

**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**  
**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

-----\*\*\*-----



Bài tập lớn môn học: Học máy

**Xây dựng hệ thống nhận diện món ăn**

*Nhóm sinh viên thực hiện : N\_13*

Lê Thành Lợi	20142729
Tống Thị Hồng	20141867
Nguyễn Đức Thắng	20144212
Lê Thị Nga	20143121
Trần Đình Huy	20142001

*Giảng viên hướng dẫn : TS.Thân Quang Khoát*

Hà Nội, 12/2017

## Thông tin nhóm

Họ và tên	MSSV	Công việc
Lê Thành Lợi	20142729	- Tìm hiểu và xây dựng mô hình
Tống Thị Hồng	20141867	- Xây dựng hệ thống website tích hợp mô hình phân loại
Nguyễn Đức Thắng	20144212	- Tìm hiểu và xây dựng mô hình
Lê Thị Nga	20143121	- Xây dựng hệ thống website tích hợp mô hình phân loại
Trần Đình Huy	20142001	- Tiền xử lí dữ liệu và viết chức năng nhận diện ảnh thực

# MỤC LỤC

Thông tin nhóm.....	2
Lời nói đầu.....	3
1. Giới thiệu bài toán.....	5
2. Bộ dữ liệu và bước tiền xử lý.....	6
3. Mô hình sử dụng.....	7
3.1 Giới thiệu mạng CNN( Convolution neural network).....	7
3.2 Các tầng được sử dụng để xây dựng mạng CNN.....	8
3.2.1 Lớp Convolution.....	9
3.2.2 Lớp Pooling.....	11
3.2.3 Lớp Fully- Connected.....	12
3.3 Ứng dụng của mô hình.....	12
4. Các thư viện sử dụng.....	14
4.1 Keras.....	14
4.2 Flask.....	14
5. Kết quả.....	15
5.1 Mô hình phân loại.....	15
5.2 Hệ thống website.....	15
5.3 Tổng kết.....	16
Tài Liệu tham khảo.....	17
Lời cảm ơn.....	18

# Lời nói đầu

Những năm gần đây, AI – Artificial Intelligence (Trí tuệ nhân tạo), và cụ thể hơn là Machine Learning (Học Máy hoặc Máy Học) nổi lên như một bằng chứng của cuộc cách mạng công nghiệp 4.0. Trí Tuệ Nhân Tạo đang len lỏi vào mọi lĩnh vực trong đời sống mà có thể chúng ta không nhận ra. Xe tự hành của Google và Tesla, hệ thống tự tag khuôn mặt của Facebook,... chỉ là một vài trong vô vàn ứng dụng của AI/ Machine Learning. Với mong muốn bắt nhịp cùng xu hướng và tìm hiểu công nghệ mới, nhóm chúng em xin lựa chọn đề tài **“Xây dựng hệ thống nhận diện món ăn”** sử dụng mạng nơ-ron tích chập (CNN).

Chúng em rất mong nhận được sự góp ý từ thầy để có thể cải thiện hơn nữa khả năng và hiệu quả của hệ thống.

# 1. Giới thiệu bài toán

**Mục tiêu:** Xây dựng một hệ thống nhận diện món ăn trên nền tảng website.

**Mô tả hệ thống:** Hệ thống cho phép người dùng lựa chọn món ăn bất kì để tiến hành dự đoán bằng cách đưa ra tên của món ăn dự đoán cũng như tỷ lệ dự đoán tên của món ăn đó dựa trên các món ăn được huấn luyện. Cùng với đó sẽ lưu lại ảnh để thu thập dữ liệu phục vụ cho mục đích phát triển hệ thống.

**Công nghệ sử dụng:** Nhóm chúng em sử dụng mô hình mạng neural tích chập (CNN) để xây dựng mô hình phân loại và sử dụng framework Flask (ngôn ngữ Python) để xây dựng hệ thống website.

## 2. Bộ dữ liệu và bước tiền xử lý

Bài toán nhận diện món ăn, với bộ dữ liệu bao gồm 101 món được lấy từ bộ dữ liệu “Food-101” . Mỗi món bao gồm 1000 ảnh có kích thước không cố định. Do tài nguyên về trang thiết bị có hạn, nhóm lựa chọn 10 món ăn một cách ngẫu nhiên để tiến hành xây dựng hệ thống.

Danh sách 10 món ăn được nhóm lựa chọn :

baby_back_ribs	0.000559866748517
fried_rice	99.1335868835
hot_dog	0.00194680560526
ice_cream	0.000145391538808
pho	0.00888289578143
pizza	0.000440025269199
red_velvet_cake	1.25718116806e-05
seaweed_salad	4.47844058726e-05
spaghetti_bolognese	0.854269135743
sushi	0.000107802804905

*Hình 1. Danh sách các món ăn được sử dụng*

### Tiền xử lý dữ liệu

Các ảnh thuộc bộ dữ liệu trên có kích thước khác nhau, nhóm tiến hành resize các ảnh về lại kích thước 96x96x3. Sau đó, nhóm tiến hành tách ra 8000 ảnh một cách ngẫu nhiên để đưa vào tập học và 2000 ảnh còn lại sẽ được sử dụng cho tập test.

