Bước 2: Đo khoảng cách (Euclidian, Manhattan, ...) từ dữ liệu mới A đến tất cả các dữ liệu khác đã được phân loại trong D.
  + Bước 3: Chọn K (K là tham số mà bạn định nghĩa) khoảng cách nhỏ nhất.
  + Bước 4: Kiểm tra danh sách các lớp có khoảng cách ngắn nhất và đếm số lượng của mỗi lớp xuất hiện.
  + Bước 5: Lấy đúng lớp (lớp xuất hiện nhiều lần nhất).
  + Bước 6: Lớp của dữ liệu mới là lớp mà bạn đã nhận được ở Bước 5.’
* Phép đo khoảng cách: Tổng quát:
* **Nhận xét**
* Ưu điểm KNN:
* Độ phức tạp tính toán của quá trình training là bằng 0.
* Việc dự đoán kết quả của dữ liệu mới rất đơn giản.
* Nhược điểm của KNN:
* KNN rất nhạy cảm với nhiễu khi K nhỏ.
* Như đã nói, KNN là một thuật toán mà mọi tính toán đều nằm ở khâu test. Trong đó việc tính khoảng cách tới từng điểm dữ liệu trong training set sẽ tốn rất nhiều thời gian, đặc biệt là với các cơ sở dữ liệu có số chiều lớn và có nhiều điểm dữ liệu. Với K càng lớn thì độ phức tạp cũng sẽ tăng lên. Ngoài ra, việc lưu toàn bộ dữ liệu trong bộ nhớ cũng ảnh hưởng tới hiệu năng của KNN.

## 3. Cách lưu trữ thuộc tính nhận dạng ảnh

- Các ảnh sẽ được lưu vào trong đối tượng dict ={key:value, …}

- Mỗi ảnh là một phần tử được lưu trữ dưới dạng key:value

- Ứng với key là đường dẫn đến file ảnh lên đó

- Value là dạng tuple với phần tử đầu tiên là 1 vector thuộc tính 10 chiều và phần tử thứ 2 là nhãn của ảnh đó

# Chương 4

# Triển khai hệ thống và đánh giá kết quả

Trong chương 4, báo cáo sẽ trình bày cách triển khai hệ thống sử dụng công cụ lập trình Visual Studio Code và ngôn ngữ lập trình python. Sau khi đã triển khai hệ thống và đưa ra được kết quả demo, hệ thống sẽ đưa ra kết luận và đánh giá kết quả đạt được.

## Triển khai cài đặt hệ thống

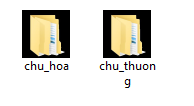
### Xậy dựng bộ dữ liệu

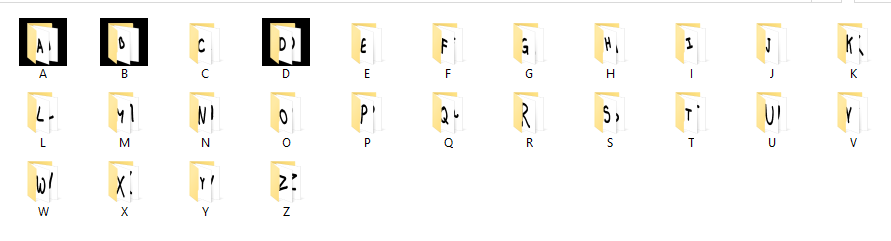
* Bộ dữ liệu nguồn

* + - Tên bộ dữ liệu: Handwritten\_character
    - Mô tả: Bộ dữ liệu gồm 2600 ảnh chia ra 52 folder (26 chữ cái in thường, 26 chữ cái in hoa), mỗi folder đều có số ảnh bằng nhau: 50 ảnh, các bức ảnh đều có cùng kích thước là 1200 x 900 pixel.

