Ghi chép bài giảng: Stanford CS231n

Lecture 5: Image Classification with CNNs

Ho Hong Phuc Nguyen

10 tháng 2, 2026

Mục lục

1 Recap: Quy trình xây dựng hệ thống Deep Learning

Thầy Justin bắt đầu bằng việc nhìn lại "công thức"(recipe) mà chúng ta đã xây dựng trong 4 bài giảng đầu. Quy trình này là một chuỗi các vấn đề và giải pháp nối tiếp nhau:

1. **Bước 1: Định nghĩa bài toán (Problem Formulation)**
   - *Vấn đề:* Làm thế nào để máy tính hiểu được dữ liệu?
   - *Giải pháp:* Chúng ta mô hình hóa bài toán dưới dạng các **Tensors** (các mảng số đa chiều).
   - Đầu vào (Input) là một tensor (ví dụ: ảnh RGB). Đầu ra (Output) cũng là một tensor (ví dụ: vector điểm số cho các lớp).
2. **Bước 2: Xây dựng mô hình (The Model)**
   - *Vấn đề:* Cần một hàm số để ánh xạ từ Input sang Output.
   - *Giải pháp:* Bắt đầu với **Linear Classifier** (Bộ phân loại tuyến tính) $f(x, W) = Wx$. Nó đơn giản là một phép nhân ma trận trọng số W với dữ liệu ảnh x.
3. **Bước 3: Đánh giá mô hình (Loss Function)**
   - *Vấn đề:* Làm sao biết ma trận W chúng ta chọn là tốt hay xấu?
   - *Giải pháp:* Định nghĩa **Hàm mất mát (Loss Function)** như Softmax hoặc SVM Loss. Hàm này cho ta một con số cụ thể thể hiện độ "tệ"của mô hình với bộ dữ liệu hiện tại.
4. **Bước 4: Tìm kiếm giải pháp tốt nhất (Optimization)**
   - *Vấn đề:* Chúng ta có hàm Loss, nhưng làm sao tìm được W để Loss nhỏ nhất?
   - *Giải pháp:* Sử dụng các thuật toán **Tối ưu hóa**. Hãy tưởng tượng không gian trọng số như một địa hình đồi núi (optimization landscape), ta cần "trượt xuống"thung lũng nơi có Loss thấp nhất.
   - Các công cụ: SGD, Momentum, RMSProp, và **Adam**.
   - *Bên lề:* Thầy có nhắc đến việc thuật toán Adam vừa nhận giải thưởng "Test of Time Award"tại ICLR 2025 (vì bài báo gốc ra đời năm 2015 và vẫn cực kỳ quan trọng sau 10 năm).
5. **Bước 5: Nhận diện hạn chế (The Limitation)**
   - *Vấn đề:* Linear Classifier quá yếu.
   - *Lý do:* Nó chỉ học được **một khuôn mẫu (template) duy nhất** cho mỗi lớp. Ví dụ: Lớp "ô tô"chỉ là một bóng mờ màu đỏ. Nếu xe ô tô màu xanh hoặc nằm ở vị trí khác, mô hình sẽ thất bại. Nó không xử lý được sự biến dạng hình học hay màu sắc phức tạp.
6. **Bước 6: Nâng cấp mô hình (Neural Networks)**
   - *Giải pháp:* Để khắc phục sự đơn điệu của Linear Classifier, ta chuyển sang **Neural Networks**.
   - Thay vì một ma trận W, ta chồng (stack) nhiều ma trận lên nhau và xen kẽ bằng các **Hàm kích hoạt phi tuyến (Non-linearities)** như ReLU. Điều này giúp mô hình mạnh hơn rất nhiều.
7. **Bước 7: Cơ chế tính toán (Backpropagation & Computational Graphs)**
   - *Vấn đề:* Mô hình giờ đây quá phức tạp, làm sao tính đạo hàm để tối ưu hóa?
   - *Giải pháp:* Sử dụng **Đồ thị tính toán**.
   - Đây là một "công thức toàn năng"(powerful recipe): Chúng ta chia nhỏ mạng thành các cổng (gate) cục bộ. Mỗi cổng chỉ cần biết cách tính đầu ra và tính đạo hàm cục bộ (local gradient). Thuật toán **Backpropagation** sẽ lo việc truyền gradient từ cuối mạng về đầu mạng để cập nhật trọng số.