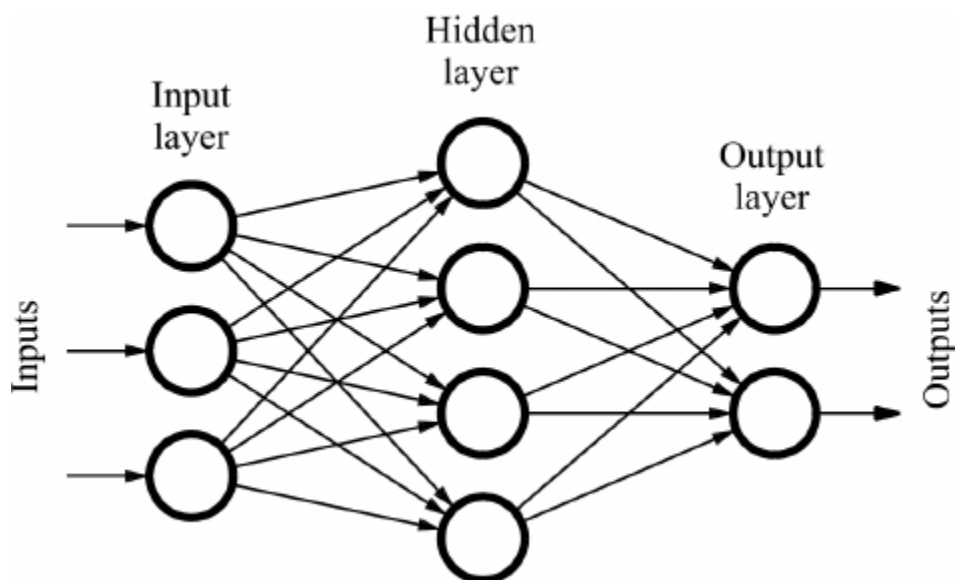
## 3. Mô hình sử dụng

### 3.1 Giới thiệu mạng CNN( Convolution neural network)

Convolutional Neural Network (CNN – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao như hiện nay như hệ thống xử lý ảnh lớn như Facebook, Google hay Amazon đã đưa vào sản phẩm của mình những chức năng thông minh như nhận diện khuôn mặt người dùng, phát triển xe hơi tự lái hay drone giao hàng tự động.

#### Tổng quan kiến trúc mạng CNN

Nhắc lại về mạng neural thường, mạng neural thường nhận đầu vào là một vector đơn( single vector) và biến đổi chúng thông qua một loạt các lớp ẩn( hidden layer). Mỗi lớp ẩn được tạo bởi một bộ các neural, mỗi neural được kết nối đầy đủ( fully connected) với tất cả các neural của lớp trước đó và các neural trong từng lớp ẩn thì hoàn toàn độc lập với nhau.



*Hình 2. Cấu trúc tổng quan của mạng neural thường*

Lớp kết nối đầy đủ cuối cùng được gọi là lớp đầu ra( output layer) và trong bài toán phân

loại thì nó đại diện cho điểm số của các lớp.

Vấn đề mà mạng neural thường gặp phải: Không thể mở rộng đối với đầu vào có kích thước lớn. Dữ liệu hình ảnh có kích thước khá lớn, một tấm ảnh xám có kích thước  $[32 \times 32]$  (pixels) sẽ cho ra vector đặc trưng có 1024 chiều, còn đối với ảnh màu cùng kích thước sẽ là 3072 chiều. Điều này cũng có nghĩa là cần tới 3072 trọng số  $\theta$  nối giữa lớp vào và một neural ở lớp ẩn kế tiếp. Số lượng trọng số sẽ càng nhân rộng hơn nữa nếu số lượng neural trong lớp ẩn tăng lên, số lượng lớp ẩn tăng lên.

Trong bài toán nhận diện món ăn, một bức ảnh đầu vào có kích thước  $[96 \times 96 \times 3]$ , như vậy chúng ta cần tới  $96 \times 96 \times 3 = 27648$  trọng số  $\theta$  nối giữa lớp vào và một neural ở lớp ẩn kế tiếp. Như vậy sẽ cần đến một mô hình khá đồ sộ. Điều này khiến cho việc thao tác với các ảnh có kích thước lớn hơn trở nên khó khăn.

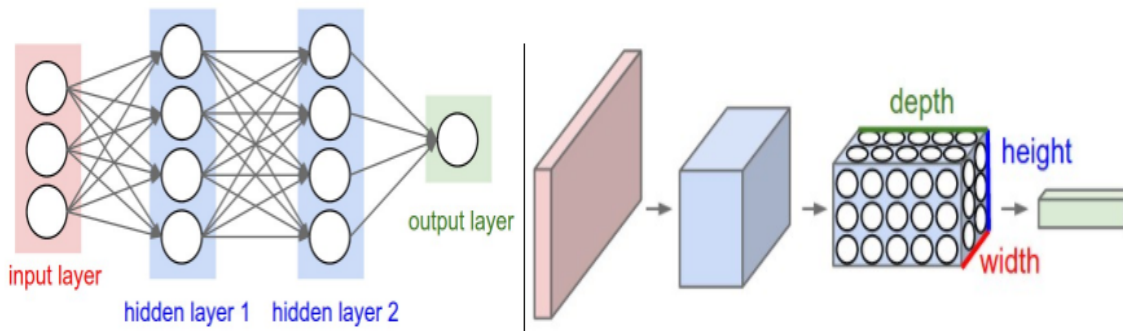
Một điều nữa là việc liên kết một cách đầy đủ các điểm ảnh vào một node trong mạng có vẻ là dư thừa vì sự phụ thuộc lẫn nhau giữa các điểm ảnh xa nhau là không nhiều mà chủ yếu là sự phụ thuộc giữa các điểm lân cận với nó. Dựa trên tư tưởng này mạng nơ-ron tích chập (Convolutional Neural Network) ra đời với một kiến trúc khác so với mạng truyền thẳng. Thay vì toàn bộ ảnh nối với một node thì chỉ có một phần cục bộ trong ảnh nối đến một node trong lớp tiếp theo (Local connectivity). Dữ liệu hình ảnh thông qua các lớp của mô hình này sẽ được “học” ra các đặc trưng để tiến hành phân lớp một cách hiệu quả.