* Dữ liệu sử dụng kiểm tra bộ phân loại:

Bộ dữ liệu tự thu thập gồm các loại là các chữ cái Latin: a,b,c,…A,B,C,…



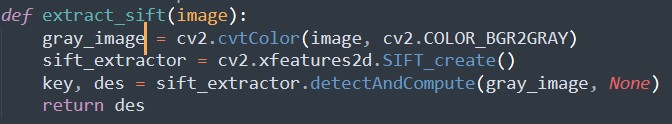




### Triển khai (mô tả một số hàm cơ bản bằng code)

**Bước 1: Tạo một bộ trích trọn đặc trưng SIFT sử dụng thư viện Opencv:**

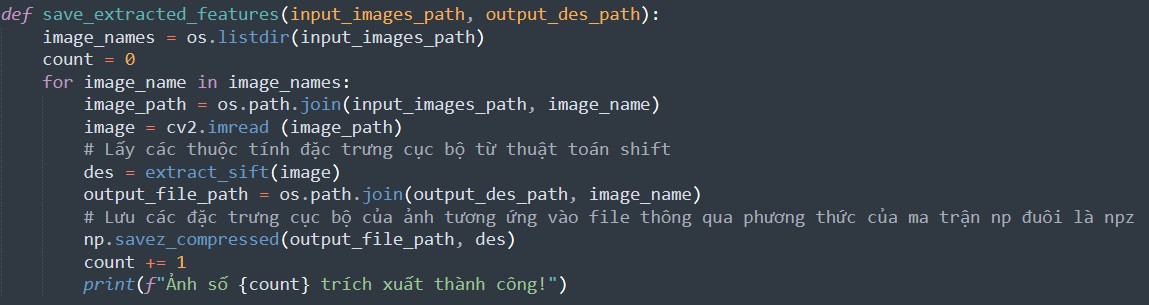
A, Sau khi chuyển đổi ảnh màu sang ảnh xám. Ta tạo một bộ một bộ trích trọn đặc trưng SIFT, và gọi đến hàm detectAndCompute với tham số truyền vào là ảnh xám vừa được chuyển đổi, và mặt nạ (mask) để None. Kết quả ta thu được một tuple gồm 2 phần tử key, des tương ứng với mảng các keypoint và mảng các vector đặc trưng mô tả cáckeypoint đó.



**Input**: Ma trận ảnh đầu vào

**Output**: Mảng các vector đặc trưng mô tả các keypoint tìm được

B, Với mỗi ảnh trong tập dữ liệu Animals, ta trích xuất đặc trưng SIFT cho mỗi ảnh. Lưu các đặc trưng trích xuất được vào file.



**Input**: Input\_images\_path: đường dẫn thư mục ảnh Animals10

**Output**: Output\_des\_path: đường dẫn thư mục lưu các file chứa các đặc trưng được trích xuất (file “.npz”).

**Bước 2: Phân cụm sử dụng thuật toán Kmean**

A, Chuyển đổi các file (“.npz”) lưu các vector đặc trưng thành (“.parquet”) để có thể đổ vào sử dụng Kmean trong thư viện pyspark. (đọc các file parquet dưới dạng Dataframe).

1. Tạo một SparkSession: với một số cấu hình:

“appName”: tên ứng dụng, xuất hiện trên UI hoặc trong dữ liệu log “memory”: Lượng bộ nhớ sử dụng (vd 512m, 4g,…)