Kết luận phần Recap: Đến đây, chúng ta đã có đủ công cụ: Tensor input/output, Loss function, Optimizer, và Backprop. Tuy nhiên, kiến trúc chúng ta đang dùng (Fully Connected Layer) phá vỡ cấu trúc không gian của ảnh (do duỗi phẳng ảnh thành vector). → *Đó là lý do chúng ta cần chuyển sang Convolutional Neural Networks (CNNs) trong bài hôm nay.*

2 Feature Extraction (Trích xuất đặc trưng)

Trước khi Deep Learning bùng nổ (giai đoạn trước 2012), người ta không đưa trực tiếp ảnh thô (Raw pixels) vào bộ phân loại tuyến tính.

2.1 Phương pháp truyền thống (Traditional Feature Extraction)

Quy trình cũ: **Image** → **Feature Extraction** → **Linear Classifier**.

- **Đặc điểm:** Phần trích xuất đặc trưng được thiết kế thủ công bởi con người (Hand-designed), cố định và không được học (not learnable). Chỉ có bộ phân loại ở cuối mới được huấn luyện.
- **Các ví dụ:**
  - **Color Histogram:** Đếm tần suất xuất hiện của các màu sắc (bỏ qua thông tin không gian).
  - **HOG (Histogram of Oriented Gradients):** Trích xuất thông tin về cạnh, hướng của đường nét (bỏ qua thông tin màu sắc).
  - **Bag of Words:** Kết hợp nhiều đặc trưng lại với nhau.

2.2 Phương pháp Deep Learning (ConvNets)

Quy trình mới: **Image** → **ConvNet Layers** → **Classifier**.

- **Sự khác biệt cốt lõi:** Thay vì dùng đặc trưng do người thiết kế, CNN tự học các đặc trưng (learned features) từ dữ liệu thô thông qua quá trình Backpropagation.
- **Ưu điểm:** Máy tính (với dữ liệu lớn) thường tìm ra các đặc trưng tốt hơn con người tự nghĩ ra. Toàn bộ hệ thống được huấn luyện **End-to-End** (từ đầu đến cuối).

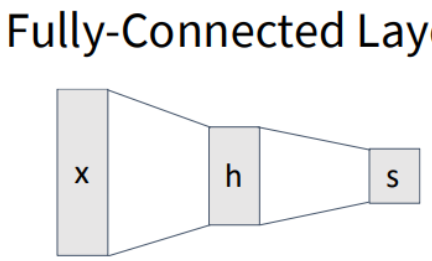
3 Convolutional Neural Networks (CNNs)

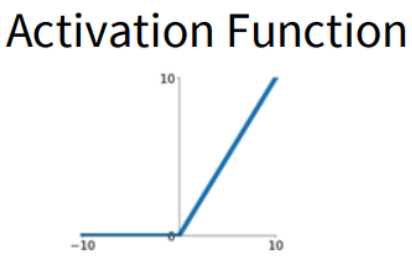
3.1 Cấu trúc tổng quát

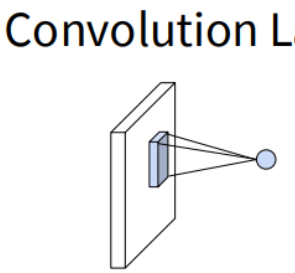
Một mạng CNN thông thường là chuỗi các lớp xếp chồng lên nhau:

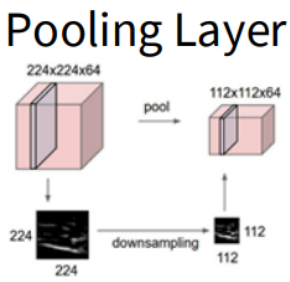
$$\text{CNN} = \text{Convolution} + \text{Activation (ReLU)} + \text{Pooling} + \text{Fully Connected (MLP)} \tag{1}$$

- **Convolutional Layer:** Trích xuất đặc trưng không gian.
- **Pooling Layer:** Giảm kích thước dữ liệu (Downsampling).
- **Fully Connected Layer:** Phân loại dựa trên đặc trưng đã trích xuất (thường ở cuối mạng).









