[Bài 6: Convolutional neural network | Deep Learning cơ bản (nttuan8.com)](https://nttuan8.com/bai-6-convolutional-neural-network/)

Convolutional Neural Networks (CNNs), là kiến trúc mạng neural nhân tạo nâng cao, được xây dựng để giải quyết các bài toán phức tạp, đặc biệt là các bài toán liên quan đến xử lý hình ảnh, chẳng hạn như nhận diện đối tượng, phân loại hình ảnh, thậm chí là tự động lái xe.

Cấu trúc cơ bản của CNNs gồm các lớp sau:

1. Lớp tích chập (Convolutional layer): Lớp này sử dụng các bộ lọc (kernels) để thực hiện các phép tích chập dựa trên ảnh đầu vào để cho ra tín hiệu mới giảm bớt các đặc trưng mà không cần thiết, giữ lại các đặc trưng chính và quan trọng. Để dễ hiểu hơn, ta có thể coi nó như là một cửa sổ trượt dưới một ma trận, và kết quả của 2 phép tích chập ma trận là một ma trận mới chứa các đặc trưng của ma trận đầu vào như ví dụ bên dưới:

Ta định nghĩa kernel là một ma trận vuông kích thước k\*k trong đó k là số lẻ. k có thể bằng 1, 3, 5, 7, 9,… Ví dụ kernel kích thước 3\*3

A number and a line

Description automatically generated with medium confidence

Với mỗi phần tử xij trong ma trận X lấy ra một ma trận có kích thước bằng kích thước của kernel W có phần tử xij làm trung tâm (đây là vì sao kích thước của kernel thường lẻ) gọi là ma trận A. Sau đó tính tổng các phần tử của phép tính element-wise của ma trận A và ma trận W, rồi viết vào ma trận kết quả Y.

A grid with numbers and symbols

Description automatically generated

Để tính giá trị của x22 ta nhân từng giá trị của ma trận A (được tô màu cam) với ma trận W, kết quả được ma trận Y. Tương tự với các giá trị khác để ra đưuọc kế quả bên dưới:

A screenshot of a grid

Description automatically generated

Vậy với các giá trị viền ngoài như x11 thì không thể tính được, vậy nên ta them một lớp bọc bên ngoài gọi là padding để giải quyết vấn đề trên.

A grid with numbers and a red square

Description automatically generated

Mục đích của phép tính convolution trên ảnh là làm mở, làm nét ảnh; xác định các đường;… Mỗi kernel khác nhau thì sẽ phép tính convolution sẽ có ý nghĩa khác nhau. Vì vậy tùy từng bài toán hoặc từng mục đích mà chúng ta chọn kernel phù hợp:

A screenshot of a computer program

Description automatically generated

Đối với các ảnh màu có tới 3 kênh là red, green, blue thì ta phải thực hiện phép tích chập bằng cách nhân một tensor 3 chiều với một kernel 3 chiều, một phần quan trọng của quá trình này là việc thêm một số gọi là "bias" sau khi thực hiện phép nhân và tổng hợp.

Bias là một giá trị số được thêm vào sau khi tích chập để tạo ra tính phi tuyến tính trong mạng neural. Nó cho phép mô hình học và biểu diễn các đặc trưng tại các vị trí không phụ thuộc vào dữ liệu đầu vào. Trong trường hợp mạng tích chập, mỗi kênh đầu ra của tích chập sẽ có một bias riêng. Chúng ta chỉ cần khởi tạo bias và giá trị bias sẽ được thực hiện tự động bởi các thư viện và framework như TensorFlow, PyTorch, Keras, …

A diagram of a program

Description automatically generated with medium confidence

Tổng quát convolution layer:

A white background with black text and red text

Description automatically generated

Output của convolutional layer sẽ qua hàm activation function trước khi trở thành input của convolutional layer tiếp theo.

Tổng số parameter của layer: Mỗi kernel có kích thước F\*F\*D và có 1 hệ số bias, nên tổng parameter của 1 kernel là F\*F\*D + 1. Mà convolutional layer áp dụng K kernel => Tổng số parameter trong layer này là K \* (F\*F\*D + 1).