“warehouse”: Kho lưu trữ dữ liệu

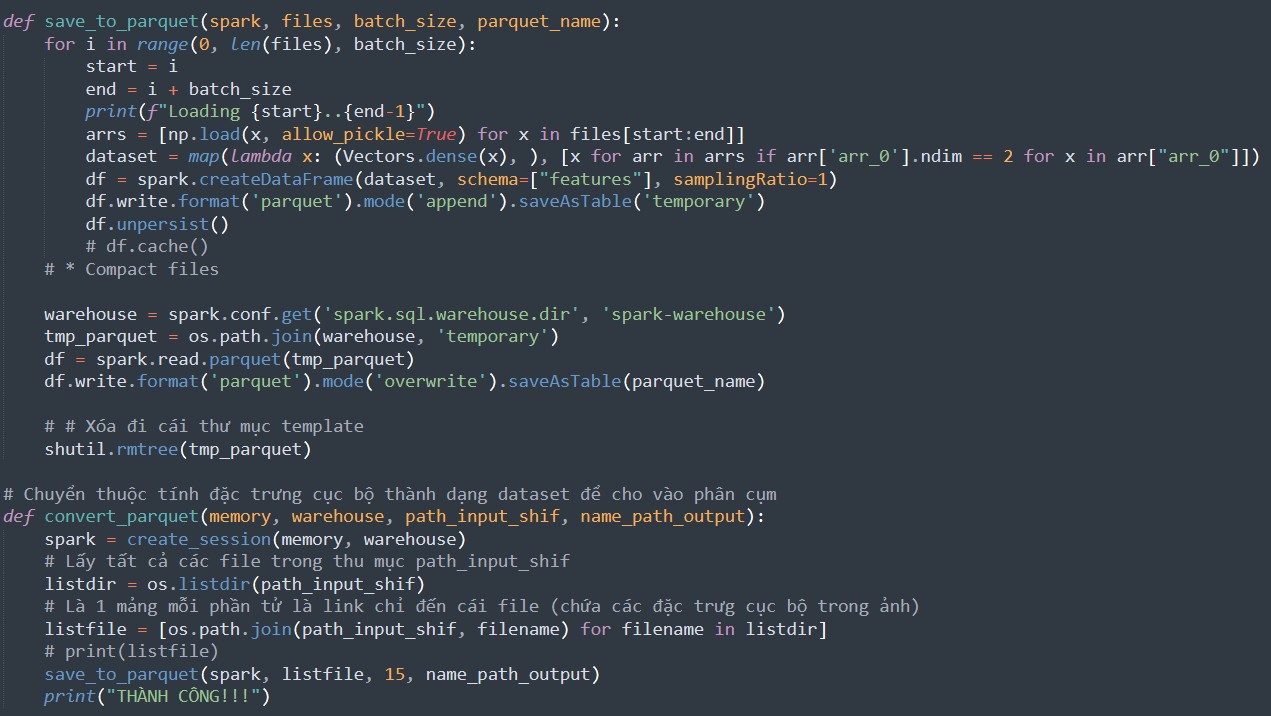


**Input**:

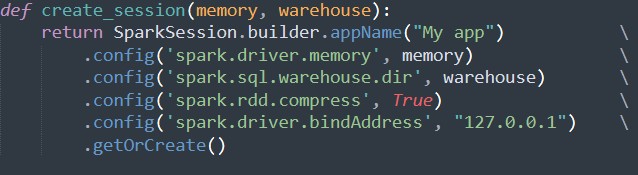
* memory: Lượng bộ nhớ sử dụng
* warehouse: Nơi lưu trữ dữ liệu “.parquet” đầu ra

**Output**: Một instance của SparkSession

1. Chuyển đổi các file “.npz” thành “.parquet”



* Hàm convert\_parquet
  + memory: Lượng bộ nhớ sử dụng
  + warehouse: Nơi lưu trữ dữ liệu “.parquet” đầu ra
  + path\_input: Thư mục lưu dữ liệu các file “.npz” lưu các vector đặc trưng trích xuất được từ các ảnh
  + name\_path\_ouput: Tên thư mục lưu các file “.parquet” sẽ được lưu trong warehouse.
* Hàm save\_to\_parquet
  + spark: Một instance của SparkSession
  + files: List các đường dẫn file “.npz”
  + batch\_size: Số lượng các file “.npz” được gộp lại để lưu thành 1 file “.parquet” trong một lần duyệt.
  + parquet\_name: Tên thư mục lưu các file parquet được lưu trong warehouse.

B, Phân cụm các vector đặc trưng sử dụng Kmean trong thư viện pyspark

* Hàm create\_session: Tạo một instance của SparkSession

**Input:**

* memory: Lượng bộ nhớ sử dụng
* warehouse: Nơi lưu trữ dữ liệu “.parquet” đầu ra

**Output**: Một instance của SparkSession



* Hàm cluster:

**Input:**

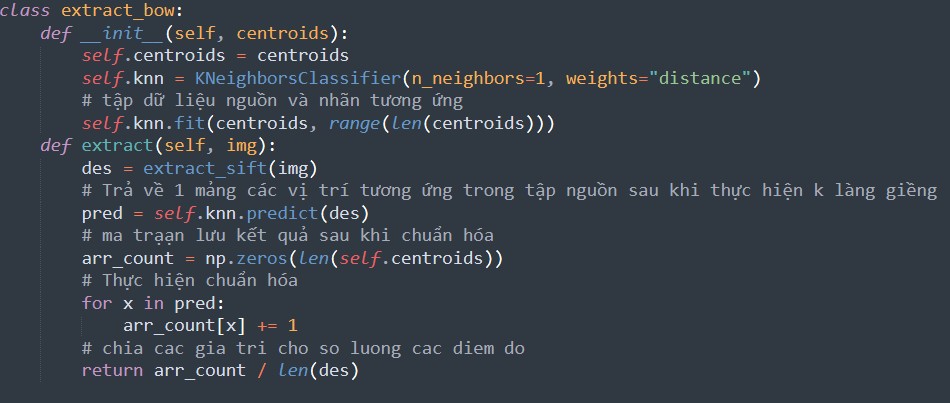
* + dataframe: Một DataFrame lưu dữ liệu của các vector đặc trưng
  + k: Số cụm cần phân chia
  + maxIter: Giới hạn số vòng lặp tối đa thực hiện nếu việc phân cụm hội tụ lâu.

**Output**: Một mảng chứa k phần tử, mỗi phần tử đại diện cho một tâm cụm

* Hàm cal\_center: Thực hiện việc khởi tạo SparkSession, tính toán tâm cụm, và lưu các tâm cụm vào file.
  + memory: Lượng bộ nhớ sử dụng
  + warehouse: Nơi lưu trữ dữ liệu “.parquet” đầu ra
  + name\_path\_parquet: tên của thư mục chứa các file “.parquet”
  + path\_centroidfile: tên của file lưu các tâm cụm
  + ncluster: số cụm

**Bước 3: Tạo một bộ trích xuất đặc trưng duy nhất cho mỗi ảnh**

A, Sử dụng kĩ thuật Bag Of Words từ tâm các cụm đã biết và sử dụng KNeigborsClassifier trong trong thư viện sklearn.



* Tạo một class extract\_bow với hàm khởi tạo nhận tham số truyền vào là:

- centroids: mảng chứa các tâm cụm tìm được sau bước Kmean

=> Tạo một bộ phân loại Knn sử dụng KNeighborsClassifier trong thư viện Sklearn.

* Định nghĩa hàm extract:

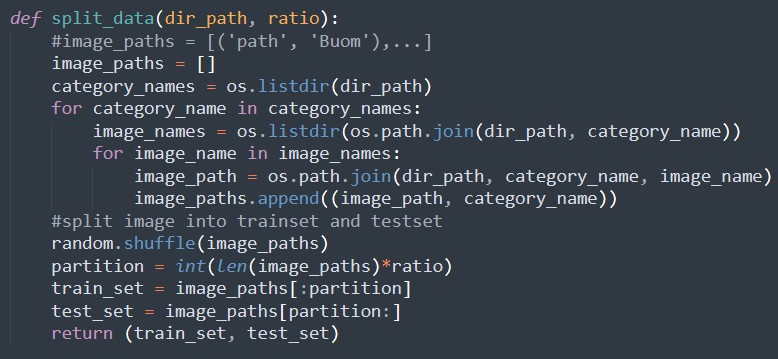
**Input**: img: ma trận ảnh đầu vào

**Output**: một vector đặc trưng duy nhất mô tả ảnh

Thực hiện bằng cách: Khởi tạo một mảng gồm k phần tử tương ứng với k cụm, mỗi vector đặc trưng được trích xuất trong ảnh sẽ được so sánh gần với tâm cụm nào nhất, thì tăng giá trị phần tử tương ứng với cụm đó lên 1,sau cùng

chia mỗi phần tử trong mảng cho tổng số đặc trưng trích xuất được trong ảnh → Ta được vector đặc trưng duy nhất mô tả ảnh).

