

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**  
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**  
**KHOA KHOA HỌC MÁY TÍNH**



**BÁO CÁO ĐỒ ÁN**  
**NHẬP MÔN THỊ GIÁC MÁY TÍNH**

**CS231.M21.KHCL**

**ĐỀ TÀI: NHẬN DIỆN MÓN ĂN VIỆT NAM**

**GIẢNG VIÊN HƯỚNG DẪN: TS. MAI TIẾN DŨNG**

**SINH VIÊN THỰC HIỆN: LÊ NGUYỄN TIẾN ĐẠT – MSSV: 20521167**  
**NGUYỄN THÀNH LÂM - MSSV: 20521517**

**TP. HỒ CHÍ MINH, 6/2022**

## MỤC LỤC

I. Giới thiệu chung về đề tài	3
1.Lý do chọn đề tài	3
2.Input và output	3
3.Bộ Data	4
II. Giới thiệu về đặc trưng HOG	4
1.Khái niệm	4
2.Ứng dụng	5
3.Một số ví dụ đặc trưng	7
III. Giới thiệu về model Support Vector Machine	8
1.Khái niệm	8
2.Cách hoạt động của thuật toán SVM	9
IV. Model CNN	10
1.Khái niệm	11
2.Các steps để áp dụng model CNN	12
V. Kết quả	14
VI. So sánh	20
Tài liệu tham khảo	21

# I.Giới thiệu chung về đề tài

## 1.Lý do chọn đề tài

Việt Nam ta là 1 quốc gia có rất nhiều món ăn truyền thống, số lượng món ăn rất nhiều và khá đa dạng về món ăn trong nước và cả món ăn của các nền ẩm thực nước khác.Vì vậy chúng ta cần 1 công cụ, 1 phần mềm để nhận diện món ăn Việt Nam để quảng bá rộng rãi, giới thiệu chúng tới cộng đồng và bạn bè quốc tế trên khắp thế giới.Có những món ăn ở nhiều vùng miền cũng chưa được phổ biến rộng rãi ở 1 số vùng trên nước ta, vì vậy việc nhận diện món ăn truyền thống của nước ta cũng giúp cho mỗi người trên mọi miền đất nước biết đến món ăn của các vùng đó.

Với đồ án môn học này,nhóm chúng em đưa ra 2 phương pháp để giải quyết vấn đề trên, cụ thể là áp dụng đặc trưng HOG trong xử lý ảnh cùng với sử dụng model SVM với nhân poly và RBF, phương pháp thứ 2 là sử dụng model CNN.Chúng em chọn 2 phương pháp này chủ yếu vì 2 lý do chính:

- + CNN là mô hình để phân lớp rất hiệu quả trên 1 tập dữ liệu lớn,nên nó có khả năng cho 1 kết quả vô cùng chính xác.
- + SVM là thuật toán sử dụng các support vector để phân bộ dữ liệu khác nhau thành các cụm nên việc cho kết quả cũng khá chính xác.

## 2.Input và output

- + Input: tập data gồm 7 món ăn gồm bánh bao, bánh mì, bánh trung thu, bún bò, bún đậu, chả giò, xôi mặn được lấy từ tập data VNfood15.Các bức ảnh chỉ gồm 1 món ăn và được chụp rõ từ nhiều góc độ khác nhau.
- + Output: Là nhãn của các món ăn như sau:  
Bánh bao, Bánh mì, Bánh trung thu, Bún bò, Bún đậu, Chả giò, Xôi mặn.

### 3. Bộ Data

Bộ data VNfood15 của nhóm nghiên cứu khoa học năm 2021 của trường Đại Học công nghệ thông tin. Chúng em đã được sự cho phép của các anh để sử dụng bộ dataset này.

Bộ Data này được thu thập từ các trang mạng lớn chứa nguồn ảnh hiện nay như Google hay Bing, Instagram. Chúng em lấy 3000 tấm ảnh từ tập data này ứng với 7 món ăn tương ứng với 2 tập data train và data test.

## II. Giới thiệu về đặc trưng HOG

### 1. Khái niệm

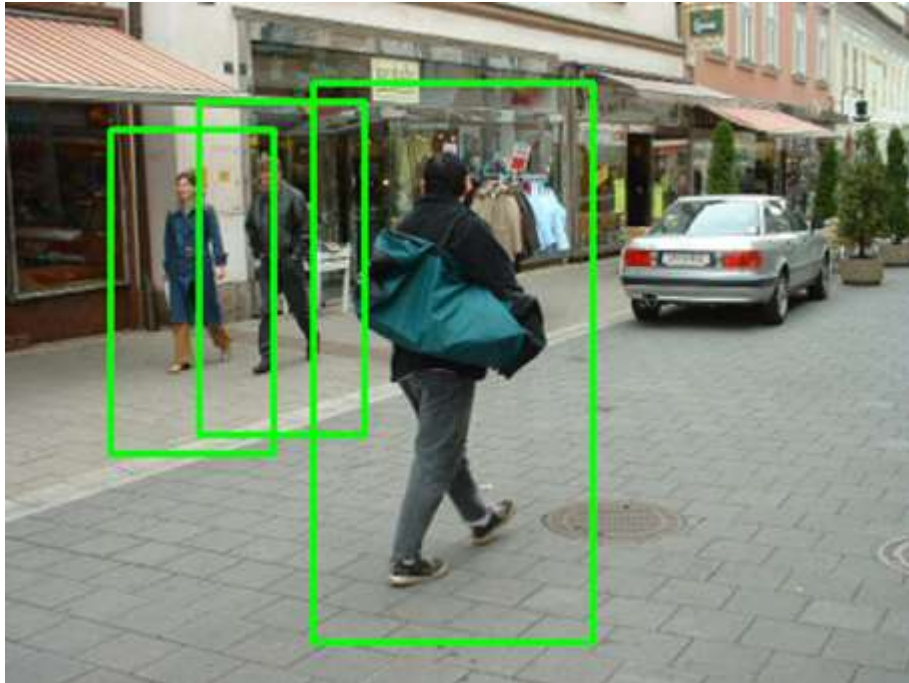
Có rất nhiều các phương pháp khác nhau trong computer vision. Khi phân loại ảnh, chúng ta có thể áp dụng họ các mô hình CNN (Inception Net, mobile Net, Resnet, Dense Net, Alexnet, Unet,...) và khi phát hiện vật thể là các mô hình YOLO, SSD, Faster RCNN, Fast RCNN, Mask RCNN.