Về cơ bản mô hình mạng nơ-ron tích chập bao gồm các lớp sau: Lớp Convolutional, Lớp RELU, Lớp Pooling, Lớp Fully connected. Sự sắp xếp về số lượng và thứ tự giữa các lớp này sẽ tạo ra những mô hình khác nhau phù hợp cho các bài toán khác nhau.

Nhận xét, đối với bài toán xử lý ảnh, việc sử dụng các kết nối đầy đủ này dường như là lãng phí và trong nhiều trường hợp, việc có quá nhiều tham số có thể sẽ dẫn đến tình trạng overfitting.

Mạng CNN có sự khác với mạng neural thường là ở các tầng của mạng bao gồm các neural được sắp xếp trong ba chiều: Chiều rộng, chiều cao, chiều sâu (chiều sâu ở đây là chiều thứ 3 của khối lượng kích hoạt chứ không phải là độ sâu của mạng). Và mỗi neural trong từng lớp sẽ chỉ kết nối tới một khu vực nhỏ của lớp trước đó thay vì kết nối đầy đủ tới tất cả các neural. Hơn thế nữa, lớp đầu ra (output layer) sẽ có kích thước  $1 \times 1 \times n$  (với  $n$  là số lượng nhân) vì tới cuối cùng của mạng CNN, chúng ta sẽ giảm chiều từ một ảnh đầy đủ tới một vector đơn chứa điểm số của các lớp, được sắp xếp dọc theo chiều depth.





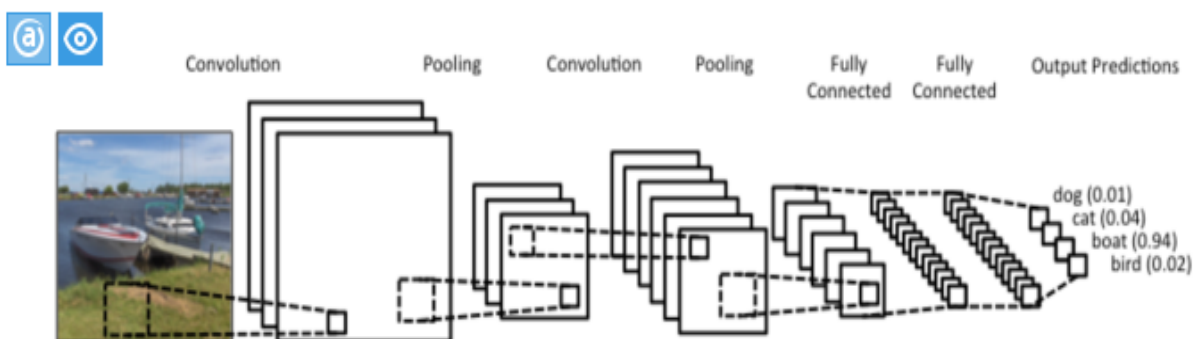
**Hình 3.** Hình bên trái là hình ảnh của 1 mạng neural thông thường với 2 lớp ẩn . Hình bên phải là minh họa cho 1 mạng CNN với việc các neural được sắp xếp theo 3 chiều . Mỗi khối input 3 chiều sẽ được biến đổi thành các output đầu ra cũng với 3 chiều ( trong ví dụ , đầu vào sẽ 1 bức ảnh với chiều sâu là 3 chiều màu : Red, Blue, Green )

## 3.2 Các tầng được sử dụng để xây dựng mạng CNN

Chúng ta sử dụng 3 loại lớp chính để xây dựng các kiến trúc convolution. Đó là lớp **Convolution**, lớp **Pooling** và lớp **Fully-Connected**. Sự sắp xếp các lớp này với số lượng và thứ tự khác nhau sẽ tạo ra các mô hình khác nhau phù hợp với những bài toán khác nhau.

Tổng quan :

- Một mô hình mạng CNN sẽ bao gồm các 1 danh sách các lớp để chuyển 1 khối đầu vào là 1 bức ảnh với đầu ra là 1 khối đầu ra ( chứa kết quả nhận diện trên các nhãn )
- Mỗi lớp sẽ nhận đầu vào là 1 khối input 3D và chuyển đổi thành khối output 3D thông qua các phương thức khác nhau



**Hình 4.** Mô hình mạng CNN

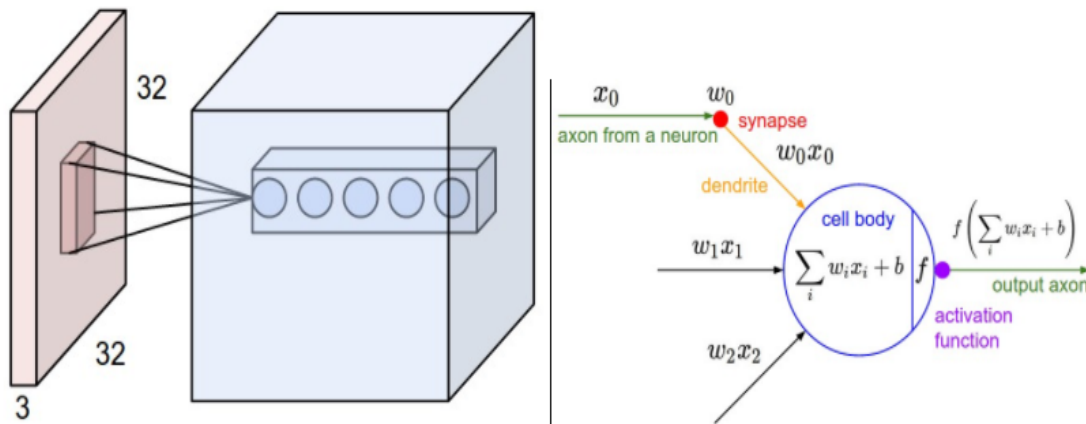
### 3.2.1 Lớp Convolution

*Các tham số của lớp Convolution* : bao gồm một bộ các bộ lọc (filter) có thể học được. Mỗi bộ lọc là một khu vực nhỏ (theo chiều rộng và chiều cao) nhưng kéo dọc theo toàn bộ chiều sâu của không gian đầu vào. Sau đó, chúng ta sẽ trượt lần lượt từng bộ lọc dọc theo chiều rộng và chiều cao của không gian đầu vào và tiến hành tính toán kết quả giữa các bộ lọc và đầu vào tại mọi khu vực. Khi chúng ta trượt các bộ lọc trong suốt chiều rộng và chiều cao của không gian đầu vào, chúng ta sẽ tạo ra bản đồ kích hoạt ( activation map) hai chiều để cho phép phản hồi của bộ lọc tại mọi vị trí.

