

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: **TS.Đình Viêt Sang**

HÀ NỘI 6-2019

PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

1. Thông tin về sinh viên

Họ và tên sinh viên: Lê Văn Mạnh

Điện thoại liên lạc: 035 985 9885

Email: levanmanhbka@gmail.com

Lớp: KSVB2-K37

Hệ đào tạo: chính quy

Đồ án tốt nghiệp được thực hiện tại:

Thời gian làm ĐATN: Từ ngày 11/03 /2019 đến 21/06/2019

2. Mục đích nội dung của ĐATN

Xây dựng hệ thống ứng dụng học sâu trong tự động xác định địa điểm du lịch nổi tiếng.

3. Các nhiệm vụ cụ thể của ĐATN :

Xây dựng mô hình học máy nhận diện địa danh. Huấn luyện mô hình học máy. Ứng dụng mô hình sau huấn luyện vào hệ thống nhận diện địa danh bằng hình ảnh.

4. Lời cam đoan của sinh viên:

Tôi Lê Văn Mạnh cam kết ĐATN là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của TS. Đinh Viết Sang.

Các kết quả nêu trong ĐATN là trung thực, không phải là sao chép toàn văn của bất kỳ công trình nào khác.

Hà Nội, ngày 10 tháng 6 năm 2019

Tác giả ĐATN

Lê Văn Mạnh

5. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ:

Hà Nội, ngày 10 tháng 6 năm 2019

Giáo viên hướng dẫn

TS. Đinh Viết Sang

LỜI CẢM ƠN

Để hoàn thành đồ án này em đã nhận được nhiều sự hỗ trợ và động viên từ thầy cô, bạn bè và đồng nghiệp.

Trước hết, em xin gửi lời chân thành cảm ơn tới thầy Đinh Viết Sang bộ môn Khoa Học Máy Tính đã định hướng cho em thực hiện đồ án từ bước chuẩn bị dữ liệu cho tới việc lựa chọn mô hình để giải bài toán mà đồ án hướng tới.

Để có điều kiện về thời gian tham gia học cũng như hoàn thành đồ án, em xin cảm ơn anh Nguyễn Minh Thành, anh Ngô Ngọc Đức cùng các anh chị em tại trung tâm SVMC đã tạo điều kiện thuận lợi cho em vừa làm việc tại công ty đồng thời tham gia học văn bằng 2.

Cuối cùng em xin cảm ơn các thầy cô tại Viện Công Nghệ Thông Tin và Truyền Thông, Đại Học Bách Khoa Hà Nội đã tận tình chỉ dạy và tạo mọi điều kiện thuận lợi cho em trong quá trình em rèn luyện tại trường.

Những trải nghiệm tại trường sẽ là nền tảng cho em đi tiếp trên con đường sự nghiệp của mình. Em xin chân thành cảm ơn!

Hà nội ngày 09/06/2019

Sinh Viên

Lê Văn Mạnh

TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP

Chương 1 - Mở đầu: Nội dung chương nói về mục tiêu của đồ án và ý nghĩa của đồ án, phương pháp tiếp cận cũng như kết quả dự kiến.

Chương 2 - Tổng quan mạng nơ-ron: Nội dung chương trình bày cơ bản nhất về định nghĩa học sâu, mối quan hệ giữa học sâu với trí tuệ nhân tạo. Chương cũng bao quát về định nghĩa mạng nơ-ron, bản chất của việc huấn luyện mạng, các phương pháp huấn luyện mạng nơ-ron nhân tạo. Tiếp theo chương nhắc tới định nghĩa mạng nơ-ron tích chập đây là nền tảng cho bài toán nhận diện ảnh ngoài ra chương cũng đưa ra một số kiến trúc mạng nơ-ron tích chập phổ biến.

Chương 3 - Ứng dụng mạng nơ-ron tích chập cho bài toán nhận diện địa danh: Chương đưa ra các cấu hình mạng gồm có tính toán số lượng tham số phải học, sơ đồ của mạng mà tác giả đã tùy biến với bài toán thực tế là nhận diện ảnh với đặc thù ảnh phong cảnh.

Chương 4 - Thử nghiệm và đánh giá kết quả: Chương nêu quá trình thử nghiệm với các mạng được đề xuất ở chương 3, kết quả thử nghiệm độ chính xác và thời gian huấn luyện với từng mạng. Cuối cùng là tổng hợp và chọn ra mạng phù hợp nhất cho mục tiêu của đồ án.

Chương 5 - Ứng dụng mô hình sau huấn luyện trong hệ thống nhận diện địa danh: Từ mô hình được xây dựng từ chương trước, tác giả đưa mô hình vào hệ thống nhận diện địa danh. Tại đây tác giả đưa ra thiết kế hệ thống và kết quả sau khi xây dựng hệ thống. Hệ thống được xây dựng trên nền tảng web.

Chương 6 - kết luận và hướng phát triển: Chương nói về những kết quả đã đạt được của đồ án cũng như những hạn chế và định hướng phát triển trong tương lai.

MỤC LỤC

| | |
|--|----|
| CHƯƠNG 1 - MỞ ĐẦU | 11 |
| 1.1. Nhiệm vụ của đồ án..... | 11 |
| 1.2. Phương pháp thực hiện..... | 11 |
| 1.3. Ý nghĩa khoa học và thực tiễn..... | 11 |
| 1.4. Kết quả dự kiến | 12 |
| CHƯƠNG 2 – TỔNG QUAN MẠNG NƠ-RON | 13 |
| 2.1 Mạng nơ-ron cơ bản | 13 |
| 2.1.1 Cấu tạo của mạng nơ-ron nhân tạo | 13 |
| 2.1.2 Huấn luyện mạng nơ-ron nhân tạo | 14 |
| 2.1.3 Một số thuật toán tối ưu hàm mất mát | 16 |
| 2.1.4 Overfitting và Underfitting | 16 |
| 2.2 Mạng nơ-ron tích chập..... | 17 |
| 2.2.1 Định nghĩa convolution | 18 |
| 2.2.2 Định nghĩa stride và padding | 19 |
| 2.2.3 Lớp pooling trong mạng CNN..... | 19 |
| 2.3 Một số kiến trúc mạng CNN phổ biến..... | 20 |
| 2.2.1 Kiến trúc mạng LeNet-5 | 20 |
| 2.2.2 Kiến trúc mạng AlexNet..... | 21 |
| 2.2.3 Kiến trúc mạng VGG-16 | 23 |
| 2.2.4 Kiến trúc mạng ResNet..... | 24 |
| 2.2.5 So sánh một số kiến trúc mạng CNN..... | 25 |
| CHƯƠNG 3 – ỨNG DỤNG MẠNG NƠ-RON TÍCH CHẬP CHO BÀI TOÁN NHẬN DẠNG ĐỊA DANH..... | 27 |
| 3.1 Đề xuất mạng dựa trên kiến trúc mạng LeNet-5 | 27 |
| 3.1 Đề xuất mạng dựa trên kiến trúc mạng AlexNet..... | 28 |
| 3.4 Đề xuất mạng dựa trên kiến trúc mạng VGG16..... | 29 |
| 3.3 Đề xuất mạng dựa trên kiến trúc mạng Resnet..... | 31 |
| 3.5 Hàm mất mát và hàm tính độ chính xác trong quá trình huấn luyện..... | 33 |
| CHƯƠNG 4 - THỬ NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ..... | 34 |
| 4.1 Bộ dữ liệu sử dụng cho bài toán | 34 |

| | |
|---|----|
| 4.2 Kết quả huấn luyện mạng dựa trên mạng lenet-5 | 35 |
| 4.3 Kết quả huấn luyện mạng dựa trên mạng alexnet..... | 37 |
| 4.4 Kết quả huấn luyện mạng dựa trên mạng vgg-16..... | 38 |
| 4.5 Kết quả huấn luyện mạng dựa trên mạng resnet-34 | 39 |
| 4.6 Tổng kết toàn bộ quá trình thử nghiệm | 40 |
| CHƯƠNG 5 – ỨNG DỤNG MÔ HÌNH SAU HUẤN LUYỆN TRONG HỆ THỐNG NHẬN DIỆN ĐỊA DANH..... | 42 |
| 5.1 Biểu đồ ca sử dụng | 42 |
| 5.2 Biểu đồ lớp | 42 |
| 5.3 Biểu đồ hoạt động..... | 43 |
| 5.4 Cơ sở dữ liệu địa danh..... | 43 |
| 5.4 Giao diện trang web..... | 45 |
| CHƯƠNG 6 – KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN..... | 46 |
| 6.1. Kết luận | 46 |
| 6.2. Hướng phát triển | 46 |
| TÀI LIỆU THAM KHẢO..... | 47 |

DANH MỤC TỪ VIẾT TẮT

| Từ viết tắt | Thuật ngữ | Giải thích |
|-------------|------------------------------|------------------------|
| CNN | Convolutional neural network | Mạng nơ-ron tích chập |
| FC | Full connected network | Mạng nơ-ron nhân tạo |
| Conv | Convolutional layer | Lớp tích chập |
| MaxPool | Max pooling layer | Lớp max pooling |
| RBF | Radial basis function | Lớp thuộc mạng lenet-5 |
| AvgPool | Average pooling layer | Lớp average pooling |

DANH MỤC BẢNG BIỂU

| | |
|---|----|
| Bảng 1 Kiến trúc mạng LeNet-5 | 21 |
| Bảng 2 Kiến trúc mạng AlexNet | 22 |
| Bảng 3 Kiến trúc mạng VGG16..... | 23 |
| Bảng 4 Chi tiết tham số của mô hình dựa trên mạng lenet-5..... | 27 |
| Bảng 5 Chi tiết tham số của mô hình dựa trên mạng alexnet | 28 |
| Bảng 6 Chi tiết tham số của mô hình dựa trên vgg16..... | 29 |
| Bảng 7 Chi tiết tham số của mô hình dựa trên Resnet-34..... | 32 |
| Bảng 8 So sánh kết quả huấn luyện mạng phân loại 5 địa danh | 40 |
| Bảng 9 Bảng dữ liệu địa danh | 43 |
| Bảng 10 Bảng thông tin người dùng | 44 |

DANH MỤC HÌNH VẼ

| | |
|--|----|
| Hình 1 Hệ thống tìm kiếm thông tin địa danh bằng hình ảnh..... | 11 |
| Hình 2 Tổng quan về trí tuệ nhân tạo..... | 13 |
| Hình 3 Mạng nơ-ron nhân tạo..... | 13 |
| Hình 4 Nơ-ron trong mạng nơ-ron network..... | 14 |
| Hình 5 Hàm Relu | 14 |
| Hình 6 Minh họa thuật toán gradient descent | 15 |
| Hình 7 Overfitting và Underfitting | 16 |
| Hình 8 Sơ đồ tổng quát CNN..... | 17 |
| Hình 9 Khái niệm convolutional | 18 |
| Hình 10 Convolutional và mảng hai chiều..... | 18 |
| Hình 11 Convolutional và mảng ba chiều..... | 19 |
| Hình 12 Minh họa max pooling | 20 |
| Hình 13 Sơ đồ kiến trúc mạng LeNet-5 | 20 |
| Hình 14 Sơ đồ kiến trúc mạng AlexNet..... | 21 |
| Hình 15 Sơ đồ kiến trúc mạng VGG16..... | 23 |
| Hình 16 Sơ đồ phần tử Residual Block..... | 24 |
| Hình 17 Sơ đồ kiến trúc mạng ResNet..... | 25 |
| Hình 18 Thiết kế chi tiết một số mạng Resnet cơ bản | 25 |
| Hình 19 Độ chính xác và khối lượng tính toán của một số mạng..... | 26 |
| Hình 20 Mô phỏng kiến trúc mạng dựa trên mạng lenet-5 | 28 |
| Hình 21 Mô phỏng kiến trúc mạng dựa trên mạng alexnet | 29 |
| Hình 22 Mô phỏng kiến trúc mạng dựa trên mạng vgg16 | 31 |
| Hình 23 Mô phỏng mạng dựa trên mạng resnet-34 | 32 |
| Hình 24 Độ chính xác của mạng số 1 trong quá trình nhận diện 5 địa danh | 35 |
| Hình 25 Giá trị lỗi của mạng số 1 trong quá trình nhận diện 5 địa danh..... | 35 |
| Hình 26 Độ chính xác của mạng số 1 trong quá trình nhận diện 48 địa danh | 36 |
| Hình 27 Giá trị lỗi của mạng số 1 trong quá trình nhận diện 48 địa danh..... | 36 |
| Hình 28 Độ chính xác của mạng số 2 trong quá trình nhận diện 5 địa danh | 37 |
| Hình 29 Giá trị lỗi của mạng số 2 trong quá trình nhận diện 5 địa danh..... | 37 |
| Hình 30 Độ chính xác của mạng số 3 trong quá trình nhận diện 5 địa danh | 38 |
| Hình 31 Giá trị lỗi của mạng số 3 trong quá trình nhận diện 5 địa danh..... | 38 |
| Hình 32 Độ chính xác của mạng số 4 trong quá trình nhận diện 5 địa danh | 39 |
| Hình 33 Giá trị lỗi của mạng số 4 trong quá trình nhận diện 5 địa danh..... | 39 |
| Hình 34 Độ chính xác của mạng số 4 trong quá trình nhận diện 48 địa danh | 40 |
| Hình 35 Giá trị lỗi của mạng số 4 trong quá trình nhận diện 48 địa danh..... | 40 |
| Hình 36 Biểu đồ ca sử dụng..... | 42 |
| Hình 37 Biểu đồ lớp cho ca sử dụng tìm kiếm thông tin địa danh | 42 |

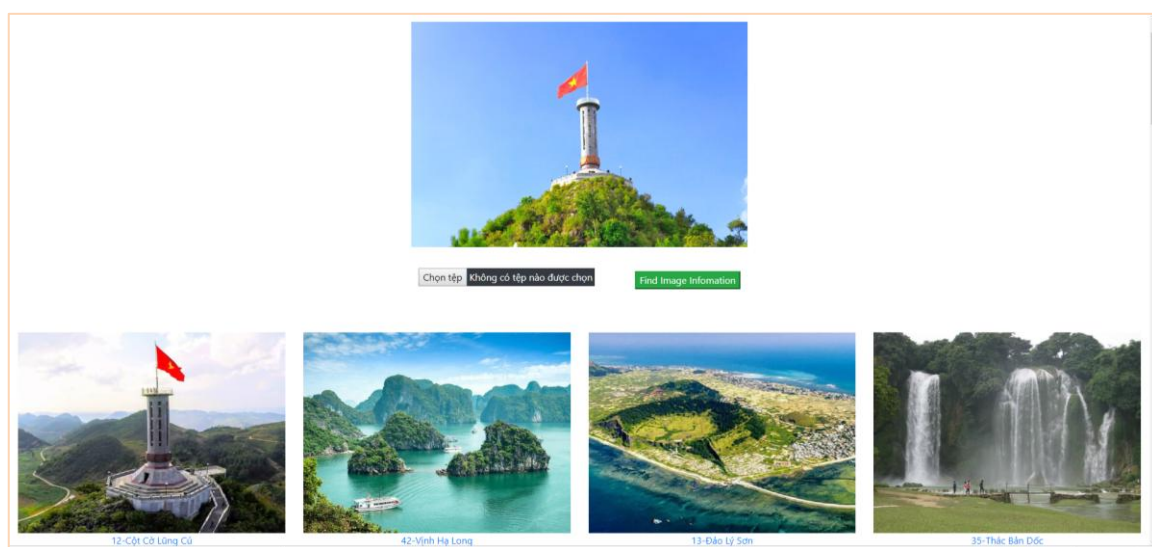
| | |
|---|----|
| Hình 38 Biểu đồ hoạt động cho ca sử dụng tìm kiếm thông tin địa danh..... | 43 |
| Hình 39 Tìm kiếm thông tin hình 1 | 45 |
| Hình 40 Tìm kiếm thông tin hình 2..... | 45 |
| Hình 41 Tìm kiếm thông tin hình 3..... | 45 |

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 ... và 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 càng phổ biến.