Các thuật toán kể trên đều là những mô hình deep learning. Vậy trước khi deep learning bùng nổ, thuật toán nào thường được sử dụng trong xử lý ảnh? Bài hôm nay chúng ta sẽ tìm hiểu về thuật toán tuy cổ điển nhưng cũng rất hiệu quả trong xử lý ảnh, đó chính là HOG (histogram of oriented gradient).

Thuật toán này sẽ tạo ra các bộ mô tả đặc trưng (feature descriptor) nhằm mục đích phát hiện vật thể (object detection). Từ một bức ảnh, ta sẽ lấy ra 2 ma trận quan trọng giúp lưu thông tin ảnh đó là độ lớn gradient (gradient magnitude) và phương của gradient (gradient orientation). Bằng cách kết hợp 2 thông tin này vào một biểu đồ phân phối histogram, trong đó độ lớn gradient được đếm theo các nhóm bins của phương gradient. Cuối cùng ta sẽ thu được véc tơ đặc trưng HOG đại diện cho histogram. Sơ khai là vậy, trên thực tế thuật toán còn hoạt động phức tạp hơn khi véc tơ HOG sẽ được tính trên từng vùng cục bộ như mạng CNN và sau đó là phép chuẩn hóa cục bộ để đồng nhất độ đo. Cuối cùng véc tơ HOG tổng hợp từ các véc tơ trên vùng cục bộ.

## 2. Các ứng dụng của HOG

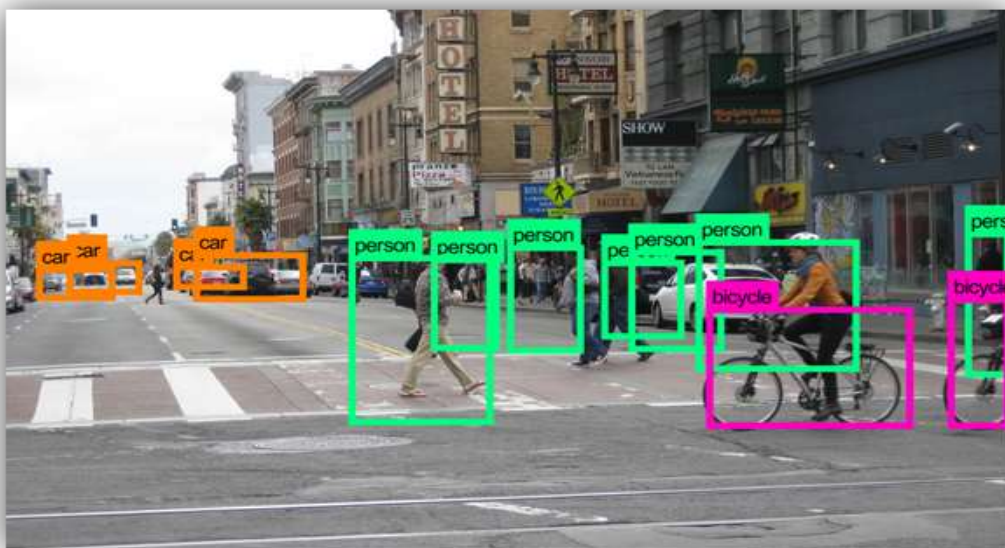
- + **Nhận diện người(human detection):** Lần đầu tiên ứng dụng này được giới thiệu trong bài báo [Histograms of Oriented Gradients for Human Detection](#) của Dalal và Trigg. HOG có thể phát hiện được một hoặc nhiều người đi bộ trên cùng một hình ảnh.



- + **Nhận diện khuôn mặt(face detection):** HOG cũng là một thuật toán rất hiệu quả được áp dụng trong bài toán này. Bởi nó có khả năng biểu diễn các đường nét chính của khuôn mặt dựa trên phương và độ lớn gradient thông qua các véc tơ trên mỗi cell như hình mô tả bên dưới.



- + **Nhận diện các vật thể khác:** Ngoài ra còn rất nhiều các trường hợp nhận diện vật thể trên ảnh tĩnh như phương tiện, tín hiệu giao thông, động vật hoặc thậm chí là ảnh động từ video.