*Kết nối cục bộ*: Khi làm việc với các dữ liệu nhiều chiều như hình ảnh, chúng đã thấy một thực tế rằng không phù hợp để kết nối đầy đủ một neural với tất cả các neural thuộc lớp trước đó. Thay vào đó, chúng ta sẽ kết nối chúng tới từng khu vực của khối lượng đầu vào. Phạm vi không gian kết nối này là một siêu tham số và được gọi là **receptive field** của neural( tương đương với kích thước bộ lọc). Một điều cần nhấn mạnh lại, đó là các kết nối nằm dọc theo không gian (theo chiều rộng và chiều cao) nhưng luôn luôn là đầy đủ đối với chiều sâu.

Ví dụ:

Giả sử rằng, không gian đầu vào có kích thước  $[32 \times 32 \times 3]$ , nếu receptive field(kích thước bộ lọc) là  $[5 \times 5]$  thì mỗi neural của lớp convolution có trọng số tương ứng với một khu vực  $[5 \times 5 \times 3]$  của không gian đầu vào, và có tổng cộng  $5 \times 5 \times 3 = 75$  trọng số (và thêm một trọng số bias).



**Hình 5.** Kết nối cục bộ tại lớp Convolution. Với mỗi neural ở tầng Conv sẽ được kết nối với 1 vùng không gian trong lớp input

*Kết quả đầu ra của lớp Conv* : Kích thước đầu ra của cũng như số lượng các neural , cách sắp xếp chúng phụ thuộc chính vào các tham số ( hyperparameters ) :

- + Chiều sâu : sẽ là số filter ( bộ lọc ) được sử dụng
- + Stride ( bước trượt ) : số bước trượt của bộ lọc
- + Zero-padding : Sẽ là các lớp input với các số 0 xung quanh viền của các input

Chúng ta có thể tính toán được kích cỡ của output thông qua các công thức sau :

- o Với kích thước của input là  $W$ , kích thước của “receptive field” là  $F$ , stride là  $S$ , số zero-padding sử dụng là  $P$ , ta sẽ tính được kích cỡ của output là :  **$(W - F + 2P) / S + 1$** .
- o Số lượng zero-padding và kích thước stride cũng phải được lựa chọn hợp lý :

Ví dụ :  $W = 5, F = 3, P = 1, S = 3$  . Lúc đó ta có kích thước của output sẽ là :

$$(5 - 3 + 2 \cdot 1) / 3 + 1 = 7 / 3 !!!$$

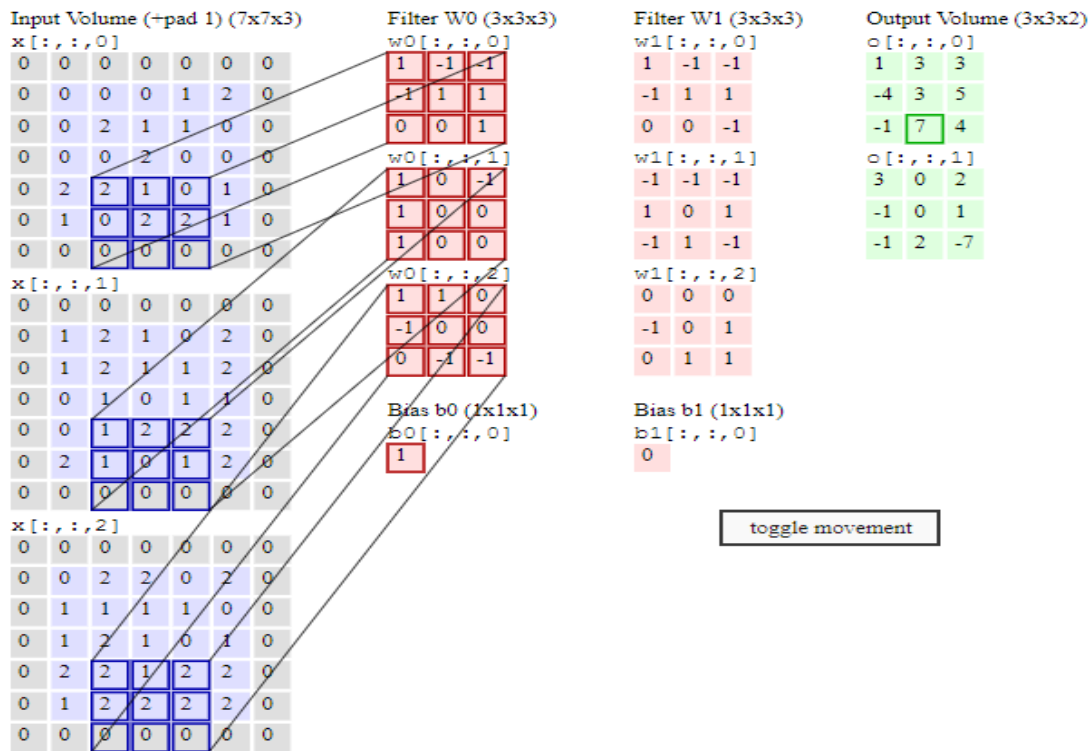
Thông thường việc xét giá trị zero-padding sẽ tính theo công thức :  **$P = (F - 1) / 2$** , với  $S = 1$  để luôn đảm bảo rằng kích thước của input và output luôn trong cùng 1 không gian như nhau.

