**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

---------------------------------------



**BÁO CÁO BÀI TẬP LỚN**

**HỆ CƠ SỞ DỮ LIỆU ĐA PHƯƠNG TIỆN**

**XÂY DỰNG HỆ CƠ SỞ DỮ LIỆU**

**NHẬN DẠNG CHỮ VIẾT TAY**

Giáo viên hướng dẫn: (Thầy) Nguyễn Đình Hóa

Sinh viên thực hiện: Nhóm 10

|  |  |
| --- | --- |
| Lê Văn Sang | B17DCCN530 |
| Nguyễn Phú Thịnh | B17DCCN578 |
| Trần Minh Ngọc | B17DCCN470 |
|  |  |
|  |  |
|  |  |
| **Hà Nội, tháng 4 năm 2021** | |

**MỤC LỤC**

[**I. Bộ dữ liệu ảnh** 3](#_Toc75085594)

[**II. Kĩ thuật xử lý và nhận dạng chữ viết tay** 3](#_Toc75085595)

[1. GaussianBlur 3](#_Toc75085596)

[2. Canny Edge Detection 3](#_Toc75085597)

[3. KNN – K-Nearest Neighbors 5](#_Toc75085598)

[4. Kỹ thuật trích rút đặc trưng HOG 6](#_Toc75085599)

[**III. Xây dựng hệ thống nhận dạng** 6](#_Toc75085600)

[a. Sơ đồ khối của hệ thống 6](#_Toc75085601)

[b. Trích chọn đặc trưng 7](#_Toc75085602)

[c. Lưu trữ 12](#_Toc75085603)

[**IV. Kết quả** 14](#_Toc75085604)

# **I. Bộ dữ liệu ảnh**

* Bộ dữ liệu ảnh về chữ cái viết tay của toàn bộ bảng chữ cái Latin
* Mỗi ảnh có kích thước 20x20

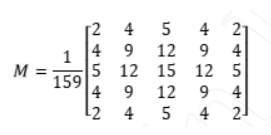


# **II. Kĩ thuật xử lý và nhận dạng chữ viết tay**

## GaussianBlur

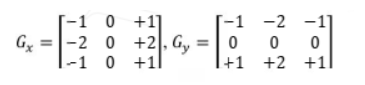
* Hiệu ứng làm mờ mang lại:
  + Lọc bớt nhiễu
  + Làm trơn ảnh (smooth). Việc làm trơn ảnh sẽ giảm sắc nét của cạnh, thay vào đó, vùng trơn sẽ lan ra

Ví dụ: Bộ lọc Gauss 5x5



## Canny Edge Detection

* Tìm biên ảnh với bộ lọc Canny
* Thực hiện qua 4 bước:
* Lọc bớt nhiễu: sử dụng bộ lọc Gauss
* Tính toán giá trị Gradient trong ảnh: Vì đường biên trong ảnh là nơi phân cách giữa các đối tượng khác nhau, nên tại đó gradient của nó sẽ có biến đổi mạnh mẽ nhất. Để tính toán gradient trong ảnh, sử dụng bộ lọc Sobel.





Trong đó, Gx và Gy chính là đạo hàm theo hướng x, y của ảnh đang xét. Góc θ sẽ được làm tròn theo các hướng thẳng đứng, nằm ngang và theo hướng chéo. nghĩa là nó sẽ được làm tròn để nhận các giá trị trong 0, 45, 90 và 135 độ.

* Loại bỏ các giá trị không phải cực đại:

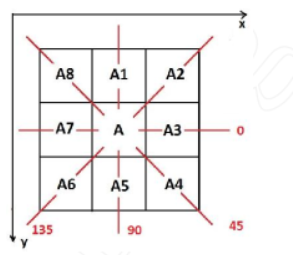
Với giá trị của góc θ ở trên thì biên của đối tượng có thể tuân theo bốn hướng, và ta có bốn khả năng sau:

Nếu θ = 0, khi đó điểm A sẽ được xem xét là một điêm trên biên độ nếu độ lớn gradient tại A lớn hơn gradient của các điểm tại A3, A7.