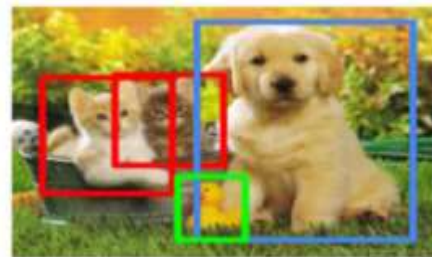
- + **Tạo feature cho các bài toán phân loại ảnh:** Nhiều bài toán phân loại ảnh được xây dựng trên một bộ dữ liệu kích thước nhỏ thì sử dụng các mạng học sâu chưa chắc đã mang lại hiệu quả và dễ dẫn tới overfitting. Nguyên nhân vì dữ liệu ít thường không đủ để huấn luyện cho máy tính nhận tốt các đặc trưng của vật thể. Khi đó sử dụng HOG để tạo đặc trưng sẽ mang lại kết quả tốt hơn.

### Classification



CAT

### Object Detection

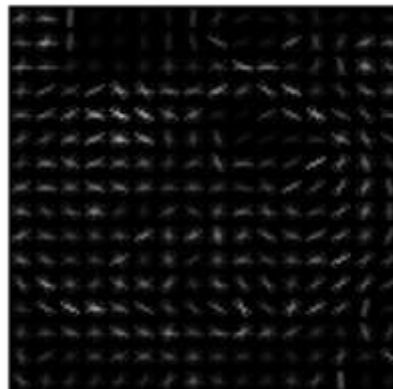


CAT, DOG, DUCK

## 3. Một số ví dụ về đặc trưng HOG

### VD1:

Sau đây nhóm em lấy ngẫu nhiên 2 hình ảnh từ tập data của mình:



Ở hình đầu tiên này ta có thể thấy các đường nét và hình ảnh của mẹt bún đậu được miêu tả rất chi tiết và rõ qua từng các gradient ngang và dọc.





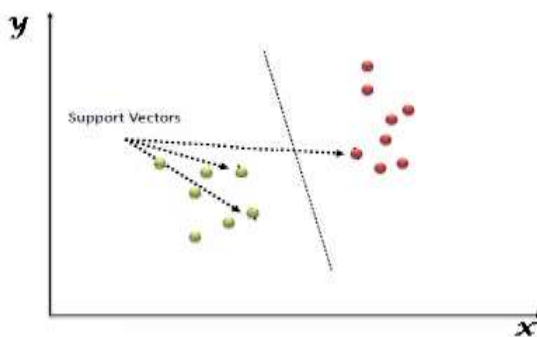
Ở hình thứ 2 này hình ảnh miêu tả đặc trưng HOG của chả giò thậm chí còn rất rõ ràng và chi tiết.

=> chính vì vậy, bọn em sử dụng đặc trưng HOG để xử lý ảnh vì nó có khả năng xử lý ảnh vô cùng tốt và chi tiết.

### III. Giới thiệu về Model Support Vector Machine

#### 1. Khái niệm

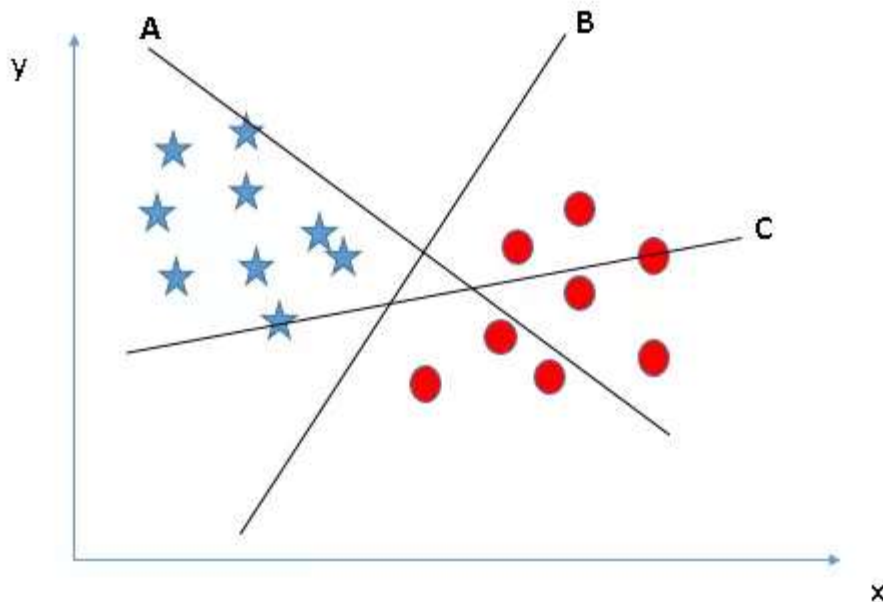
SVM là một thuật toán giám sát, nó có thể sử dụng cho cả việc phân loại hoặc đệ quy. Tuy nhiên nó được sử dụng chủ yếu cho việc phân loại. Trong thuật toán này, chúng ta vẽ đồ thị dữ liệu là các điểm trong  $n$  chiều (ở đây  $n$  là số lượng các tính năng bạn có) với giá trị của mỗi tính năng sẽ là một phần liên kết. Sau đó chúng ta thực hiện tìm "đường bay" (*hyper-plane*) phân chia các lớp. Hyper-plane nó chỉ hiểu đơn giản là 1 đường thẳng có thể phân chia các lớp ra thành hai phần riêng biệt.