B, Chia tập dữ liệu tự thu thập thành hai phần: trainset và testset theo một tỉ lệ đặt trước:



**Input:**

* + dir\_path: Đường dẫn thư mục chứa tập ảnh thu thập
  + ratio: tỉ lệ chia bộ trainset, testset: (vid ratio=0.8 → 80% trainset,

20% testset)

**Output**: Một tuple gồm 2 phần tử tương ứng của tập train\_set và testset. Mỗi phần tử là một mảng của các tuple (đường dẫn ảnh, nhãn ảnh).

C, Với mỗi ảnh trong trainset: tìm encode ( vector đặc trưng mô tả ảnh sau BoW) và label của ảnh → Lưu vào file (csdl)



* Hàm extract\_encode\_image:

**Input:**

* + img\_path: Đường dẫn ảnh đầu vào
  + centroids: Mảng chứa các tâm cụm

**Output**: img\_encode: Một vector đặc trưng mô tả ảnh

* Hàm cal\_encodes\_trainningset:

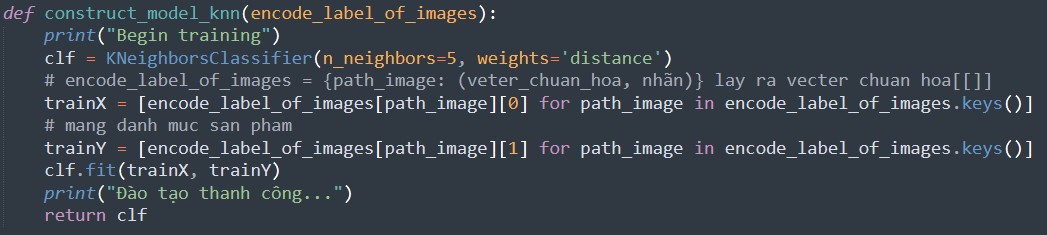
**Input:**

* + train\_set: Mảng chứa các phần tử (đường dẫn ảnh, nhãn) của các ảnh trong tập train
  + centroids: Mảng chứa các tâm cụm

**Output**: encode\_label\_of\_images: Một dictionary(dict) chứa các (key: value) tương ứng là: img\_path : (image\_encode, category)

**Bước 4: So sánh đối chiếu ảnh sử dụng thuật toán KNN**

A, Xây dựng model KNN



**Input:**

* + encode\_label\_of\_images: Một dictionary(dict) chứa các (key: value) tương ứng là: img\_path : (image\_encode, category)

**Output:**

* + clf: Bộ phân loại (instance của KneighborsClassifier) với dữ liệu train là vector đặc trưng chuẩn hóa tương ứng với các nhãn loài vật của ảnh đó trong tập các ảnh train