4 Convolutional Layer (Lớp Tích chập)

4.1 Cơ chế hoạt động

Khác với Fully Connected Layer (kéo giãn ảnh thành vector 1D làm mất cấu trúc không gian), Convolutional Layer **bảo toàn cấu trúc 3D** của ảnh ($Height \times Width \times Channel$).

- **Input:** Một tensor 3D kích thước $H \times W \times C$.
- **Filter (Kernel):** Các bộ lọc nhỏ có kích thước $K \times K \times C$. *Lưu ý: Độ sâu (Depth) của Filter luôn bằng độ sâu của Input.*
- **Thao tác:** Trượt Filter qua từng vị trí không gian của Input, tính tích vô hướng (Dot Product) giữa Filter và vùng ảnh cục bộ + Bias.
- **Output (Activation Map):** Mỗi Filter tạo ra một bản đồ kích hoạt 2D (Activation Map). Nếu dùng N filters, ta sẽ có Output là chồng của N bản đồ này.

4.2 Các thành phần và công thức toán học

Giả sử:

- Input size: W (giả sử ảnh vuông cho đơn giản).
- Filter size (Kernel size): K.
- Padding (Lề): P.
- Stride (Bước nhảy): S.

Công thức tính kích thước đầu ra (Output Size):

$$\text{Output Size} = \frac{W - K + 2P}{S} + 1 \tag{2}$$

4.2.1 Padding (Lề)

Vấn đề: Khi thực hiện tích chập, kích thước ảnh sẽ bị thu nhỏ dần qua các lớp (ví dụ: Input 32, Filter 5 $\rightarrow$ Output 28). Sau nhiều lớp, ảnh sẽ biến mất. **Giải pháp (Zero Padding):** Thêm các viền giá trị 0 xung quanh ảnh Input.

- Giúp bảo toàn kích thước không gian (Spatial Size).

- Công thức phổ biến: Để Output bằng Input (với $S = 1$), ta chọn $P = \frac{K-1}{2}$.
- Ví dụ: $K = 3 \rightarrow P = 1$; $K = 5 \rightarrow P = 2$.

4.2.2 Stride (Bước nhảy)

Là khoảng cách trượt của Filter.

- $S = 1$: Trượt từng pixel (độ phân giải cao).
- $S = 2, 3...$: Nhảy cóc (giảm kích thước Output - một dạng Downsampling).

4.3 Số lượng tham số (Parameters)

Với một lớp Conv có N filters, kích thước $K \times K \times C_{in}$:

$$\text{Total Parameters} = (\underbrace{K \times K \times C_{in}}_{\text{Weights}} + \underbrace{1}_{\text{Bias}}) \times N \tag{3}$$

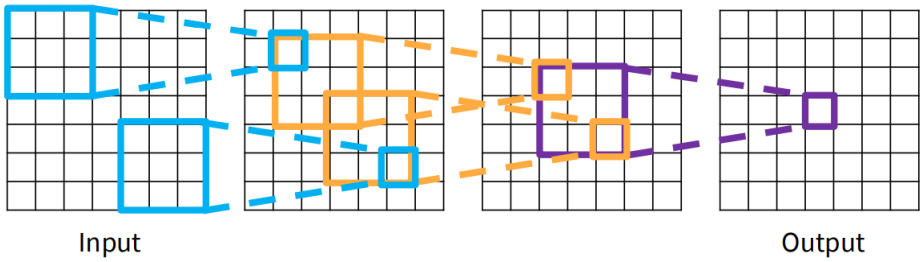
5 Receptive Field (Vùng cảm nhận)

Đây là khái niệm quan trọng giải thích tại sao mạng sâu lại hiệu quả.