*Parameter Sharing*: Lược đồ Parameter sharing được sử dụng để kiểm soát số lượng các tham số. Giả sử với , chúng ta có đầu vào với kích thước là  $[55 \times 55 \times 96]$  , vậy chúng ta sẽ có tổng cộng 290,400 neural trong tầng Conv, mỗi 1 neural lại có  $11 \times 11 \times 3 = 363 + 1 \text{bias}$  ( với filter có kích thước là  $11 \times 11 \times 3$  ) trọng số. Vậy tổng cộng sẽ có  $290400 \times 364 = 105,705,600$  tham số trong 1 lớp Conv !!!

Thay vào đó, chúng ta sẽ sử dụng việc giảm thiểu các tham số bằng cách sử dụng chúng chung cho từng chiều của 1 input đầu vào.

Giả sử : với đầu vào là  $[55 \times 55 \times 96]$ , ta sẽ tách được 96 slices ,với kích thước mỗi slice là  $[55 \times 55]$  , chúng ta sẽ sử dụng cho mỗi neural các trọng số và bias giống nhau, tất cả  $55 \times 55$  neural trong mỗi slice sẽ sử dụng chung tham số. Với việc giảm thiểu như vậy, chúng ta sẽ có 96 bộ trọng số khác nhau cho mỗi slice , và sẽ có tổng cộng là  $96 \times 11 \times 11 \times 3 = 34,848$  trọng số, hay 34,944 tham số (+96 biases) .

## Ví dụ lớp Convolution



Hình 6. Ví dụ lớp Convolution

Trong ví dụ trên, lớp Convolution nhận dữ liệu đầu vào có kích thước là  $[5 \times 5 \times 3]$  với việc sử dụng zero-padding là  $P = 1$  thì ta sẽ có không gian đầu vào là  $[7 \times 7 \times 3]$ . Sử dụng 2 bộ lọc có kích thước là  $3 \times 3 \times 3$  (chiều sâu là 3 vì không gian đầu vào có chiều sâu bằng 3). Mỗi bộ lọc chạy lần lượt bằng cách trượt trên từng khu vực có kích thước là  $3 \times 3$  (theo chiều rộng và chiều cao) dọc theo toàn bộ chiều sâu trên không gian đầu vào. Mỗi bộ lọc sẽ cho ra một vector 2 chiều  $3 \times 3$ . Cuối cùng, không gian đầu ra của lớp Convolution sẽ có kích thước là  $[3 \times 3 \times 2]$

### 3.2.2 Lớp RELU – rectified linear unit

Lớp này thường được cài đặt ngay sau lớp Convolutional. Lớp này sử dụng hàm kích hoạt  $f(x) = \max(0, x)$ . Nói một cách đơn giản, lớp này có nhiệm vụ chuyển toàn bộ giá trị âm trong kết quả lấy từ lớp Convolutional thành giá trị 0. Ý nghĩa của cách cài đặt này chính là tạo nên

tính phi tuyến cho mô hình. Tương tự như trong mạng truyền thẳng, việc xây dựng dựa trên các phép biến đổi tuyến tính sẽ khiến việc xây dựng đa tầng đa lớp trở nên vô nghĩa. Có rất nhiều cách để khiến mô hình trở nên phi tuyến như sử dụng các hàm kích hoạt sigmoid, tanh, ... nhưng hàm  $f(x) = \max(0, x)$  dễ cài đặt, tính toán nhanh mà vẫn hiệu quả.

### 3.2.3 Lớp Pooling

Thông thường, các lớp Pooling được chèn vào giữa các lớp Convolution liên tiếp trong kiến trúc mạng CNN. Chức năng của nó là để giảm dần kích thước không gian của biểu diễn để giảm số lượng các tham số và tính toán trong mạng, và do đó cũng kiểm soát overfitting.

Lớp Pooling hoạt động một cách độc lập theo chiều sâu của không gian đầu vào. Các phương thức lấy phổ biến trong lớp Pooling là MaxPooling (lấy giá trị lớn nhất), MinPooling (lấy giá trị nhỏ nhất) và AveragePooling (lấy giá trị trung bình). Các tham số yêu cầu là cửa sổ trượt [WxH] (thông thường là 2x2) và bước trượt (là giá trị để trượt các bộ lọc trên không gian đầu vào). Lớp này sử dụng một cửa sổ trượt quét qua toàn bộ ảnh dữ liệu, mỗi lần trượt theo một bước trượt (stride) cho trước.

