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

———————— * ————————

ĐỒ ÁN
TỐT NGHIỆP ĐẠI HỌC

NGÀNH CÔNG NGHỆ THÔNG TIN

Ứng dụng học sâu trong việc tự động xác định địa điểm du lịch nổi tiếng

Sinh viên thực hiện : **Lê Văn Mạnh**
Lớp KSVB2 – K37
Giáo viên hướng dẫn: **GV.Đinh Viết Sang**

HÀ NỘI 6-2019

MỤC LỤC

DANH MỤC BẢNG BIỂU

DANH MỤC HÌNH VẼ

CHƯƠNG 1 - MỞ ĐẦU

1.1. Nhiệm vụ của đồ án

Hiện nay để biết về một địa danh hay một điểm du lịch thông thường người dùng sẽ lên các trang tìm kiếm ví dụ google.com, bing.com …sau đó gõ từ khóa tên hoặc địa điểm du lịch muốn tới và sau đó đọc các thông tin liên quan tới địa điểm du lịch. Tuy nhiên, với sự bùng nổ của các mạng xã hội, các ứng dụng di động cùng với sự đa dạng về loại dữ liệu đặc biệt là dữ liệu ảnh nhiều trường hợp người dùng chỉ có một bức ảnh về địa điểm du lịch hoặc muốn tới một nơi có phong cảnh đẹp như trong bức ảnh mình đang có. Điều trên dẫn tới nhu cầu tìm kiếm thông tin địa danh thông qua hình ảnh ngày càn phổ biến.

Đồ án thực hiện trên dữ liệu ảnh về các địa điểm du lịch nổi tiếng, mỗi địa danh sẽ chứa khoảng 1000 ảnh kèm với thông tin về địa lý liên quan tới địa danh đó. Do giới hạn về thời gian và nền tảng phần cứng nên hệ thống xây dựng để nhận diện và gợi ý 64 địa điểm du lịch khác nhau trên lãnh thổ Việt Nam.

1.2. Phương pháp thực hiện

Với bài toán nhân diện thông tin qua ảnh việc lập trình truyền thống sẽ khó có độ chính xác cao do độ phức tạp và đặc thù của thông tin dữ liệu. Qua một thời gian tìm hiểu công nghệ, em lựa chọn phương pháp xây dựng hệ thống với phần lõi nhận diện ảnh sẽ sử dụng trí tuệ nhân tạo để đạt độ chính xác cao nhất có thể.

1.3. Ý nghĩa khoa học và thực tiễn

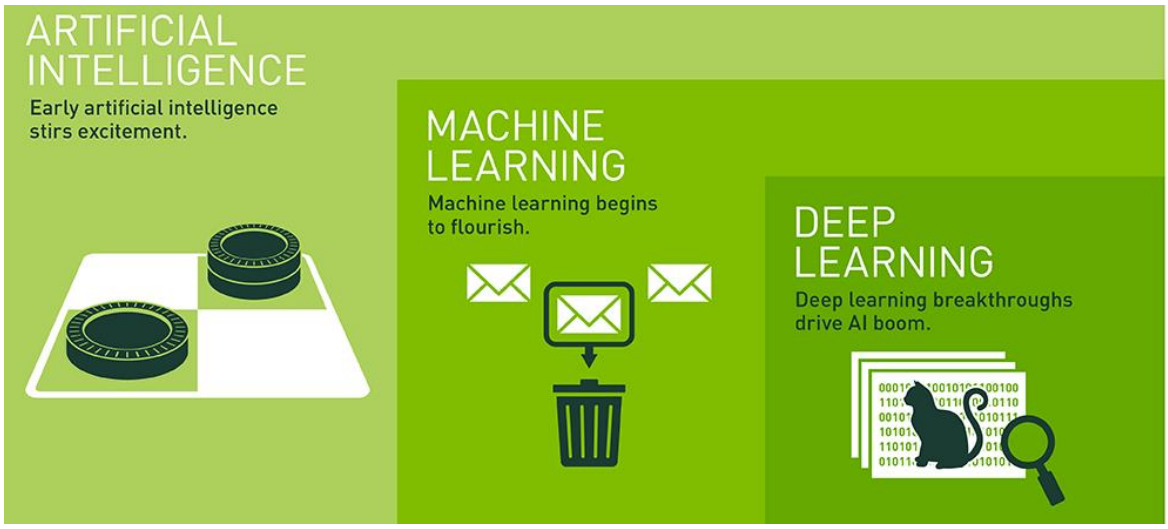
Người dùng khi xem ảnh có thể ngay lập tức tìm kiếm thông tin địa danh thông qua hình ảnh mình đang xem. Mọi thứ sẽ trờ lên nhanh chóng và thuận tiện cho người sử dụng góp phần thúc đẩy ngành dịch vụ và du lịch của Việt Nam.

1.4. Kết quả dự kiến

Xây dựng mạng neuron nhân tạo nhận diện 64 địa danh thông qua hình ảnh với độ chính xác trên 80%, triển khai hệ thống trên nền tăng web cho người dùng truy cập và upload ảnh lên sau đó trả lại kết quả cho người dùng trên giao diện website.

CHƯƠNG 2 - CONVOLUTIONAL NEURAL NETWORK

2.1. Tổng quan về mạng neural



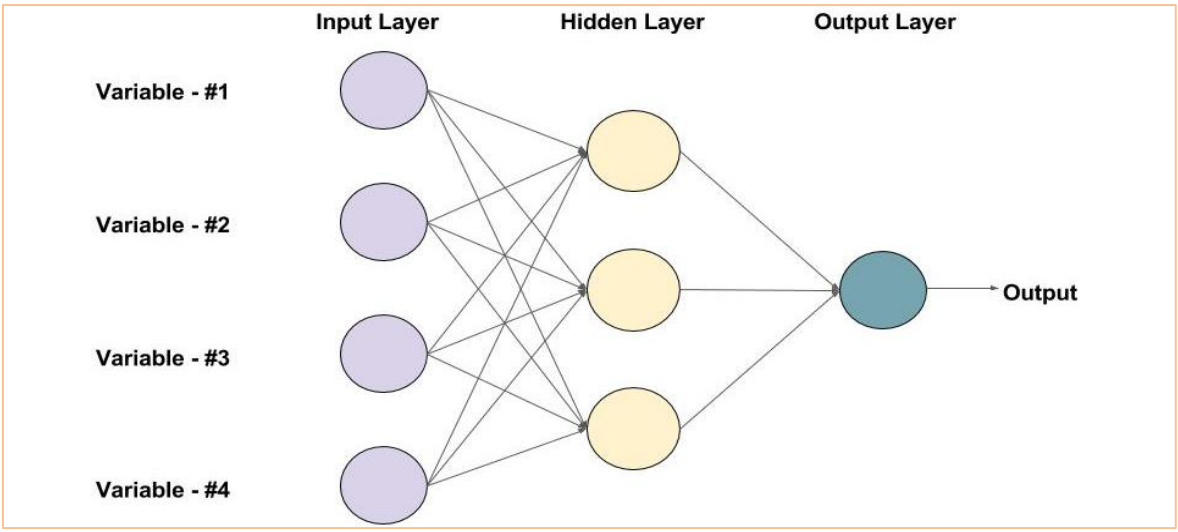
Hình 1 Tổng quan về trí tuệ nhân tạo

Trí tuệ nhân tạo là bất kỳ kỹ thuật nào cho phép máy tính để bắt chước hành vi của con người.

Học máy là tập con của trí tuệ nhân tạo, là việc máy có khả năng học mà không cần lập trình cụ thể.

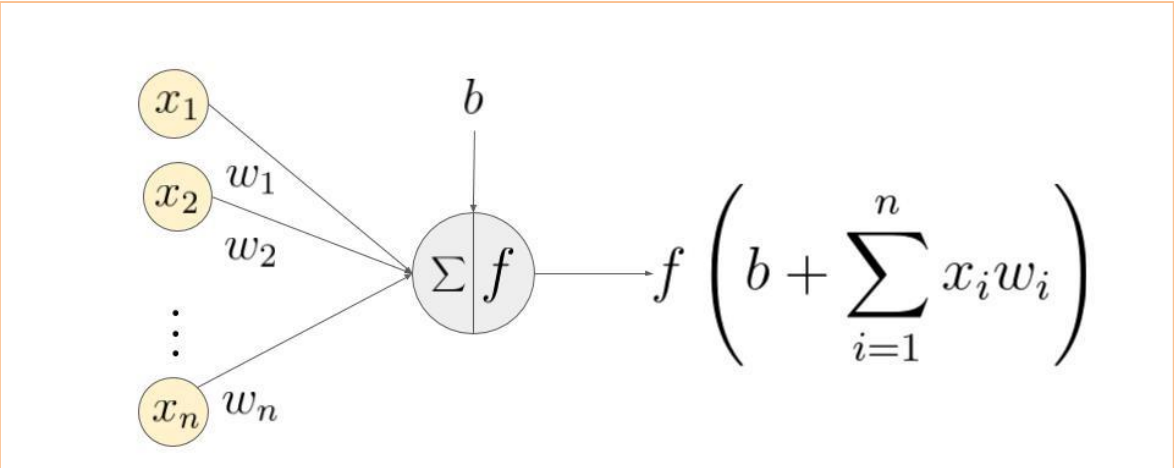
Học sâu là tập con của học máy, là việc máy có thể tự trích xuất ra các đặc trưng của dữ liệu thông qua mạng neural nhân tạo. Sau đây là định nghĩa về mạng neural network.

2.1.1 Cấu tạo của mạng neural nhân tạo



Hình 2 Mạng neural nhân tạo

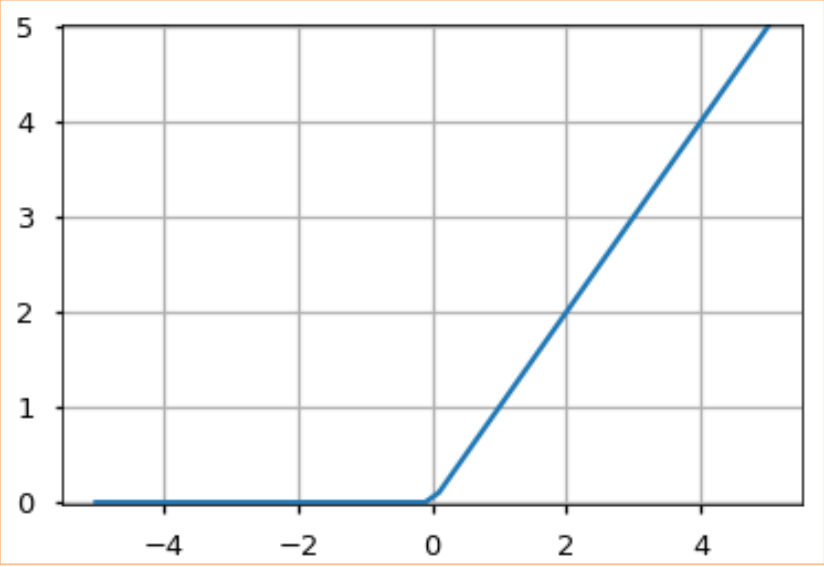
Mạng gồm lớp đầu vào, lớp đầu ra và một hoặc nhiều lớp ở giữa gọi là lớp ẩn (didden layer) các lớp được kết nối với nhau theo mô tả như hình trên. Mỗi một phần tử trong lớp ẩn gọi là một neural.



Hình 3 Neural trong mạng neural network

Một neural có đầu vào được kết nối với mọi phần tử đầu vào hoặc các neural ở lớp ẩn trước đó. Đầu ra của neural sẽ được nối với đầu ra của mạng hoặc các neural hớp lớp ẩn phía sau.

Kết quả đầu ra của neural sẽ là kết quả của hàm f được mô tả trên hình. Trong đó các giá gị x là đầu vào của neural, w là các con số đại diện cho một kết nối gọi là **weight**, b là **bias** đóng vai trò hiệu chỉnh, f là một hàm số gọi là hàm **activation**. Có nhiều hàm activation trong đó phổ biến nhất là hàm Relu.



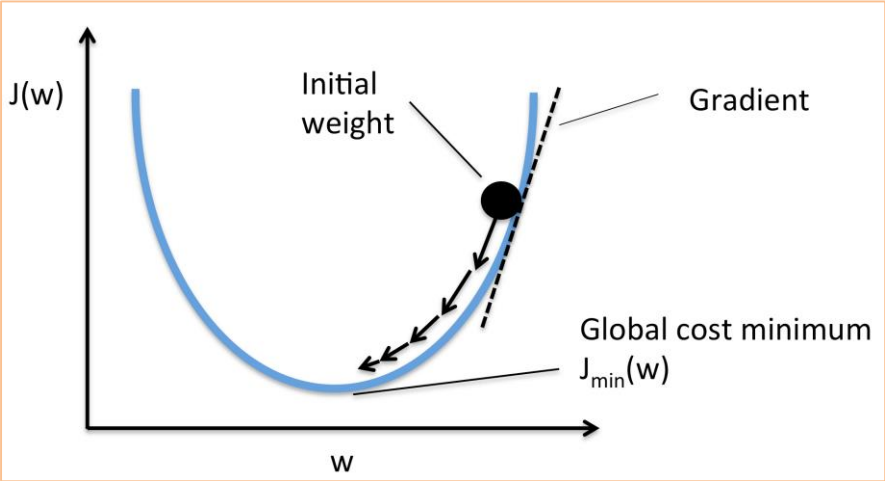
Hình 4 Hàm Relu

Giá trị đầu ra của hàm sẽ bằng giá trị đầu vào khi giá trị đầu vào lớn hơn 0. Giá trị đầu ra của hàm sẽ là 0 nếu giá trị của đầu vào nhỏ hơn hoặc bằng 0.

2.1.2 Huấn luyện mạng neural nhân tạo

+ Định nghĩa hàm mất mát: Với bài toán học có giám sát, trong quá trình huấn luyện, mạng neural nhân tạo ta sử dụng dữ liệu đã được gán nhãn sẵn, với mỗi dữ liệu đầu vào x sẽ có một nhãn tương ứng là y. Kết quả dự đoán của mạng đối với đầu vào x sẽ là $\bar{y}$. Dựa vào $\bar{y}$ và y ta thu được một hàm gọi là hàm mất mát J, hàm này sẽ là hàm số của w và b được nêu ở trên.

+ Cơ sở của việc hấn luyện mạng chính là: việc máy tìm ra các giá trị của w và b sao cho giá trị hàm mất mát (loss function) nhỏ nhất đối với một tập dữ liệu học cho trước. Để tối ưu hàm mất mát ta sử dụng thuật toán có tên gọi là **Gradient Descent**.



Hình 5 Minh họa thuật toán gradient descent

Trên hình minh họa việc tìm w để hàm J có giá trị nhỏ nhất, đây là trường hợp cho hàm một biết. Ban đầu, w nhận một giá trị bất kì sau khi di chuyển theo chiều ngược với chiều của đạo hàm ở điểm hiện tại thì ta luôn tìm được giá trị w sao cho hàm J có giá trị bé hơn. Lặp lại việc di chuyển theo nguyên tắc ngược chiều của đạo hàm phía trên ta sẽ tìm được w sao cho J đạt giá trị nhỏ nhất.

