HCMUS
Viet Nam National University
Ho Chi Minh City
University of Science

Khoa Toán - Tin học
Fac. of Math. & Computer Science

CHƯƠNG 4

Neural Networks

Lưu Giang Nam

Bộ môn Ứng dụng Tin học
Khoa Toán - Tin học
Trường Đại học KHTN, ĐHQG TPHCM

09/2025

Bối cảnh và vấn đề

- Trong Chương 3–4: xem xét các mô hình hồi quy và phân loại dựa trên tổ hợp tuyến tính của các hàm cơ sở cố định.
- Ưu điểm: giải tích tốt, tính toán thuận lợi.
- Hạn chế: chịu ảnh hưởng của lời nguyền chiều.
- Mạng nơ-ron truyền thẳng (Feed-forward NN)
  - Cố định số lượng hàm cơ sở, nhưng cho phép chúng thích nghi thông qua các tham số huấn luyện.
  - Mô hình thành công nhất: Feed-forward neural network hay Multilayer Perceptron (MLP).
  - Thực chất MLP là chuỗi các mô hình logistic liên tục, không phải các perceptron rời rạc.
  - Ưu điểm: mô hình gọn hơn, tính toán nhanh hơn SVM khi có cùng năng lực tổng quát hóa.

Phần 1

Feed-forward Network

Mô hình tuyến tính cơ sở

- Bắt đầu từ mô hình tuyến tính hồi quy hoặc phân loại:

$$y(x, w) = f\left(\sum_{j=1}^{M} w_j \phi_j(x)\right) \tag{1}$$

- Trong đó:
  - $f(\cdot)$ là hàm kích hoạt (activation function).
  - $\phi_j(x)$ là các hàm cơ sở cố định
- Mục tiêu: mở rộng mô hình bằng cách cho phép $\phi_j(x)$ phụ thuộc tham số, huấn luyện cùng w_j
- Neural Network: mỗi $\phi_j(x)$ là một hàm phi tuyến của tổ hợp tuyến tính các đầu vào (inputs) với trọng số (weights) và độ lệch (bias).

Hàm kích hoạt

- Ở mỗi tầng ẩn (hidden unit) ta định nghĩa a_j:

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}, \quad j = 1, \ldots, M \tag{2}$$

- Từ dạng linear ta chuyển qua dạng phi tuyến qua thông qua hàm kích hoạt (khả vi, phi tuyến) ta được hàm:

$$z_j = h(a_j) \tag{3}$$

- Các hàm kích hoạt phổ biến:
  - Logistic sigmoid: $\sigma(a) = \frac{1}{1+\exp(-a)}$
  - Tanh: $h(a) = \tanh(a)$
- Chức năng:
  - Tạo phi tuyến giữa input và output
  - Đảm bảo khả vi

Lớp đầu ra

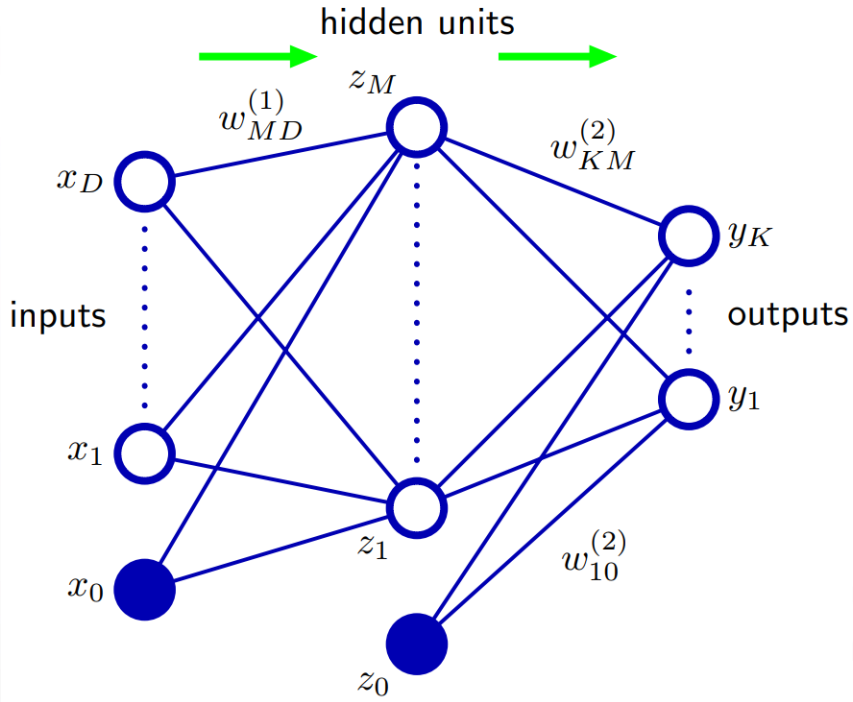
- Tầng đầu ra (output) với giá trị kích hoạt đầu ra ở tầng thứ 2:

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)}, \quad k = 1, \ldots, K$$

- Cuối cùng, các kích hoạt đầu ra được biến đổi bằng cách sử dụng hàm kích hoạt thích hợp để đưa ra một tập hợp các đầu ra y_k.
  - Regression: identity, $y_k = f(a_k) = a_k$
  - Binary classification: sigmoid, $y_k = f(a_k) = \sigma(a_k)$
  - Multiclass: softmax, $y_k = f(a_k) = \text{softmax}(a_k)$
- Lan truyền thuận: tuần tự từ input $\rightarrow$ hidden $\rightarrow$ output

$$y_k(x, w) = f\left(\sum_{j=0}^{M} w_{kj}^{(2)} h\left(\sum_{i=0}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right) \quad (4)$$

Sơ đồ mạng cho mạng nơ-ron hai lớp



Hình: Các biến đầu vào, biến ẩn và biến đầu ra được biểu diễn bằng các nút, và các trọng số được biểu diễn bằng các liên kết giữa các nút, trong đó các tham số độ lệch được biểu thị bằng các liên kết đến từ các biến đầu vào và biến ẩn bổ sung x_0 và z_0. Các mũi tên biểu thị hướng của luồng thông tin qua mạng trong quá trình lan truyền thuận (forward propagation).

Bài tập 1

Câu hỏi

Với điều kiện $x_0 = 1$, hãy biểu diễn a_j và $y_k(x, w)$ theo dạng sau:

$$a_j = \sum_{i=0}^{D} w_{ji}^{(1)} x_i, \quad y_k(x, w) = \sigma\left(\sum_{j=0}^{M} w_{kj}^{(2)} h\left(\sum_{i=0}^{D} w_{ji}^{(1)} x_i\right)\right) \tag{5}$$

- Lợi ích:
  - Viết gọn công thức
  - Chuẩn hóa forward propagation

MLP và kiến trúc mạng

- Hai tầng = Multilayer Perceptron (MLP)
- Hidden units liên tục, khả vi $\rightarrow$ hỗ trợ gradient-based training
- Linear hidden units $\rightarrow$ khả năng xấp xỉ hạn chế
- Nonlinear hidden units: $\rightarrow$ xấp xỉ mọi hàm liên tục trên miền compact. Số lượng hidden units quyết định độ chính xác
- Mỗi hidden unit: tính tuyến tính $\rightarrow$ phi tuyến $\rightarrow$ chuyển sang tầng đầu ra.
- Feed-forward: không có cycles, đảm bảo output xác định
- Mạng có thể mở rộng:
  - Thêm layers
  - Skip connections (input $\rightarrow$ output)
  - Sparse connections

Vai trò hidden units

- Hidden units trích xuất đặc trưng phi tuyến
- Hợp tác để xấp xỉ hàm mục tiêu
- Cấu trúc hai tầng + nonlinear hidden units $\rightarrow$ xấp xỉ nhiều hàm phức tạp
- Thách thức: tìm tập tham số w tối ưu từ dữ liệu huấn luyện
- Giải pháp:
  - Maximum likelihood
  - Bayesian approach
- Mở rộng: Skip-layer connections, sparse connections
- Có thể thêm layers $\rightarrow$ deep network nhưng luôn tuân thủ feed-forward $\rightarrow$ output xác định
- Mỗi unit:

$$z_k = h\left(\sum_j w_{kj} z_j\right) \tag{6}$$

Phần 2

Huấn luyện mô hình

Giới thiệu

- Neural network là một lớp hàm phi tuyến tham số từ input vector $x \rightarrow$ output vector y
- Mục tiêu: tìm các tham số w sao cho mạng mô hình hóa tốt dữ liệu huấn luyện
- Phương pháp đơn giản: Cực tiểu hóa tổng bình phương lỗi:

$$E(w) = \frac{1}{2} \sum_{n=1}^{N} \|y(x_n, w) - t_n\|^2$$

- Nhưng cách tiếp cận tổng quát hơn là dùng góc nhìn xác suất cho outputs. Làm rõ ràng hơn cho cả việc lựa chọn tính phi tuyến tính của output unit và lựa chọn hàm lỗi.