B, Tìm k ảnh giống nhất từ đàu vào



**Input:**

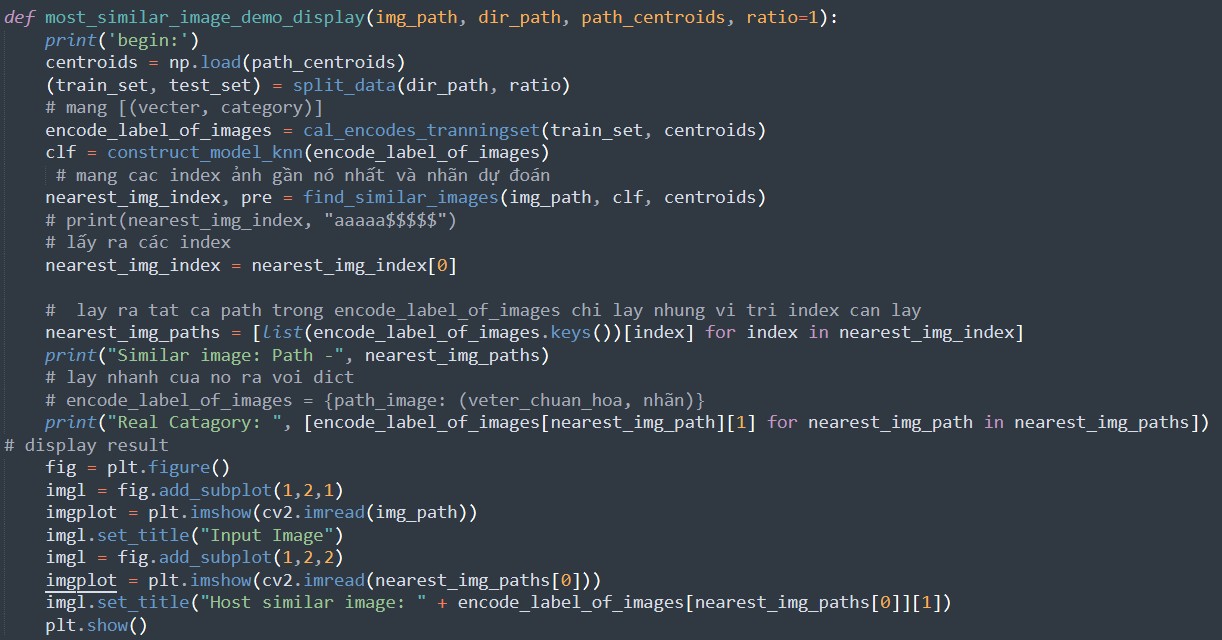
* + img\_path: Đường dẫn ảnh đầu vào
  + clf: Bộ phân loại (instance của KneighborsClassifier) với dữ liệu train là vector đặc trưng chuẩn hóa tương ứng với các nhãn loài vật của ảnh đó trong tập các ảnh train.
  + centroids: Mảng chứa các tâm cụm

**Output:**

* + (nearest\_img\_index, pre): Một tuple chứa 2 phần tử. phần tử thứ nhất là mảng chỉ số của các ảnh giống ảnh đầu vào nhất trong tập trainning, phần tử thứ hai là nhãn dự đoán của hình ảnh.

C, Demo bộ phân loại

Với một ảnh động vật bất kỳ, ta sẽ tìm ra ảnh giống nhất với ảnh đầu vào trong cơ sở dữ liệu.