Vậy giả sử ta cần tìm tham số $\theta \in R^n$ để hàm mất mát J(θ) đạt giá trị nhỏ nhất ta tiến hành bước lặp như sau

$$\theta^{(k+1)} = \theta^{(k)} - \eta \nabla_\theta J(\theta)$$

Trong bài toán tối ưu mạng neural η được gọi là **learning rate**

Với hàm nhiều biến thì thay vì tính đạo hàm thông thường, ta sẽ tính đạo hàm riêng để tinh chỉnh từng biến số của hàm nhiều biến. Kết quả là sẽ thu được giá trị cho từng biến số để hàm mất mát có giá trị nhỏ nhất.

Với mạng có nhiều lớp ẩn thì việc tính giá trị đạo hàm của hàm mất mát theo từng weight và bias được thực hiện thông qua phép tính đạo hàm của hàm hợp.

$$J = g(f(x)) \Rightarrow \frac{dJ}{dx} = \frac{dJ}{df}.\frac{df}{dx}$$

Có nhiều thuật toán tối ưu hàm mất mát nhưng đa số đều dựa vào nguyên lý của thuật toán gradient descent.

Quá trình để mạng tìm ra các tham số weight và bias được gọi là quá trình huấn luyện mạng neural. Kết thúc quá trình huấn luyện ta thu được tập hợp các weight và bias. Các giá trị này sẽ được lưu trong một tập tin gọi là **model** cùng với cá tham số cấu hình mạng.

2.1.3 Một số thuật toán tối ưu hàm mất mát

+ Biến thể của gradient descent: Gradient descent truyền thống sẽ tính giá trị điều chỉnh tham số của một tham số cần học cho toàn bộ tập dữ liệu học.

Trong thực tế điều này trong nhiều trường hợp là bất khả thi do dữ liệu học có thể sẽ trở lên quá lớn so với tài nguyên bộ nhớ. Điều này dẫn tới việc chúng ta phải cho từng phần dữ liệu vào để huấn luyện mô hình dẫn tới hàm mất mát theo các trọng số học máy sẽ được tối ưu theo từng lần đưa dữ liệu vào (gọi là vòng lặp) khác nhau dưới đây là hai cách huấn luyện chính.

+ Stochastic gradient descent: Là thuật toán tối ưu hàm mất mát theo từng phần tử của tập dữ liệu. Nhược điểm của phương pháp này là giá trị đạo hàm biến thiên lớn dẫn tới mạng không ổn đinh. Các tham số trong mạng có khoảng biến thiên lớn.

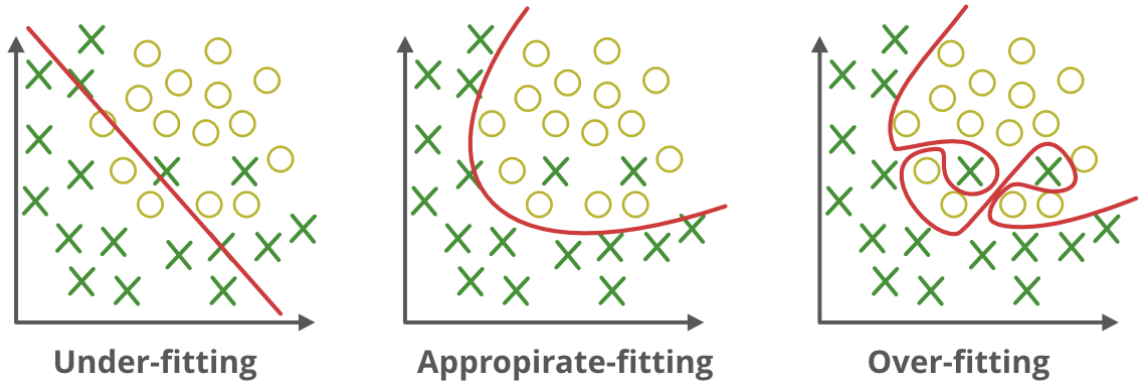
+ Mini Batch Gradient Descent: Thay vì dùng chỉ một phần tử đem ra huấn luyện để cập nhật giá trị cho các trọng số trong mỗi bước lặp. mạng sẽ được đưa vào một lượng các phần tử nhất định để cập nhật các trọng số. ưu điểm của phương pháp này là giảm độ lệch chuẩn của các tham số cập nhật.

+ Các thuật toán tối ưu hàm mất mát: Trong quá trình học máy, giá trị đạo hàm theo hàm mất mát sẽ biến thiên theo từng thời điểm. Có nhưng lúc giá trị

đạo hàm rất thấp nhưng đó không phải là điểm tối ưu của hàm mất mát. Việc đặt hệ số học (learning rate) cố định cho toàn bộ các bước lặp dẫn tới việc đạo hàm lớn quá gây hiện tượng bùng nổ đạo hàm (exploding gradient) hoặc hiện tượng biến mất của đạo hàm (vanishing gradient).

Hai hiện tượng phía trên dẫn tới việc không tìm được điểm cực tiểu cho hàm mất mát. Các hàm tối ưu hiện đại sẽ giải quyết cho ta hiện tượng trên giá trị đạo hàm của vòng lặp sẽ phụ thuộc vào giá trị đao hàm tính được từ hàm mất mát và giá trị đạo hàm của các vòng lặp trước đó. Mộ số thuật toán tối ưu hàm mất mát là Momentum, Adagrad, AdaDelta, Adam ... trong đó thuật toán Adam được dùng phổ biến nhất.

2.1.4 Overfitting và Underfitting



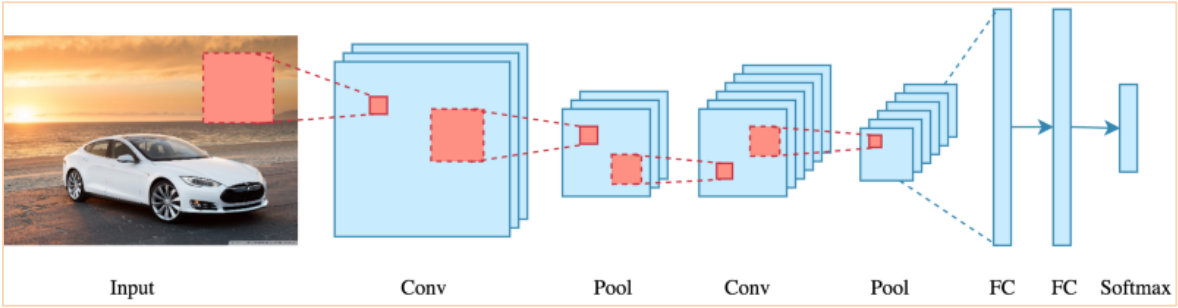
Hình 6 Overfitting và Underfitting

Underfitting là hiện tượng mạng quá đơn giản và không học được các đặc trưng của dữ liệu. Khi huấn luyện mạng thì mạng sẽ đưa ra dự đoán sai lớn trên cả tập dữ liệu học và tập dữ liệu kiểm thử

Overfitting là hiện tượng mạng quá phức tạp dẫn tới việc dự đoán rất tốt trên tập huấn luyện nhưng sai số trên tập kiểm thử lại lớn.

2.2. Tổng quan về mạng CNN

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. CNN được sử dụng nhiều trong các bài toán nhận dạng các object trong ảnh.



Hình 7 Sơ đồ tổng quát CNN

Sau đây là một số đặc điểm trong cách thức hoạt động của mạng CNN.

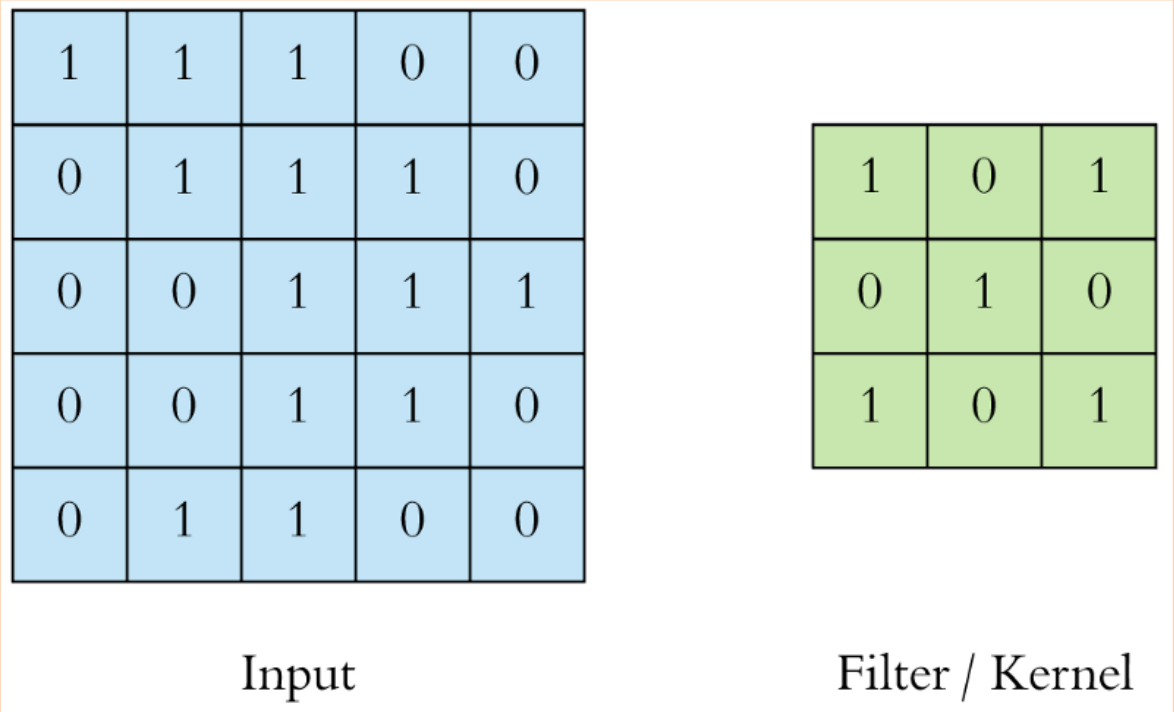
Local receptive fields: Trong mạng neural network truyền thống mỗi một neural trong input layer kết nối với một neural trọng hidden layer. Tuy nhiên trong CNN chỉ một vùng xác định trọng các neural trong input layer kết nối với một neural trong hidden layer. Những vùng xác định nêu trên gọi là Local receptive fields. Sự kết nỗi giữa input layer và hidden được chính là viêc từ Local receptive fields trên một ảnh đầu vào được biến đổi thông qua một phép toán được gọi là convolution để thu được một điểm trên hidden layer.

Shared weights và biases: Giống với mạng neural network truyền thống CNN cũng có tham số weights và biases. Các tham số này được học trong suốt quá trình trainning và liên tục cập nhật giá trị với mỗi mẫu mới (new trainning example). Tuy nhiên, các trọng số trong CNN là giống nhau đối với mọi neural trọng cùng một lớp (layer). điều này có nghĩ là tất cả các hidden neural trong cùng một lớp đang cùng tìm kiếm trung một đặc trưng (ví dụ như cạnh của ảnh) trong các vùng khác nhau của ảnh đầu vào.

Activation và pooling: Activation là một bước biến đổi giá trị đầu ra của mỗi neural thông qua việc sử dụng một số hàn ví dụ hàm ReLU. Giá trị thu được sau phép biến đổi là giá trị dương nhất có thể của output, trong trường hợp ouput mang giá trị âm thì giá trị nhận được là 0.

pooling là một bước nhằm giảm số chiều của ma trận, các thức phổ biến nhất là từ một vùng trên ma trận ta chọn ra số có giá trị lớn nhất làm kết quả thu được sau bước pooling (max pooling)

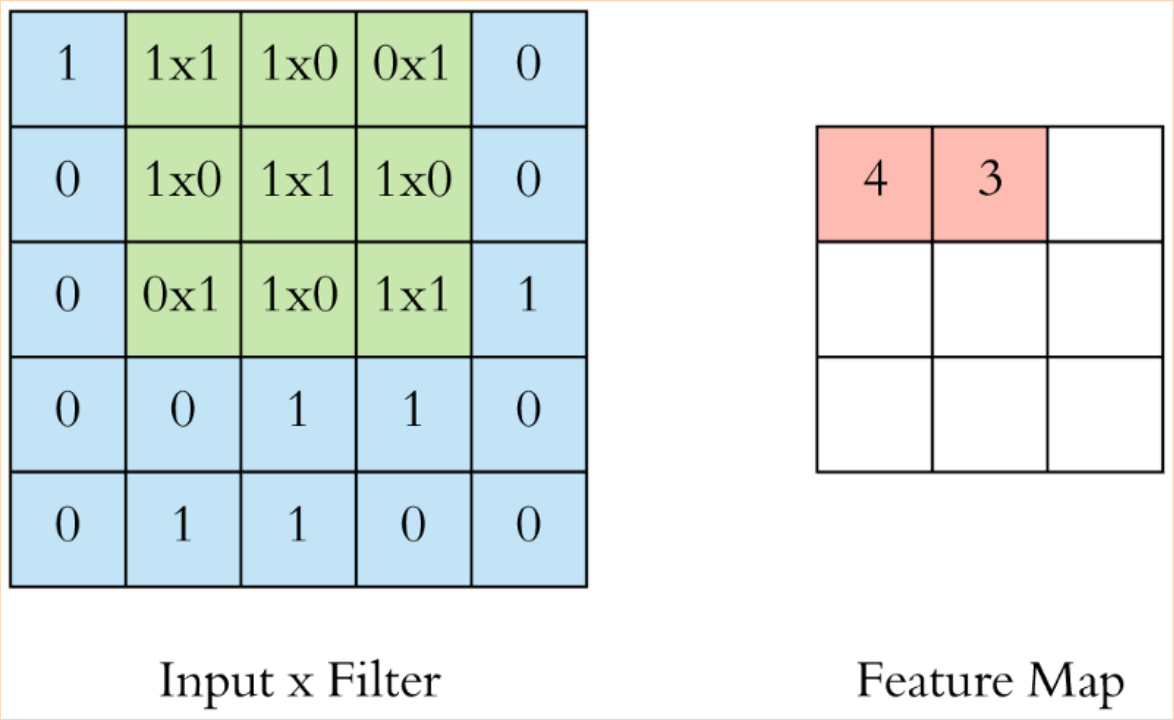
2.2.1 Định nghĩa convolution



Hình 8 Khái niệm convolutional

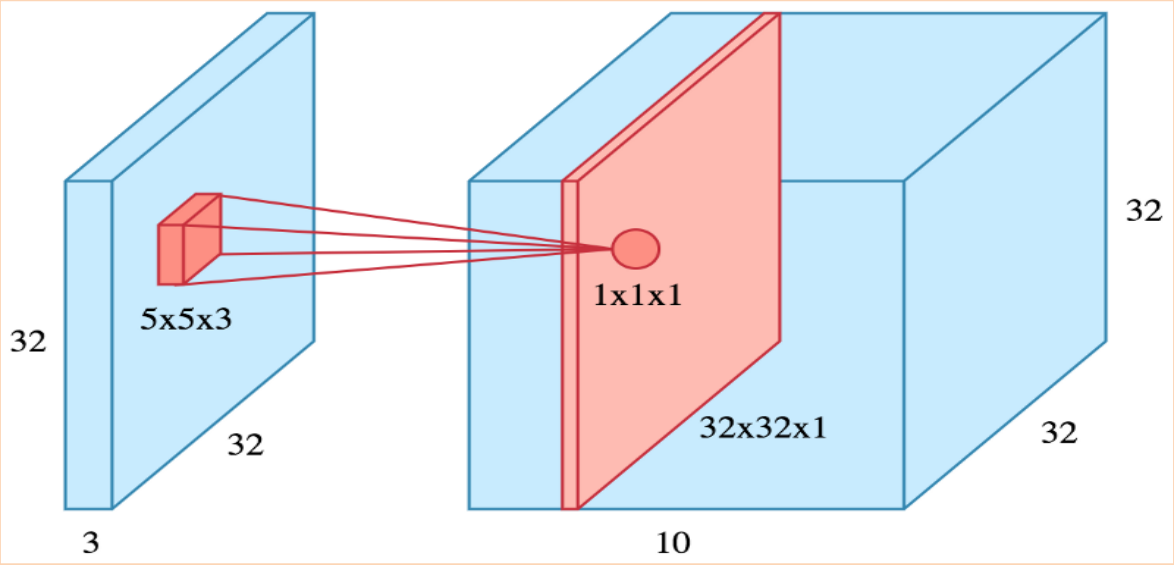
Khối cơ bản tạo lên CNN là convolutional layer. Convolution là một phép toán học để kết hợp hai khối thông tin với nhau. Trong trường hợp này, convolution được áp dụng trên dữ liệu đầu vào (ma trận) và sử dụng một mặt lạ gọi là convolution filter để tạo ra một mảng mới gọi là feature map.