Mục tiêu của đồ án là giải bài toán tự động xác định địa điểm du lịch thông qua hình ảnh người dùng cung cấp.



Hình 1 Hệ thống tìm kiếm thông tin địa danh bằng hình ảnh

1.2. Phương pháp thực hiện

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 địa điểm du lịch khác nhau trên lãnh thổ Việt Nam.

Với bài toán nhận 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

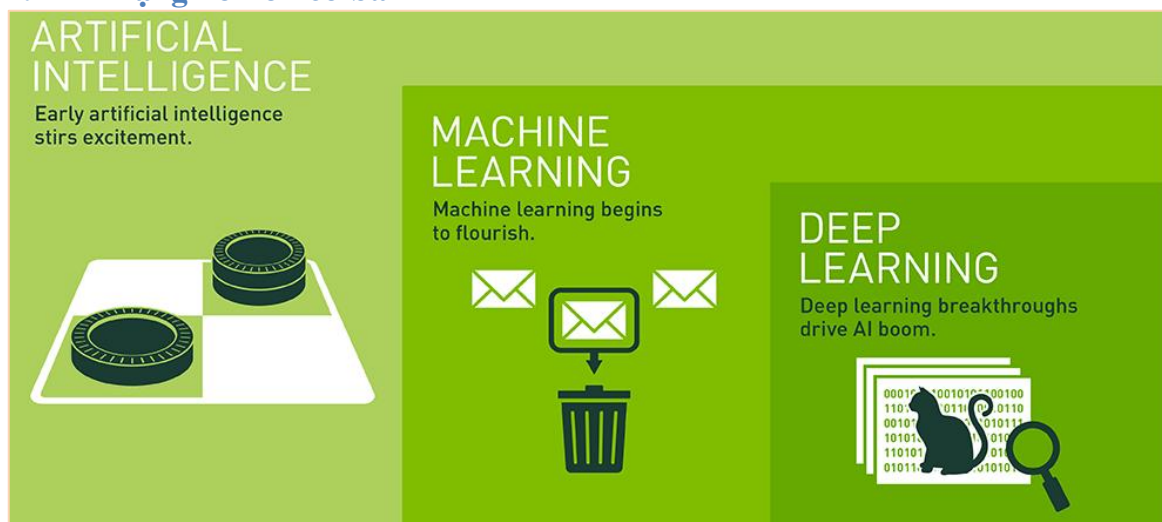
Người dùng khi xem ảnh có thể ngay lập tức tìm kiếm thông tin địa danh thông qua hình ảnh mình đang xem. Mọi thứ sẽ trở nên nhanh chóng và thuận tiện cho người sử dụng góp phần thúc đẩy ngành dịch vụ và du lịch của Việt Nam.

1.4. Kết quả dự kiến

Xây dựng mạng neuron nhân tạo nhận diện 48 địa danh thông qua hình ảnh với độ chính xác trên 80%, triển khai hệ thống trên nền tảng 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 – TỔNG QUAN MẠNG NƠ-RON

2.1 Mạng nơ-ron cơ bản



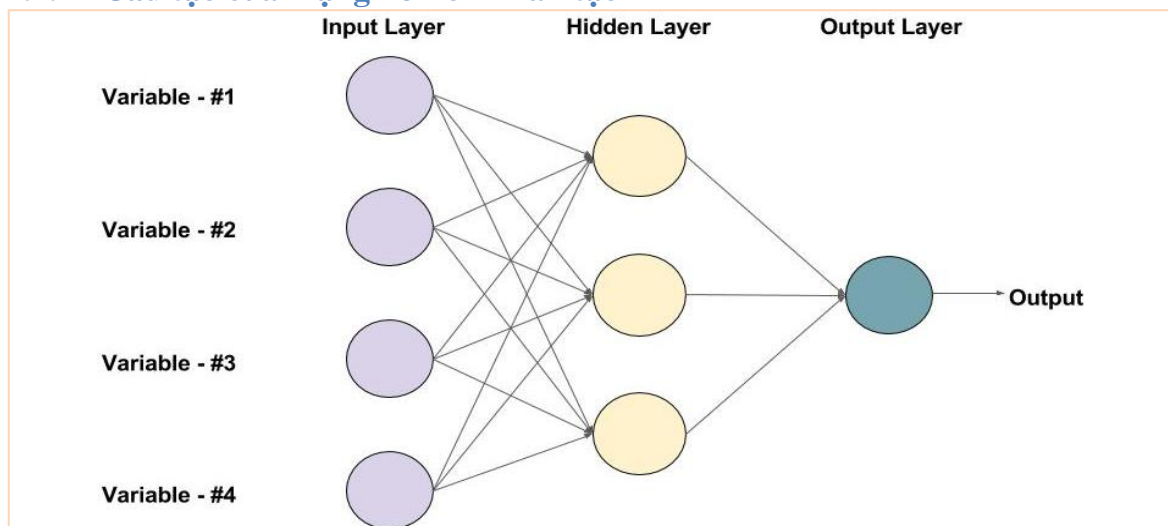
Hình 2 Tổng quan về trí tuệ nhân tạo

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.

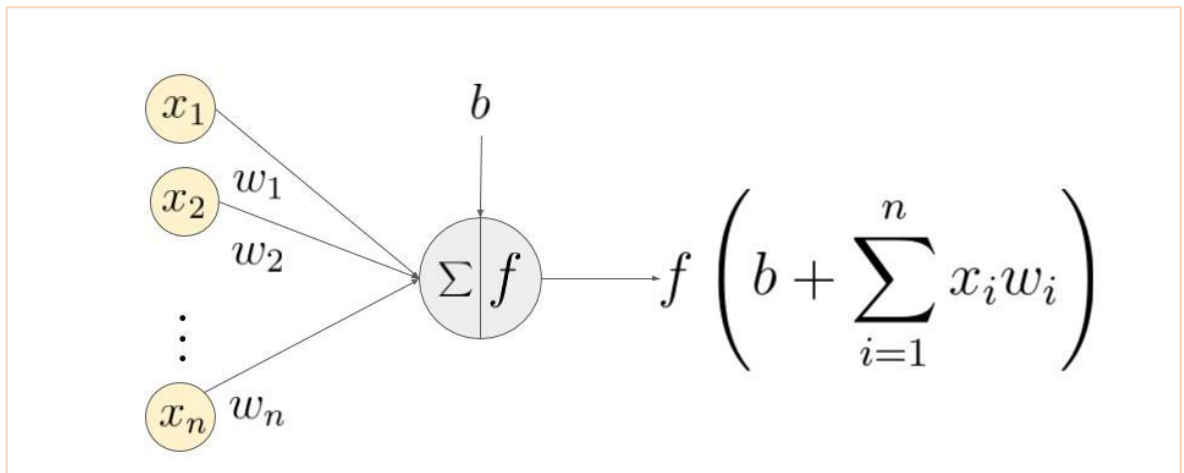
Học sâu là tập con của học máy, là việc máy có thể tự trích xuất ra các đặc trưng của dữ liệu thông qua mạng nơ-ron nhân tạo. Sau đây là định nghĩa về mạng nơ-ron.

2.1.1 Cấu tạo của mạng nơ-ron nhân tạo



Hình 3 Mạng nơ-ron nhân tạo

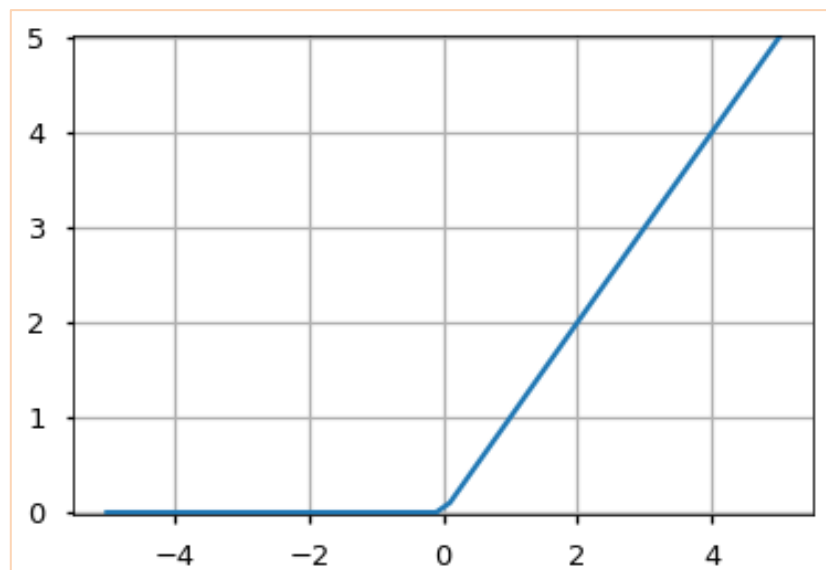
Mạng gồm lớp đầu vào, lớp đầu ra và một hoặc nhiều lớp ở giữa gọi là lớp ẩn (hidden layer) các lớp được kết nối với nhau theo mô tả như hình trên. Mỗi một phần tử trong lớp ẩn gọi là một nơ-ron.



Hình 4 Nơ-ron trong mạng nơ-ron network

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

Kết quả đầu ra của nơ-ron sẽ là kết quả của hàm f được mô tả trên hình. Trong đó các giá trị x là đầu vào của nơ-ron, 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 5 Hàm Relu

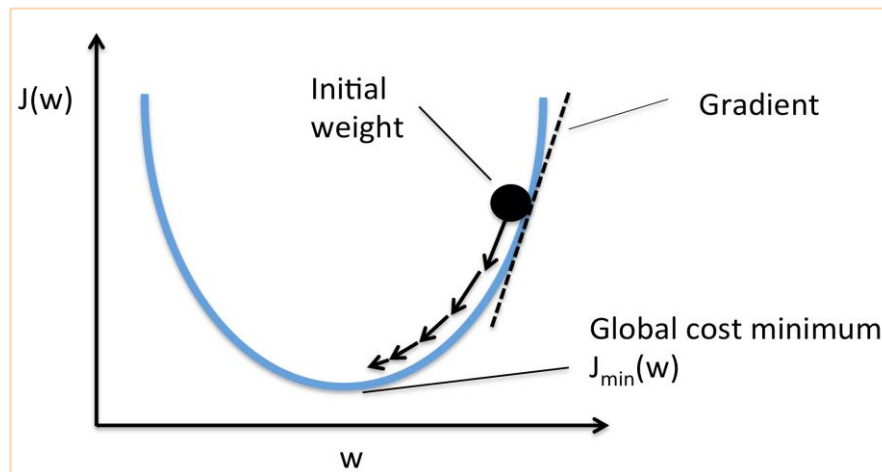
Giá trị đầu ra của hàm sẽ bằng giá trị đầu vào khi giá trị đầu vào lớn hơn 0. Giá trị đầu ra của hàm sẽ là 0 nếu giá trị của đầu vào nhỏ hơn hoặc bằng 0.

2.1.2 Huấn luyện mạng nơ-ron nhân tạo

Định nghĩa hàm mất mát: Với bài toán học có giám sát, trong quá trình huấn luyện, mạng nơ-ron 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 huấn 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 6 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 biến. Ban đầu, w nhận một giá trị bất kỳ 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 $J(\theta)$ 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 nơ-ron 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 tính chính 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.

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

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 nơ-ron. 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 các 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 (iterations) khác nhau dưới đây là hai cách huấn luyện chính.

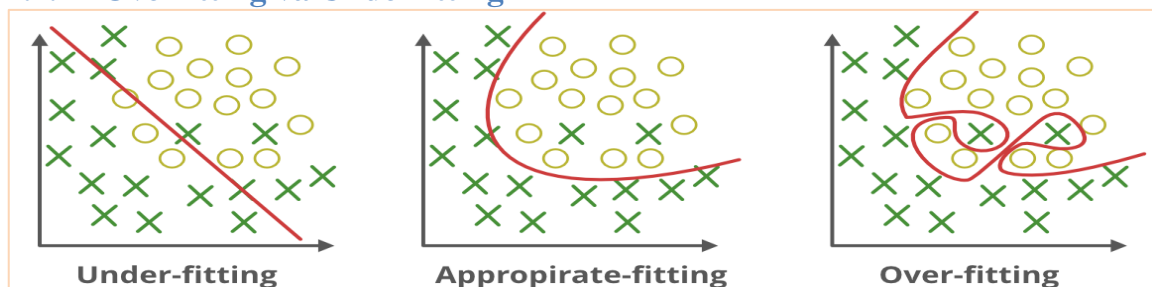
Stochastic gradient descent: Là thuật toán tối ưu hàm mất mát theo từng phần tử của tập dữ liệu. Nhược điểm của phương pháp này là giá trị đạo hàm biến thiên lớn dẫn tới mạng không ổn định. Các tham số trong mạng có khoảng biến thiên lớn.