1. Lớp tổng hợp (Pooling layer): Lớp tổng hợp giúp giảm kích thước khối ma trận đầu vào thông qua việc tìm ra 1 giá trị đại diện cho mỗi vùng không gian mà bộ lọc đi qua, không làm thay đổi đường nét chính của bức ảnh nhưng giảm được kích thước. Điều này giúp giảm độ phức tạp của mô hình và giúp tránh overfitting.

Gọi pooling size kích thước K\*K. Input của pooling layer có kích thước H\*W\*D, ta tách ra làm D ma trận kích thước H\*W. Với mỗi ma trận, trên vùng kích thước K\*K trên ma trận ta tìm maximum hoặc average của dữ liệu rồi viết vào ma trận kết quả.

A close-up of a grid

Description automatically generated

Có 2 loại pooling layer phổ biến là: max pooling và average pooling.

A diagram of a number

Description automatically generated

1. Lớp kích hoạt (Activation layer): Các lớp này thường được sử dụng hàm kích hoạt như ReLU (Rectified Linear Unit) để thêm tính phi tuyến tính vào mô hình
2. Lớp kết nối đầy đủ (Fully Connected Layer): Sau khi qua các lớp tích chập và tổng hợp, các đặc trưng được duyệt qua các lớp kết nối đầy đủ để thực hiện phân loại hoặc dự đoán.

A screenshot of a computer

Description automatically generated

Tổng quát quá trình trong CNNs:

A diagram of a rectangular object

Description automatically generated

Ưu điểm của CNNs

CNNs có nhiều ưu điểm quan trọng, bao gồm:

Khả năng học đặc trưng tự động: CNNs tự động học các đặc trưng quan trọng từ dữ liệu, giúp giảm bớt sự phụ thuộc vào việc tạo ra các đặc trưng thủ công.

Invariance về dịch chuyển: CNNs có khả năng nhận diện đối tượng trong ảnh bất kể vị trí của chúng, điều này có ích trong nhiều ứng dụng thực tế.

Hiệu suất cao trên dữ liệu hình ảnh: CNNs đã đạt được hiệu suất xuất sắc trong nhiều cuộc thi và ứng dụng thực tế, chẳng hạn như phát hiện khuôn mặt, phân loại hình ảnh và xe tự động lái.

Ứng dụng của CNNs

CNNs có nhiều ứng dụng quan trọng, bao gồm:

Nhận diện đối tượng: CNNs có thể được sử dụng để nhận diện đối tượng trong ảnh và video, từ nhận dạng khuôn mặt đến phát hiện xe hơi.

Phân loại hình ảnh: CNNs có thể phân loại ảnh vào các lớp khác nhau, ví dụ như phân loại loại động vật hoặc thực phẩm.

Xử lý ảnh y tế: Trong lĩnh vực y tế, CNNs đã được sử dụng để xác định bệnh từ hình ảnh chẩn đoán như tia X hoặc hình ảnh MRI.

Xe tự động lái: CNNs đóng một vai trò quan trọng trong phát triển công nghệ xe tự động lái, giúp xe ô tô tự động nhận biết và phản ứng với môi trường xung quanh.

Một số hang nổi tiếng đang sử dụng CNNs như: Apple đã sử dụng CNNs trong Face ID, công nghệ nhận dạng khuôn mặt trên các phiên bản Iphone mới nhất. Google sử dụng CNNs vào Google Photos để phân loại và tìm kiếm hình ảnh dựa trên nội dung, và cũng đang đưuọc áp dụng vào Waymo, công ty con của Alphabet để phát triển công nghệ xe tự động lái. Facebook cũng sử dụng CNNs để nhận dạng khuôn mặt và phân loại hình ảnh trên nền tảng của họ. NVIDIA công ty chuyên về đồ họa cũng sử dụng CNNs cho mục đích xử lý hình ảnh và thời gian thực bao gồm các giải pháp tự động lái xe.