Việc thực hiện phép toán convolution được mô tả như hình dưới dây với đầu vào là một mảng hai chiều 5x5 phần tử là filter có kích thước là 3x3 phần tử. Của sổ filter sẽ được trượt từ trái qua phải, từ trên xuống dưới. Tại mỗi vị trí của cửa sổ filter ta thực hiện nhân tương ứng từng phần tử trong ma trận đầu vào với từng phần tử trong filter, sau đó cộng tổng các tích với nhau ta thu được kết quả là một phần tử trên feature map. Quá trình thực hiện được mô tả trong hình minh họa sau đây.



Hình 9 Convolutional và mảng hai chiều

Trên đây là mô tả thực hiện phép toán convolution với ma trận hai chiều với một filter duy nhất. Trong thực tế đối với ảnh RGB ta thực hiện convolution với ma trận ba chiều ví dụ như ảnh RGB và với cùng một ảnh đầu vào ta áp dụng phép toán convolution với nhiều filter khác nhau. Mỗi một filter được áp dụng cho ta một feature layer. Nhiều feature layer xếp trồng lên nhau ta thu được một convolution layer. Ví dụ sau thể hiện ảnh có kích thước 32x32 và có ba kênh mầu, ta sử dụng 10 filter và thu được convolution layer là một ma trận 32x32x10.



Hình 10 Convolutional và mảng ba chiều

2.2.2 Định nghĩa stride và padding

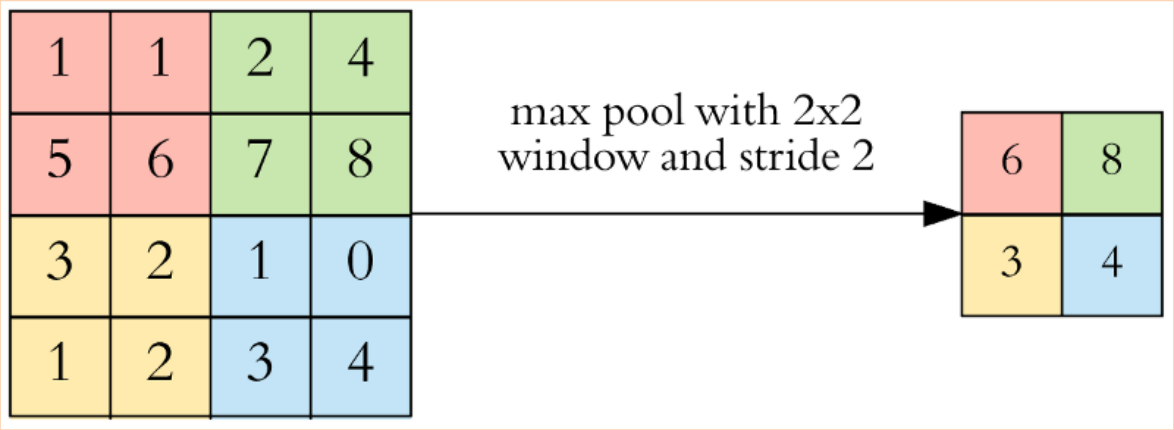
Stride là số bước nhảy của mỗi lần dịch chuyển convolution filter, trong ví dụ đầu tiên về convolution ta nhận thấy kích thước của feature map nhỏ hơn kích thước của ma trận đầu vào. Để kích thước của feature map bằng kích thước của ma trận đầu vào ta cần phải bổ xung thêm một số điểm bao quanh ma trận đầu vào thường là thêm các phần tử 0 vào xung quanh ma trận đầu vào, thao tác trên được gọi là padding.

Khi thực hiện phép toán convolution với đầu vào là ma trận vuông có kích thước là nxn, stride là s, kích thước filter là fxf, vùng padding có kích thước là p ta có kích thước của feature map thu được là:

$$Output\ size = \left(\frac{n+2p-f}{s}+1\right) x \left(\frac{n+2p-f}{s}+1\right)$$

2.2.3 Lớp pooling trong mạng CNN

Sau khi thực hiện phép toán convolution chúng ta thường sủ dụng pooling nhằm giảm số chiều của dữ liệu. Loại pooling thông dụng nhất là max pooling tức là trong một vùng được chọn (pooling window) được chọn của ma trận, ta lấy phần tử có kích thước lớn nhất, cũng giống với convolution thì pooling window cũng được định nghĩa kích thước (size) và bước nhảu (tride). Dưới đây là ví dụ việc áp dụng max pooling sử dụng 2x2 window và stride là 2.



Hình 11 Minh họa max pooling

Các lớp phía sau cùng FC viết tắt của fully connection neural, chính là các lớp ẩn trong mạng neural truyền thống. Cuối cùng đầu ra thường được cho qua hàm có tên là softmax. Hàm này có tác dụng chuyển một vector đầu vào thành một vector đầu ra có cùng kích thước, các phần tử nằm trong khoảng 0 và 1, tổng các phần tử sẽ có giá trị là 1. Trong bài toán phân loại ảnh thì giá trị đầu ra của softmax chính là sác xuất rơi vào mỗi loại tương ứng khi đem một ảnh làm đầu vào của mạng để dự đoán.

CHƯƠNG 3 – BÀI TOÁN NHẬN DẠNG ĐỊA DANH

3.1. Đầu vào và đầu ra của bài toán

3.1.1 Phân tích dữ liệu đầu vào

Đầu vào bài toán em chia thành hai dạng đó là đầu vào dữ liệu (dataset) dùng cho việc huấn luyện mạng neuron và đầu vào của bài toán cần phải nhận diện.

Dữ liệu đầu vào dùng cho huấn luyện mạng hay còn gọi là dataset là tập ảnh có kích tướng 480x480x3, đây là ảnh RGB về các địa danh du lịch nổi tiếng của Việt Nam, được download từ trang https://challenge.zalo.ai .Như đã nói ở phần mởi đầu trong phần pham vi của đồ án, dữ liệu đầu vào chứa 64 loại ảnh về 64 địa danh tương ứng khác nhau. Các ảnh được lưu chữ trong từng thư mục khác nhau với mỗi thư mục chứa khoảng 1000 ảnh. Quá trình huấn luyện mạng sẽ chia tập dữ liệu thành hai phần, trong đó 70% dùng cho việc huấn luyện mạng và 30% dùng cho việc kiểm định độ chính xác của mạng. Trong quá trình huấn luyện và kiểm định độ chính xác của ảnh, các ảnh sẽ được thay đổi kích thước một cách hợp lý để phù hợp với thiết kế của mạng neuron. Do

tập ảnh với số lượng 1000 cho mỗi địa danh là chưa đủ lớn để tăng số lượng ảnh cho việc huấn luyện em sẽ sử dụng một số kĩ thuật như lật, xoay ảnh gốc để thu được một ảnh khác nhằm tăng số lượng ảnh cho việc huấn luyện.

Dữ liệu đầu vào dành cho việc nhận dạng địa danh là ảnh của địa danh có kích thước bất kìa với định dạng ảnh yêu cầu là RGB.

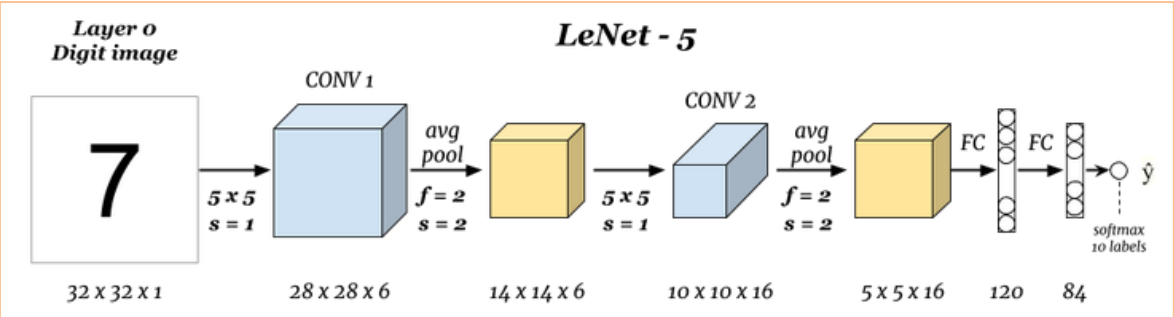
3.1.2 Một số kĩ thuật augmentation

Các kĩ thuật augmentation được sử dụng nhằm mục đích làm cho phong phú tập dữ liệu có sẵn. Ví dụ trong dataset có số lượng phần tử của mỗi loại không đồng đều hay lượng ảnh đại diện cho mỗi loại đó có số lượng ít. Một số kỹ thuật phổ biến như Flip, Rotation, Scale, Crop, Translation … Trong đồ án này em dùng kĩ thuật Crop để dữ liệu đầu vào dược cân bằng giữa các loại (class) khá nhau.

3.2. Một số kiến trúc mạng convolutional neural network

3.2.1 Kiến trúc mạng LeNet-5

LeNet-5 là một mạng cổ điển và cơ bản nhất cho bài toán nhận diện ảnh, được ứng dụng cho bài toán nhận diện số và chữ viết tay với đặc điêm dữ liệu đầu là ảnh có độ chi tiết đơn giản và kích thước nhỏ. Kiến trúc được công bố trong bài báo của Y. Lecun, L. Bottou, Y. Bengio and P. Haffner vào năm 1998.



Hình 12 Sơ đồ kiến trúc mạng LeNet-5

Sau đây là bảng tóm tắt kiến trúc của mạng, gồm các lớp cùng với các tham số cần phải học tương ứng.

Layer	Filters	Biases	Stride	Tensor	Parameteres
Input	0	0	0	32x32x1	0
Conv-1	5x5x1x6	6	1	28x28x6	156

MaxPool-1	2x2	0	2	14x14x6	0
Conv-2	5x5x6x16	16	1	10x10x16	2416
MaxPool-2	2x2	0	2	5x5x16	0
FC-1	400x120	120	0	120	48120
FC-2	120x84	84	0	84	10164
RBF	0	0	0	10	0
Total					60856

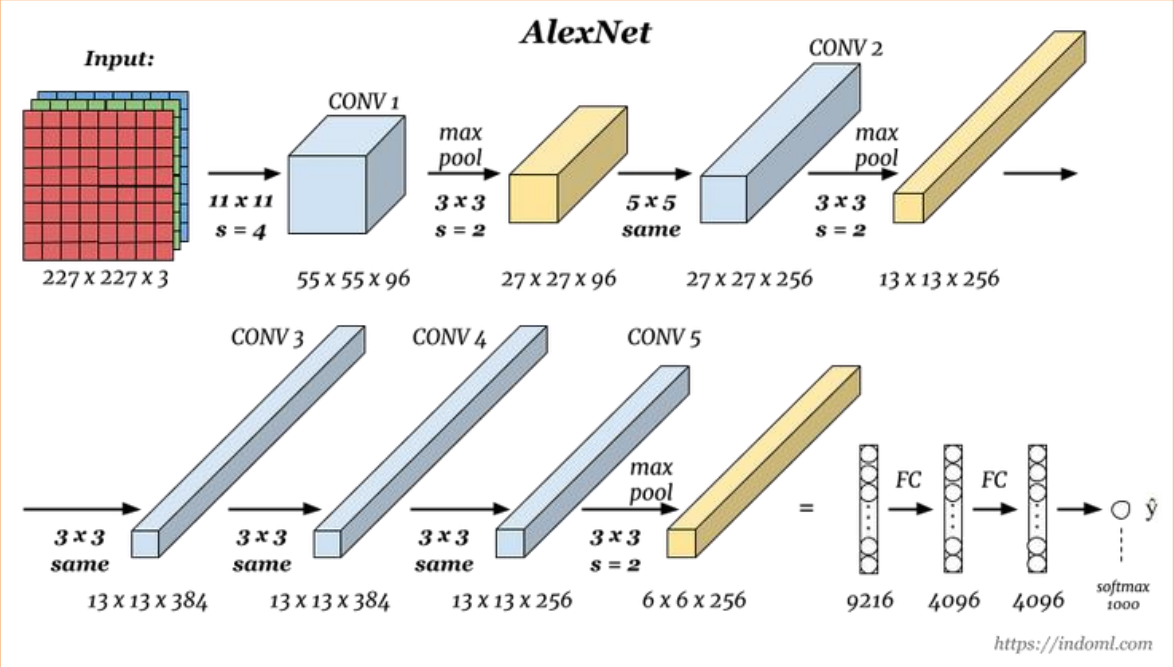
Bảng 1 Kiến trúc mạng LeNet-5

Đầu vào là ảnh Gray với kích thước 32x32 pixel và đâu ra là khoảng các Euclidean giữa mỗi vector đầu vào vector trọng số tức tensor đầu ra của FC-2.

Ta có thể thấy đâu là mạng rất đơn giản với kích thước đầu vào cũng như số lượng các tham số phải học. Tiếp theo chúng ta tìm hiểu một kiến trúc phổ biến khác nhưng phức tạp hơn mạng LeNet-5.