Nếu θ = 45, khi đó điểm A được xem là một điểm trên biên độ nếu độ lớn gradient tại A lớn hơn độ lớn gradient của các điểm tại A4, A8

Nếu θ = 90, khi đó điểm A sẽ được coi là một điểm nằm trên biên độ nếu độ lớn gradient tại A lớn hơn độ lớn gradient của các điểm tại A1, A5.

Nếu θ = 135, khi đó điểm A được xem là một điểm nằm trên biên độ nếu độ lớn gradient tại A lớn hơn độ lớn gradient của các điểm tại A2, A6



* Chọn biên

Vì những điểm có giá trị gradient lớn bao giờ cũng có xác suất biên thật sự hơn những điểm có gradient bé, do đó để xác định chính xác hơn nữa biên của các đối tượng, ta sử dụng các ngưỡng. Theo đó, bộ lọc Canny sẽ sử dụng một ngưỡng trên (upper threshold) và một ngưỡng dưới (lower threshold):

* Nếu gradient tại một điểm trong ảnh có giá trị lớn hơn ngưỡng trên thì ta xác nhận điểm đó là một điểm biên trong ảnh.
* Nếu giá trị này bé hơn ngưỡng dưới thì ta xác nhận điểm đó không phải là điểm biên.
* Nếu giá trị gradient nằm giữa ngưỡng trên và ngưỡng dưới thì nó chỉ được tính là điểm trên biên khi các điểm liên kế bên cạnh của nó có giá trị gradient lớn hơn ngưỡng trên.

## KNN – K-Nearest Neighbors

Thuật toán K-láng giềng gần nhất (KNN) là một loại thuật toán ML có giám sát có thể được sử dụng cho cả các bài toán phân loại. Tuy nhiên, nó chủ yếu được sử dụng cho các bài toán phân loại trong ngành.

Một số điểm quan trọng liên quan đến thuật toán KNN:

* K-Nearest Neighbor là một trong những thuật toán Machine Learning đơn giản nhất dựa trên kỹ thuật Supervised Learning.
* Thuật toán K-NN giả định sự giống nhau giữa trường hợp / dữ liệu mới và các trường hợp có sẵn và đặt trường hợp mới vào danh mục giống nhất với các danh mục có sẵn.
* Thuật toán K-NN lưu trữ tất cả dữ liệu có sẵn và phân loại một điểm dữ liệu mới dựa trên sự tương đồng. Điều này có nghĩa là khi dữ liệu mới xuất hiện thì nó có thể dễ dàng được phân loại vào loại tốt bằng cách sử dụng thuật toán K-NN.
* K-NN là một Non-parametric learning algorithm (thuật toán phi tham số), có nghĩa là nó không đưa ra bất kỳ giả định nào về dữ liệu cơ bản.
* K-NN còn được gọi là Lazy learning algorithm vì nó không học từ tập huấn luyện ngay lập tức thay vào đó nó lưu trữ tập dữ liệu và tại thời điểm phân loại, nó thực hiện một hành động trên tập dữ liệu.
* Thuật toán KNN ở giai đoạn huấn luyện chỉ lưu trữ tập dữ liệu và khi nó nhận được dữ liệu mới, sau đó nó sẽ phân loại dữ liệu đó thành một loại gần giống với dữ liệu mới.
* Ưu điểm
* Nó là một thuật toán rất đơn giản để hiểu và giải thích.
* Nó rất hữu ích cho dữ liệu phi tuyến vì không có giả định về dữ liệu trong thuật toán này.
* Nó là một thuật toán linh hoạt vì chúng ta có thể sử dụng nó để phân loại cũng như hồi quy.
* Nó có độ chính xác tương đối cao nhưng có nhiều mô hình học có giám sát tốt hơn KNN.
* Nhược điểm
* Nó là một thuật toán hơi tốn kém về mặt tính toán vì nó lưu trữ tất cả các dữ liệu huấn luyện.
* Yêu cầu bộ nhớ lưu trữ cao so với các thuật toán học có giám sát khác.
* Dự đoán chậm trong trường hợp N lớn.