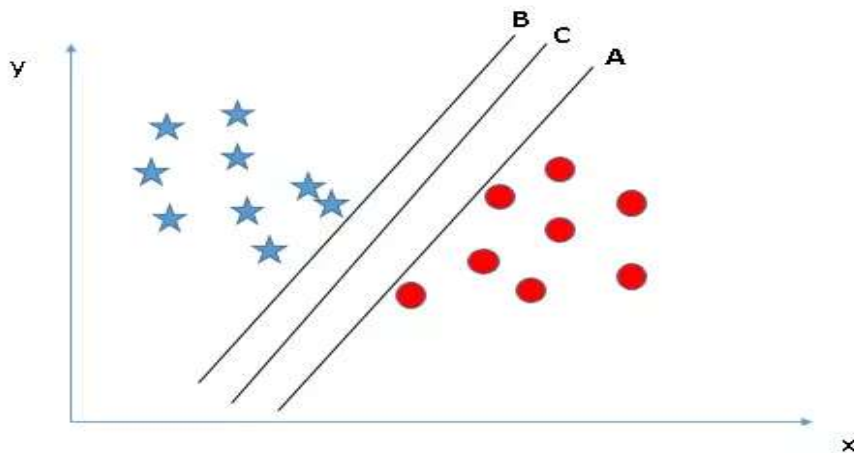
#### 2. Cách hoạt động của SVM

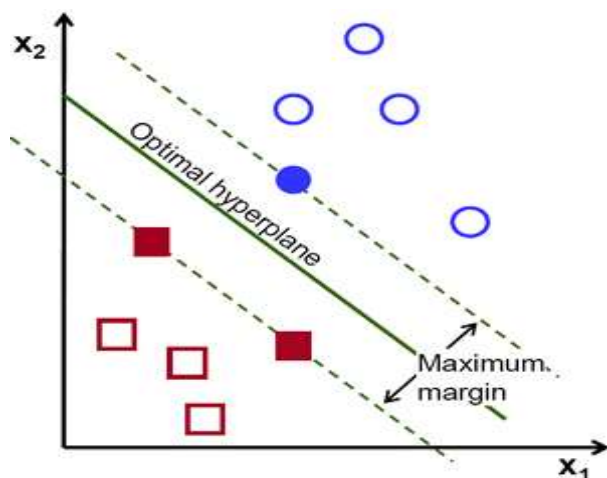
Quy tắc số một để chọn 1 hyper-plane, chọn một hyper-plane để phân chia hai lớp tốt nhất. Trong ví dụ này chính là đường B.





Quy tắc thứ hai chính là xác định khoảng cách lớn nhất từ điều gần nhất của một lớp nào đó đến đường hyper-plane. Khoảng cách này được gọi là "Margin", Hãy nhìn hình bên dưới, trong đây có thể nhìn thấy khoảng cách margin lớn nhất đây là đường C. Cần nhớ nếu chọn làm hyper-lane có margin thấp hơn thì sau này khi dữ liệu tăng lên thì sẽ sinh ra nguy cơ cao về việc xác định nhầm lớp cho dữ liệu.





Margin là khoảng cách giữa một hyperbol trong không gian 2D đến 2 điểm dữ liệu gần nhất tương ứng với các phân lớp.

phương pháp SVM luôn cố gắng cực đại hóa margin này, từ đó thu được một siêu phẳng tạo khoảng cách xa nhất so với 2 lớp. Nhờ vậy, SVM có thể giảm thiểu việc phân lớp sai (misclassification) đối với điểm dữ liệu mới đưa vào.

## IV. Model CNN

### 1. Khái niệm

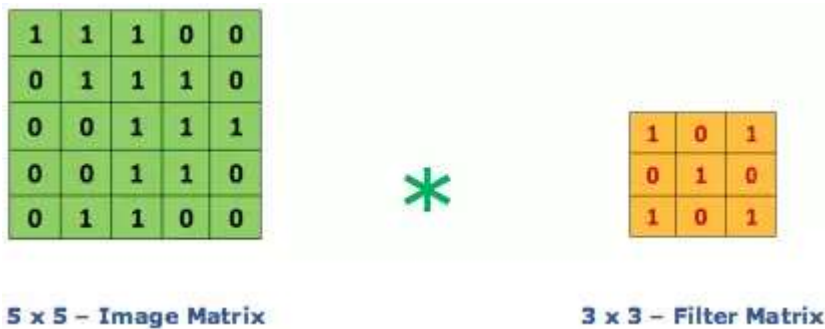
Trong mạng neural, mô hình mạng neural tích chập (CNN) là 1 trong những mô hình để nhận dạng và phân loại hình ảnh. Trong đó, xác định đối tượng và nhận dạng khuôn mặt là 1 trong số những lĩnh vực mà CNN được sử dụng rộng rãi.

Về kỹ thuật, mô hình CNN để training và kiểm tra, mỗi hình ảnh đầu vào sẽ chuyển nó qua 1 loạt các lớp tích chập với các bộ lọc (Kernels), tổng hợp lại các lớp được kết nối đầy đủ (Full Connected) và áp dụng hàm Softmax để phân loại đối tượng có giá trị xác suất giữa 0 và 1. Hình dưới đây là toàn bộ luồng CNN để xử lý hình ảnh đầu vào và phân loại các đối tượng dựa trên giá trị.