3.2.2 Kiến trúc mạng AlexNet

Mạng AlexNet. Dùng để phân loại 1000 loại ảnh có kích thước 227x227x3. Kiến trúc được công bố bởi Alex Krizhevsky, Geoffrey Hinton, and Ilya Sutskever vào năm 2012



Hình 13 Sơ đồ kiến trúc mạng AlexNet

Sau đây là bảng tóm tắt các thông số của mạng chứa thông tin về kích thước tensor đầu ra của từng lớp cùng với số lượng tương ứng các tham số mà mạng cần phải học.

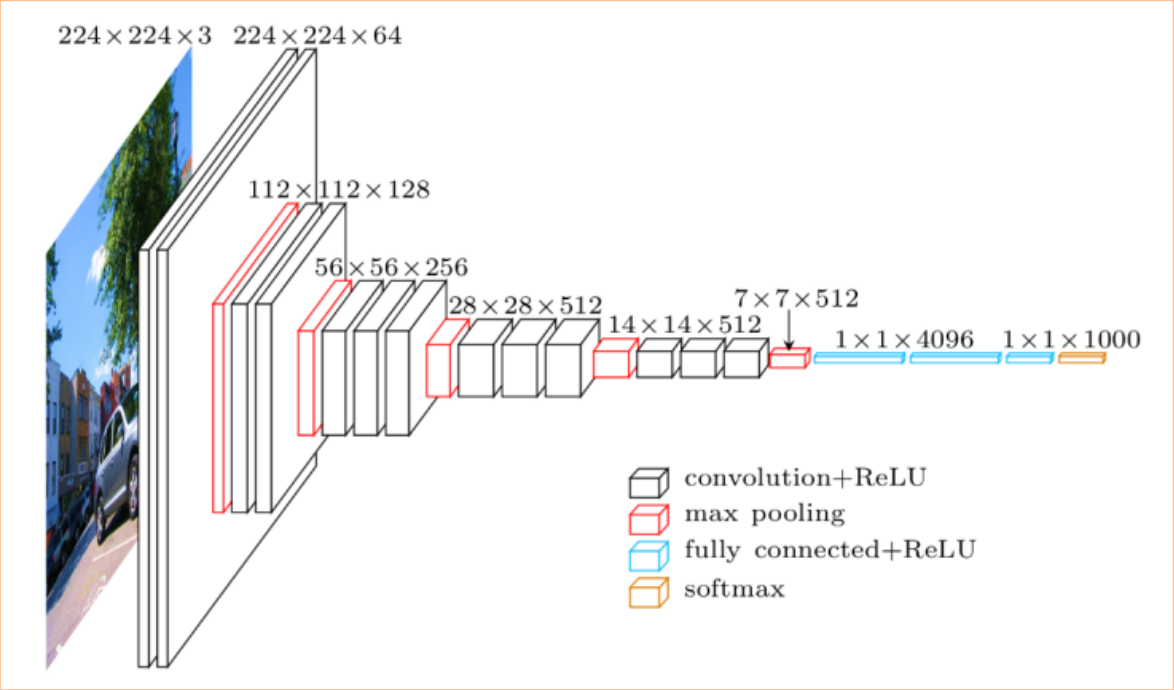
Layer	Filters	Biases	Stride	Tensor	Parameteres
Input	0	0	0	227x227x3	0
Conv-1	11x11x3x96	96	4	55x55x96	34944
MaxPool-1	3x3	0	2	27x27x96	0
Conv-2	5x5x96x256	256	1	27x27x256	614656
MaxPool-2	3x3	0	2	13x13x256	0
Conv-3	3x3x256x384	384	1	13x13x384	885120
Conv-4	3x3x384x384	384	1	13x13x384	1327488
Conv-5	3x3x384x256	256	1	13x13x256	884992
MaxPool-3	3x3	0	2	6x6x256	0
FC-1	9216x4096	4096	0	4096	37752832
FC-2	4096x4096	4096	0	4096	16781312
FC-3	4096x1000	1000	0	1000	4097000
Output	0	0	0	1000	
Total					62378344

Bảng 2 Kiến trúc mạng AlexNet

Mạng với đầu vào là ảnh có kích thước 227x227x3 và kết quả đầu cần thực hiện là phân loại 100 ảnh khác nhau, tổng số tham số cần phải học của mạng là 62378344 lớn hơn so với mạng LeNet-5. Các trọng số được cập nhập thông qua quá trình tranning mạng bởi thuật toán backpropagation.

3.2.3 Kiếm trúc mạng VGG-16

Mạng VGG-16 được công bố trong bài báo của Karen Simonyan and Andrew Zisserman vào năm 2014.



Hình 14 Sơ đồ kiến trúc mạng VGG16

Thông tin các tham số về mạng được mô tả tron bản sau, với đầu vào của mạng là ảnh RGB và đầu ra là vector chứa 1000 phần tử tương ứng với 1000 class được phân loại.

Layer	Filters	Biases	Stride	Tensor	Parameteres
Input	0	0	0	224x224x3	0
Conv-11	3x3x3x64	64	1	224x224x64	1792
Conv-12	3x3x64x64	64	1	224x224x64	36928
MaxPool-1	2x2	0	2	112x112x64	0
Conv-21	3x3x64x128	128	1	112x112x128	73856
Conv-22	3x3x128x128	128	1	112x112x128	147584
MaxPool-2	2x2	0	2	56x56x128	0
Conv-31	3x3x128x256	256	1	56x56x256	295168
Conv-32	3x3x256x256	256	1	56x56x256	590080
Conv-33	3x3x256x256	256	1	56x56x256	590080
MaxPool-3	2x2	0	2	28x28x256	0
Conv-41	3x3x256x512	512	1	28x28x512	1180160
Conv-42	3x3x512x512	512	1	28x28x512	2359296

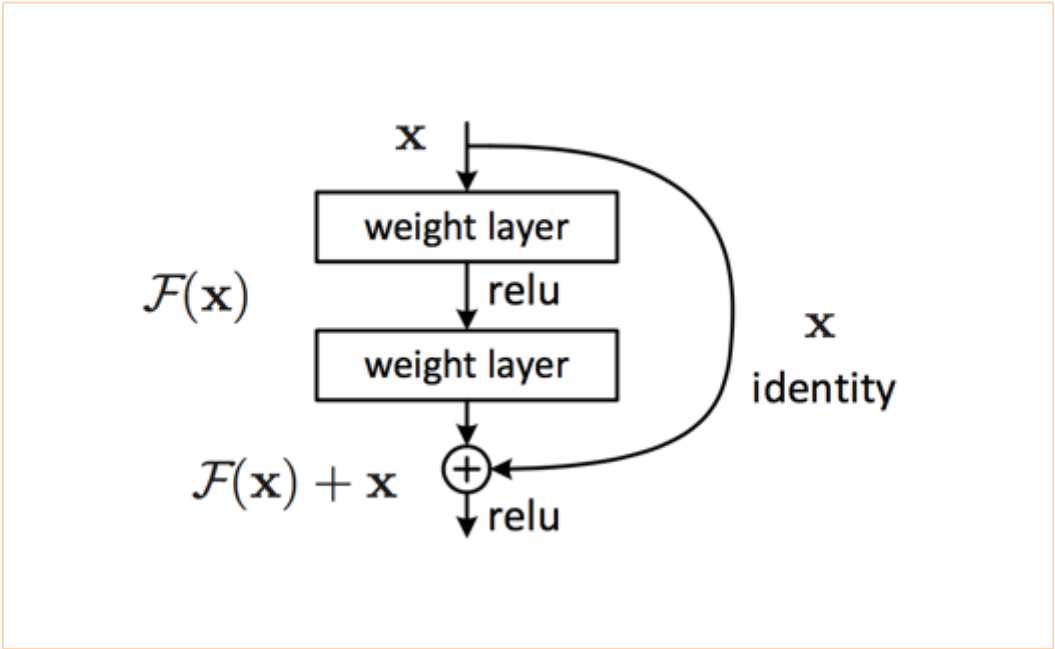
Conv-43	3x3x512x512	512	1	28x28x512	2359296
MaxPool-4	2x2	0	2	14x14x512	0
Conv-51	3x3x512x512	512	1	14x14x512	2359296
Conv-52	3x3x512x512	512	1	14x14x512	2359296
Conv-53	3x3x512x512	512	1	14x14x512	2359296
MaxPool-5	2x2	0	2	7x7x512	0
FC-1	25088x4096	4096	0	4096	102764544
FC-2	4096x4096	4096	0	4096	16781312
FC-3	4096x1000	1000	0	1000	4097000
Soft-max				1000	0
Total					138354984

Bảng 3 Kiến trúc mạng VGG16

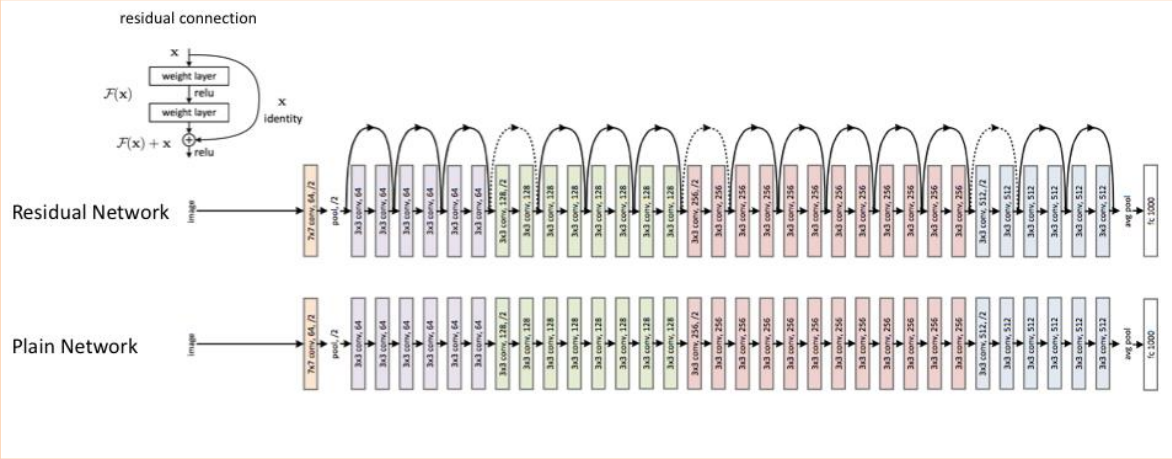
Ta có thể thấy số lượng các tham số phải học trong mạng VGG16 với cùng một đầu ra là 1000 class đã lớn hơn mạng AlexNet 2 lần.

3.2.4 Kiến trúc mạng ResNet

Mạng ResNet được công bố bởi Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun vào năm 2015. Đặc điểm của mạng là số lượng layer lớn, với phần tử cơ bản có tên gọi là Residual Block được mô tả trong hình dưới đây.



Hình 15 Sơ đồ phần tử Residual Block



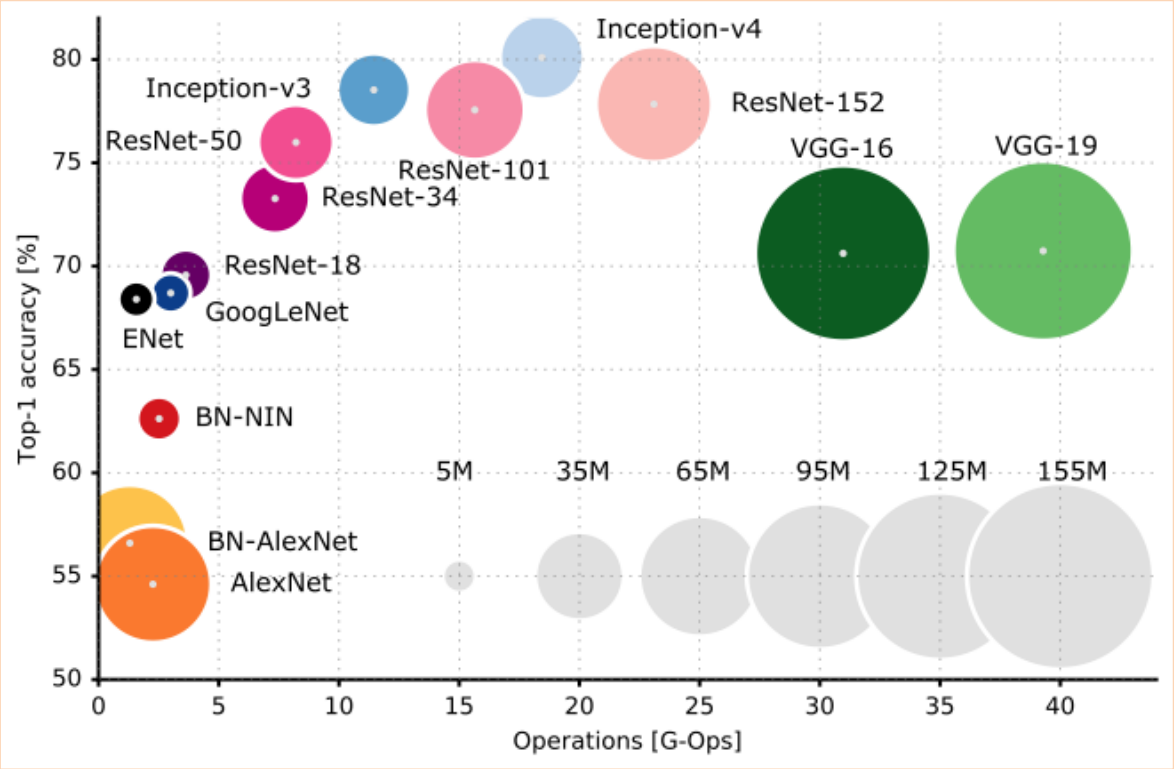
Hình 16 Sơ đồ kiến trúc mạng ResNet

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		[3×3, 64; 3×3, 64] ×2	[3×3, 64; 3×3, 64] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3
conv3_x	28×28	[3×3, 128; 3×3, 128] ×2	[3×3, 128; 3×3, 128] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×8
conv4_x	14×14	[3×3, 256; 3×3, 256] ×2	[3×3, 256; 3×3, 256] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×23	[1×1, 256; 3×3, 256; 1×1, 1024] ×36
conv5_x	7×7	[3×3, 512; 3×3, 512] ×2	[3×3, 512; 3×3, 512] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Hình 17 Thiết kế chi tiết một số mạng Resnet cơ bản

3.2.5 So sánh một số kiến trúc mạng CNN

Trên đây là một số mạng CNN phổ biến được dùng mà em đã tìm hiểu, ngoài ra còn rất nhiều kiến trúc mạng khác nữa. Nhu cầu đặt ra cần có một bảng thống kê tổng quan về độ chính xác và yêu cầu tài nguyên tính toán cho các loại mạng