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 những 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 đạo hàm tính được từ hàm mất mát và giá trị đạo hàm của các vòng lặp trước đó. Một 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



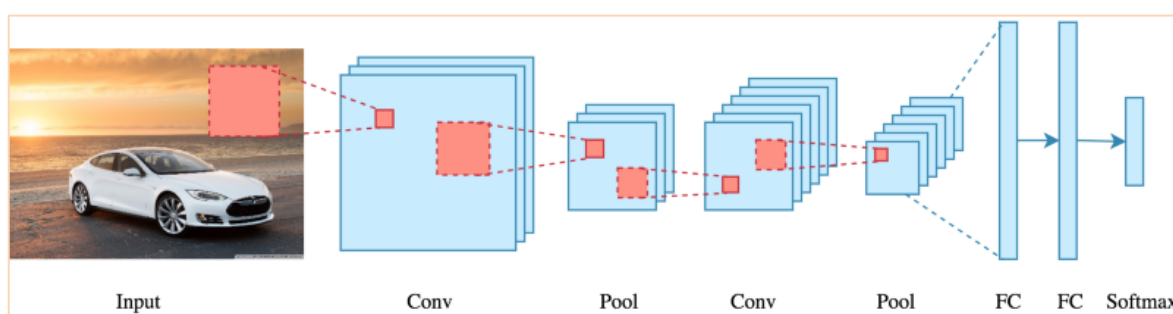
Hình 7 Overfitting và Underfitting

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 Mạng nơ-ron tích chập

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 8 Sơ đồ tổng quát CNN

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

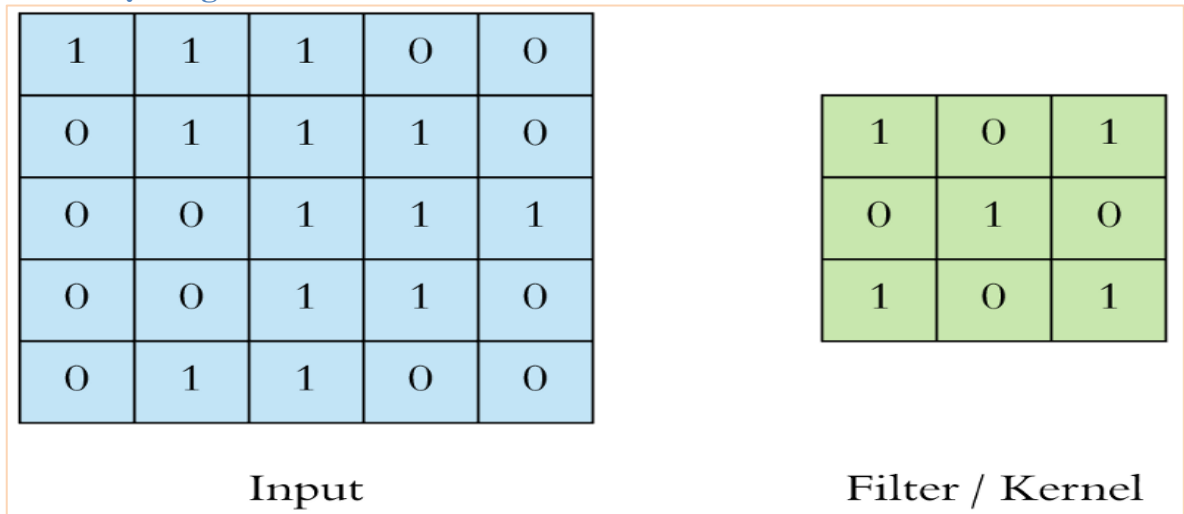
Local receptive fields: Trong mạng nơ-ron network truyền thống mỗi một nơ-ron trong lớp đầu vào kết nối với một nơ-ron trong lớp ẩn. Tuy nhiên trong CNN chỉ một vùng xác định trong các nơ-ron trong lớp đầu vào kết nối với một nơ-ron trong lớp ẩn. Những vùng xác định nêu trên gọi là local receptive fields. Sự kết nối giữa lớp đầu vào và lớp ẩn được chính là việc 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 lớp ẩn.

Shared weights và biases: Giống với mạng nơ-ron truyền thống CNN cũng có tham số weights và biases. Các tham số này được học trong suốt quá trình huấn luyện mạng và liên tục cập nhật giá trị với mỗi mẫu mới (new training example). Tuy nhiên, các trọng số trong CNN là giống nhau đối với mọi neural trong cùng một lớp (layer). Điều này có nghĩa là tất cả các nơ-ron trong cùng một lớp ẩn 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 hàm ví dụ hàm ReLU. Giá trị thu được sau phép biến đổi là giá trị dương nhất có thể của đầu ra, trong trường hợp đầu ra mang giá trị âm thì giá trị nhận được là 0. Pooling là một bước nhằm giảm số chiều của

ma trận. Cách 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)

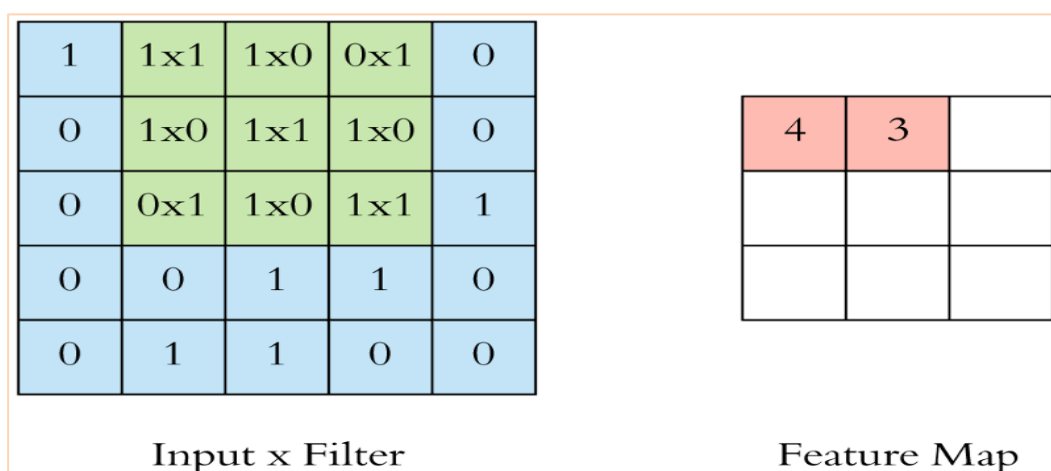
2.2.1 Định nghĩa convolution



Hình 9 Khái niệm convolution

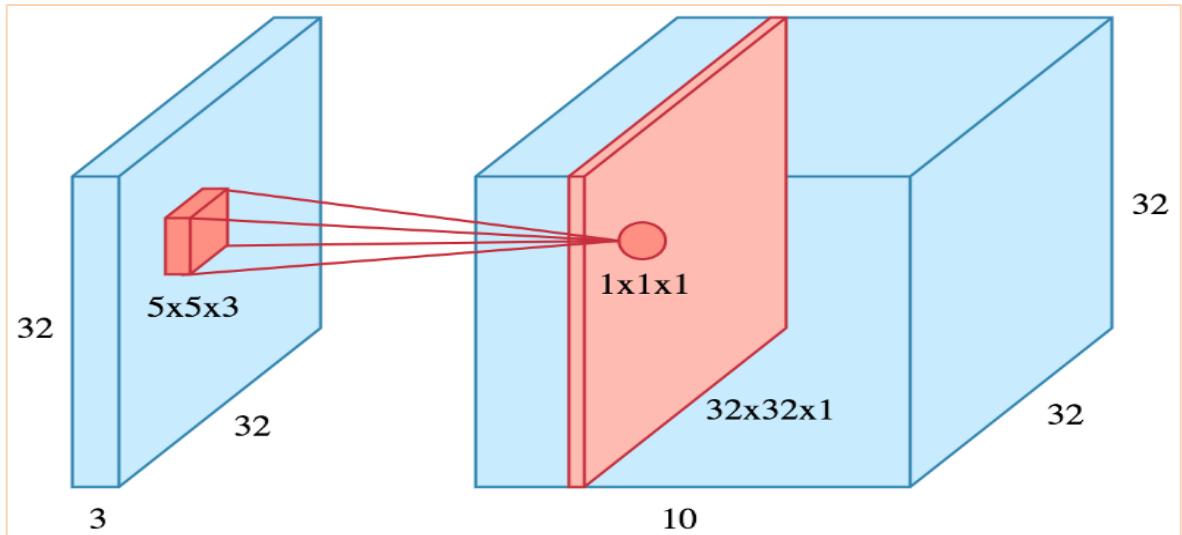
Khối cơ bản tạo lên CNN là các lớp tích chập (convolutional layer). Convolution là một phép toán học để kết hợp hai khối thông tin với nhau. Trong trường hợp này, convolution được áp dụng trên dữ liệu đầu vào (ma trận) và sử dụng một mặt nạ gọi là convolution filter để tạo ra một mảng mới gọi là feature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

Việc thực hiện phép toán convolution được mô tả như hình dưới đây với đầu vào là một mảng hai chiều 5x5 phần tử là filter có kích thước là 3x3 phần tử. Cửa 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 10 Convolutional và mảng hai chiều

Trên đây là mô tả thực hiện phép toán convolution với ma trận hai chiều dùng một bộ lọc (filter) duy nhất. Trong thực tế đối với ảnh RGB ta thực hiện convolution với ma trận ba chiều ví dụ như ảnh RGB. 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 chồng lên nhau ta thu được một convolution layer. Ví dụ sau thể hiện ảnh có kích thước 32x32 và có ba kênh màu, ta sử dụng 10 filter và thu được convolution layer là một ma trận 32x32x10.



Hình 11 Convolutional và mảng ba chiều

2.2.2 Định nghĩa stride và padding

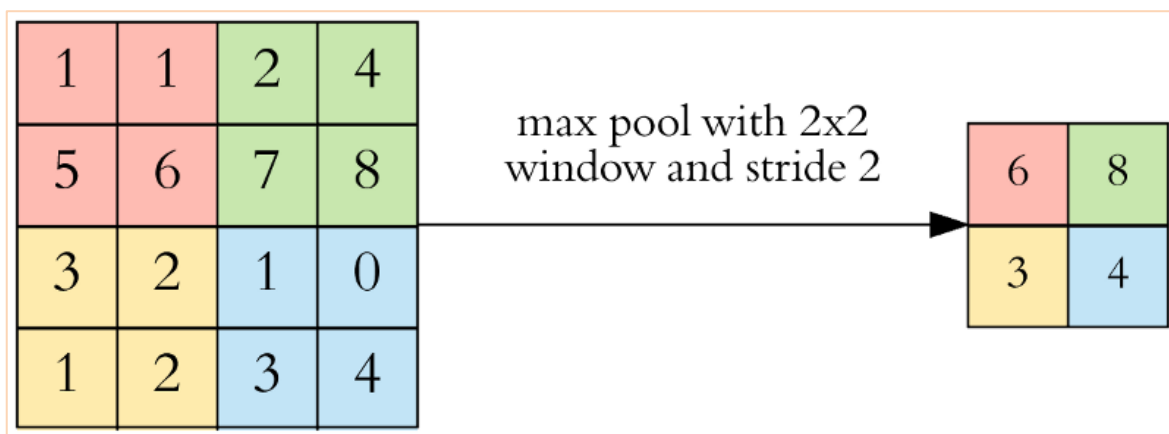
Stride là số bước nhảy của mỗi lần dịch chuyển convolution filter, trong ví dụ đầu tiên về convolution ta nhận thấy kích thước của feature map nhỏ hơn kích thước của ma trận đầu vào. Để kích thước của feature map bằng kích thước của ma trận đầu vào ta cần phải bổ xung thêm một số điểm bao quanh ma trận 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 $n \times n$, stride là s , kích thước filter là $f \times f$, vùng padding có kích thước là p ta có kích thước của feature map thu được là:

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

2.2.3 Lớp pooling trong mạng CNN

Sau khi thực hiện phép toán convolution chúng ta thường sử 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 nhảy (stride). Dưới đây là ví dụ việc áp dụng max pooling sử dụng 2x2 window và stride là 2.



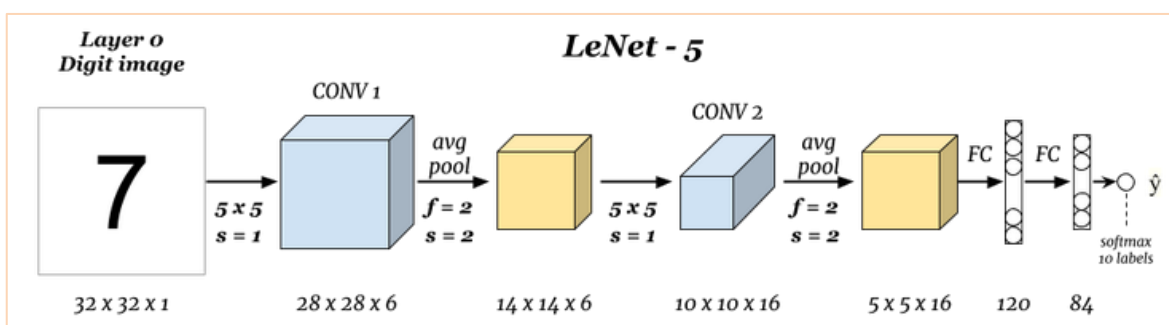
Hình 12 Minh họa max pooling

Các lớp phía sau cùng FC viết tắt của fully connected neural, chính là các lớp ẩn trong mạng nơ-ron 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 xác suất rơi vào mỗi loại tương ứng khi đem một ảnh làm đầu vào của mạng để dự đoán.

2.3 Một số kiến trúc mạng CNN phổ biến

2.2.1 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 điểm 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 13 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.