## Kỹ thuật trích rút đặc trưng HOG

HOG là viết tắt của Histogram of Oriented Gradient - một loại “feature descriptor”. Mục đích của “feature descriptor” là trừu tượng hóa đối tượng bằng cách trích xuất ra những đặc trưng của đối tượng đó và bỏ đi những thông tin không hữu ích. Vì vậy, HOG được sử dụng chủ yếu để mô tả hình dạng và sự xuất hiện của một đối tượng trong ảnh.

Bản chất của phương pháp HOG là sử dụng thông tin về sự phân bố của các cường độ gradient (intensity gradient) hoặc của hướng biên (edge directions) để mô tả các đối tượng cục bộ trong ảnh. Các toán tử HOG được cài đặt bằng cách chia nhỏ một bức ảnh thành các vùng con, được gọi là “tế bào” (cells) và với mỗi cell, ta sẽ tính toán một histogram về các hướng của gradients cho các điểm nằm trong cell. Ghép các histogram lại với nhau ta sẽ có một biểu diễn cho bức ảnh ban đầu. Để tăng cường hiệu năng nhận dạng, các histogram cục bộ có thể được chuẩn hóa về độ tương phản bằng cách tính một ngưỡng cường độ trong một vùng lớn hơn cell, gọi là các khối (blocks) và sử dụng giá trị ngưỡng đó để chuẩn hóa tất cả các cell trong khối. Kết quả sau bước chuẩn hóa sẽ là một vector đặc trưng có tính bất biến cao hơn đối với các thay đổi về điều kiện ánh sáng.

Có 5 bước cơ bản để xây dựng một vector HOG cho hình ảnh, bao gồm:

* Tiền xử lý
* Tính gradient
* Tính vector đặc trưng cho từng ô (cells)
* Chuẩn hóa khối (blocks)
* Tính toán vector HOG

# **III. Xây dựng hệ thống nhận dạng**

## a. Sơ đồ khối của hệ thống

Tập ảnh đầu vào

Lưu mô hình

Huấn luyện mô hình

Trích chọn đặc trưng

* Tập ảnh đầu vào được trích chọn đặc trưng sau đó gắn nhãn và đưa vào mô hình huẩn luyện

Ảnh nhận dạng

Trích chọn đặc trưng

Nhận dạng

* Ảnh cần nhận dạng được trích chọn đặc trưng sau đó đưa vào model để nhận dạng.

## b. Trích chọn đặc trưng

- Trích chọn đặc trưng là hình dạng.

- Sử dụng thuật toán HOG để trích xuất đặc trưng hình dạng của ảnh đầu vào

- Đầu ra là một vecto biểu diễn đặc trưng

\* Thuật toán HOG

HOG được sử dụng chủ yếu để mô tả hình dạng và sự xuất hiện của một đối tượng trong ảnh. Điểm mấu chốt trong nguyên lý hoạt động của HOG đó là hình dạng của một vật thể cục bộ có thể được mô tả thông qua hai ma trận đó là ma trận độ lớn gradient và ma trận phương gradient.

Bước 1: Tính Gradient

- Giả sử I là ma trận ảnh gốc ban đầu. Sau đó sử dụng bộ lọc Sobel.

- Tính đạo hàm theo chiều ngang

=

- Tính đạo hàm theo chiều dọc

= \* I

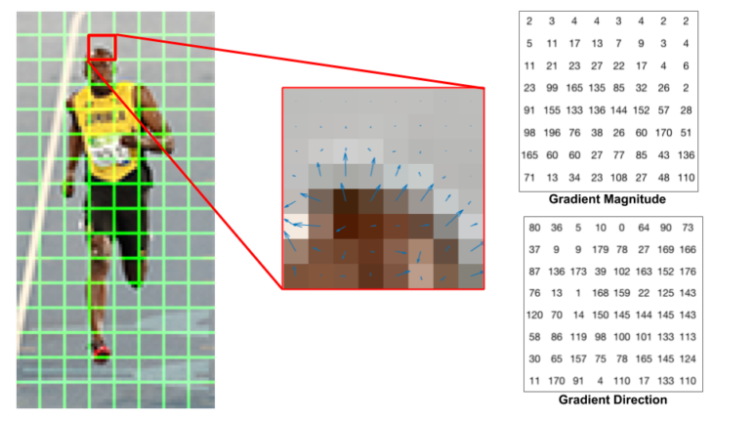
- Tính độ lớn gradient là chiều dài của véc tơ gradient theo phương x và phương y):