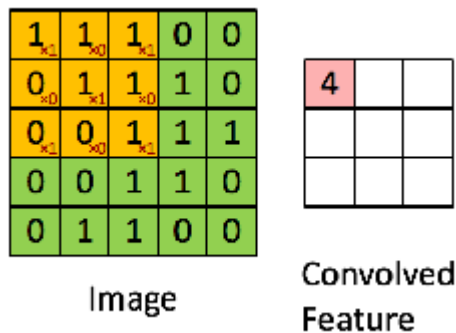
### 2. Các steps để áp dụng model CNN

#### Step1: Lớp tích chập - Convolution Layer

- Tích chập là lớp đầu tiên để trích xuất các tính năng từ hình ảnh đầu vào. Tích chập duy trì mối quan hệ giữa các pixel bằng cách tìm hiểu các tính năng hình ảnh bằng cách sử dụng các ô vuông nhỏ của dữ liệu đầu vào. Nó là 1 phép toán có 2 đầu vào như ma trận hình ảnh và 1 bộ lọc hoặc hạt nhân.
- Xem xét 1 ma trận 5 x 5 có giá trị pixel là 0 và 1. Ma trận bộ lọc 3 x 3 như hình bên dưới.

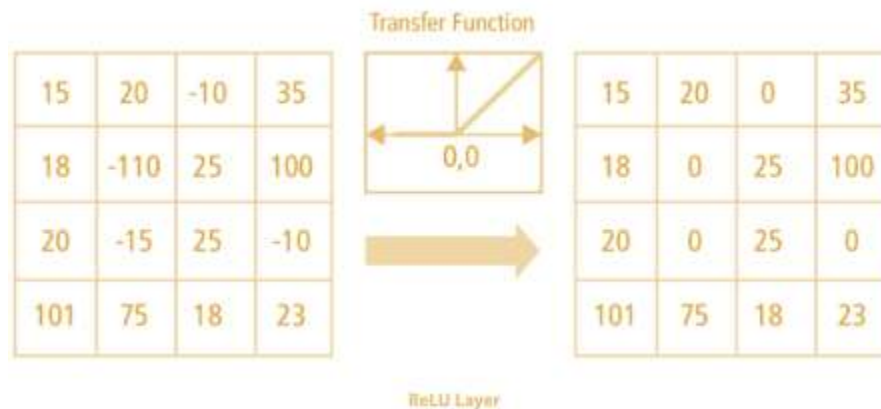


- Sau đó, lớp tích chập của ma trận hình ảnh 5 x 5 nhân với ma trận bộ lọc 3 x 3 gọi là 'Feature Map' như hình bên dưới.



## Step2: Hàm phi tuyến - ReLU

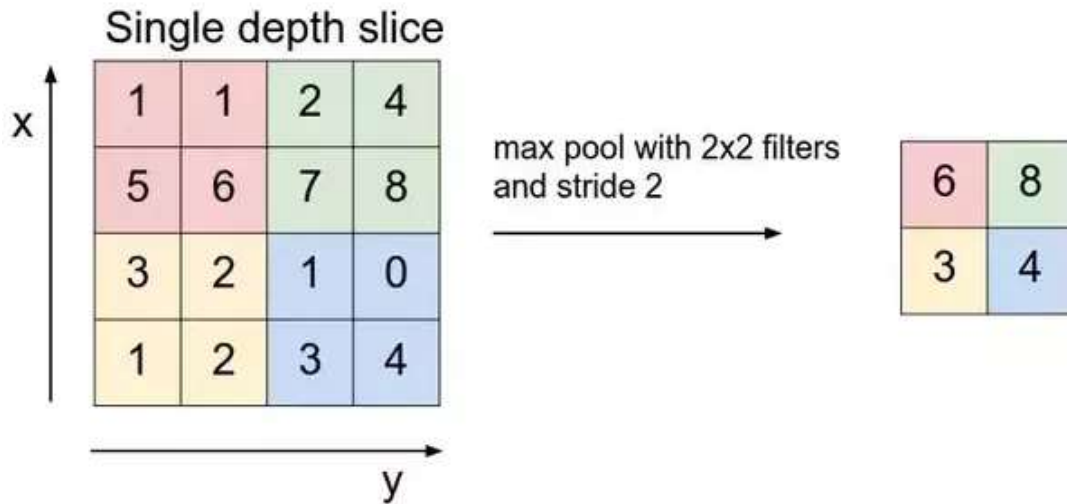
- + ReLU viết tắt của Rectified Linear Unit, là 1 hàm phi tuyến. Với đầu ra là:  $f(x) = \max(0, x)$ .
- + Tại sao ReLU lại quan trọng: ReLU giới thiệu tính phi tuyến trong ConvNet. Vì dữ liệu trong thế giới mà chúng ta tìm hiểu là các giá trị tuyến tính không âm.



- + Có 1 số hàm phi tuyến khác như tanh, sigmoid cũng có thể được sử dụng thay cho ReLU. Hầu hết người ta thường dùng ReLU vì nó có hiệu suất tốt.

## Step3: Pooling layer

- Lớp pooling sẽ giảm bớt số lượng tham số khi hình ảnh quá lớn. Không gian pooling còn được gọi là lấy mẫu con hoặc lấy mẫu xuống làm giảm kích thước của mỗi map nhưng vẫn giữ lại thông tin quan trọng. Các pooling có thể có nhiều loại khác nhau:
- Max Pooling
- Average Pooling
- Sum Pooling
- Max pooling lấy phần tử lớn nhất từ ma trận đối tượng, hoặc lấy tổng trung bình. Tổng tất cả các phần tử trong map gọi là sum pooling