Hình 18 Độ chính xác và khối lương tính toán của một số mạng

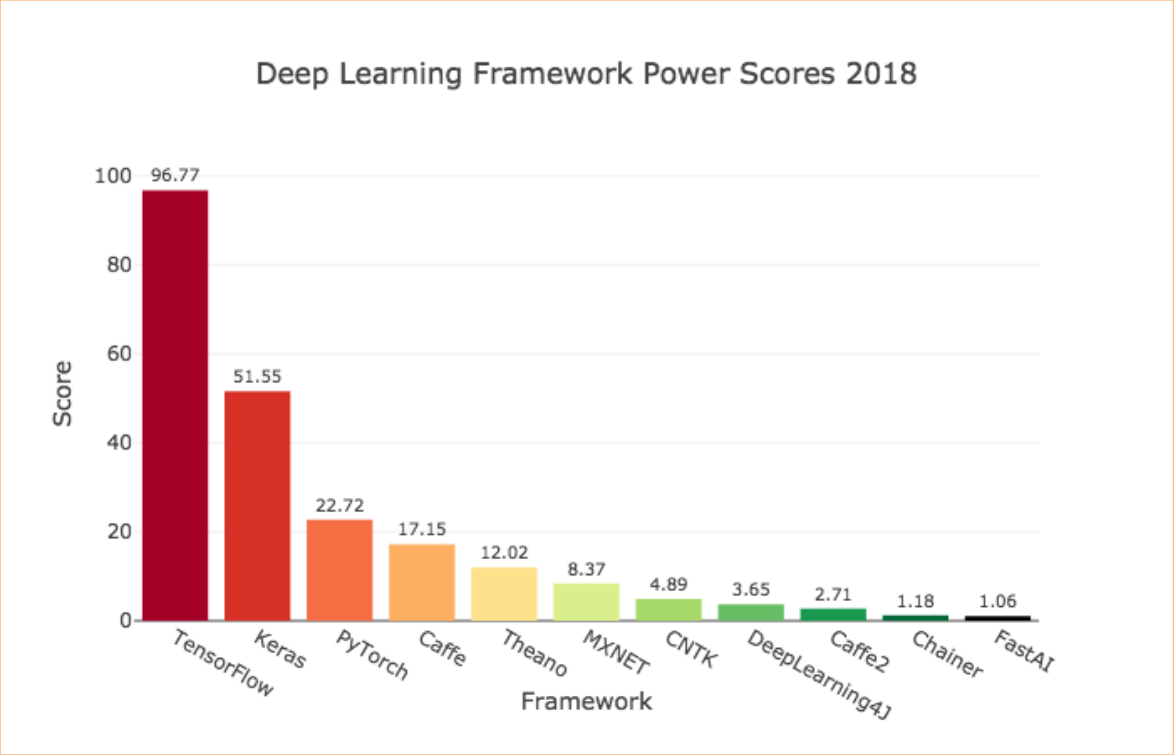
Hình trên được lấy từ kaggle.com, chúng ta có thể thấy được sự so sánh về khối lượng tính toán và độ chính xác giữa các cách xây dựng mạng CNN phổ biến hiện nay, trong đó ResNet 50 có độ chính xác khá cao trong khi khối lượng tính toán không nhiều. AlexNet cần khối lượng tính toán thấp nhưng độ chính xác tương đối thấp so với các kiến trúc mạng khác.

Trên đây là một số kiến trúc mạng phổ biến trên thế giới đã được xây dựng và kiểm nghiệm cũng như dùng trong các cuộc thi nhận diện ảnh lớn. Dựa vào các kiến trúc đã có cũng như cơ sở lý thuyết vê machine learning em đi tiến hành xây dựng và kiểm nghiệm với mạng được tùy biến đối với bài toán cụ thể của em là nhận dạng địa danh.

CHƯƠNG 4 – TRIỂN KHAI NHẬN DẠNG ĐỊA DANH

Việc xây dựng và triển khai một mạng CNN trên máy tính là một công việc phức tạp đối với lập trình viên, ngoài ra mạng phải đảm bảo tận dụng tối đa nền tảng phần cứng của máy tính đặc biệt là GPU của máy tính do khối lượng tính toán trong quá trình huấn luyện mạng rất lợn. Vì lý do trên để đạt được mục tiêu xây dựng được mạng CNN trong khoảng thời gian giới hạn của việc thực hiện đồ án này em đã tìm hiểu một số thư viện hỗ trợ triển khai việc xây

dựng các thuật toán học máy. Trong số đó có một số thư viện rất phổ biến được đưa ra trong hình sau.



Hình 19 Các framework được dùng trong deep learning

Hình tên được lấy trên trang towardsdatascience.com, trong bài "Deep Learning Framework Power Scores 2018" cùng với nhiều nguồn khác có thể thấy TensorFlow là một thư viện được dùng rất phổ biến. TensorFlow là một thư viện rất nổi tiếng được phát triển bởi Google được sử dụng để xây dựng lên các hệ thống học máy, thư viện này được sử dụng phổ biến trong công nghiệp phù hợp cho việc xây dựng lên các hệ thống lớn hoặc hệ thống mà lập trình viên muốn có sự tùy biến cao.

Từ các đặt điểm trên em lựa chọn TensorFlow cho việc xây dựng mạng CNN của mình.

Do giới hạn của phần cứng máy tính nên em lựa chọn đầu vào của mạng có kích thước là ảnh 256 x 256 pixel.

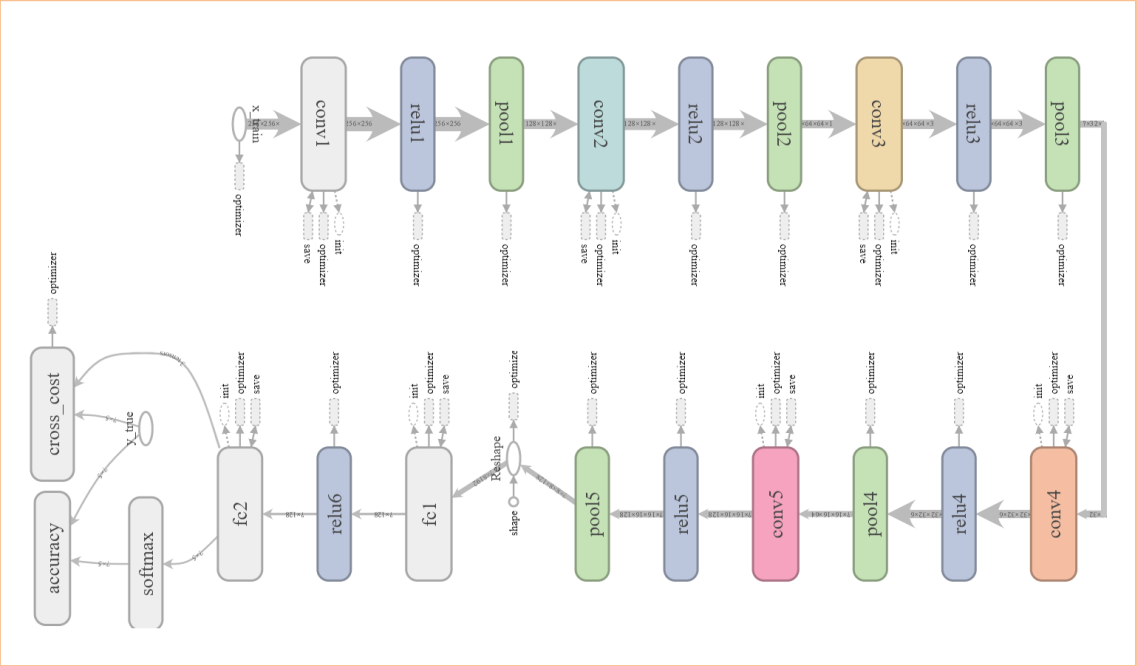
4.1. Triển khai mạng dựa trên kiến trúc mạng LeNet-5

Bảng thiết kế chi tiết các lớp của mạng với bài toán nhận diện địa danh.

Layer	Filters	Biases	Stride	Tensor	Parameteres
Input	0	0	0	256x256x3	0
Conv-1	3x3x3x8	8	1	256x256x8	224
MaxPool-1	2x2	0	2	128x128x8	0
Conv-2	3x3x8x16	16	1	128x128x16	1178
MaxPool-2	2x2	0	2	64x64x16	0
Conv-3	3x3x16x32	32	1	64x64x32	4640
MaxPool-3	2x2	0	2	32x32x32	0
Conv-4	3x3x32x64	64	1	32x32x64	18496
MaxPool-4	2x2	0	2	16x16x64	0
Conv-5	3x3x64x128	128	1	16x16x128	73856
MaxPool-5	2x2	0	2	8x8x128	0
FC-1	8192x128	128	0	128	1048704
FC-2	128x64	64	0	64	8256
Total					1155354

Bảng 4 Chi tiết thiết kế dựa trên mạng lenet-5

Sau đây là kết quả triển khai mạng bằng tensorflow, hình ảnh trên được xuất ra bởi tensorboard



Hình 20 Mô phỏng kiến trúc mạng dựa trên mạng lenet-5

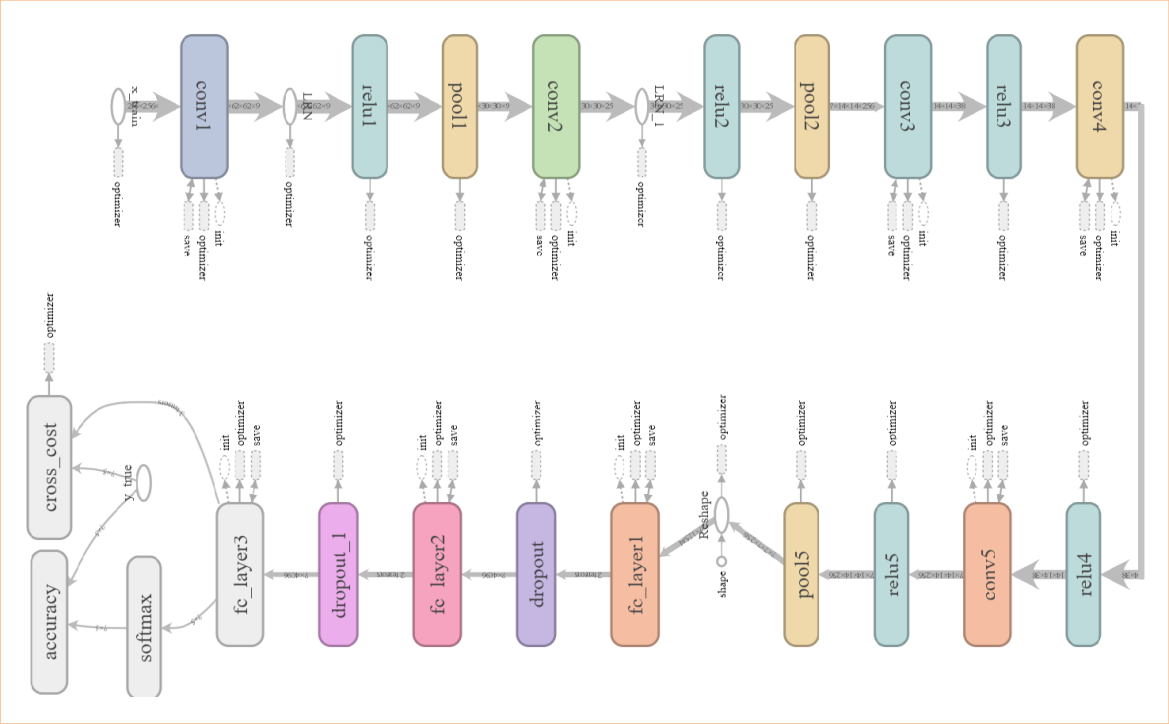
4.2. Triển khai mạng dựa trên kiến trúc mạng AlexNet

Dưới đây là bảng thiết kế các lớp của bài toán khi triển khai dựa trên mạng AlextNet.

Layer	Filters	Biases	Stride	Tensor	Parameteres
Input	0	0	0	256x256x3	0
Conv-1	11x11x3x96	96	4	62x62x96	34944
MaxPool-1	3x3	0	2	30x30x96	0
Conv-2	5x5x96x256	256	1	30x30x256	614656
MaxPool-2	3x3	0	2	14x14x256	0
Conv-3	3x3x256x384	384	1	14x14x384	885120
Conv-4	3x3x384x384	384	1	14x14x384	1327488
Conv-5	3x3x384x256	256	1	14x14x256	884992
MaxPool-3	3x3	0	2	7x7x256	0
FC-1	12544x4096	4096	0	4096	51384320
FC-2	4096x4096	4096	0	4096	16781312
FC-3	4096x64	64	0	64	262208
Total					72175040

Bảng 5 Chi tiết thiết kế dựa trên mạng alexnet

Với các tính toán ở phía trên, mạng được triển khai trên tensorfow và sử dụng tensorboard để kiểm tra một cách trực quan cấu trúc của mạng.



Hình 21 Mô phỏng kiến trúc mạng dựa trên mạng alexnet

4.3. Triển khai mạng dựa trên kiến trúc mạng VGG16

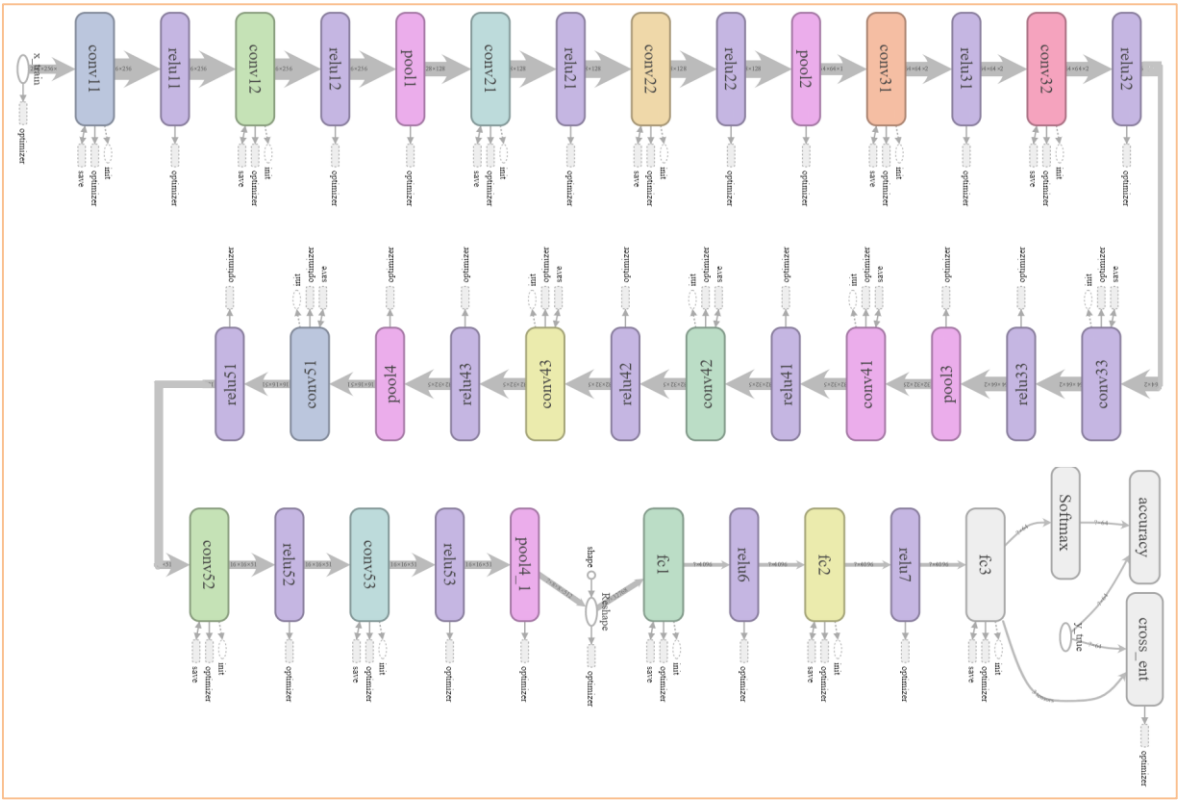
Dưới đây là bảng thiết kế các lớp của bài toán khi triển khai dựa trên mạng VGG16