Bảng 1 Kiến trúc mạng LeNet-5

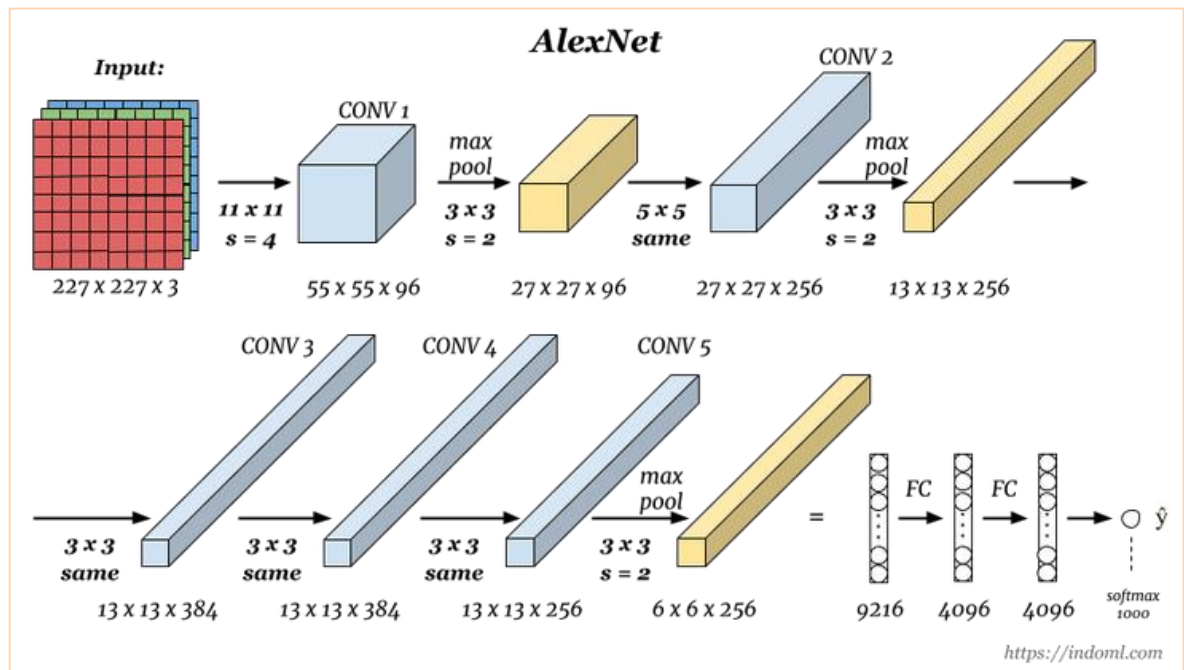
| Layer | Filters | Biases | Stride | Tensor | Parameters |
|--------------|----------|--------|--------|----------|--------------|
| Input | 0 | 0 | 0 | 32x32x1 | 0 |
| Conv-1 | 5x5x1x6 | 6 | 1 | 28x28x6 | 156 |
| MaxPool-1 | 2x2 | 0 | 2 | 14x14x6 | 0 |
| Conv-2 | 5x5x6x16 | 16 | 1 | 10x10x16 | 2416 |
| 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 |

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

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

2.2.2 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 14 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.

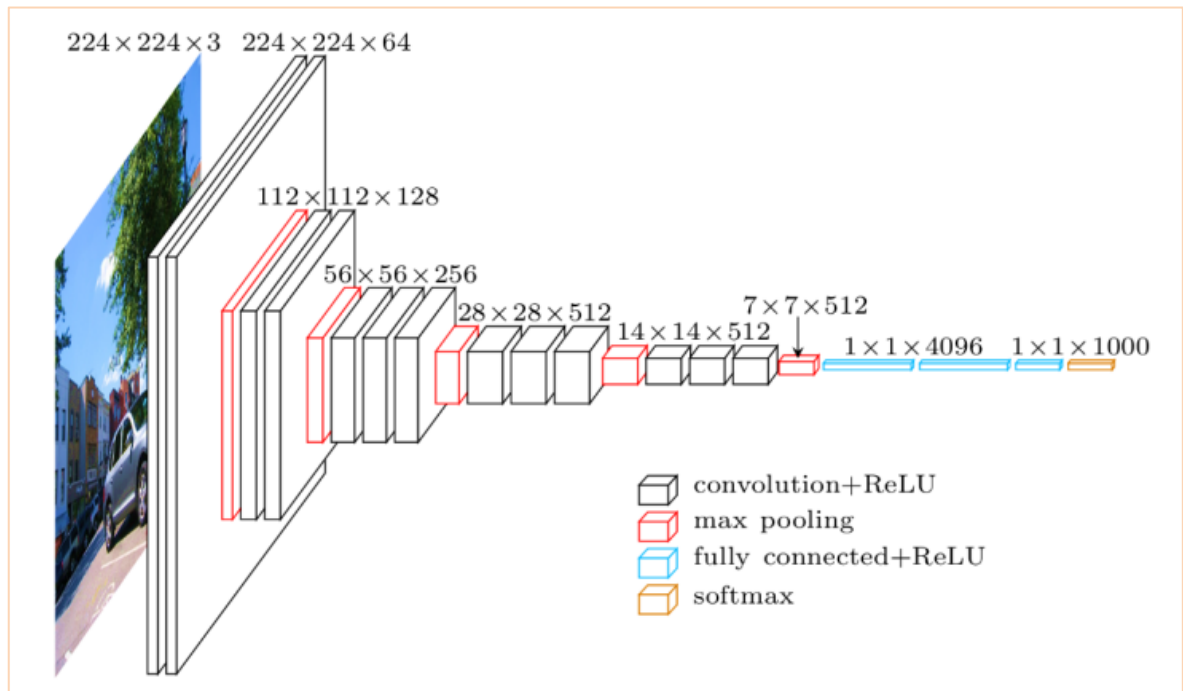
Bảng 2 Kiến trúc mạng AlexNet

| Layer | Filters | Biases | Stride | Tensor | Parameters |
|--------------|-------------|--------|--------|-----------|-----------------|
| 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 |

Mạng với đầu vào là ảnh có kích thước 227x227x3 và kết quả đầu cần thực hiện là phân loại 1000 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 training mạng bởi thuật toán backpropagation.

2.2.3 Kiến trúc mạng VGG-16

Mạng VGG-16 được công bố trong bài báo của Karen Simonyan and Andrew Zisserman vào năm 2015.



Hình 15 Sơ đồ kiến trúc mạng VGG16

Thông tin các tham số về mạng được mô tả trong bảng 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.

Bảng 3 Kiến trúc mạng VGG16

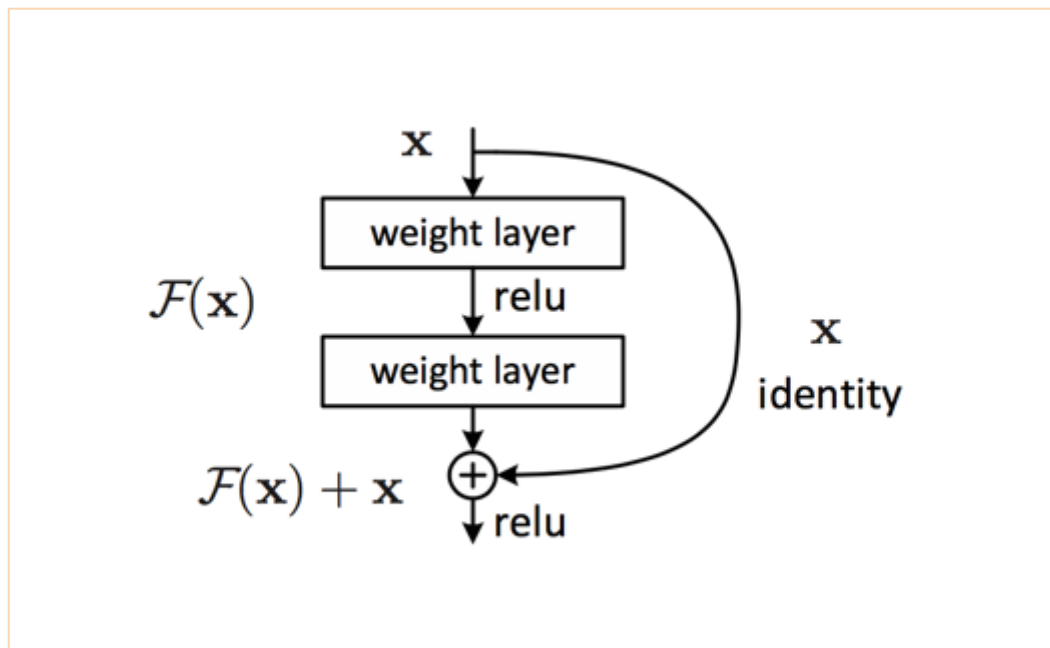
| Layer | Filters | Biases | Stride | Tensor | Parameters |
|-----------|-------------|--------|--------|-------------|------------|
| Input | 0 | 0 | 0 | 224x224x3 | 0 |
| Conv-11 | 3x3x3x64 | 64 | 1 | 224x224x64 | 1792 |
| Conv-12 | 3x3x64x64 | 64 | 1 | 224x224x64 | 36928 |
| MaxPool-1 | 2x2 | 0 | 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 |

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.

2.2.4 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 16 Sơ đồ phân tử Residual Block



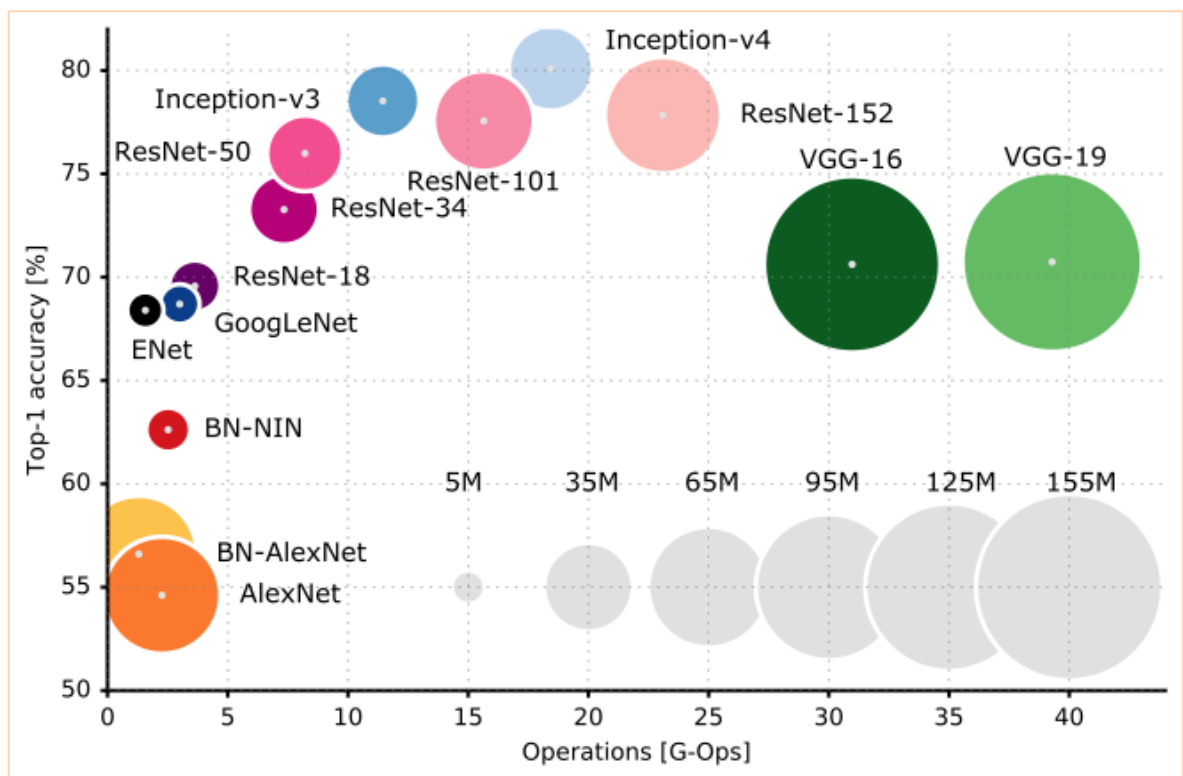
Hình 17 Sơ đồ kiến trúc mạng ResNet

| layer name | output size | 18-layer | 34-layer | 50-layer | 101-layer | 152-layer |
|------------|------------------|---|---|---|--|--|
| conv1 | 112×112 | $7 \times 7, 64, \text{stride } 2$ | | | | |
| conv2_x | 56×56 | $3 \times 3 \text{ max pool, stride } 2$ | | | | |
| | | $\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$ | $\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$ | $\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$ | $\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$ | $\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$ |
| conv3_x | 28×28 | $\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$ | $\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$ | $\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$ | $\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$ | $\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 8$ |
| conv4_x | 14×14 | $\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$ | $\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$ | $\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$ | $\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$ | $\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 36$ |
| conv5_x | 7×7 | $\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$ | $\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$ | $\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$ | $\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$ | $\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$ |
| | 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 18 Thiết kế chi tiết một số mạng Resnet cơ bản

2.2.5 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 19 Độ chính xác và khối lượng tính toán của một số mạng

Hình trên được lấy từ <https://www.researchgate.net>, 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 và ResNet-34 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 về 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 3 – ỨNG DỤNG MẠNG NƠ-RON TÍCH CHẬP CHO BÀI TOÁN NHẬN DẠNG ĐỊA DANH

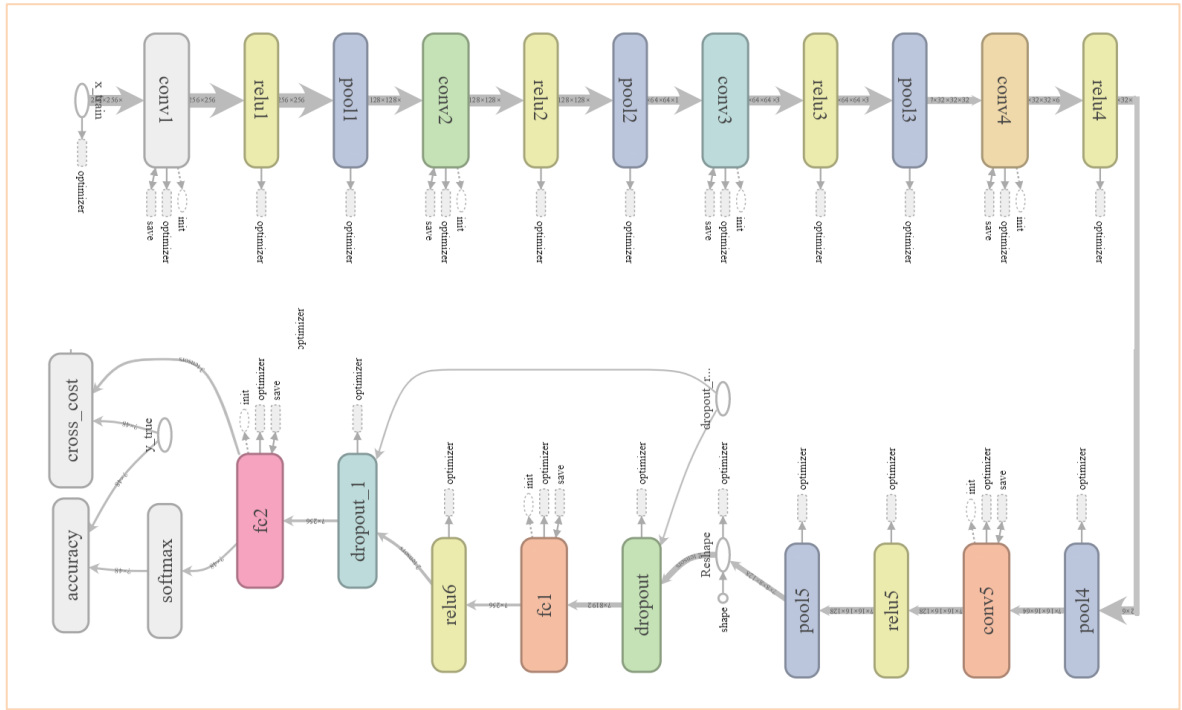
3.1 Đề xuất mạng dựa trên kiến trúc mạng LeNet-5