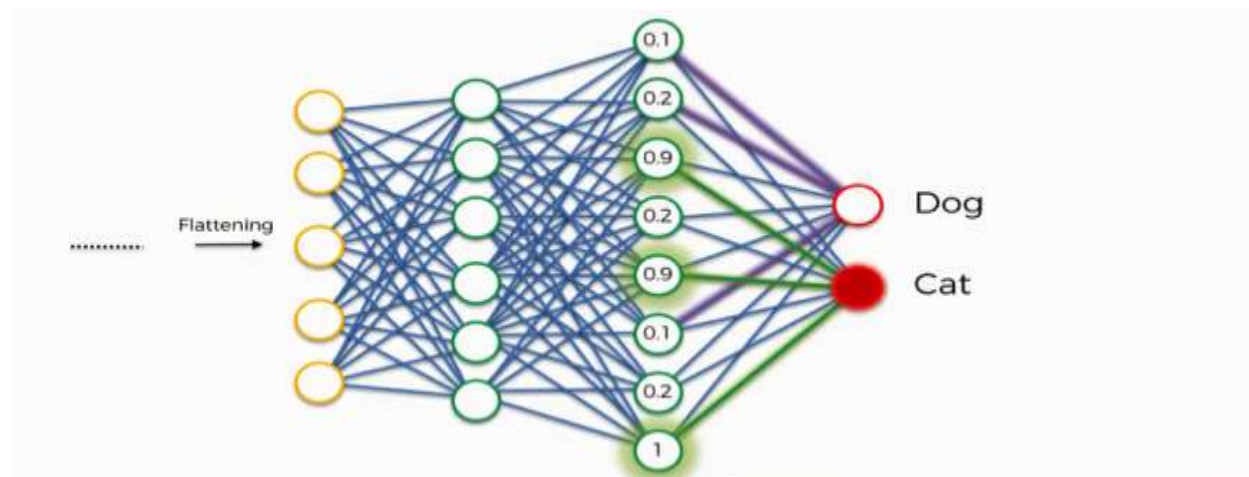


#### Step4: Fully Connection

Fully Connected Layer thường sử dụng để đưa ra các kết quả.

**Ví dụ:** Sau khi các lớp Convolutional Layer và Pooling Layer nhận được các ảnh đã truyền qua chúng, bạn sẽ thu được kết quả là Model đã đọc được khá nhiều thông tin về ảnh. Do đó, để có thể liên kết các đặc điểm này lại và cho ra Output, bạn cần dùng đến Fully Connected Layer.

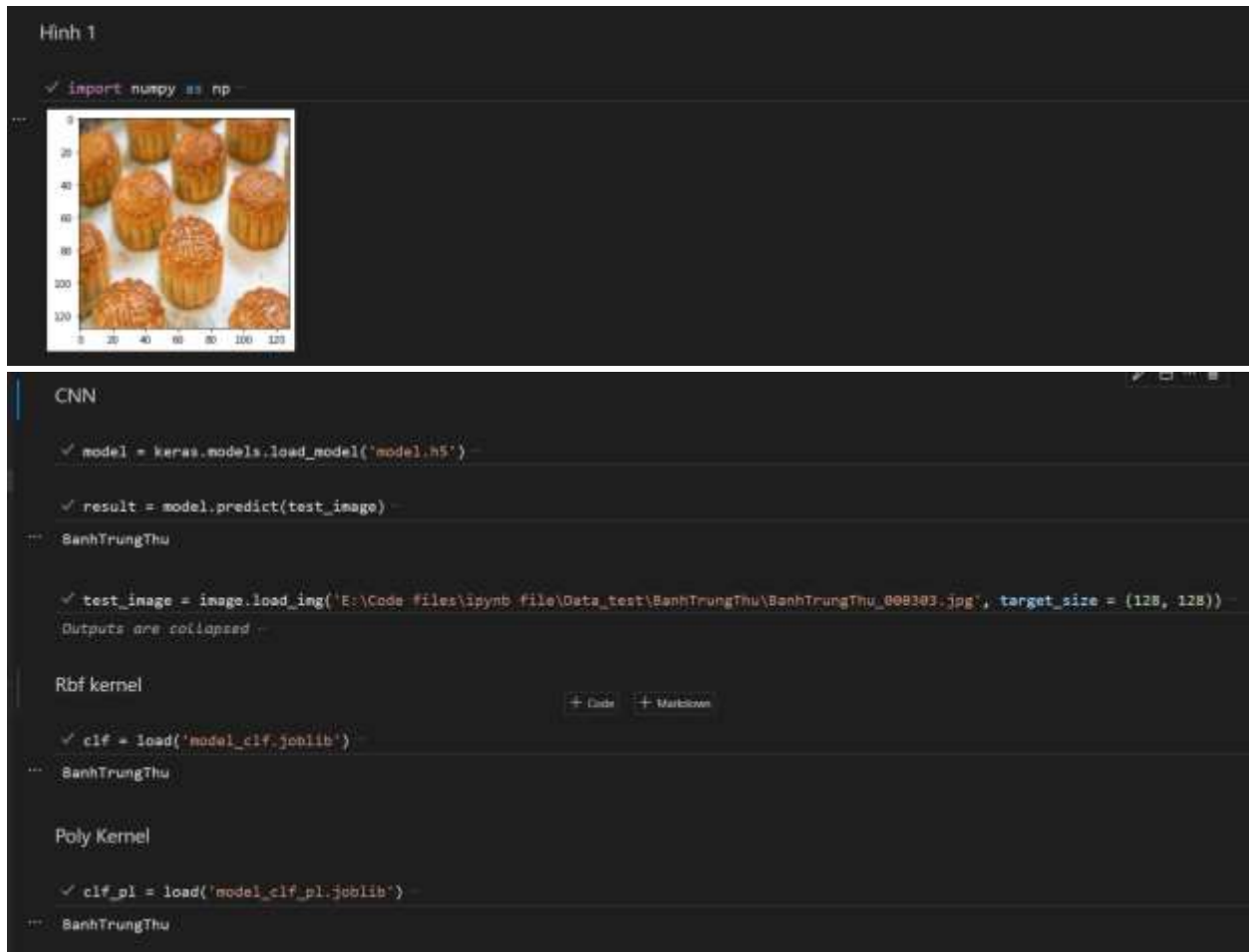
Bên cạnh đó, khi có được các dữ liệu về hình ảnh, Fully Connected Layer sẽ chuyển đổi chúng thành những mục có phân chia chất lượng. Tương tự như kiểu chia chúng thành các phiếu bầu và đánh giá để chọn ra hình ảnh đạt chất lượng tốt nhất. Dù vậy, quá trình này không được coi là quá trình dân chủ nên rất ít sử dụng.