G =

- Phương gradient là độ lớn góc giữa vectơ gradient x và y giúp xác định phương thay đổi cường độ màu sắc

ɵ = arctan()

- Ví dụ: Hình ảnh được chia thành các cell có kích thước 8x8 pixels. Như vậy mỗi cell có tổng cộng 64 ô pixels. Trên mỗi một ô trong 64 pixels ta sẽ cần tính ra 2 tham số đó là độ lớn gradient (gradient magnitute) và phương gradient (gradient direction).



Bước 2: Xác định histogram

- Mapping độ lớn gradient vào các bins tương ứng của phương gradient. Mỗi một phương gradient sẽ ghép cặp với một độ lớn gradient ở cùng vị trí tọa độ. Khi biết được phương gradient thuộc bins nào trong véc tơ bins, ta sẽ điền vào giá trị giá trị của độ lớn gradient vào chính bin đó.

- Đầu mút là các giá trị chia hết cho độ rộng của một bin (chẳng hạn 0, 20, 40 là những đầu mút bin). Trong trường hợp độ lớn phương gradients không rơi vào các đầu mút, ta sẽ sử dụng linear interpolation để phân chia độ lớn gradient về 2 bins liền kề mà giá trị phương gradient rơi vào.

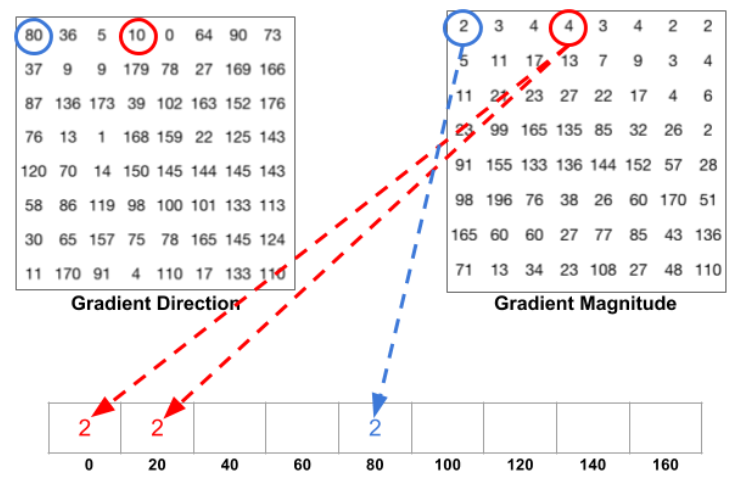
Ví dụ: giá trị phương gradient bằng x ghép cặp với độ lớn gradient bằng y.

x ∈ [x0, x1] tức là phương gradients rơi vào khoảng giữa bin thứ (l−1) và bin thứ l. Khi đó tại 2 bins (l−1) và l được điền vào giá trị cường độ theo công thức.

*Giá* trị bins tại *l*-1:

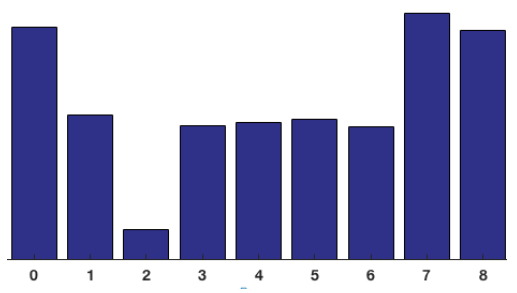
Giá trị bins tại *l*:

Ví dụ: Sắp xếp các giá trị phương gradient theo thứ tự từ nhỏ đến lớn và chia chúng vào 9 bins. Độ lớn của phương gradient sẽ nằm trong khoảng [0, 180] nên mỗi bins sẽ có độ dài là 20 như hình bên dưới.