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

Bảng 4 Chi tiết tham số của mô hình dựa trên mạng lenet-5

| 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 | 8192x256 | 256 | 0 | 256 | 2097408 |
| FC-2 | 256x48 | 48 | 0 | 48 | 12336 |
| Total | | | | | 2208138 |

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

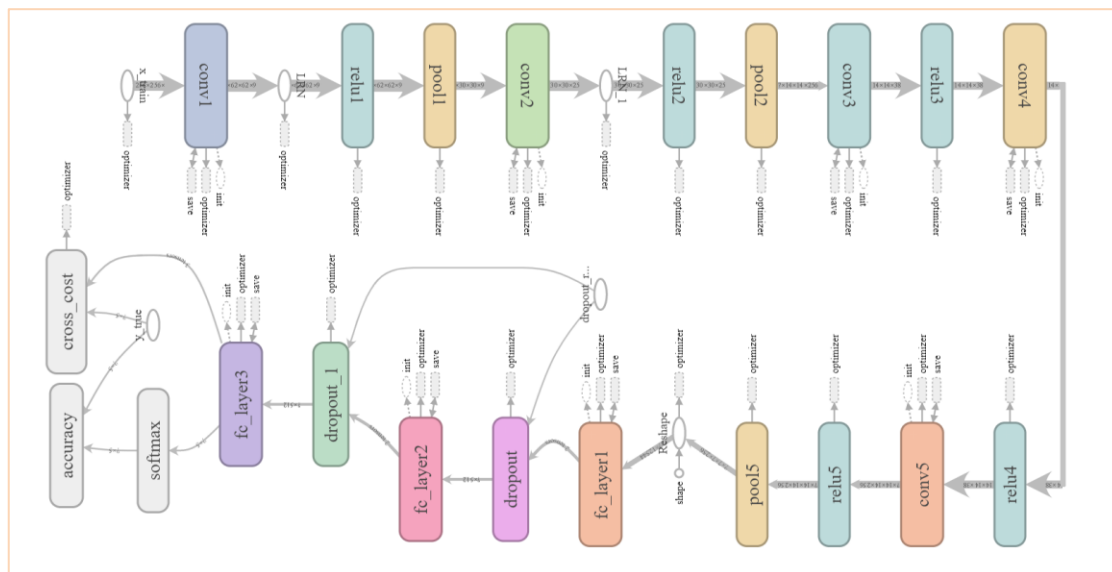
3.1 Đề xuất mạng dựa trên kiến trúc mạng AlexNet

Dưới đây là tham số các lớp của mô hình triển khai dựa trên mạng AlexNet.

Bảng 5 Chi tiết tham số của mô hình dựa trên mạng alexnet

| Layer | Filters | Biases | Stride | Tensor | Parameters |
|-----------|-------------|--------|--------|-----------|------------|
| 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 | 12544x512 | 512 | 0 | 512 | 6422528 |
| FC-2 | 512x512 | 512 | 0 | 512 | 262656 |
| FC-3 | 512x5 | 5 | 0 | 5 | 2560 |
| Total | | | | | 10434944 |

Với các tính toán ở phía trên, mạng được triển khai trên tensorflow 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

3.4 Đề xuất 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.

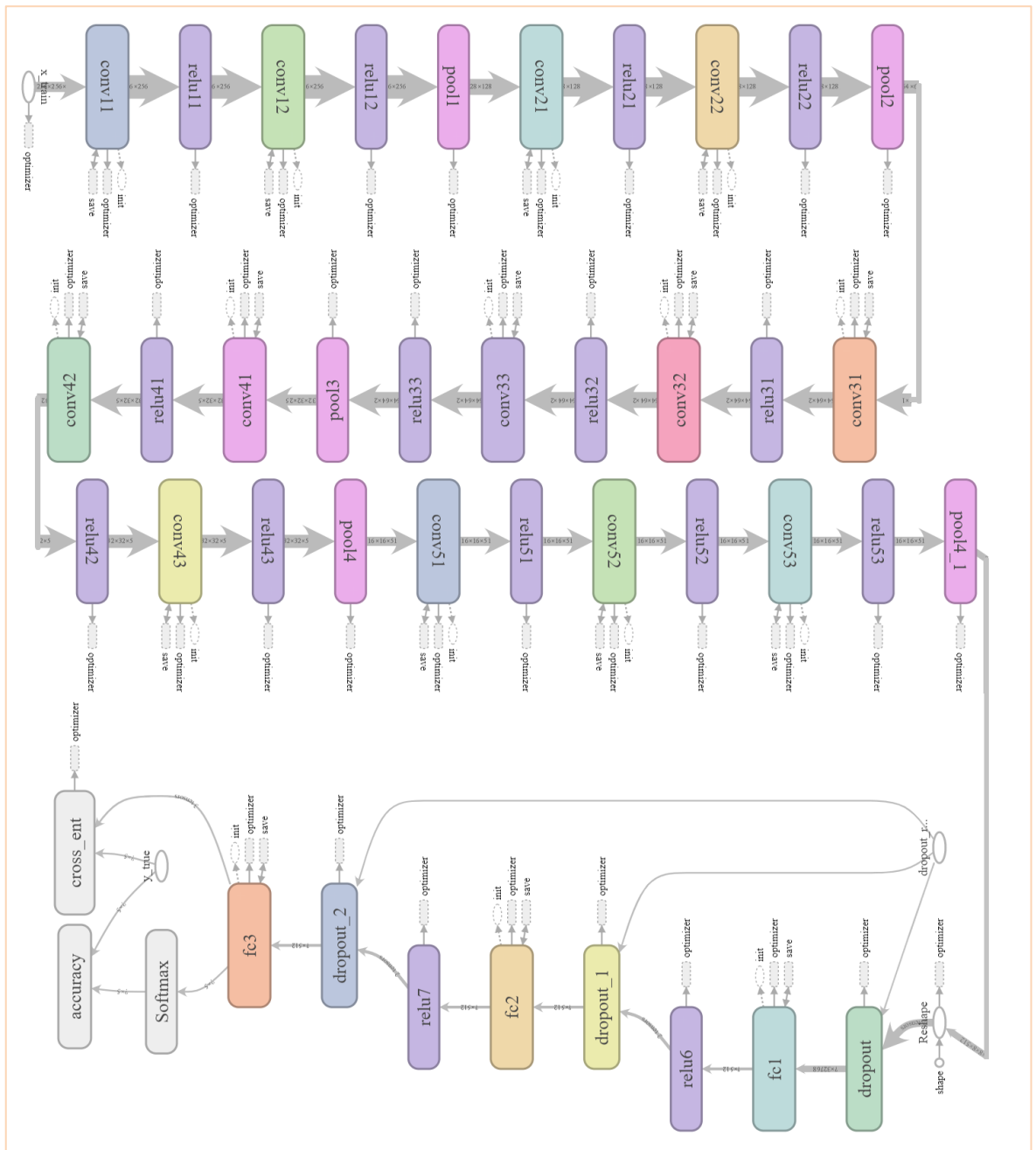
Bảng 6 Chi tiết tham số của mô hình dựa trên vgg16

| Layer | Filters | Biases | Stride | Tensor | Parameters |
|-----------|-------------|--------|--------|-------------|------------|
| 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 | 32768x512 | 512 | 0 | 512 | 16777728 |
| FC-2 | 512x512 | 512 | 0 | 512 | 262656 |
| FC-3 | 512x5 | 5 | 0 | 5 | 2565 |
| Total | | | | | 31755077 |

Để giảm số lượng tham số của mạng giúp việc huấn luyện và tối ưu mạng nhanh hơn. Em điều chỉnh số lượng nút ở các lớp thuộc mạng nơ-ron cơ bản về 512 nơ-ron.

Sau đây là kiến trúc mạng được triển khai trong chương trình dùng thư viện tensorflow. Và được mô phỏng trên bộ công cụ tensorboard của tensorflow.



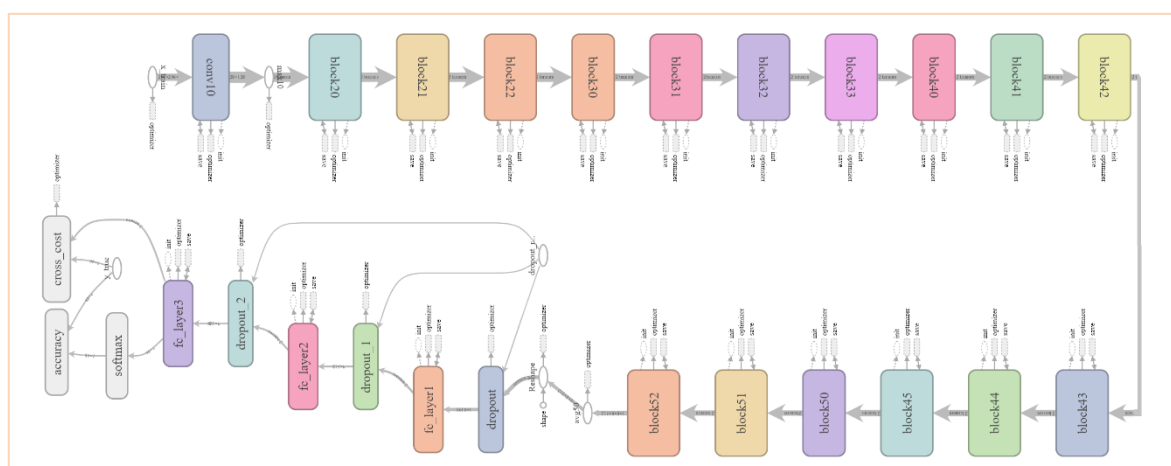
Hình 22 Mô phỏng kiến trúc mạng dựa trên mạng vgg16

3.3 Đề xuất mạng dựa trên kiến trúc mạng Resnet

Dưới đây là bảng thiết kế các lớp của bài toán khi triển khai dựa trên mạng Resnet-34. 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 Resnet-34 để thử nghiệm, với đầu vào là ảnh 256x256x3. Chi tiết các lớp được mô tả trong bảng sau.

Bảng 7 Chi tiết tham số của mô hình dựa trên Resnet-34

| 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 | 4x4x512x512 | 512 | | 512 | 4194816 |
| FC-2 | 512x128 | 128 | | 128 | 65664 |
| FC-3 | 128x48 | 48 | | 48 | 6192 |
| Total | | | | | 4302384 |



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. Trong chương 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.

3.5 Hàm mất mát và hàm tính độ chính xác trong quá trình huấn luyện

Hàm mất mát được sử dụng trong quá trình huấn luyện mô hình là hàm Loss. Hàm có phương trình như sau.

$$\text{Loss} = \sum_{(x,y) \in D} -y \log(y') - (1-y) \log(1-y')$$

Trong đó D là tập dữ liệu, x là đối tượng đầu vào của mạng, y là nhãn của tương ứng của dữ liệu đó, y' là giá trị dự đoán của mạng đối với dữ liệu đầu vào là x .

Độ chính xác của mô hình được đo bởi công thức

$$\text{Accuracy} = \frac{1}{N} \sum_{(x,y) \in D} \text{equal}(y', y)$$

N là số lượng phần tử trong tập dữ liệu, hàm `equal` trả về giá trị là 1 khi y' và y bằng nhau, trả về giá trị 0 khi y' và y có giá trị khác nhau.

CHƯƠNG 4 - THỬ NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ

4.1 Bộ dữ liệu sử dụng cho bài toán

Bộ dữ liệu zalo được download từ trang <https://challenge.zalo.ai> dữ liệu gồm tập hợp các ảnh có kích thước 480x480x3 định dạng ảnh RGB về các địa danh du lịch nổi tiếng của Việt Nam. Bộ dữ liệu có khoảng 30000 ảnh cho 103 địa danh. Số lượng ảnh phân bố cho mỗi địa danh không đều nằm trong khoảng từ 50 đến 3000 ảnh. Trong 103 địa danh có nhiều địa danh có phong cảnh tương tự nhau ví dụ các địa danh ở vùng Tây Bắc thường có hình ảnh giống nhau mà ngay bản thân mắt người cũng khó phân biệt các địa danh đó với nhau. Trong phạm vi đồ án này em chọn ra 48 địa danh có nét đặc trưng phong cảnh đôi một khác nhau để huấn luyện mô hình.

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.

Do số lượng ảnh trên mỗi địa danh phân bố không đều, để tăng độ chính xác của mô hình cần tăng số lượng ảnh cho mỗi địa danh và làm cho số lượng ảnh cho mỗi địa danh cân bằng nhau. Để giải quyết vấn đề trên cần sử dụng một số kỹ thuật augmentation.

Các kỹ 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 kỹ thuật Crop để dữ liệu đầu vào được cân bằng giữa các địa danh (class) khác nhau.

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 48 địa danh.

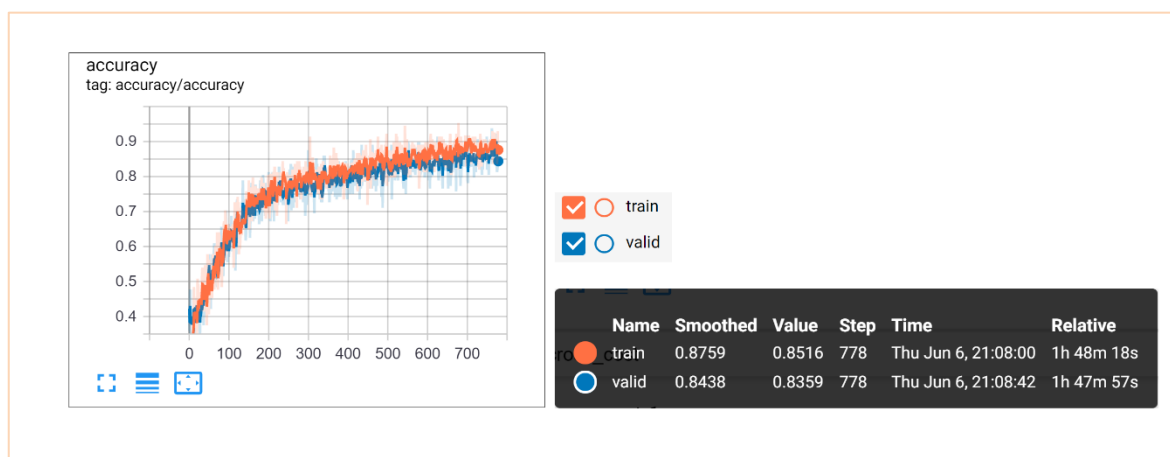
Sau khi áp dụng kỹ thuật augmentation, mỗi địa danh có 1200 ảnh dùng để huấn luyện, 400 ảnh để kiểm chứng độ chính xác.