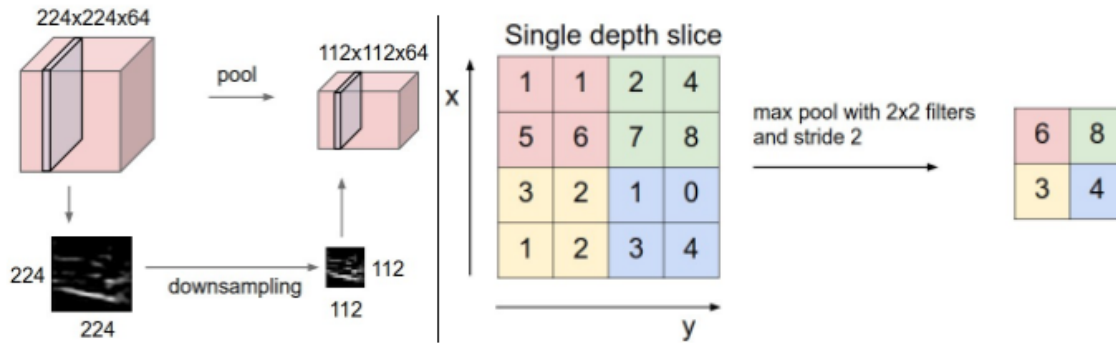
Trong bài toán nhận diện món ăn, nhóm chúng em sử dụng cửa sổ trượt có kích thước [2x2] và bước trượt = 2 và phương pháp sử dụng là MaxPooling. Filter sẽ lần lượt duyệt qua ảnh, với mỗi lần duyệt chỉ có giá trị lớn nhất trong 4 giá trị nằm trong vùng cửa sổ [2x2] của filter được giữ lại và đưa ra đầu ra. Chiều sâu của không gian đầu vào không bị thay đổi.

Một cách tổng quát, tầng Pooling hoạt động như sau:

- Nhận dữ liệu đầu vào với kích thước không gian là: **W1xH1xD1**

- Nhận giá trị trượt và kích thước bộ lọc
- Tiến hành tổng hợp và đưa ra đầu ra có kích thước không gian là:  **$W2 \times H2 \times D2$**  với:

$$\begin{aligned} W2 &= (W1 - \text{bước trượt}) / 2 + 1, \\ H2 &= (H1 - \text{bước trượt}) / 2 + 1, \\ D2 &= D1. \end{aligned}$$



**Hình 7.** Ví dụ tính toán lớp MaxPooling

### 3.2.4 Lớp Fully-Connected

Lớp này tương tự với lớp trong mạng neural truyền thẳng, các giá trị ảnh được liên kết đầy đủ vào node trong lớp tiếp theo. Sau khi ảnh được xử lý và rút trích đặc trưng từ các lớp trước đó, dữ liệu ảnh sẽ không còn quá lớn so với mô hình truyền thẳng nên ta có thể sử dụng mô hình truyền thẳng để tiến hành nhận dạng. Tóm lại, lớp fully-connected đóng vai trò như một mô hình phân lớp và tiến hành dựa trên dữ liệu đã được xử lý ở các lớp trước đó.

## 3.3 Ứng dụng của mô hình

Mạng nơ-ron tích chập được áp dụng khá nhiều trong các bài toán nhận dạng như nhận dạng vật thể trong ảnh, nhận dạng chữ viết tay (chuyển đổi chữ viết trong hình ảnh thành văn bản thô trong máy tính), nhận dạng vật thể 3D, xử lý tiếng nói, xử lý ngôn ngữ tự nhiên, .... với độ chính xác trên 90%.

Đã có rất nhiều bài báo khoa học đề xuất ra nhiều kiến trúc mạng nơ-ron tích chập khác nhau cho các bài toán khác nhau như: LeNet, AlexNet, VGGNet, GoogleNet, .... Tuy nhiên, sự phát triển của mạng nơ-ron tích chập đến một ngưỡng nào đó bị chậm dần do những hạn chế của mô hình. Kích thước và độ sâu của mạng nơ-ron khiến mạng nơ-ron trở nên không linh hoạt. Giả sử ta đã huấn luyện thành công mô hình nhận diện 10 đối tượng

khác nhau, một nhu cầu cần nhận diện đối tượng thứ 11 nảy sinh và để có thể nhận diện đối tượng thứ 1 này ta phải xây dựng một kiến trúc khác và huấn luyện lại từ đầu.

Dù vậy, trong thực tế, ta có thể chọn giải quyết các bài toán cụ thể mà nhu cầu mở rộng không quá lớn và không quá đòi hỏi độ linh hoạt cao (vd nhận diện các nhân viên cấp cao có quyền truy cập các hồ sơ quan trọng trong công ti, số lượng nhân viên cấp cao không quá nhiều và cũng không phải là thường xuyên thay đổi). Với những bài toán như vậy thì mạng nơ-ron tích chập vẫn là một mô hình hiệu quả.

## **4. Các thư viện sử dụng**

### **4.1 Keras**

Keras là thư viện nguồn mở được viết bằng Python dùng để xây dựng các mạng neural ở cấp độ cao cấp của interface. Thư viện này đơn giản và có khả năng mở rộng cao.

Keras sử dụng backend là Theano hoặc TensorFlow nhưng gần đây, Microsoft đã cố gắng tích hợp CNTK (Cognitive Toolkit của Microsoft) thành back-end mới.

Cách tiếp cận đơn giản về thiết kế nhằm đến quy trình experimentation dễ dàng, nhanh chóng từ việc build các compact systems.

Bắt đầu dùng Keras rất đơn giản, tiếp theo là prototyping nhanh. Keras được viết bằng Python, theo mô-đun và mở rộng được. Không chỉ đơn giản và có tính định hướng cao, Keras vẫn hỗ trợ modeling rất mạnh mẽ.

Ý tưởng chung về Keras là dựa trên các layers và mọi thứ khác cũng đều được xây dựng xung quanh các layer này. Data được chuẩn bị trong các tensors, layer đầu tiên chịu trách nhiệm về input của các tensors, layer cuối cùng chịu trách nhiệm output và model được build ở giữa.

Ưu điểm:

- Được tùy chọn chạy trên nền Theano hay TensorFlow
- Interface ở mức high level.
- Dễ xây dựng mô hình, dễ training.

Nhược điểm:

- Kém linh hoạt: Khó có thể tùy chỉnh mạng ở mức độ sâu.

## 4.2 Flask

Flask là một framework có tuổi đời khá trẻ và đang phát triển vô cùng mạnh mẽ. Điểm mạnh của Flask là sự nhỏ gọn, bạn có thể tạo nhanh một trang web để thử nghiệm các ý tưởng của mình một cách nhanh chóng.