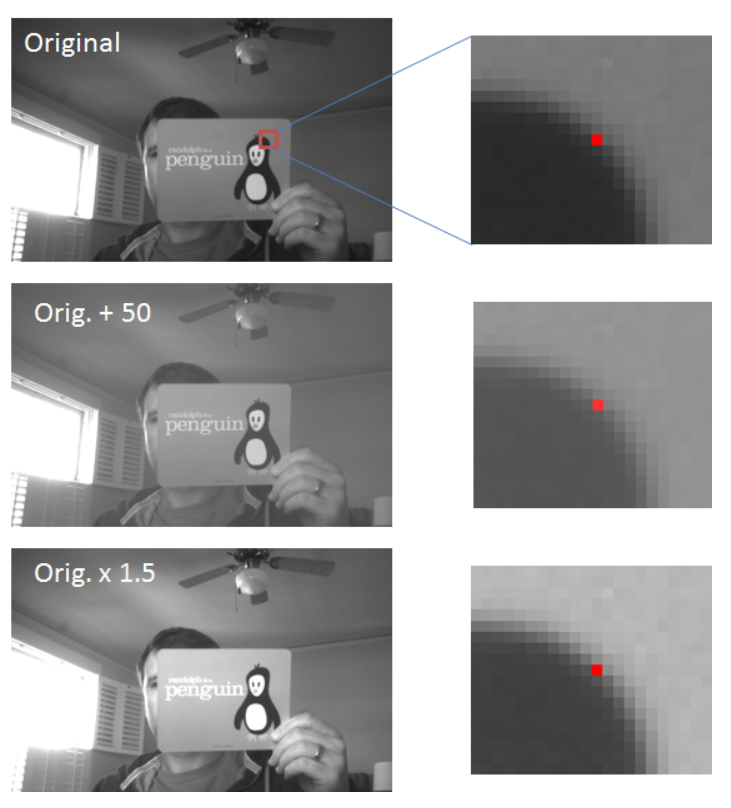
Như trong hình ảnh trên, đầu tiên là pixel có bao quanh màu xanh lam. Nó có hướng 80 độ và cường độ là 2, vì vậy ta thêm 2 vào bin thứ 5 (hướng 80 độ). Tiếp theo là pixel có bao quanh màu đỏ. Nó có hướng 10 độ và cường độ 4. Vì không có bin 10 độ, nên ta vote cho bin 0 độ và 20 độ, mỗi bin thêm 2 đơn vị.

- Tính tổng tất cả các độ lớn gradient thuộc cùng 1 bins của véc tơ bins ta thu được biểu đồ Histogram.



- Như vậy khi chia ảnh đầu vào thành các cells kích thước 8x8 pixels ta thu được một vecto histogram có kích thước 1x9.

Bước 3: Chuẩn hóa Histogram



- Trong hình ảnh trên, trường hợp đầu tiên là một ô của hình ảnh ban đầu. Trường hợp thứ hai, tất cả các giá trị pixel đã được tăng lên 50. Trong trường hợp thứ ba, tất cả các giá trị pixel được nhân với 1.5. Dễ dàng thấy được, trường hợp thứ ba hiển thị độ tương phản gia tăng. Ảnh hưởng của phép nhân là làm các điểm ảnh sáng trở nên sáng hơn nhiều, trong khi các điểm ảnh tối chỉ sáng hơn một chút, do đó làm tăng độ tương phản giữa các phần sáng và tối của hình ảnh.