Đường màu vàng cam biểu thị cho độ chính xác và lỗi trên tập dữ liệu huấn luyện, đường màu xanh biểu thị cho độ chính xác và lỗi trên tập dữ liệu kiểm chứng.

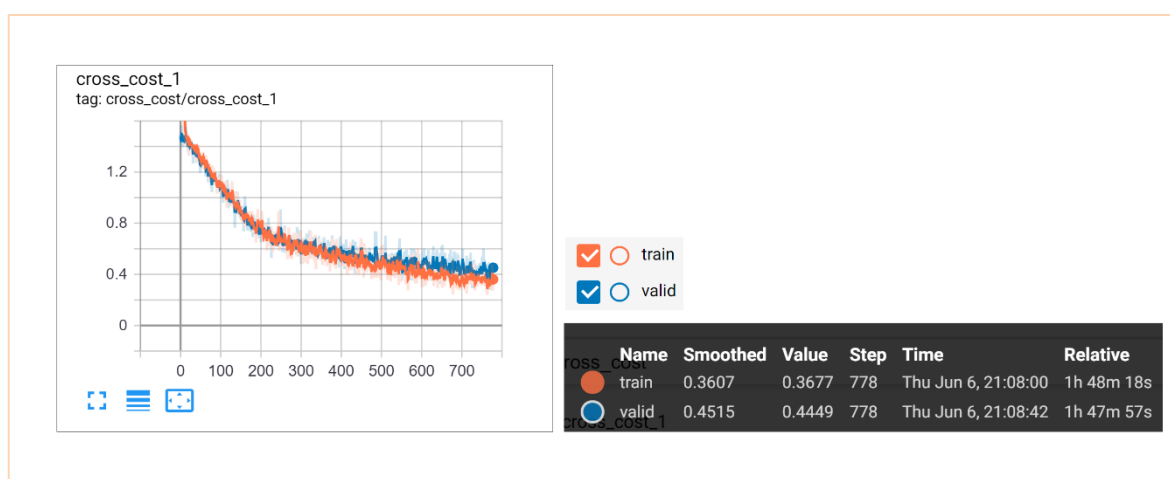
Quá trình huấn luyện đều được thực hiện trên máy tính hệ điều hành window-10, CPU Core-i7, GPU 1080-TI.

4.2 Kết quả huấn luyện mạng dựa trên mạng lenet-5

Để thuận tiện cho việc nhắc tới mạng này, em gọi mạng này là mạng số 1.



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



Hình 25 Giá trị lỗi của mạng số 1 trong quá trình nhận diện 5 địa danh

Trên đây là kết quả của mạng được huấn luyện với tập dữ liệu có 5 địa danh, mục tiêu của việc chọn ra 5 địa danh để huấn luyện nhằm mục đích thử nghiệm mạng.

Dữ liệu gồm có 6000 ảnh cho huấn luyện và 2000 ảnh cho việc kiểm thử chia đều cho 5 địa danh.

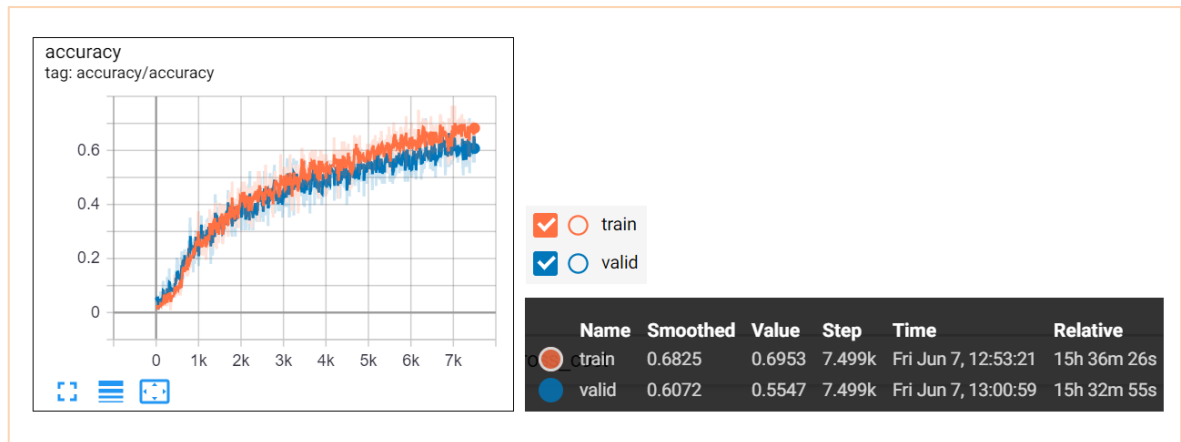
Số lần lặp (epoch) của quá trình huấn luyện là 51 tức là toàn bộ dữ liệu ảnh được đem vào huấn luyện 51 lần. Trong mỗi lần lặp, toàn bộ dữ liệu ảnh sẽ chia thành từng phần gọi là batch để đưa vào mạng. Kích thước của batch là 128 ảnh. Hệ số học (learning rate) là $1E-4$.

Thời gian huấn luyện mạng là 108 phút.

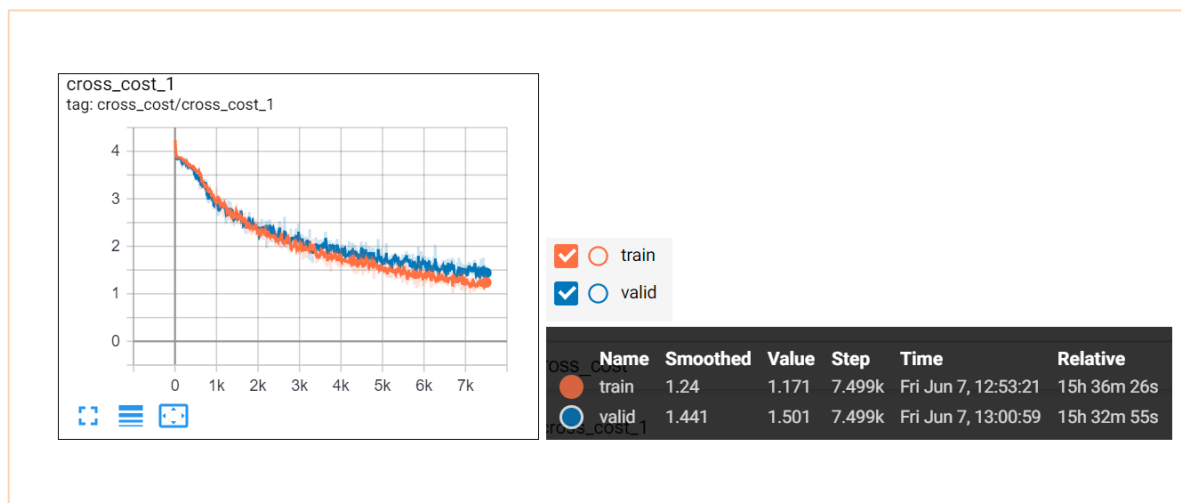
Độ chính xác cuối cùng là 87% trên tập dữ liệu huấn luyện và 84% trên tập dữ liệu kiểm thử. Giá trị lỗi cuối cùng là 0.36 trên tập dữ liệu huấn luyện và 0.45 trên tập dữ liệu kiểm thử.

Chúng ta có thể thấy giá trị của độ chính xác tăng dần một cách đều đặn và giá trị của hàm mất mát giảm đều đặn. Giá trị của độ chính xác và giá trị lỗi trên hai tập huấn luyện và kiểm thử gần với nhau cho thấy mạng đã được cấu hình tốt.

Từ kết quả trên em quyết định thử nghiệm mạng với tập dữ liệu 48 địa danh.



Hình 26 Độ chính xác của mạng số 1 trong quá trình nhận diện 48 địa danh



Hình 27 Giá trị lỗi của mạng số 1 trong quá trình nhận diện 48 địa danh

Số lượng ảnh gồm có 1200x48 ảnh cho huấn luyện, 400x48 cho kiểm thử.

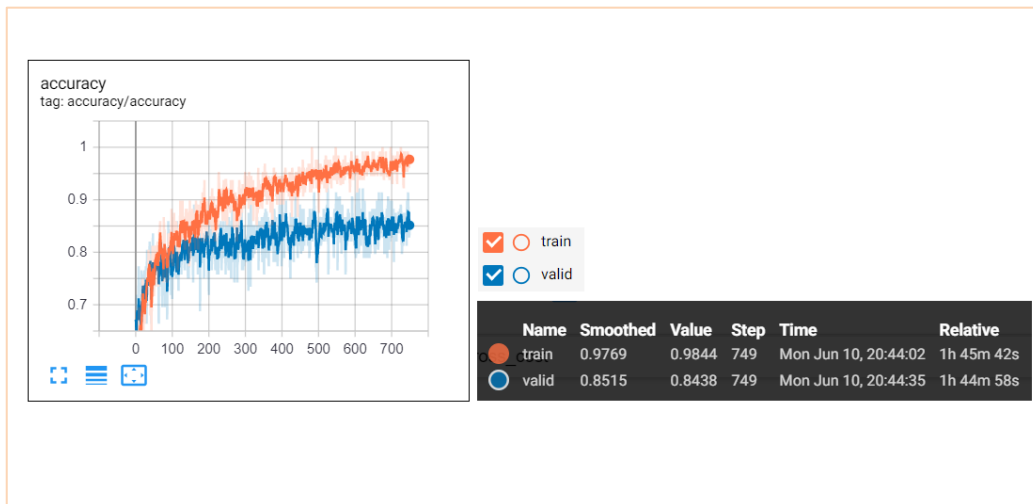
Số lần lặp (epoch) là 50 lần. Độ chính xác cuối cùng 68% trên tập huấn luyện và 60% trên tập kiểm thử.

Thời gian huấn luyện là 15 giờ 30 phút.

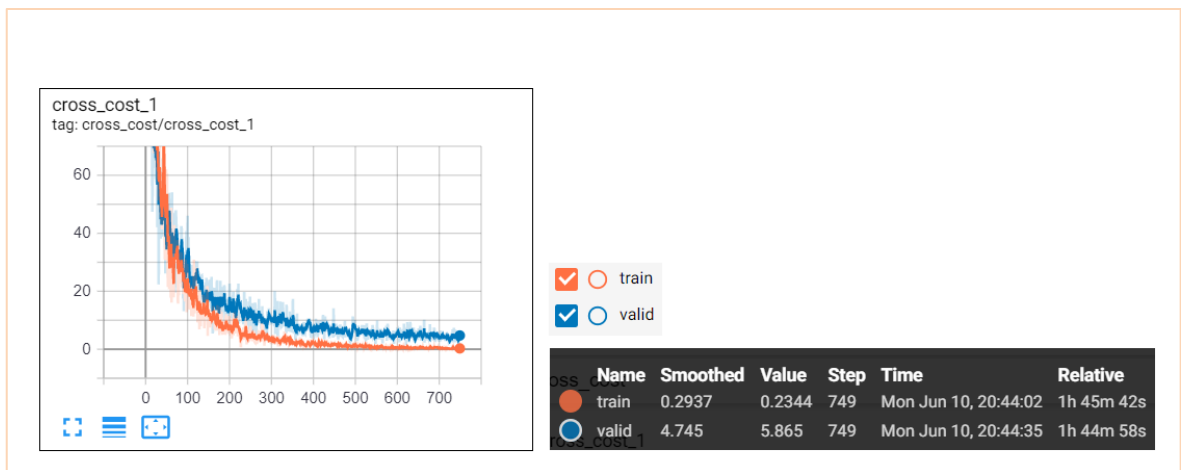
Nhìn vào đồ thị có thể thấy nếu tăng thời gian huấn luyện mạng thì độ chính xác của mạng sẽ tăng. Đồ thị của lần thử nghiệm thứ hai có xu hướng giống với đồ thị thử nghiệm của lần thứ nhất cho thấy khi đã huấn luyện tốt với số lượng đầu ra nhỏ thì khả năng mạng tốt cho số lượng đầu ra lớn là cao.

4.3 Kết quả huấn luyện mạng dựa trên mạng alexnet

Để thuận tiện cho việc nhắc tới mạng này, em gọi mạng này là mạng số 2



Hình 28 Độ chính xác của mạng số 2 trong quá trình nhận diện 5 địa danh



Hình 29 Giá trị lỗi của mạng số 2 trong quá trình nhận diện 5 địa danh

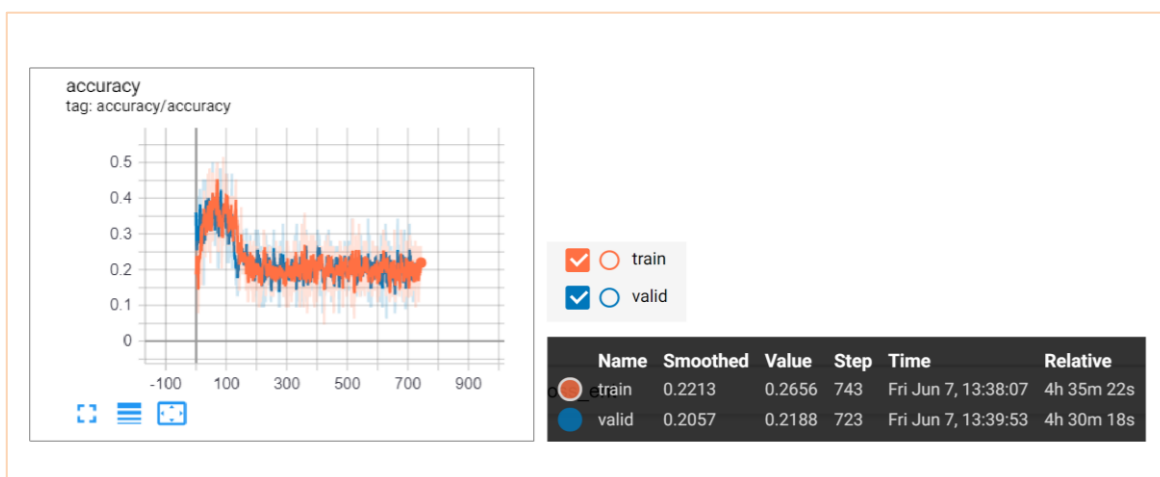
Kết quả cho thu được sau quá trình huấn luyện với 5 địa danh, số lượng ảnh cho huấn luyện là 1200x5 ảnh, số lượng ảnh cho kiểm thử là 400x5 ảnh.