Layer	Filters	Biases	Stride	Tensor	Parameteres
Input	0	0	0	256x256x3	0
Conv-1	3x3x3x64	64	1	256x256x64	1792
Conv-2	3x3x64x64	64	1	256x256x64	36928
MaxPool-1	2x2	0	2	128x128x64	0
Conv-3	3x3x64x128	128	1	128x128x128	73856
Conv-3	3x3x128x128	128	1	128x128x128	147584
MaxPool-2	2x2	0	2	64x64x128	0
Conv-4	3x3x128x256	256	1	64x64x256	295168
Conv-5	3x3x256x256	256	1	64x64x256	590080
Conv-6	3x3x256x256	256	1	64x64x256	590080

MaxPool-3	2x2	0	2	32x32x256	0
Conv-7	3x3x256x512	512	1	32x32x512	1180160
Conv-8	3x3x512x512	512	1	32x32x512	2359296
Conv-9	3x3x512x512	512	1	32x32x512	2359296
MaxPool-4	2x2	0	2	16x16x512	0
Conv-10	3x3x512x512	512	1	16x16x512	2359296
Conv-11	3x3x512x512	512	1	16x16x512	2359296
Conv-12	3x3x512x512	512	1	16x16x512	2359296
MaxPool-5	2x2	0	2	8x8x512	0
FC-1	32768x4096	4096	0	4096	134221824
FC-2	4096x4096	4096	0	4096	16781312
FC-3	4096x64	64	0	64	262208
Total					165977472

Bảng 6 Chi tiết thiết kế dựa trên vgg16

Kiến trúc mạng được triển khai trong chương trình dùng thư viện tensorflow.



Hình 22 Mô phỏng kiến trúc mạng dựa trên mạng vgg16

Mạng sau khi triển khai sẽ được huấn luyện với tập dữ liệu ảnh về các địa danh.

Độ chính xác của mạng được mô tả trong hình dưới đây.

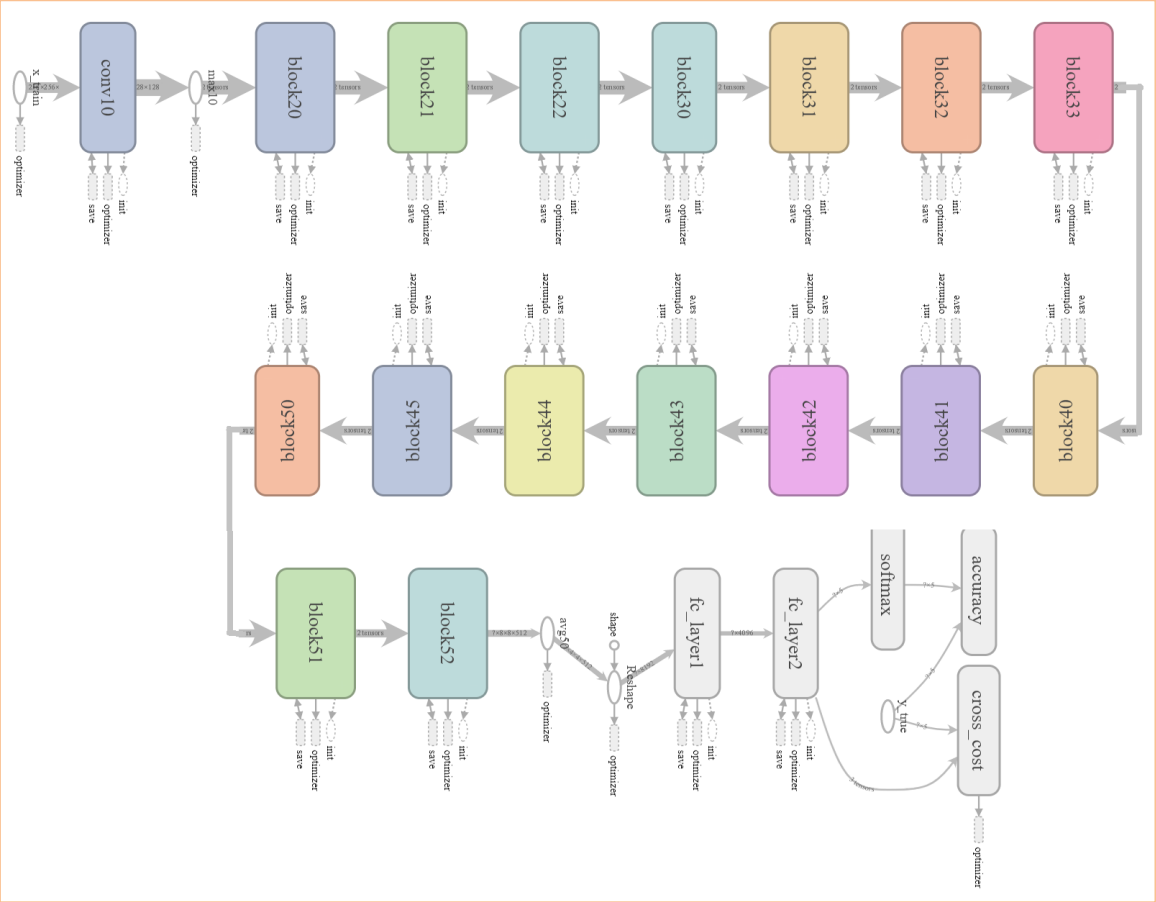
4.4. Triển khai mạng dựa trên kiến trúc mạng Resnet

Dưới đây là bảng thiết kế các lớp của bài toán khi triển khai dựa trên mạng Resnet34. Mạng resnet có nhiều phiên bản với số lượng các lớp khác nhau có thể lên tới 152 lớp. Với giới hạn của nền tảng phần cứng về bộ nhớ máy tính trong quá trình huấn luyện mạng. Em lựa chọn mạng Resnet34 để thử nghiệm, với đầu vào là ảnh 256x256x3. Chi tiết các lớp được mô tả trong bảng sau.

Layers	Filters	Stride	Number	Tensor	Parameters
Input	0	0	1	256x256x3	0
Conv-0	7x7x3x64	2	1	128x128x64	9408
MaxPool-0	2x2	2	1	64x64x64	0

Block-2	3x3x64 3x3x64 3x3x64	1	3	64x64x64	1728
Block-3	3x3x128 3x3x128 3x3x128	1	4	32x32x128	3072
Block-4	3x3x256 3x3x256 3x3x256	1	6	16x16x256	9216
Block-5	3x3x512 3x3x512 3x3x512	1	4	8x8x512	12288
AvgPool	2x2	2	1	4x4x512	0
FC-1	4x4x512x1024			1024	8388608
FC-2	1024x64			64	65536
Total					8489856

Bảng 7 Chi tiết thiết kế dựa trên Resnet-34



Hình 23 Mô phỏng mạng dựa trên mạng resnet-34

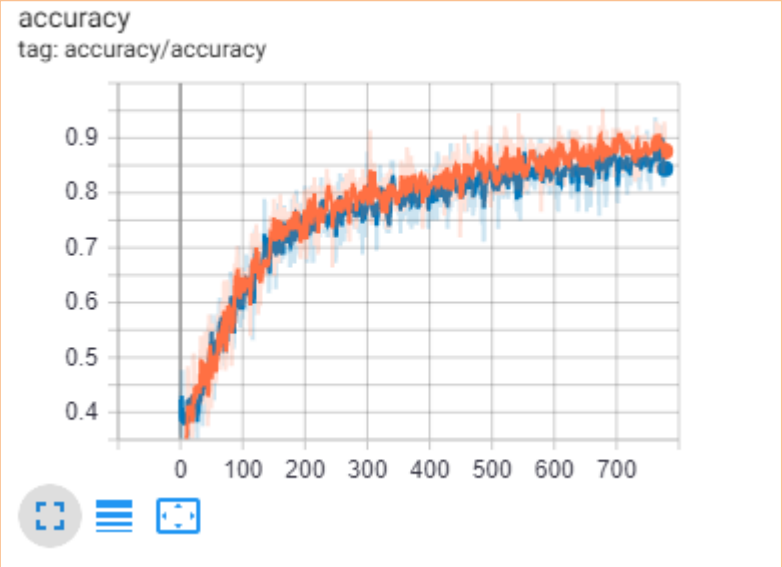
Trên đây là kết quả mô phỏng mạng Resnet cùng với một số mạng cơ bản nhất được đã được thử nghiệm và sử dụng trong CNN. Tiếp theo em đi tới phần chạy thử nghiệm để chọn ra kiến trúc mạng phù hợp nhất với bài toán nhận dạng địa danh.

4.5. Kết quả thử nghiệm các mạng với đầu ra 5 địa danh

Việc huấn luyện mạng mất nhiều thời gian, nhằm chọn lựa ra mạng phù hợp nhất em sẽ cho mạng huấn luyện với số lượng đầu ra của mạng tăng dần. Dựa vào kết quả huấn luyện với việc nhận diện 5 địa danh em sẽ chọn ra mạng phù hợp nhất cho việc nhận diện 64 địa danh.

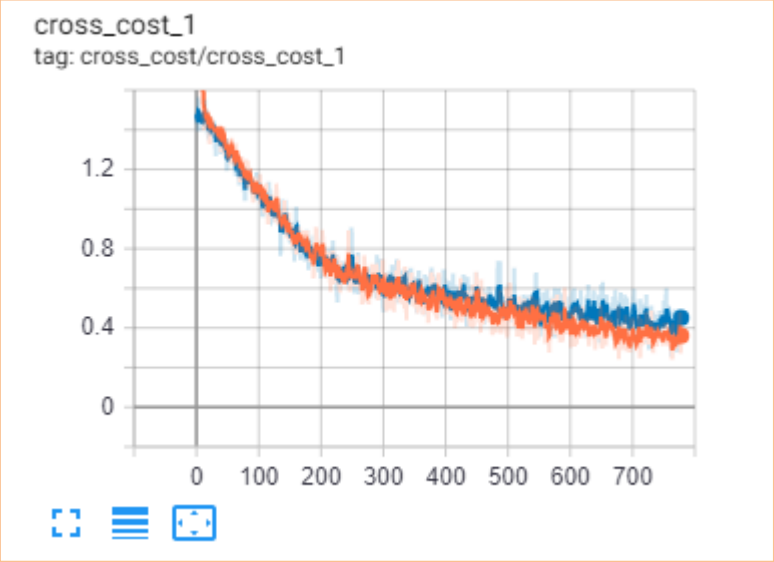
Ở mỗi địa danh có 1200 ảnh dùng để huấn luyện, 400 ảnh để kiểm chứng độ chính xác. Đường màu xanh biểu thị cho độ chính xác và lỗi trên tập dữ liệu huấn luyện, đường màu vàng cam biểu thị cho đo chính xác và lỗi trên tập dữ liệu kiểm chứng.

- Kết quả huấn luyện mạng dựa trên mạng lenet-5. Em gọi là mạng số 1



Hình 24 Độ chính xác của mạng số 1 trong quá trình nhận diện 5 địa danh

Sau đây là biểu đồ mô tả giá trị lỗi trong việc dự đoán của mạng tại mỗi thời điểm huấn luyện.



Hình 25 Lỗi của mạng số 1 trong quá trình nhận diện 5 địa danh

Kết thúc quá trình huấn luyện mạng độ chính xác trên tập huấn luyện là 90% và trên tập kiểm chứng là 82%.

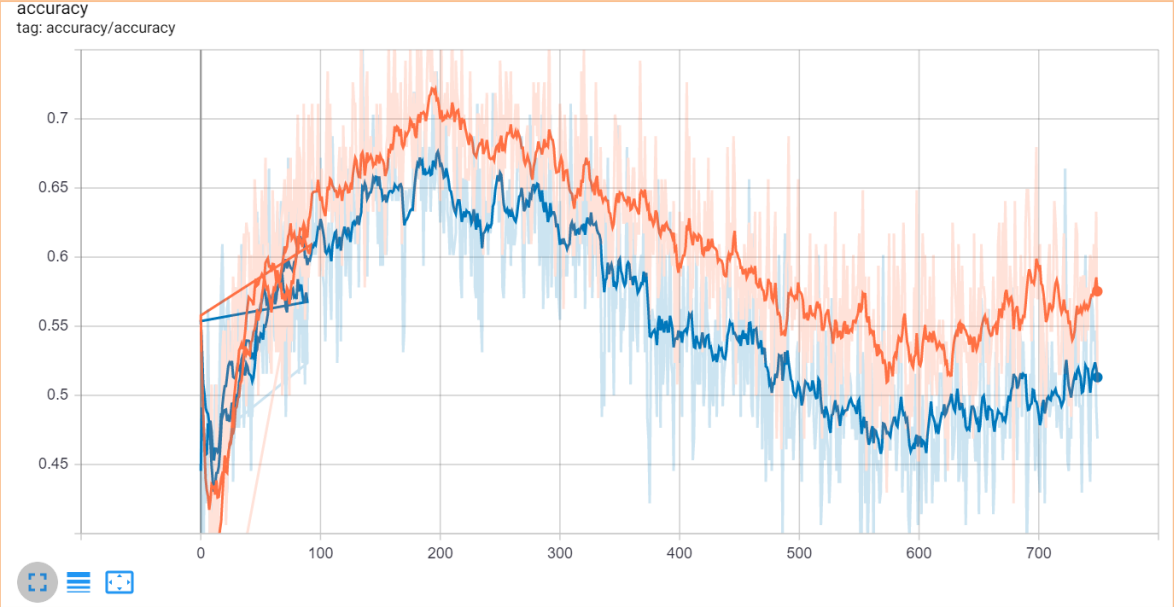
Giá trị lỗi ở cuối quá trình huấn luyện là 0.2 trên tập dữ liệu huấn luyện và 0.5 trên tập dữ liệu kiểm chứng.