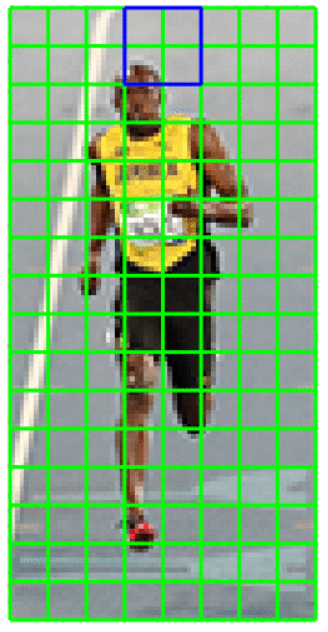
- Sử dụng chuẩn hóa norm chuẩn bậc 2. Giả sử vecto histogram của các gradient như sau:

V = [x1, x2, x3, …, x36]

Sẽ có norm bậc 2 như sau:



Khi này chuẩn hóa norm bậc 2: normalize(V) =



Quá trình chuẩn hóa sẽ thực hiện trên một block kích thước 2x2 trên lưới ô vuông ban đầu (mỗi ô kích thước 8x8 pixels) tức 16x16 pixels. Như vậy chúng ta sẽ có 4 vectơ histogram kích thước 1x9, hay một vectơ histogram tổng hợp kích thước là 1x36 và sau đó chuẩn hóa theo norm chuẩn bậc 2 trên véc tơ này.

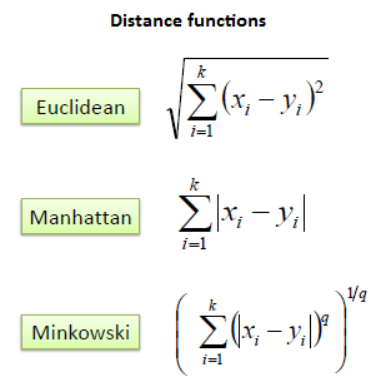
Bước 4: Tính toán vecto đặc trưng HOG

* Với mỗi hình ảnh kích thước 64x128, chia thành các block 16x16 chồng nhau, sẽ có 7 block ngang và 15 block dọc, nên sẽ có 7x15 = 105 blocks.
* Mỗi block gồm 4 cell. Khi áp dụng biểu đồ 9-bin cho mỗi cell, mỗi block sẽ được đại diện bởi một vector có kích thước 1x36.
* Vì vậy, khi nối tất cả các vector trong một block lại với nhau, ta sẽ thu được vector đặc trưng HOG của ảnh có kích thước 1x3780.

## c. Lưu trữ

* Trong bài toán của nhóm dùng mỗi ảnh đầu vào có kích thước 20x20, mỗi cell có kích thước 10x10. Sau khi được trích xuất thì đặc trưng mỗi ảnh là một mảng một chiều 36 phần tử.
* Mô hình sau khi train được lưu vào file “.pkl” theo cặp (dữ liệu, nhãn).
* Cách nhận dạng chữ viết sử dụng thuật toán KNN

Ý tưởng của thuật toán là những dữ liệu tương tự nhau sẽ tồn tại gần nhau trong một không gian, từ đó ta sẽ tìm k điểm gần với dữ liệu cần kiểm tra nhất. Việc tìm khoảng cách giữa 2 điểm củng có nhiều công thức có thể sử dụng, tùy trường hợp mà chúng ta lựa chọn cho phù hợp. Đây là 3 cách cơ bản để tính khoảng cách 2 điểm dữ liệu x, y có k thuộc tính:

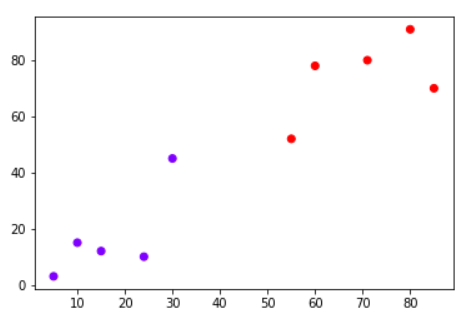


Lớp (nhãn) của một đối tượng dữ liệu mới có thể dự đoán từ các lớp (nhãn) của k hàng xóm gần nó nhất.