- **Định nghĩa:** Receptive Field là vùng không gian trên ảnh gốc (Input ban đầu) mà một nơ-ron ở lớp hiện tại có thể "nhìn thấy"(chịu ảnh hưởng).
- **Quy luật:** Các lớp sâu hơn (Deeper layers) sẽ có Receptive Field lớn hơn.
  - Lớp Conv 1: Nhìn thấy 3x3 pixel.
  - Lớp Conv 2 (trên Conv 1): Một nơ-ron nhìn thấy 3x3 của Conv 1, tức là nhìn thấy 5x5 của ảnh gốc.
  - Lớp Conv 3: Nhìn thấy 7x7 của ảnh gốc...

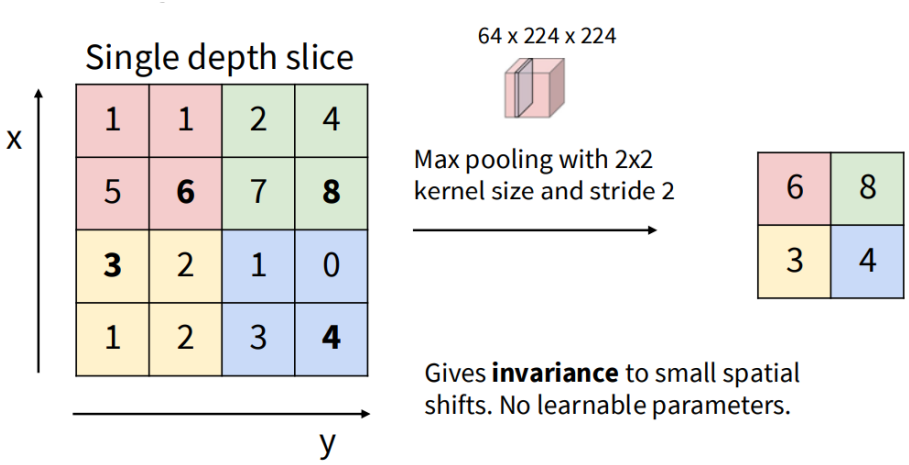
Vấn đề với ảnh lớn (High Resolution Images): Nếu ảnh đầu vào rất lớn (ví dụ 4K), để nơ-ron cuối cùng nhìn thấy toàn bộ ảnh (Global Context), ta cần một Receptive Field cực lớn.

- Nếu chỉ dùng Conv 3×3 với Stride 1, ta cần hàng nghìn lớp mới bao phủ hết ảnh $\rightarrow$ Tốn kém tính toán, bộ nhớ.
- **Giải pháp:** Cần giảm kích thước không gian nhanh chóng (**Downsampling**) để tăng Receptive Field theo cấp số nhân.



6 Pooling Layer (Lớp Gộp)

Pooling là giải pháp chính cho việc Downsampling.



6.1 Chức năng

- Giảm kích thước không gian (W, H) để giảm lượng tham số và tính toán.
- Tăng nhanh Receptive Field.
- Tạo ra tính bất biến với các dịch chuyển nhỏ (Translation Invariance).
- **Lưu ý:** Pooling hoạt động độc lập trên từng kênh (Channel/Depth slice). Input depth = Output depth.

6.2 Các loại Pooling

- **Max Pooling (Phổ biến nhất):** Chọn giá trị lớn nhất trong vùng filter (ví dụ 2×2).
- **Average Pooling:** Tính trung bình cộng (ít dùng hơn trong các mạng hiện đại).

6.3 Tổng kết sự khác biệt về trích xuất đặc trưng

Đặc điểm	**Chi tiết**
Tầng thấp (Low-level)	Học các đặc trưng đơn giản: Cạnh (Edges), Màu sắc (Colors).
Tầng giữa (Mid-level)	Học các hình dạng cơ bản: Góc, đốm tròn, kết cấu.
Tầng cao (High-level)	Học các vật thể phức tạp: Mắt, bánh xe, mặt người (nhờ Receptive Field lớn).

Bảng 1: Sự phân cấp đặc trưng trong CNN