# 5. Xây dựng mô hình

## 5.1 Mô hình CNN

Cấu trúc mô hình:

- Trong hệ thống, nhóm đã sử dụng 8 Lớp Conv với các số lượng các filter là khác nhau



cho mỗi cặp tầng :

- + Hai lớp Conv đầu tiên với filter có kích thước  $[64, (3 \times 3)]$  : 64 filter với kích thước là  $[3 \times 3]$

- + Hai lớp Conv thứ 2 với filter có kích thước  $[128, (3 \times 3)]$  : 128 filter với kích thước là  $[3 \times 3]$

- + Hai lớp Conv tiếp theo với filter có kích thước  $[256, (3 \times 3)]$  : 256 filter với kích thước là  $[3 \times 3]$

- + Hai lớp Conv cuối cùng với filter có kích thước  $[512, (3 \times 3)]$  : 512 filter với kích thước là  $[3 \times 3]$

Hàm kích hoạt *reLu*:  $f(x) = \max(0, x)$ . Được sử dụng trong mỗi 1 lớp Conv ở trên. Mục đích chính là đưa các giá trị âm trong kết quả của lớp Conv về giá trị 0, nhờ đó giúp cho việc training các *Deep Networks* nhanh hơn rất nhiều.

*Lớp Pooling* : Sau mỗi cặp lớp Conv ở trên, nhóm sẽ sử dụng 1 lớp pooling với kích thước là  $[2 \times 2]$  và với phương thức lấy là MaxPooling

- Nhóm sử dụng 2 lớp Fully-Connected :

- + Lớp đầu tiên với 512 neural , hàm activation là ReLU và sử dụng ***regularizers.l2*** với giá trị là 0.01, giá trị này được sử dụng để tránh cho mạng Neural Network tránh bị overfitting

- +Lớp cuối cùng sử dụng 10 neural tương ứng với số lượng các nhãn của tập train, và sử dụng hàm activation là softmax.

Hàm kích hoạt softmax:

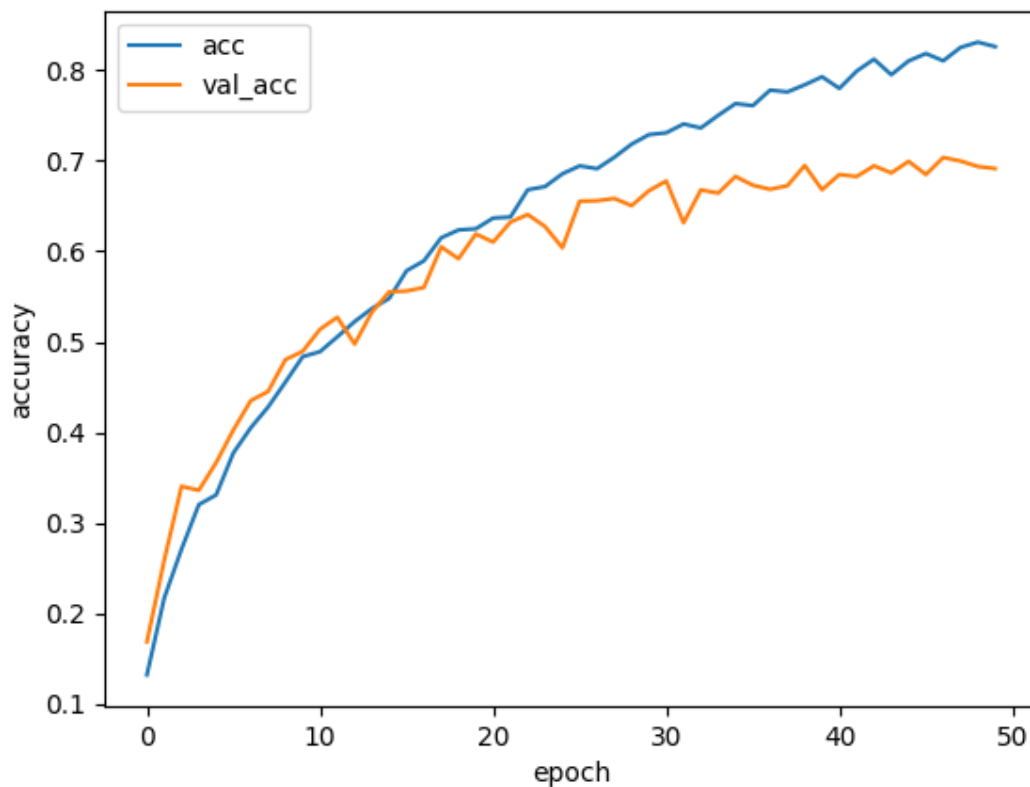
$$a_i = \frac{\exp(z_i)}{\sum_{j=1}^C \exp(z_j)}, \quad \forall i = 1, 2, \dots, C$$