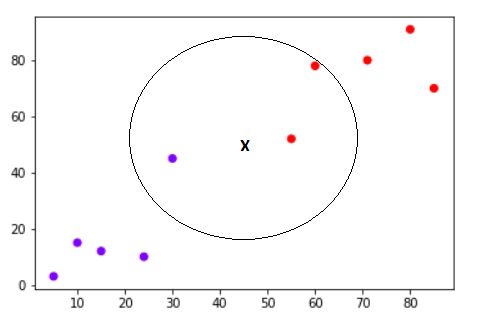
- Các bước trong thuật toán KNN:

1. Ta có D là tập các điểm dữ liệu đã được gắn nhãn và A là dữ liệu chưa được phân loại.
2. Đo khoảng cách (Euclidian, Manhattan, Minkowski, Minkowski hoặc Trọng số) từ dữ liệu mới A đến tất cả các dữ liệu khác đã được phân loại trong D.
3. Chọn K (K là tham số mà bạn định nghĩa) khoảng cách nhỏ nhất.
4. Kiểm tra danh sách các lớp có khoảng cách ngắn nhất và đếm số lượng của mỗi lớp xuất hiện.
5. Lấy đúng lớp (lớp xuất hiện nhiều lần nhất).
6. Lớp của dữ liệu mới là lớp mà bạn đã nhận được ở bước 5.

- Ví dụ: Có một tập dữ liệu được biểu diễn bằng 2 thuộc tính được biểu diễn bằng biểu đồ sau:



Ta phân loại một điểm dữ liệu mới với 'X' thành lớp "Xanh" hoặc lớp "Đỏ". Các giá trị tọa độ của điểm dữ liệu là x = 45 và y = 50. Giả sử giá trị của K là 3. Thuật toán KNN bắt đầu bằng cách tính khoảng cách của điểm X từ tất cả các điểm. Sau đó, nó tìm 3 điểm gần nhất có khoảng cách nhỏ nhất đến điểm X.



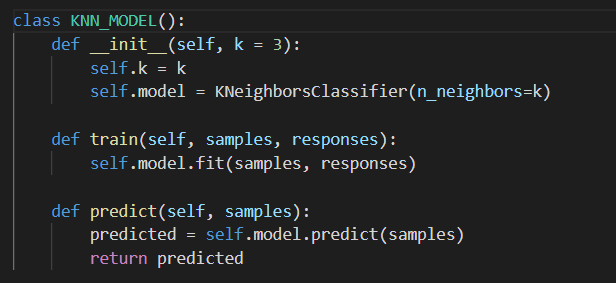
Bước cuối cùng của thuật toán KNN là gán điểm mới cho lớp mà phần lớn trong ba điểm gần nhất thuộc về lớp. Từ hình trên, chúng ta có thể thấy rằng hai trong ba điểm gần nhất thuộc lớp "Đỏ" trong khi một điểm thuộc lớp "Xanh lam". Do đó điểm dữ liệu mới sẽ được phân loại là "Đỏ".

# **IV. Kết quả**

Trong code, nhóm em sử dụng thư viện skimage để dùng hàm trích xuất đặc trưng hog của ảnh.