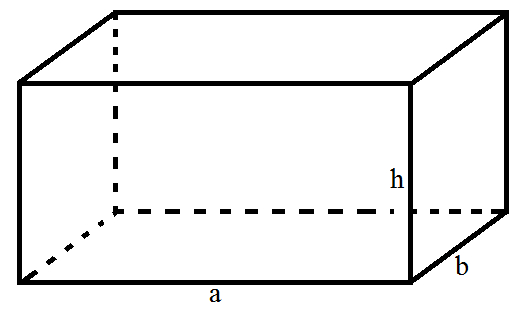
Tensor là gì

Khi dữ liệu biểu diễn dạng 1 chiều, người ta gọi là vector, mặc định khi viết vector sẽ viết dưới dạng cột.

Khi dữ liệu dạng 2 chiều, người ta gọi là ma trận, kích thước là số hàng \* số cột.

Khi dữ liệu nhiều hơn 2 nhiều thì sẽ được gọi là tensor, ví dụ như dữ liệu có 3 chiều.

Để ý thì thấy là ma trận là sự kết hợp của các vector cùng kích thước. Xếp n vector kích thước m cạnh nhau thì sẽ được ma trận m\*n. Thì tensor 3 chiều cũng là sự kết hợp của các ma trận cùng kích thước, xếp k ma trận kích thước m\*n lên nhau sẽ được tensor kích thước m\*n\*k.



AUTOENCODER

Autoencoders là một loại mạng neural sâu được sử dụng trong học không giám sát để học biểu diễn hiệu quả của dữ liệu. Autoencoders thường được sử dụng

trong nhiều tác vụ như phân tích phân loại, nén dữ liệu và tái tạo hình ảnh. Chúng hoạt động bằng cách ánh xạ dữ liệu từ không gian đầu vào sang không gian ẩn và sau đó tái tạo lại dữ liệu từ không gian ẩn

Một autoencoder cơ bản bao gồm hai phần chính:

Encoder (Bộ mã hóa): Phần này ánh xạ dữ liệu đầu vào vào không gian ẩn (latent space). Nó thường bao gồm một hoặc nhiều lớp neural để giảm chiều dữ liệu và biểu diễn nó trong không gian ẩn. Quá trình này tương tự như việc nén dữ liệu.

Decoder (Bộ giải mã): Phần này ánh xạ từ không gian ẩn trở lại không gian đầu vào. Nó có nhiệm vụ tái tạo dữ liệu từ biểu diễn ẩn. Decoder thường chứa các lớp neural để phục hồi dữ liệu ban đầu từ không gian ẩn.

Autoencoder cố gắng học biểu diễn sao cho đầu ra của quá trình giải mã gần giống với dữ liệu đầu vào ban đầu. Điều này thúc đẩy mô hình học cách biểu diễn dữ liệu một cách hiệu quả bằng cách tìm ra các đặc trưng quan trọng và loại bỏ nhiễu.

Ứng dụng của autoencoders bao gồm:

Nén dữ liệu: Autoencoders có thể được sử dụng để nén dữ liệu mà vẫn giữ lại thông tin quan trọng. Ví dụ, trong nén hình ảnh để giảm dung lượng lưu trữ.

Phát hiện nhiễu và ngoại lai: Autoencoders có khả năng phát hiện nhiễu hoặc các điểm dữ liệu ngoại lai bằng cách so sánh đầu ra của mô hình với dữ liệu ban đầu.

Trích xuất đặc trưng: Autoencoders có thể được sử dụng để trích xuất các đặc trưng quan trọng từ dữ liệu, chẳng hạn như trong việc phân loại hình ảnh hoặc âm thanh.

Autoencoders là một phần quan trọng của Deep Learning và có nhiều biến thể như Variational Autoencoders (VAEs) và Denoising Autoencoders (DAEs), mở ra nhiều cơ hội cho nghiên cứu và ứng dụng trong nhiều lĩnh vực khác nhau.