**Input:**

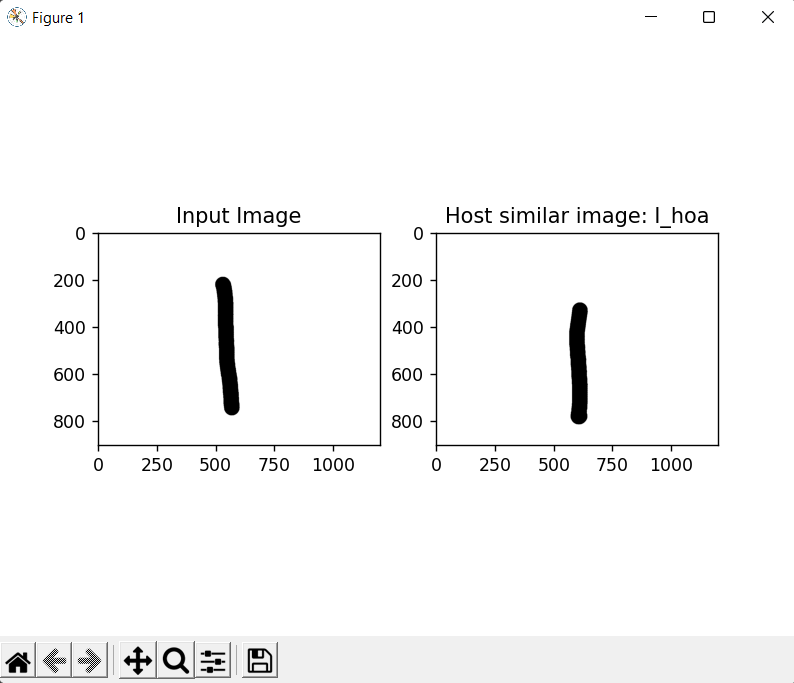
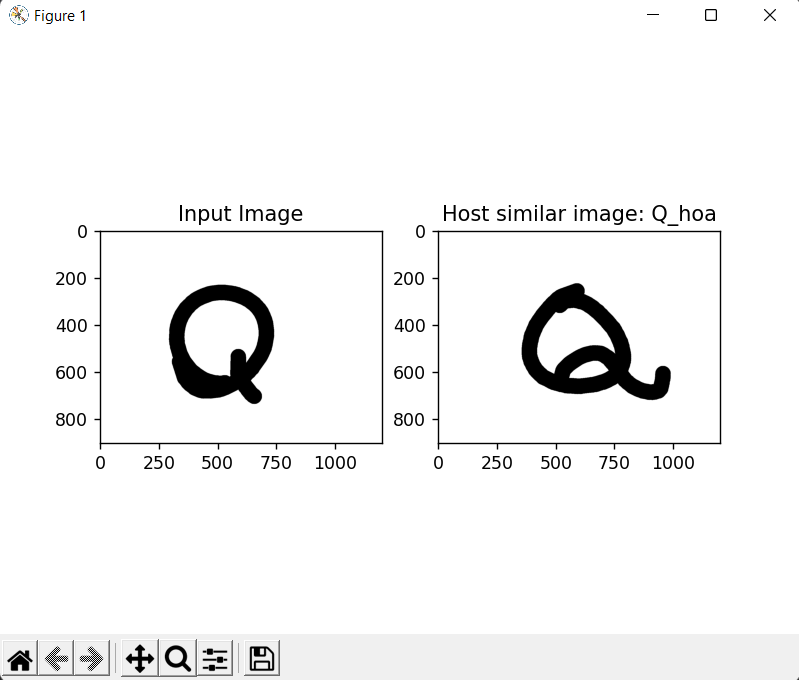
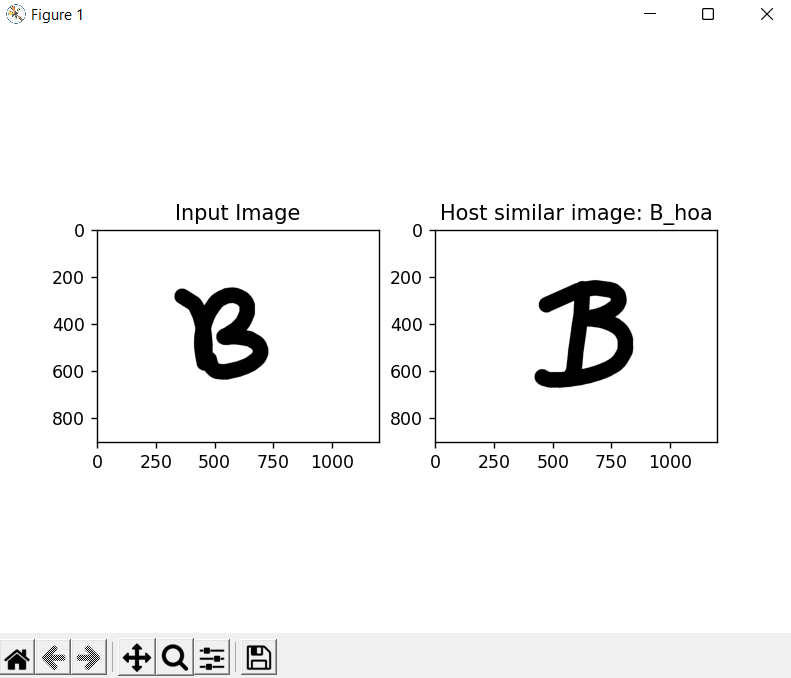
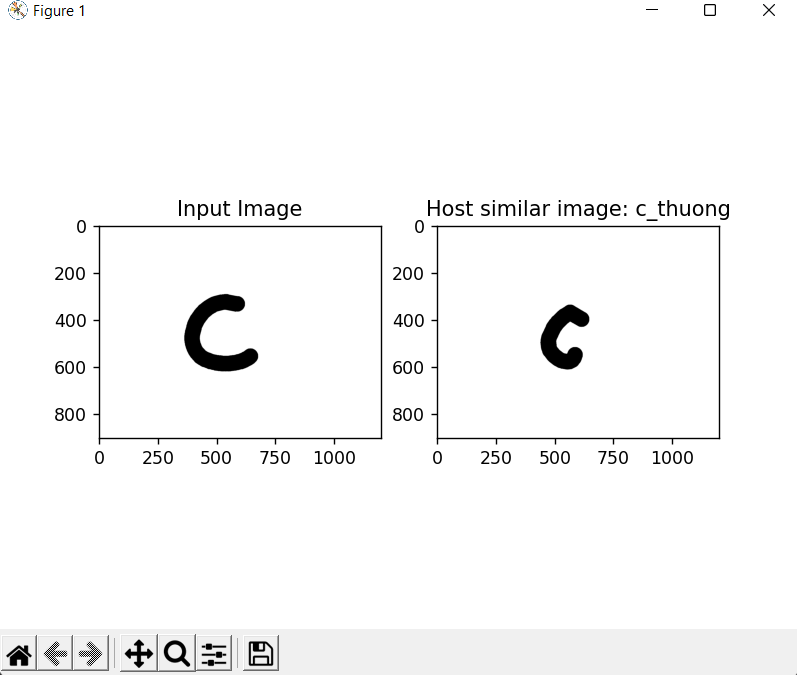
* + img\_path: Đường dẫn ảnh đầu vào, ảnh test
  + dir\_path: Đường dẫn bộ dữ liệu thu thập (tập các ảnh trong CSDL)
  + path\_centroids: Đường dẫn file lưu tâm cụm
  + ratio: Tỉ lệ phân chia trainset, testset (Trong trường hợp này không cần thiết, ta sẽ coi toàn bộ dữ liệu thu thập là trainset → đặt ratio