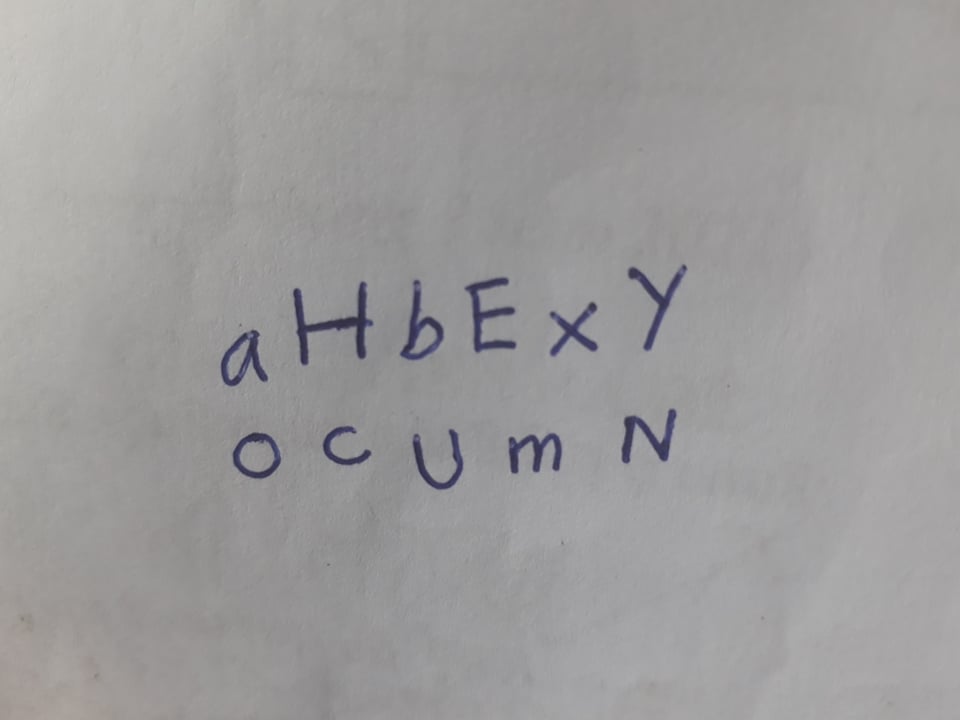


Và sử dụng thư viện sklearn để dùng KneighborsClassifier cho thuật toán KNN.

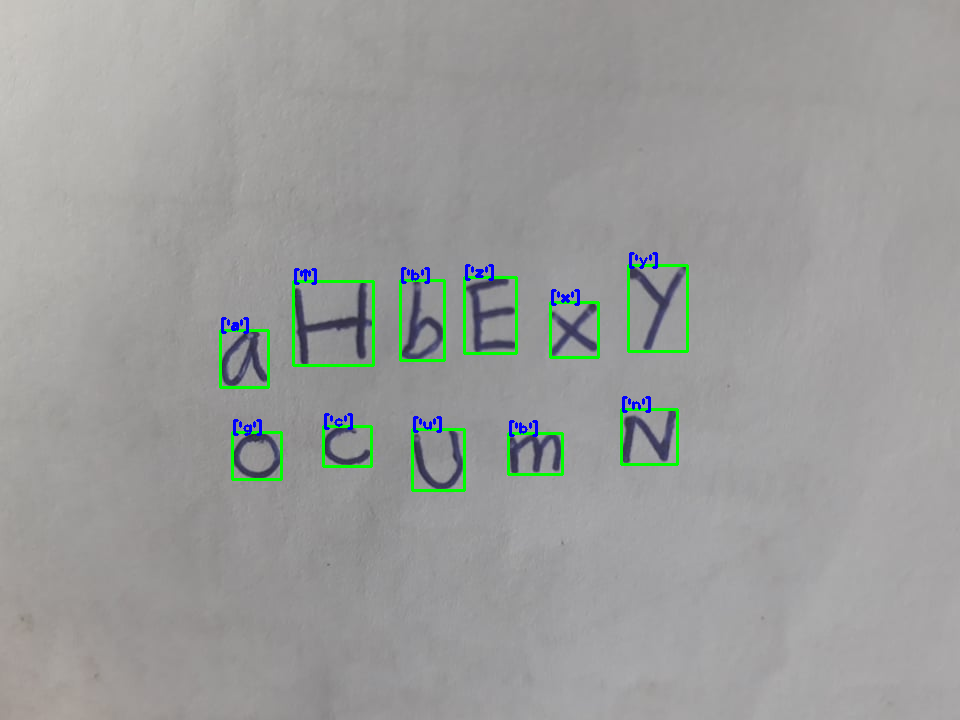


Kết quả nhận được:

* Ảnh đầu vào:



* Kết quả:



Trong ảnh nhận dạng, có các chữ ‘a’, ‘b’, ‘x’, ‘y’, ‘c’, ‘u’, ‘n’ là ra đúng nhãn. Các kí tự còn lại có nhãn chưa đúng.

* Nhận xét:
  + Sai số xảy ra có thể do việc chọn kích thước cell chưa phù hợp với bài toán
  + Thuật toán KNN chưa phải là cách tốt nhất để giải quyết bài toán