Đồ thị cho kết quả huấn luyện tốt với độ chính xác tăng dần và giá trị lỗi giảm dần. Trong quá trình huấn luyện độ chính xác trên tập dữ liệu huấn luyện gần

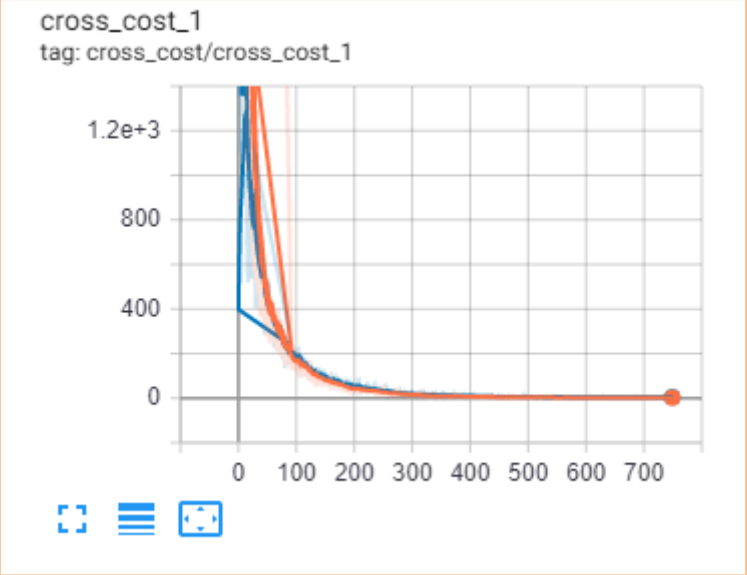
với độ chính xác trên tập dữ liệu kiểm chứng. Tương tự với giá trị lỗi trên tập kiểm chứng gần với giá trị lỗi trên tập huấn luyện.

Mạng được huấn luyện bằng việc đưa toàn bộ tập dữ liệu tham gia huấn luyện 50 lần, thời gian huấn luyện 2 giờ 20 phút trên cạc màn hình GTX-1080TI.

+ Kết quả huấn luyện mạng dựa trên mạng alexnet. Em gọi là mạng số 2.



Hình 26 Độ chính xác của mạng số 2 trong quá trình nhận diện 5 địa danh



Hình 27 Lỗi của mạng số 2 trong quá trình nhận diện 5 địa danh

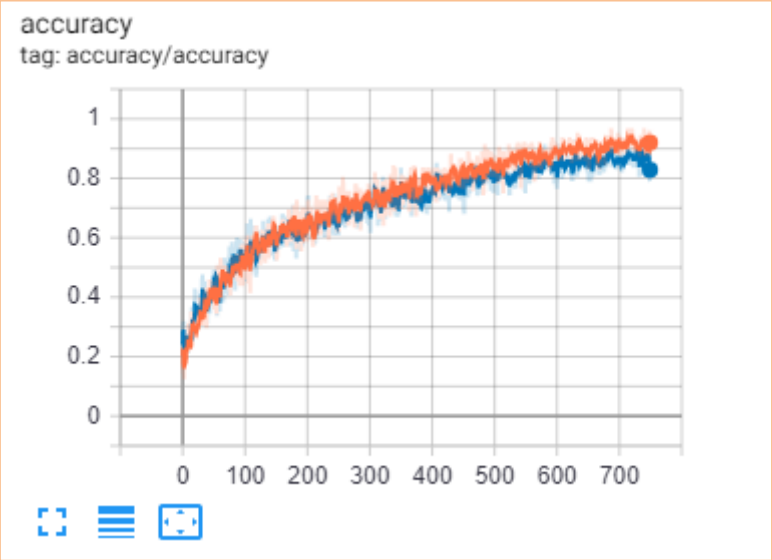
Độ chính xác lớn nhất của mạng với tập huấn luyện là 60% với tập kiểm chứng là 50% lằm ở giữa quá trình huấn luyện. qua biểu đồ về độ chính xác

cho thấy mạng học trên tập dữ liệu không tốt, có hiện tượng bùng nổ đào hàm trong quá trình huấn luyện mạng.

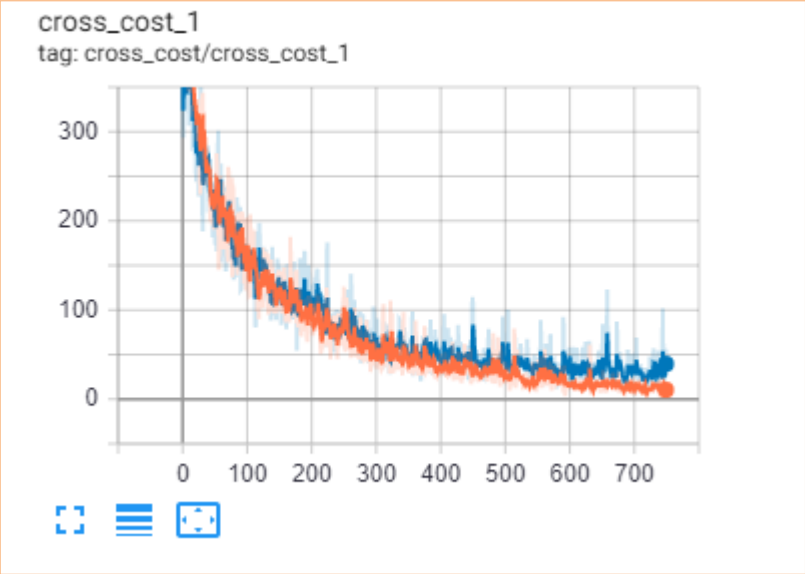
Giá trị lỗi ở cuối quá trình huấn luyện mạng là 2 với tập dữ liệu huấn luyện và 4 với tập dữ liệu kiểm chứng.

Dựa vào hai đồ thị phía trên cho thấy mạng có cấu trúc đơn giản dẫn tới học không chính xác dữ liệu.

+ Kết quả huấn luyện mạng dựa trên mạng resnet-43. Em gọi là mạng số 3.



Hình 28 Độ chính xác của mạng số 3 trong quá trình nhận diện 5 địa danh



Hình 29 Lỗi của mạng số 3 trong quá trình nhận diện 5 địa danh

Kết thúc quá trình huấn luyện độ chính xác trên tập dữ liệu trên tập huấn luyện là 99%, trên tập dữ liệu kiểm chứng là 82%. Giá trị lỗi trên tập huấn luyện là 0.001, trên tập kiển chứng là 3.8.

Giá trị của độ chính xác và lỗi trên hai tập dữ liệu huấn luyện và kiểm chứng cùng tăng và cùng giảm.

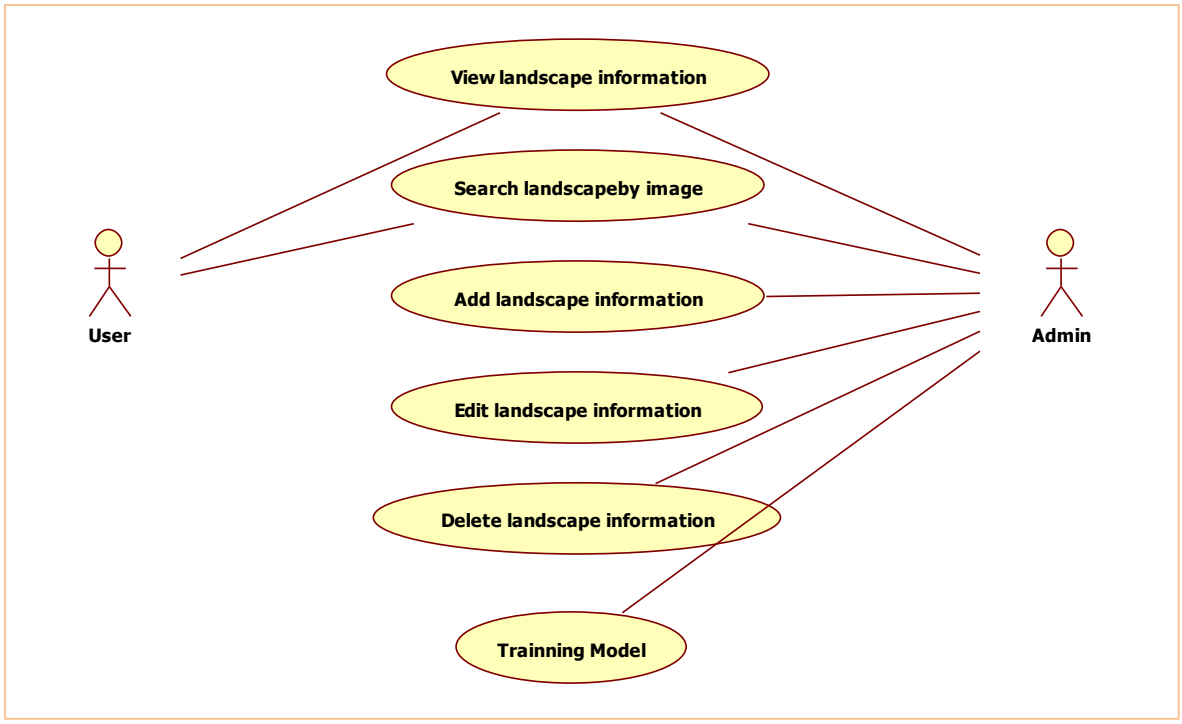
Tuy nhiên hai giá trị các xa nhau như trên hình cho thấy quá trình học không tốt. Mạng quá phức tạp dẫn tới bị overfit.

Thời gian huấn luyện mạng với 50 lần huấn luyện toàn bộ tập dữ liệu là 1 giờ 30 phút trên cạc màn hình GTX-1080TI.

4.6. Kết quả thử nghiệm mạng với đầu ra 48 địa danh

CHƯƠNG 6 - THIẾT KẾ HỆ THỐNG

6.1. Biểu đồ ca sử dụng



Hình 30 biểu dồ ca sử dụng

Hệ thống gồm các chức năng cơ bản giúp người dùng có thể xem thông tin và tìm kiếm địa danh bằng hình ảnh. Người quản trị có thêm các quyền liên quan tới trang quản trị như thêm, sửa, xóa thông tin địa danh.

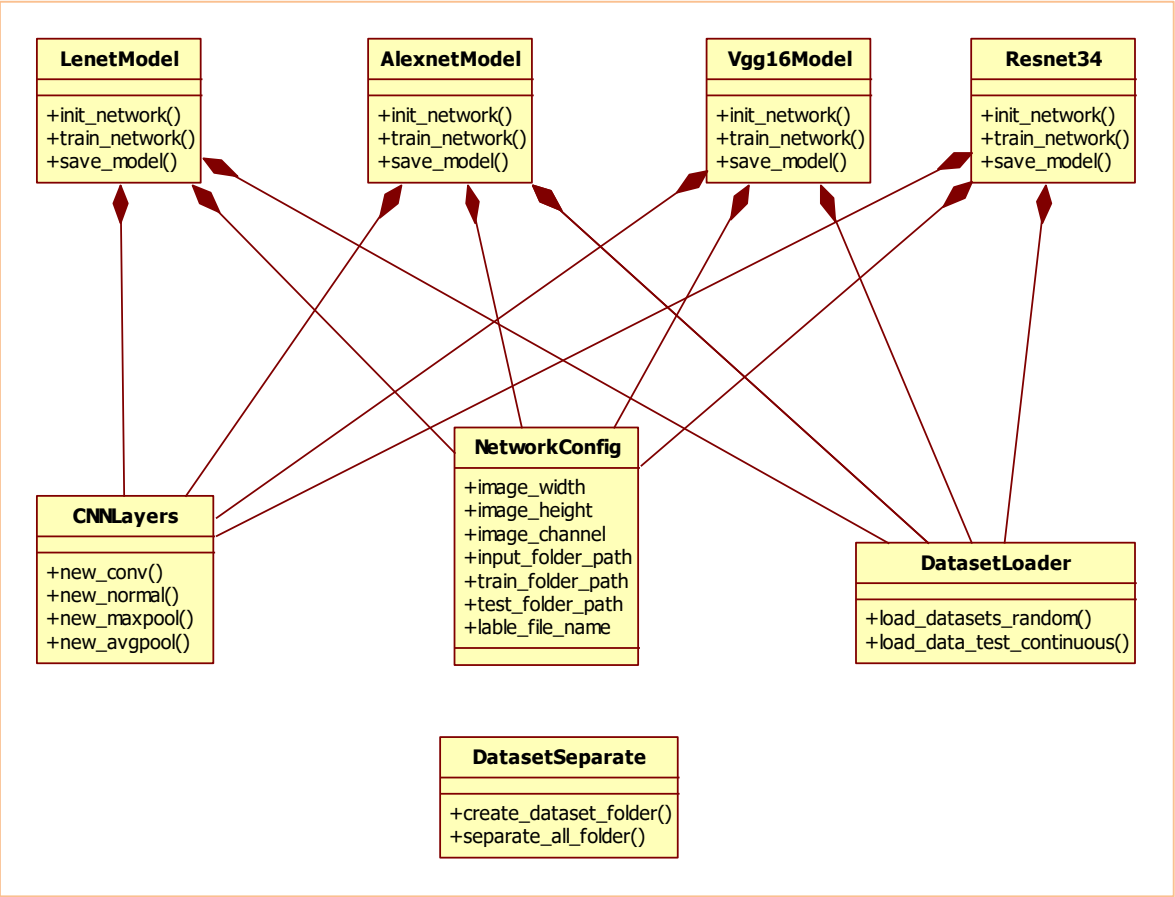
6.2. Biểu dồ lớp

Biểu đồ lớp trong trường hợp huấn luyện mạng nhận diện địa danh gồm ba thành phần chính:

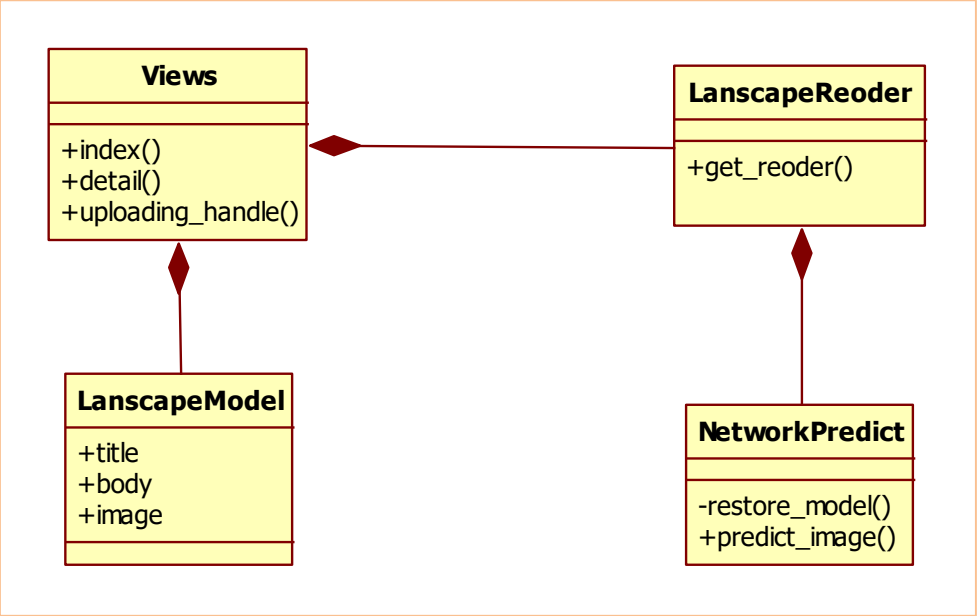
+ Các class liên quan tới tiền xử lý dữ liệu như phân chia dữ liệu train và dữ liệu test, kiểm duyệt dữ liệu và loại bỏ những ảnh không đúng format ví dụ ảnh không phải rgb.