Số lần huấn luyện (epoch) là 50 lần. Hệ số học (learning rate) là $1E-4$. Đồ thị cho thấy các tham số cấu hình chưa được tối ưu, xảy ra hiện tượng bùng nổ đạo hàm. Dẫn tới độ chính xác giảm ở cuối quá trình huấn luyện.

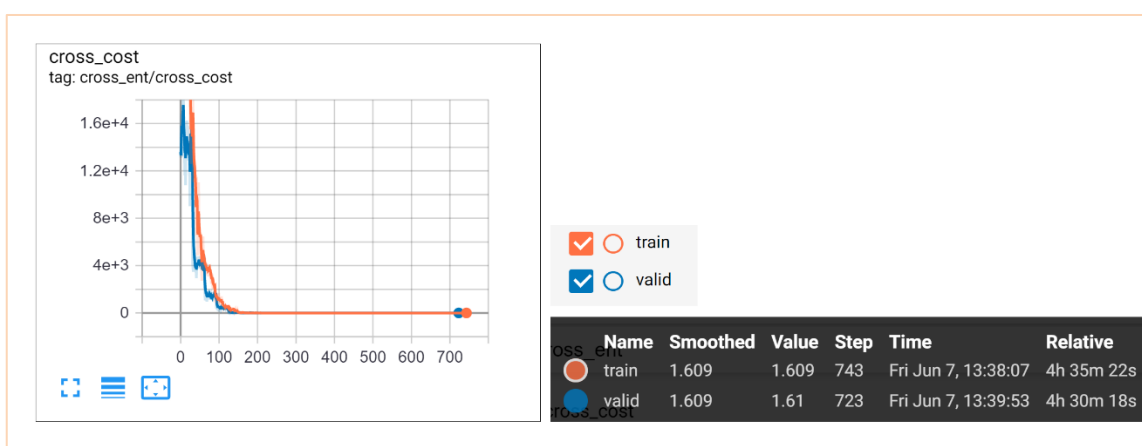
Mạng có độ chính xác trên tập huấn luyện là 97%, độ chính xác trên tập kiểm thử là 84%. Nhìn vào đồ thị ta có thể thấy mạng có hiện tượng overfitting do độ chính xác trên tập huấn luyện khá lớn, còn độ chính xác trên tập kiểm thử lại nhỏ hơn nhiều.

4.4 Kết quả huấn luyện mạng dựa trên mạng vgg-16

Để thuận tiện cho việc nhắc tới mạng này, em gọi mạng này là mạng số 3.



Hình 30 Độ chính xác của mạng số 3 trong quá trình nhận diện 5 địa danh



Hình 31 Giá trị lỗi của mạng số 3 trong quá trình nhận diện 5 địa danh

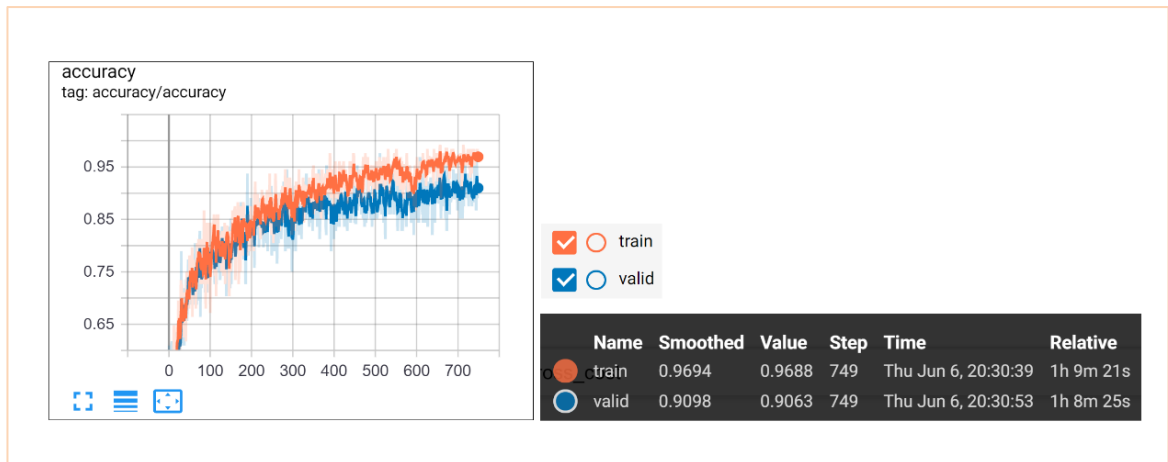
Trên đây là hai biểu đồ thể hiện cho giá trị độ chính xác và lỗi trong quá trình huấn luyện mạng dựa trên mạng vgg16 với việc phân loại 5 địa danh. Với 6000 ảnh huấn luyện và 1200 ảnh kiểm thử. Hệ số học (learning rate) là $1e-4$.

Mạng thực hiện khoảng 47 lần huấn luyện trên toàn bộ tập dữ liệu. Thời gian huấn luyện là 4 giờ 30 phút.

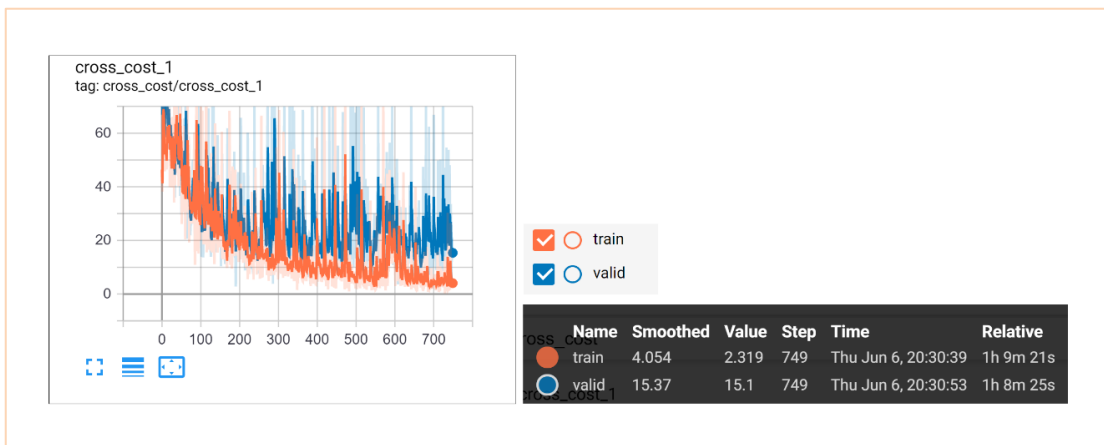
Nhìn vào đồ thị có thể thấy giá trị độ chính xác tăng nhanh thời điểm ban đầu và giảm sau đó. Trong quá trình huấn luyện mạng đã xảy ra hiện tượng bùng nổ đạo hàm dẫn tới độ chính xác của mạng bị giảm ở cuối quá trình huấn luyện. Cần giảm hệ số học xuống. Độ chính xác cao nhất của mạng rất thấp chưa tới 50% nên mạng không thích hợp cho việc tiếp tục tối ưu trong khi giới hạn về thời gian và tài nguyên.

4.5 Kết quả huấn luyện mạng dựa trên mạng resnet-34

Để thuận tiện cho việc nhắc tới mạng này, em gọi mạng này là mạng số 4.



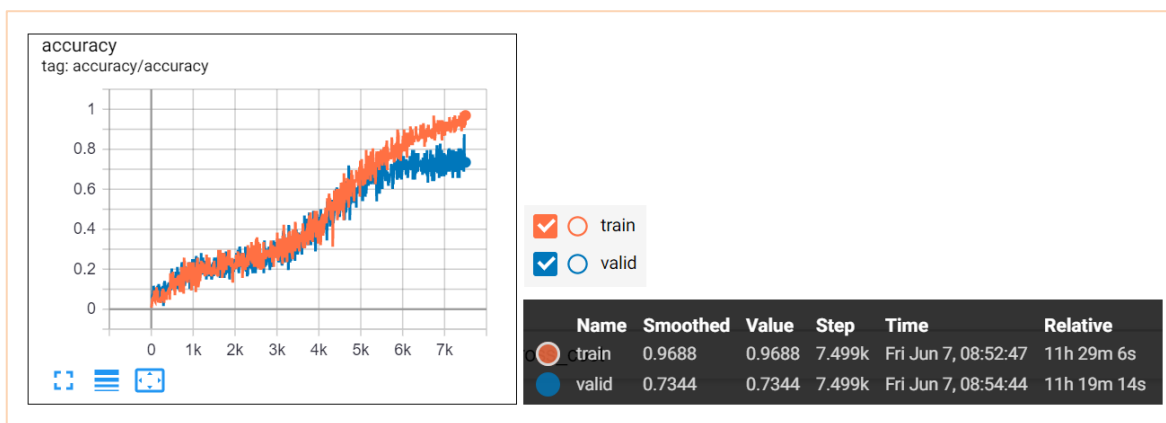
Hình 32 Độ chính xác của mạng số 4 trong quá trình nhận diện 5 địa danh



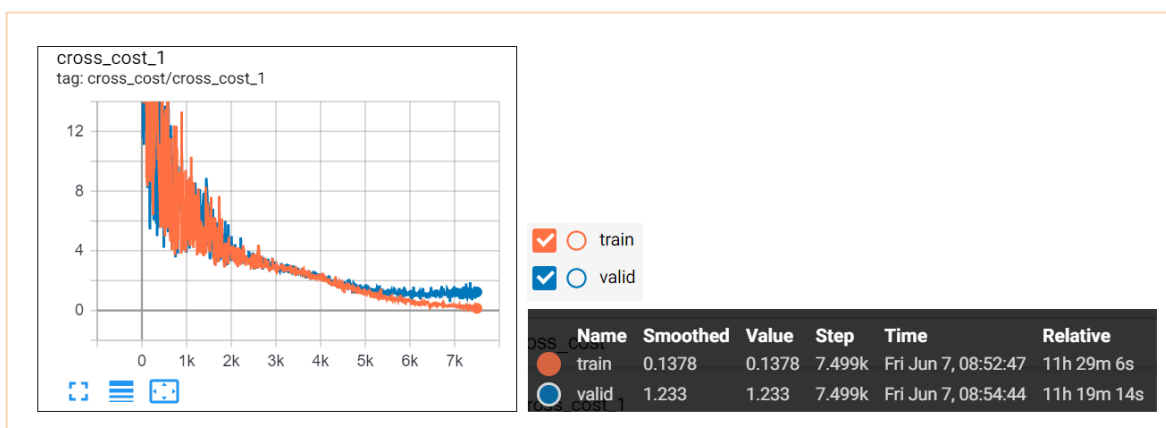
Hình 33 Giá trị lỗi của mạng số 4 trong quá trình nhận diện 5 địa danh

Số lượng ảnh và số lần huấn luyện với 5 địa danh của mạng giống với các thử nghiệm phía trên. Đặc điểm nổi bật của mạng so với các mạng tham gia thử nghiệm là thời gian huấn luyện mạng chỉ bằng 2/3 thời gian huấn luyện mạng số 1. Cùng với việc độ chính xác trong quá trình huấn luyện của mạng tăng nhanh.

Sau đây là kết quả cho việc huấn luyện mạng với đầu ra nhận diện 48 địa danh khác nhau.



Hình 34 Độ chính xác của mạng số 4 trong quá trình nhận diện 48 địa danh



Hình 35 Giá trị lỗi của mạng số 4 trong quá trình nhận diện 48 địa danh

Số lượng ảnh dùng huấn luyện mạng 1200x48 ảnh, số lượng ảnh dùng cho kiểm chứng lại mạng là 400x48 ảnh. Số lần lặp là 50 cho toàn bộ dữ liệu ảnh đầu vào. Hệ số học (learning rate) là $1e-4$.

Giá trị của độ chính xác tăng và giá trị lỗi giảm trên hai tập dữ liệu huấn luyện và tập dữ liệu kiểm thử. Tuy nhiên ở giai đoạn cuối của quá trình huấn luyện, độ chính xác trên tập huấn luyện tăng trong khi trên tập kiểm thử bị chững lại cho thấy mô hình đang có dấu hiệu bị over-fitting. Mạng cần tăng tỉ lệ dropout hoặc áp dụng nhiều kỹ thuật tránh over-fitting khác.

4.6 Tổng kết toàn bộ quá trình thử nghiệm

Bảng 8 So sánh kết quả huấn luyện mạng phân loại 5 địa danh

| Tên mạng | Huấn luyện | Kiểm thử |
|------------------------------|------------|----------|
| Mạng dựa trên mạng lenet-5 | 87% | 84% |
| Mạng dựa trên mạng alexnet | 97% | 84% |
| Mạng dựa trên mạng vgg16 | 22% | 20% |
| Mạng dựa trên mạng resnet-34 | 96% | 90% |

Quá trình thử nghiệm đối với từng mạng cùng với việc điều chỉnh tham số số học, cấu trúc mạng, sử dụng kỹ thuật tránh overfitting. Mạng có độ chính xác cao và đồ thị thể hiện quá trình huấn luyện tốt nhất là mạng dựa trên mạng resnet-34. Em chọn ra được hai mô hình có độ chính xác tốt nhất đó là mô hình xây dựng dựa trên mạng lenet-5 và mô hình dựa trên mạng resnet-34.