mặc đinh bằng 1)

**Output:**

* + Kết quả ảnh giống nhất và nhãn phân loại ảnh

### Kết quả thử nghiệm



## Đánh giá kết quả

* + Độ chính xác là 60%
  + Độ chính xác của chúng em vẫn chưa cao
  + Nhiều hình ảnh vẫn nhận sai nhãn ứng với con vật
  + Ưu điểm nếu ảnh mà xoay hoặc thu nhỏ phóng to thì bài chúng em vẫn nhận ra được.

# Chương 5

# Kết luận

Trong chương 5, chúng em sẽ đưa ra kết luận của mình thông qua quá trình tìm hiểu chủ đề, thuật toán và bài học, kinh nghiệm rút ra.

1. **Nhận xét**

Kết quả đạt được với SIFT tương đối tốt, nguyên nhân do hình ảnh chỉ chứa duy nhất đối tượng động vật. Và tận dụng được tính chất của các đặc trưng bất biến trong SIFT.

**2. Thu hoạch đạt được**

- Các thành viên biết thêm được thế nào là các đặc trưng của ảnh, các loại đặc trưng của ảnh, và làm thế nào để sử dụng các đặc trưng ảnh để nhận dạng và phân loại ảnh

- Hiểu biết thêm kĩ thuật sử dụng giải thuật Kmean trong bài toán phân cụm

- Kỹ thuật Bag of Words ứng dụng trong hình ảnh

- Hiểu thêm về ứng dụng của giải thuật K-Nearest Neighbors trong bài toán phân loại.

# TÀI LIỆU THAM KHẢO

1. David G. Lowe *“Distinctive Image Features from Scale-Invariant Keypoints ”*

<https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf>

1. Giải thích chi tiết về giải thuật SIFT:

[https://aishack.in/tutorials/sift-scale-invariant-feature-transform-](https://aishack.in/tutorials/sift-scale-invariant-feature-transform-%20eliminate-low-contrast/) [eliminate-low-contrast/](https://aishack.in/tutorials/sift-scale-invariant-feature-transform-%20eliminate-low-contrast/)

1. Tham khảo một trong những cách triển khai giải thuât SIFT bằng python<https://github.com/rmislam/PythonSIFT>
2. Tài liệu tham khảo về kĩ thuật Bag Of Words

<https://machinelearningcoban.com/general/2017/02/06/featureengineering>

/

1. Tài liệu tham khảo về Kmean<https://machinelearningcoban.com/2017/01/01/kmeans/>
2. Tài liệu tham khảo về K-nearest neighbors<https://machinelearningcoban.com/2017/01/08/knn/>