+ Class liên quan tới các tham số cấu hình của mạng và tải dữ liệu ảnh từ bộ nhớ

- Class định nghĩa các model dựa trên các kiến trúc khác nhau có hàm huấn luyện và lưu lại model sau khi được huấn luyện



Hình 31 sơ đồ lớp cho user case huấn luyện mạng

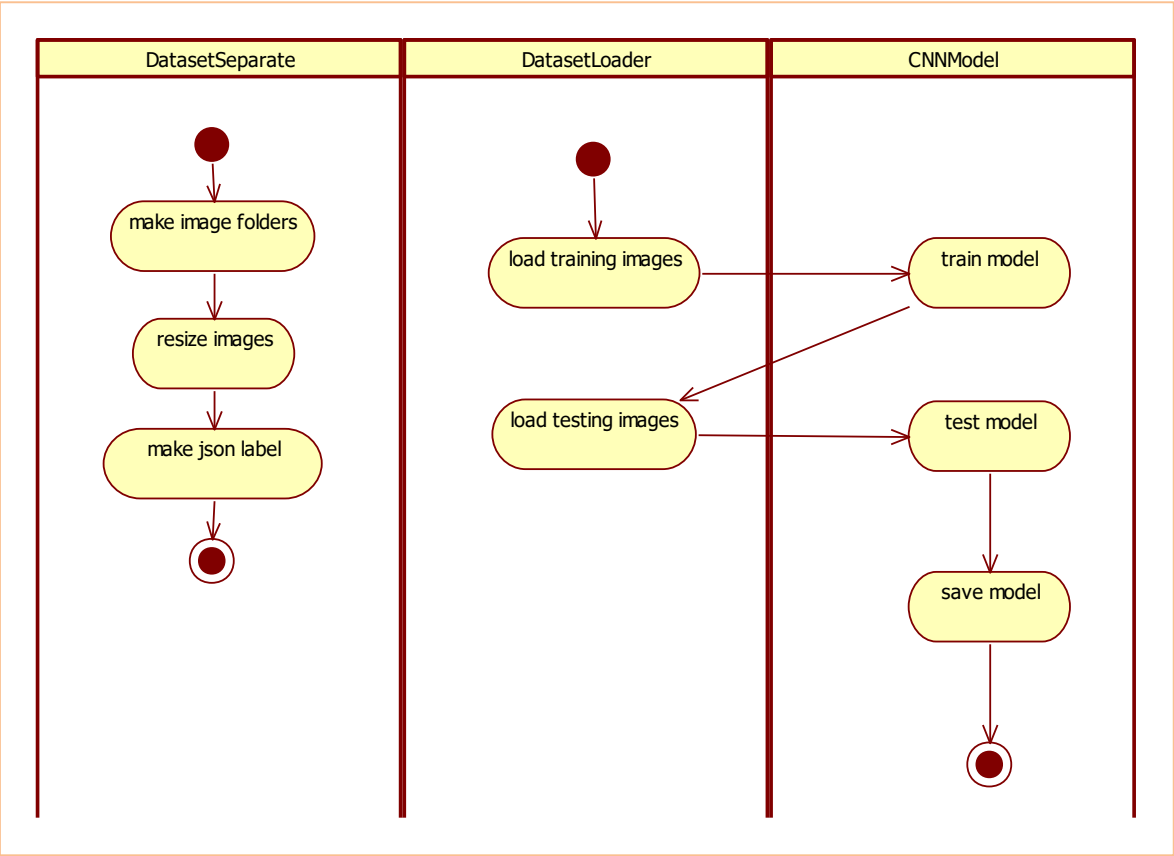


Hình 32 sơ đồ lớp cho user case tìm kiếm địa danh bằng hình ảnh

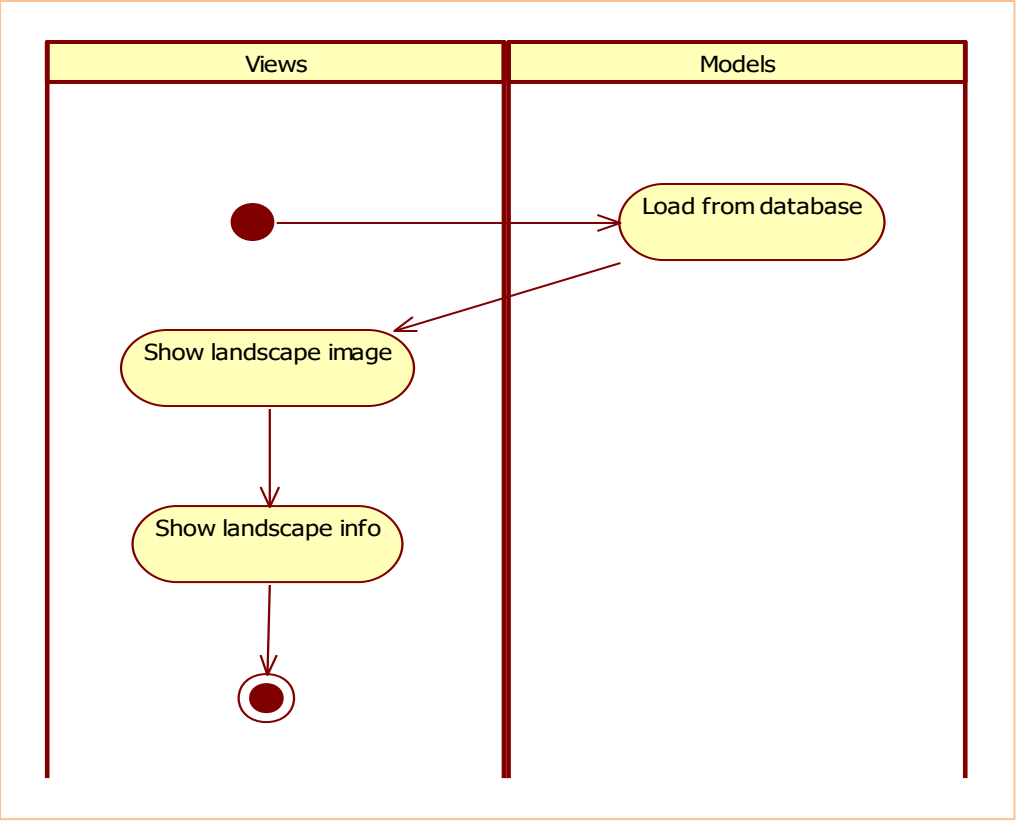
Trong trường hợp tìm kiếm thông tin bằng hình ảnh, chức năng này được thực hiện trên môi trường web. Trong đó có các lớp chính tham gia vào user case được mô tả trong hình dưới đây

6.3. Biểu đồ hoạt động

Biểu đồ hoạt động cho việc huấn luyện mạng neuron, kết quả thu được sau quá trình huấn luyện là một model tương ứng với cấu trúc mạng đã cấu hình có thể là AlexNet hoặc VGG16Net …



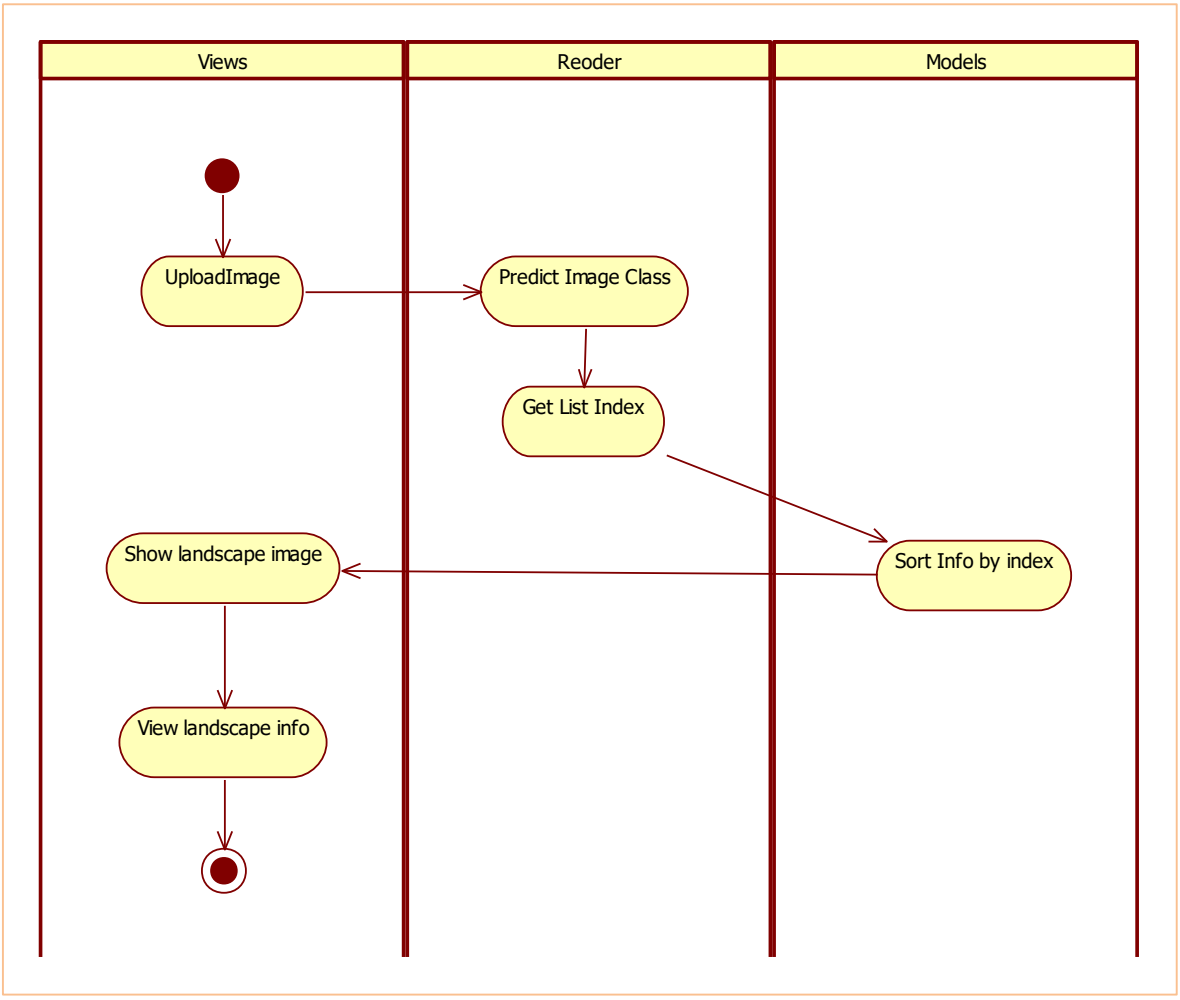
Hình 33 biểu đồ hoạt động cho quá trình huấn luyện mạng



Hình 34 biểu dồ hoạt động xem thông tin địa danh

Dưới đây là biểu đồ hoạt động cho hai chức năng quan trọng nhất của hệ thống đó là chức năng xem thông tin địa danh và tìm kiếm thông tin địa danh.

Chức năng tìm kiếm thông tin địa danh bằng hình ảnh là chức năng quan trọng nhất. Trong đó sử dụng mô hình học máy được đào tạo ở phần 4 để trích chọn đặc trưng, từ đó đưa ra danh sách các địa danh có đặc điểm giống với địa danh được tìm kiếm nhất. Dựa vào đó hệ thống sẽ sắp xếp lại thứ tự danh sách các địa danh và đưa các địa danh có sác xuất giống nhất lên vị trí đầu tiên.



Hình 35 biểu đồ hoạt động tìm kiếm thông tin địa danh

Các chắc năng về thêm, sửa, xóa thông tin địa danh được Django hỗ trợ trong trang quản trị của frame work. Người quản trị đăng nhập vào hệ thống và sửa thông tin theo yêu cầu tương ứng.

6.4. Biểu đồ tuần tự

6.5. Cơ sở dữ liệu địa danh

Cơ sở dữ liệu được sử dụng trong trương trình là cơ sở dữ liệu sqlite được tích hợp sẵn trong Django Frame Work. Với phạm vi chương trình thì cơ sở dữ liệu sqlite đủ đáp ứng được nhu cầu sử dụng cũng như tính gọn nhẹ của cơ sở dữ liệu.

Dưới đây là bảng sữ liệu thông tin địa danh trong cơ sở dữ liệu.

Tên trường	Kiể dữ liệu	Ghi chú
id	interger	primary key
title	varchar(1000)	
body	text	
date	datetime	
num	varchar(3)	
image	varchar(100)	

Bảng 8 bảng dữ liệu địa danh

Trường dữ liệu "title" được dùng để lưu tên của địa danh, trường dữ liệu "body" được dùng để lưu mô tả cũng như cung cấp thông tin chi tiết về địa danh. Trường dữ liệu num để lưu thông tin về ID của địa danh, trường này dùng để tra cứu thông tin địa danh sau khi hàm nhận diện địa danh trả về kết quả là các ID của các địa danh theo thứ tự từ gần giống nhất với đặc điểm địa danh được tìm kiếm. "Image" là trường lưu đường dẫn ảnh minh họa cho địa danh, ảnh minh họa được lưu tại một thư mục trên server.

Tiếp theo là bảng chứa tài khoản người dùng của website

Tên trường	Kiểu dữ liệu	Ghi chú
id	interger	primary key
password	varchar(128)	
last_login	datetime	
is_superuser	bool	
username	varchar(150)	
first_name	varchar(30)	
email	varchar(254)	
is_staff	bool	
is_active	bool	
date_joined	datetime	
last_name	varchar(150)	

Bảng 9 bảng thông tin người dùng

Bảng được Django hỗ trợ tạo ra trong hệ thống admin, trong đó các trường quan trọng như is_superuser, is_staff được dùng để phân biệt tài khoản admin và user thông thường. Trong đồ án này thông tin user em đang để chỉ gồm một tài khoản admin để đăng nhập vào trang quản trị từ đó thêm, sửa xóa các thông tin về địa danh

CHƯƠNG 7 – ĐÁNH GIÁ KẾT QUẢ

7.1. Giao diện trương trình

7.2. Minh họa chức năng tìm kiếm địa danh bằng hình ảnh

7.3. Đánh giá kết quả đạt được

7.4. Hướng phát triển trong tương lai

CHƯƠNG 8 - TÀI LIỆU THAM KHẢO

[1]https://www.learnopencv.com/understanding-feedforward-neural-networks/

[1] https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

[2] https://www.mathworks.com/videos/introduction-to-deep-learning-what-are-convolutional-neural-networks--1489512765771.html

[3] https://medium.com/machine-learning-bites/deeplearning-series-convolutional-neural-networks-a9c2f2ee1524

[4] https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/

[5] http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/

[6] https://www.kaggle.com/shivamb/cnn-architectures-vgg-resnet-inception-tl

[7] https://www.learnopencv.com/number-of-parameters-and-tensor-sizes-in-convolutional-neural-network/

[8] https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/

[9] https://towardsdatascience.com/deep-learning-framework-power-scores-2018-23607ddf297a