## V. Kết Quả

Nhóm chúng em đã sử dụng 6 trường hợp khác nhau để đưa ra kết quả chính xác nhất.

**Trường hợp 1 và 2:** Cả 2 mô hình đều train đúng



## CNN

```
✓ result = model.predict(test_image)
```

XoiMan

```
✓ test_image = image.load_img('E:\Code_files\ipynb_file\Data_test\XoiMan\XoiMan_000015.jpg', target_size = (128, 128))
```

C:\Users\nguye\AppData\Local\Temp\ipykernel\_4168\1418635869.py:6: FutureWarning: "multichannel" is a deprecated argument name for "hog". It will be removed in version 1.0. Please use "channel\_axis" instead.

```
test_image, hog_image = hog(test_image, orientations=9, pixels_per_cell=(8, 8),
```

## RBF

```
✓ output(clf.predict(test_image))
```

XoiMan

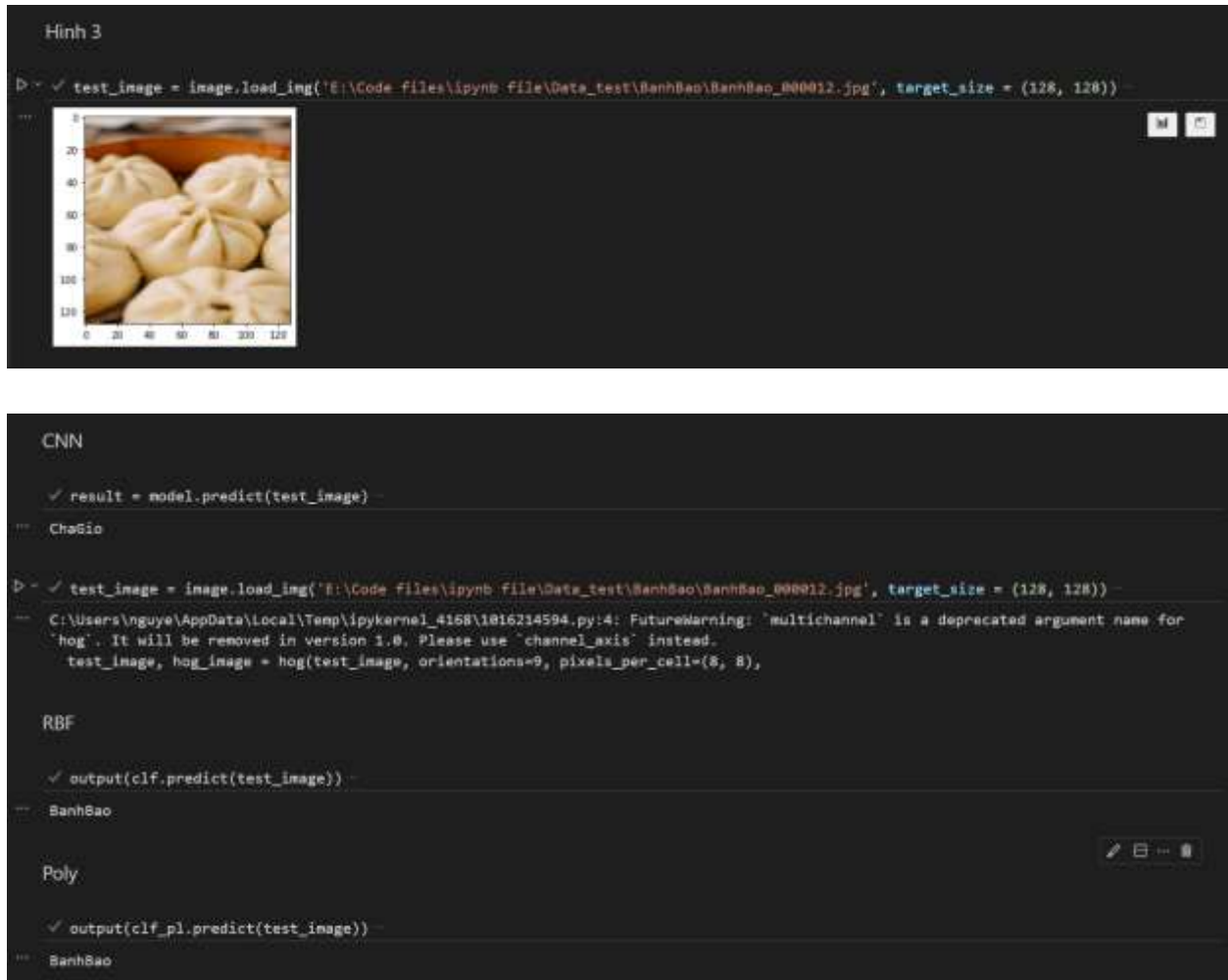
## Poly

```
✓ output(clf_pl.predict(test_image))
```