Với  $\mathbf{z}_i = \mathbf{w}_i^T \mathbf{x}$

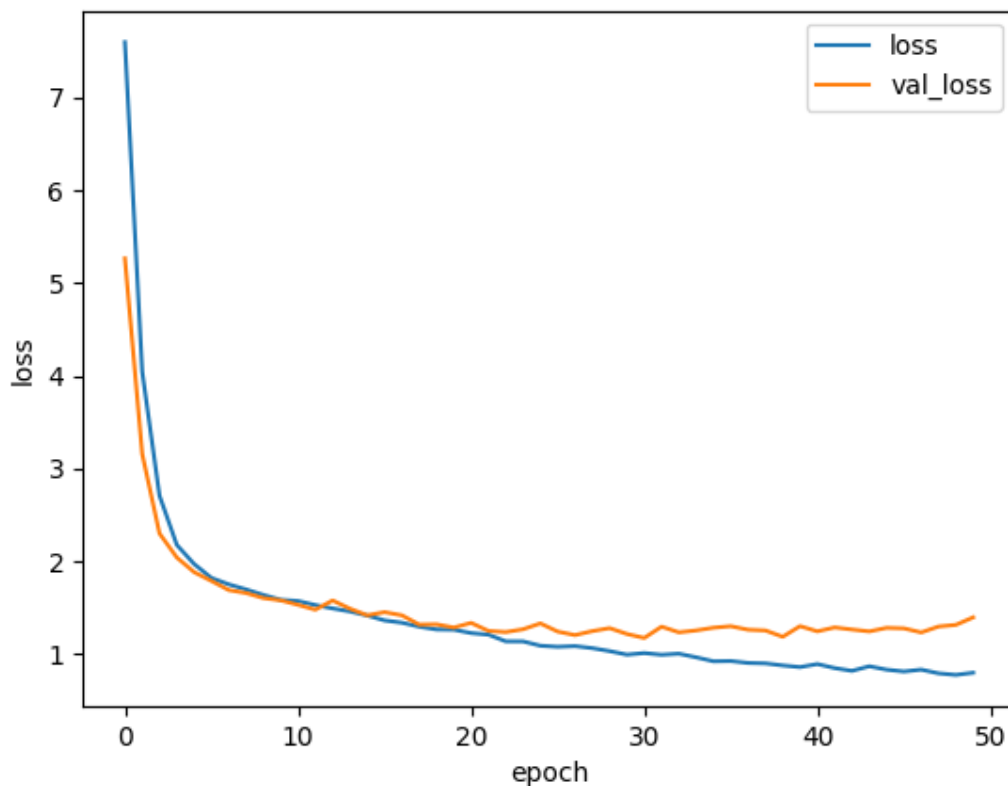
- Nhóm sử dụng hàm tối ưu là : Stochastic Gradient Descent. Với learning rate = 0.01 và momentum=0.9
- Hàm đánh giá lỗi là hàm : Cross Entropy

Cross entropy giữa hai phân phối p và q được định nghĩa là:  
 **$H(p,q)=E_p[-\log q]$**

### Kết quả dự đoán:




**Hình 8.** Kết quả của hàm đánh giá độ chính xác sau mỗi epoch



Hình 9. Kết quả thực nghiệm của hàm loss function sau mỗi epoch

## 5.2 Hệ thống website

Giao diện hệ thống:



HỆ THỐNG NHẬN DIỆN MÓN ĂN

Chọn món ăn cần nhận diện

Choose Files No file chosen

PREDICT

Tên món ăn: fried\_rice



Kết quả:

baby_back_ribs	0.00000006748517
<b>fried_rice</b>	<b>99.132888836</b>
hot_dog	0.00194680760526
ice_cream	0.000146391138068
pho	0.0000829678163
pizza	0.000440125269199
red_velvet_cake	1.25718110006e-05
seaweed_salad	4.47644059726e-05
spaghetti_bolognese	0.854289135743
sushi	0.000107802048005

## *Hình 10. Giao diện hệ thống website*

### 5.3 Tổng kết

Nhóm đã xây dựng thành công hệ thống nhận diện món ăn trên nền tảng website.

Mã nguồn của nhóm được lưu tại [https://github.com/lethanhloiHD/predict\\_food](https://github.com/lethanhloiHD/predict_food)

Tuy nhiên, độ chính xác của hệ thống chưa cao.

Lí do:

- Hệ thống chưa có độ sâu, độ lớn phù hợp.
- Dữ liệu cho việc học chưa nhiều
- Sử dụng thư viện Keras dẫn đến việc khó có thể tùy chỉnh ở mức độ cao tới các tầng.

Phương hướng phát triển:

- Đối với hệ thống nhận diện:
  - Tăng cường độ sâu, độ lớn cho hệ thống
  - Thu thập dữ liệu để đưa vào training hệ thống
  - Tìm hiểu công nghệ để có thể tùy chỉnh sâu hơn vào từng tầng
  - Xây dựng một hệ thống phù hợp cho toàn bộ 101 món của bộ dữ liệu và có thể là nhiều hơn nữa (hiện tại là 10 món).
- Đối với hệ thống website
  - Xây dựng giao diện tương tác mạnh mẽ với người sử dụng
  - Xây dựng hệ thống trên nhiều nền tảng khác nhau, chú trọng hơn nữa vào các ứng dụng mobile.

## Tài Liệu tham khảo

Slide môn Học máy – thầy Thân Quang Khoát, Viện Công nghệ thông tin và Truyền thông, Trường Đại học Bách khoa Hà Nội

<http://cs231n.github.io/convolutional-networks/>

<http://nhiethuyettre.net/mang-no-ron-tich-chap-convolutional-neural-network/>

Wikipedia.org

<https://blog.topdev.vn/top-15-thu-vien-python-tot-nhat-cho-data-science-trong-2017/>

<https://keras.io/>

## Lời cảm ơn

Sau một thời gian tìm hiểu, thảo luận, nghiên cứu đề tài cùng với sự giảng dạy của thầy **Thân Quang Khoát**, nhóm chúng em phần nào đã có cái nhìn tổng quan về học máy, tìm hiểu một công nghệ mới và ứng dụng nó vào một hệ thống website.

Tuy nhiên trong quá trình nghiên cứu, tìm hiểu và xây dựng hệ thống, nhóm chúng em không tránh khỏi những thiếu sót và nhiều vấn đề chưa được xử lý tối ưu. Nhóm rất mong nhận được sự đóng góp nhận xét của thầy cô và các bạn để đề tài này được hoàn thiện hơn.

Nhóm chúng em xin trân thành cảm ơn!