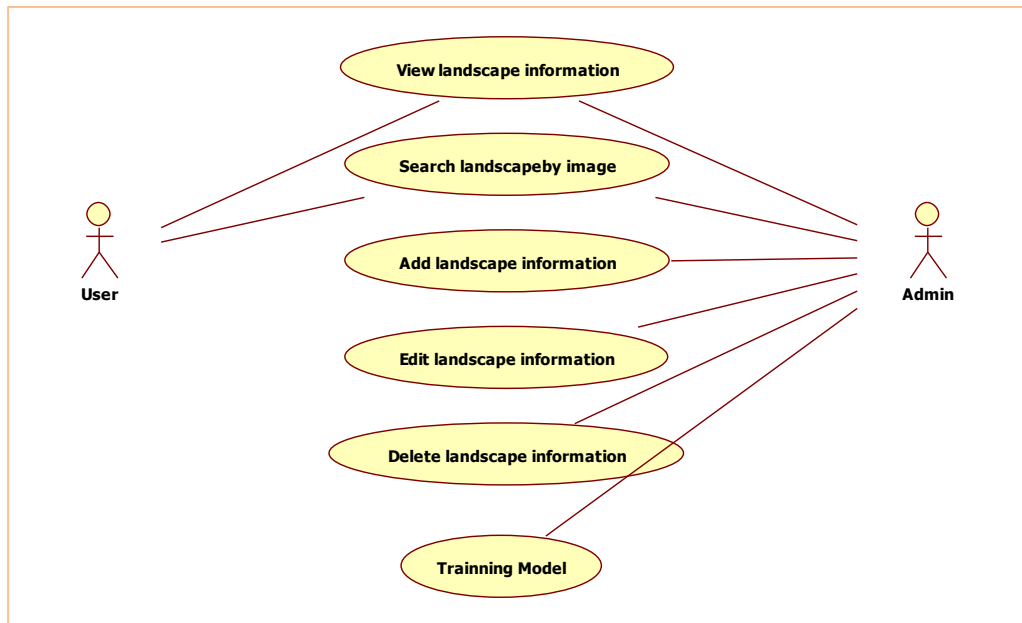
Độ chính xác của hai mô hình này có thể được cải thiện hơn nữa bằng việc tăng thời gian huấn luyện và sử dụng nhiều hơn các kỹ thuật tránh overfitting cho mô hình.

Một ưu điểm của hai mô hình trên so với hai mô hình mạng dựa trên mạng alexnet và mạng dựa trên mạng vgg-16 là thời gian huấn luyện thấp hơn. Trong khi vẫn đảm bảo được độ chính xác như mong muốn.

Trong chương sau em sẽ ứng dụng mô hình được huấn luyện vào hệ thống tìm kiếm thông tin địa danh bằng hình ảnh. Hệ thống được triển khai trên nền tảng web.

CHƯƠNG 5 – ỨNG DỤNG MÔ HÌNH SAU HUẤN LUYỆN TRONG HỆ THỐNG NHẬN DIỆN ĐỊA DANH

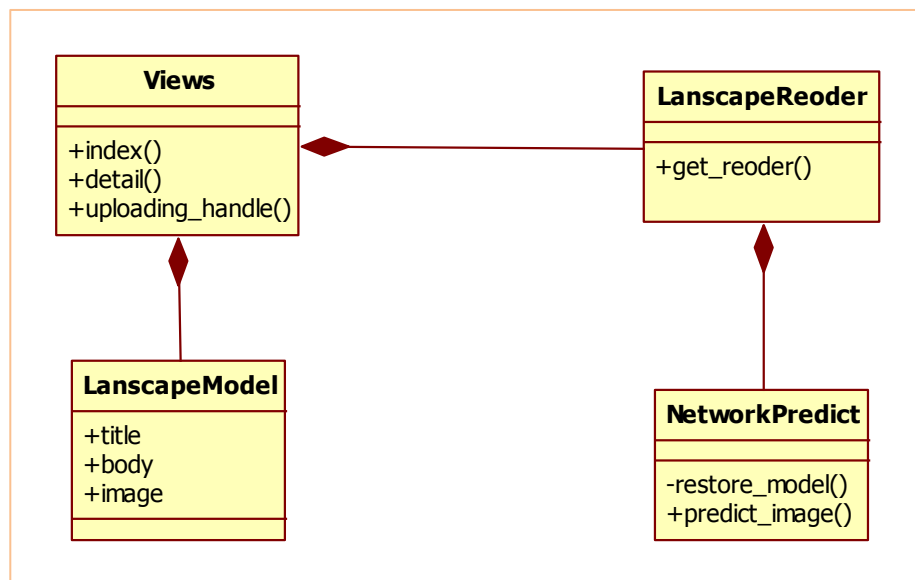
5.1 Biểu đồ ca sử dụng



Hình 36 Biểu đồ ca sử dụng

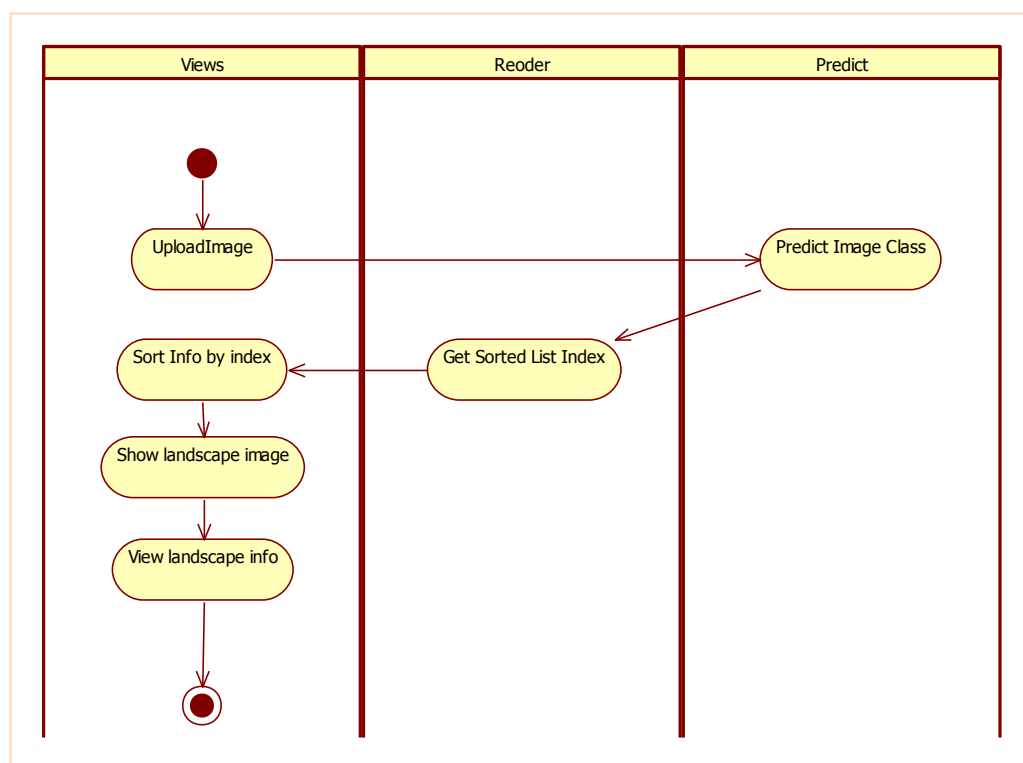
Hệ thống gồm các chức năng cơ bản giúp người dùng có thể xem thông tin và tìm kiếm địa danh bằng hình ảnh. Người quản trị có thêm các quyền liên quan tới trang quản trị như thêm, sửa, xóa thông tin địa danh.

5.2 Biểu đồ lớp



Hình 37 Biểu đồ lớp cho ca sử dụng tìm kiếm thông tin địa danh

5.3 Biểu đồ hoạt động



Hình 38 Biểu đồ hoạt động cho ca sử dụng 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 sắc xuất giống nhất lên vị trí đầu tiên.

Các chức 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.

5.4 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 sử liệu thông tin địa danh trong cơ sở dữ liệu.

Bảng 9 Bảng dữ liệu địa danh

| Tên trường | Kiểu dữ liệu | Ghi chú |
|------------|---------------|-------------|
| id | integer | primary key |
| title | varchar(1000) | |

| | | |
|-------|--------------|--|
| body | text | |
| date | datetime | |
| num | interger | |
| image | varchar(100) | |

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

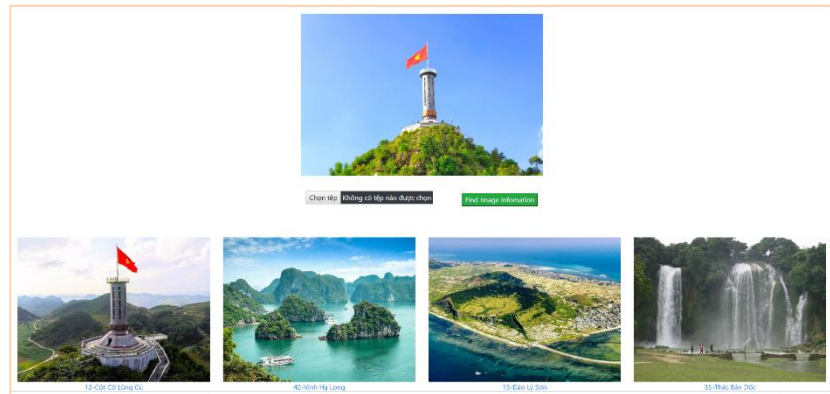
Bảng 10 Bảng thông tin người dùng

| 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) | |

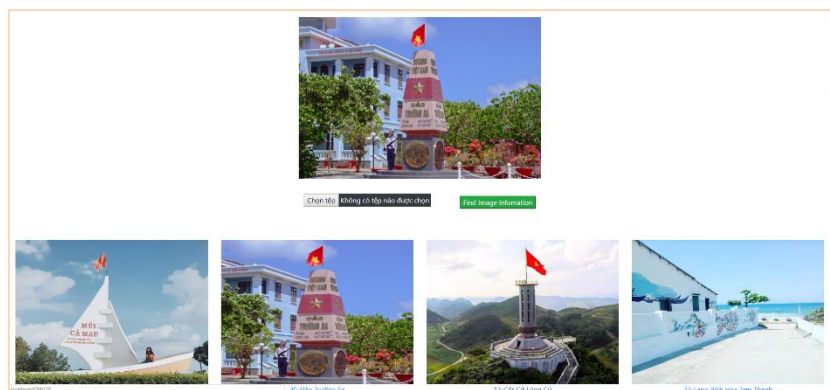
Bảng được Django hỗ trợ tạo ra trong hệ thống admin, trong đó các trường quan trọng như is_superuser, is_staff được dùng để phân biệt tài khoản admin và user thông thường. Trong đồ án này thông tin user em đang đề chỉ gồm một tài khoản admin để đăng nhập vào trang quản trị từ đó thêm, sửa xóa các thông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

5.4 Giao diện trang web

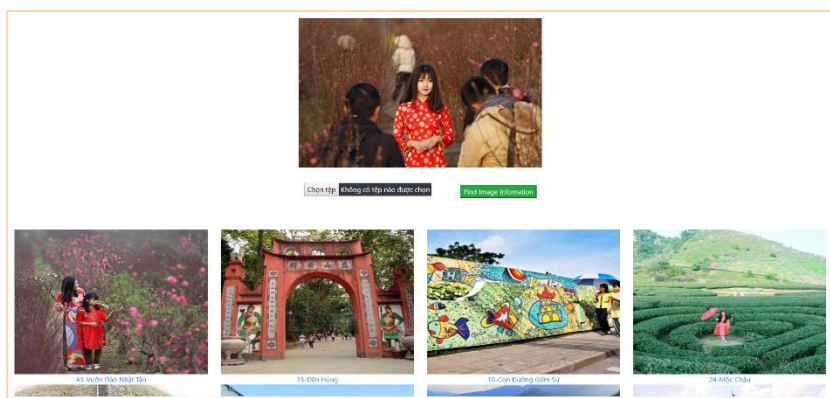
Dưới đây là kết quả xây dựng hệ thống với chức năng tìm kiếm thông tin bằng hình ảnh. Hình ảnh người dùng upload lên và hệ thống nhận diện sau đó trả về thông tin của địa danh tương ứng.



Hình 39 Tìm kiếm thông tin hình 1



Hình 40 Tìm kiếm thông tin hình 2



Hình 41 Tìm kiếm thông tin hình 3

CHƯƠNG 6 – KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1. Kết luận

Việc huấn luyện mô hình nhận dạng địa danh hiện tại cho kết quả tốt nhất là 90% với đầu ra 5 địa danh và 73% với kết quả phân loại 48 địa danh, do thời gian huấn luyện mô hình lâu cũng như giới hạn về thời gian. Kết quả phân loại 48 địa danh vẫn còn có thể cải thiện thêm được ít nhất là bằng cách tăng thời gian huấn luyện do như em đã trình bày xu hướng của đồ thị độ chính xác trên mô hình dựa trên mạng lenet-5 và resnet-34 vẫn đang theo chiều tăng. Cùng với đó độ chính xác trên tập huấn luyện và trên tập kiểm thử vẫn đang gần nhau giúp mô hình tránh được lỗi phổ biến là hiện tượng over-fitting.

Trang web ứng dụng mô hình được huấn luyện cho hoạt động ổn định. Mô hình sau khi huấn luyện cùng với trang web có thể chạy trên máy cấu hình thấp chỉ cần điều kiện là trong môi trường có django và tensorflow giúp cho việc triển khai trong thực tế thuận tiện.

Thời gian đáp ứng của trang web nhận diện hình ảnh là khoảng 2 giây, hiện nay trang web nhận diện đang hoạt động theo mô hình web tĩnh.

6.2. Hướng phát triển

Hướng phát triển sẽ nâng cao độ chính xác của việc nhận diện địa danh bằng việc tập trung tối ưu các tham số trên hai mô hình dựa trên mạng lenet-5 và resnet-34 do hai mô hình này cho thấy ưu điểm nổi trội là thời gian huấn luyện mạng ngắn hơn hai mô hình còn lại trong khi vẫn đảm bảo độ chính xác. Ngoài ra em sẽ sử dụng nhiều kỹ thuật tối ưu mạng khác như L2 regularization ... cùng với các kỹ thuật augmentation trên tập dữ liệu.

Bổ xung vào dữ liệu nhiều địa danh nổi tiếng hơn thay vì đang dùng là 48 địa danh.

Tối ưu lại trang web để chuyển trang web về dạng web động giúp cho trang web hoạt động trơn chu hơn.

TÀI LIỆU THAM KHẢO

1. Giancarlo Zaccone, Getting Started with TensorFlow, published by Packt Publishing Ltd , published 2016.
2. Yann LeCun, Léon Bottou, Yoshua Bengio and Patrick Haffner, Gradient-Based Learning Applied to Document Recognition, The IEEE, November 1998.
3. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, published 2012.
4. Karen Simonyan, Andrew Zisserman, Very Deep Convolutional Networks For Large-Scale Image Recognition, Submitted on 4 Sep 2014, Last revised 10 Apr 2015.
5. Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep Residual Learning for Image Recognition, arXiv:1512.03385v1 [cs.CV] 10 Dec 2015.
6. Get Started with TensorFlow, url: <https://www.tensorflow.org/tutorials/>, last visited day june 9.
7. Django Documentation, <https://docs.djangoproject.com/en/2.2/>, last visited day june 8.
8. Gesture Recognition for Robotic Control Using Deep Learning, url: https://www.researchgate.net/publication/320084139_Gesture_Recognition_for_Robotic_Control_Using_Deep_Learning, last visited june 9.