XoiMan



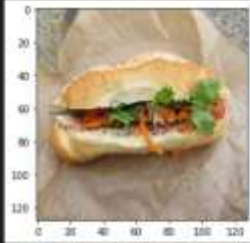
### Trường hợp 3: Model CNN nhận diện sai, SVM nhận diện đúng



#### Trường hợp 4: Model CNN nhận diện đúng, model SVM nhận diện sai.

Hình 4

```
D - ✓ test_image = image.load_img('E:\Code_files\ipynb_file\Data_test\BanhMi\BanhMi_000878.jpg', target_size = (128, 128))
```



CNN

```
✓ result = model.predict(test_image)
```

BanhMi

```
✓ test_image = image.load_img('E:\Code_files\ipynb_file\Data_test\BanhMi\BanhMi_000878.jpg', target_size = (128, 128))
```

C:\Users\nguye\AppData\Local\Temp\ipykernel\_4168\267152674.py:4: FutureWarning: 'multichannel' is a deprecated argument name for 'hog'. It will be removed in version 1.0. Please use 'channel\_axis' instead.

```
test_image, hog_image = hog(test_image, orientations=9, pixels_per_cell=(8, 8),
```

RBF

```
D - ✓ output(clf.predict(test_image))
```

XoiMan

Poly

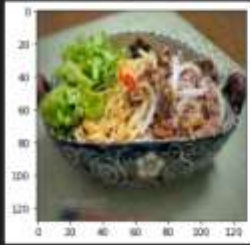
```
D - ✓ output(clf_pl.predict(test_image))
```

XoiMan

## Trường hợp 5: Cả 2 model đều nhận dạng sai

Hình 5

```
> ✓ test_image = image.load_img('E:\Code_files\ipynb_file\Data_test\BunBo\BunBo_000005.jpg', target_size = (128, 128))
```



CNN

```
✓ result = model.predict(test_image)
```

XoiMan

```
✓ test_image = image.load_img('E:\Code_files\ipynb_file\Data_test\BunBo\BunBo_000005.jpg', target_size = (128, 128))
```

C:\Users\nguye\AppData\Local\Temp\ipykernel\_4168\1745234913.py:4: FutureWarning: 'multichannel' is a deprecated argument name for 'hog'. It will be removed in version 1.0. Please use 'channel\_axis' instead.

```
test_image, hog_image = hog(test_image, orientations=9, pixels_per_cell=(8, 8),
```

RBF

```
✓ output(clf.predict(test_image))
```

XoiMan

Poly

```
✓ output(clf_pl.predict(test_image))
```

XoiMan

## Trường hợp 6: Nhận dạng 1 ảnh từ bên ngoài không có trong tập data

```
Đoàn ảnh tải từ bên ngoài

D> ✓ test_image = image.load_img('C:/Users/nguye/Downloads/download.jpg', target_size = (128, 128))

[128] ✓ 0.1s Python
[127] ✓ 0.1s Python

CNN

✓ result = model.predict(test_image)

BanhBao

✓ test_image = image.load_img('C:/Users/nguye/Downloads/download.jpg', target_size = (128, 128))

C:\Users\nguye\AppData\Local\Temp\ipykernel_4168\2820005383.py:4: FutureWarning: 'multichannel' is a deprecated argument name for 'hog'. It will be removed in version 1.0. Please use 'channel_axis' instead.
test_image, hog_image = hog(test_image, orientations=9, pixels_per_cell=(8, 8),

RBF

output(clf.predict(test_image))

[128] ✓ 0.1s Python
[127] ✓ 0.1s Python

Poly

output(clf_pl.predict(test_image))

[127] ✓ 0.1s Python
[128] ✓ 0.1s Python

BanhBao
```

## VI. So sánh

Sau khi thực nghiệm, chúng em có 1 bảng so sánh về 2 phương pháp như sau:

	HOG + SVM	CNN
Độ chính xác	SVM(kernel = poly):0,45 SVM(kernel = rbf): 0,42	0,83

=> Trong bài toán nhận dạng này thì model CNN tỏ ra hiệu quả hơn khá nhiều so với model SVM, tuy vậy cũng có những trường hợp model SVM lại đúng.

**Tài liệu tham khảo:**

<https://wiki.tino.org/convolutional-neural-network-la-gi/>

<https://www.superdatascience.com/blogs/deep-learning-a-z-convolutional-neural-networks-cnn-step-3-flattening/>

<https://viblo.asia/p/gioi-thieu-ve-support-vector-machine-svm-6J3ZgPVEImB>

<https://viblo.asia/p/deep-learning-tim-hieu-ve-mang-tich-chap-cnn-maGK73bOKj2>

<https://phamdinhkhanh.github.io/2019/11/22/HOG.html#1-gi%E1%BB%9Bi-thi%E1%BB%87u-v%E1%BB%81-thu%E1%BA%ADt-to%C3%A1n-hog>

<https://viblo.asia/p/tim-hieu-ve-phuong-phap-mo-ta-dac-trung-hog-histogram-of-oriented-gradients-V3m5WAwxZO7>