Probabilistic Interpretation - Regression

- Giả sử một target $t \in \mathbb{R}$ có phân phối Gaussian với

$$p(t|x, w) = \mathcal{N}(t|y(x, w), \beta^{-1}) \tag{7}$$

- $\beta =$ đô chính xác (precision hay inverse variance) của nhiễu Gauss.
- Đối với phân phối có điều kiện được cho bởi (7), chỉ cần lấy hàm kích hoạt đơn vị đầu ra là đồng nhất thức, vì một mạng như vậy có thể xấp xỉ bất kỳ hàm liên tục nào từ x đến y.
- Cho một tập dữ liệu gồm N quan sát độc lập, phân phối đồng nhất $X = \{x_1, ..., x_N\}$, cùng với các giá trị mục tiêu tương ứng $t = \{t_1, ..., t_N\}$ ta có thể xây dựng được likelihood cho toàn bộ dataset:

$$p(t|X, w, \beta) = \prod_{n=1}^{N} p(t_n|x_n, w, \beta) \tag{8}$$

Bài tập 2:

Câu hỏi

1. Chứng minh hàm sai số (error function) được cho bằng hàm đối của log-likelikelihood là (với w và β là các tham số):

$$\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n, w) - t_n\}^2 - \frac{N}{2}\ln\beta + \frac{N}{2}\ln(2\pi) \qquad (9)$$

2. Sau khi tìm được w_{ML}, giá trị của β có thể được tìm thấy bằng cách tương tự.

$$\frac{1}{\beta_{ML}} = \frac{1}{N}\sum_{n=1}^{N}\{y(x_n, w_{ML}) - t_n\}^2 \qquad (10)$$

Nhiễn biến mục tiêu

- Giả sử K biến mục tiêu (independent conditionally) trên x và w, khi đó ta xét:

$$p(t|x,w) = \mathcal{N}(t|y(x,w), \beta^{-1}I) \tag{11}$$

- Khi đó để tối đại likelihood ta sẽ tối tiểu $E(w)$:

$$E(w) = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}\{y_k(x_n,w) - t_{nk}\}^2 \tag{12}$$

với

$$\frac{1}{\beta_{ML}} = \frac{1}{NK}\sum_{n=1}^{N}\sum_{k=1}^{K}\{y_k(x_n,w_{ML}) - t_{nk}\}^2 \tag{13}$$

Với hàm đơn nhất

- Nếu Output activation = identity, tức là $y_k = a_k$
- Ta sẽ có:

$$\frac{\partial E}{\partial a_k} = y_k - t_k \tag{14}$$

- Tương thích tự nhiên giữa output activation và error function
- Max likelihood $w \leftrightarrow$ minimize $E(w)$:

$$E(w) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, w) - t_n\}^2 \tag{15}$$

- Nonlinear network $\rightarrow$ E(w) nonconvex $\rightarrow$ local minima

Với phân loại nhị phân

- Với hàm mục tiêu đơn biến t: lớp C_1 gán $t = 1$, lớp C_2 gán $t = 0$.
- Sử dụng kích hoạt cho hàm đầu ra là hàm logistic sigmoid:

$$y = \sigma(a) = \frac{1}{1 + \exp(-a)} \tag{16}$$

- Khi đó ta có $0 \leq y(x, w) \leq 1$ và $y(x, w) = p(C_1|x)$, $p(C_2|x) = 1 - y(x, w)$.
- Khi đó phân phối của điều kiện của biến mục tiêu sẽ tuân theo phân phối Bernoulli

$$p(t|x, w) = y(x, w)^t \{1 - y(x, w)\}^{1-t} \tag{17}$$

Cross-entropy

- Khi đó, sử dụng "negative log-likelihood" ta sẽ được một hàm lỗi được gọi là "cross-entropy":

$$E(w) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\} \qquad (18)$$

- Không có noise precision β
- Ưu điểm: huấn luyện nhanh hơn, tổng quát hóa tốt hơn.

Cross-entropy

Đối với bài toán đa biến (K biến) ta đơn giản là tích (K) hàm sigmoid lại:

$$p(\mathbf{t}|\mathbf{x},\mathbf{w}) = \prod_{k=1}^{K} y_k^{t_k}(\mathbf{x},\mathbf{w})(1 - y_k(\mathbf{x},\mathbf{w}))^{1-t_k} \qquad (19)$$

Sử dụng "negative log-likelihood" để chứng minh hàm lỗi $E(\mathbf{w})$ sẽ có dạng:

$$E(\mathbf{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K}\{t_{nk} \ln y_{nk} + (1 - t_{nk}) \ln(1 - y_{nk})\} \qquad (20)$$

với $y_{nk} = y_K(\mathbf{x}_n, \mathbf{w})$

Phần 2

Huấn luyện mô hình

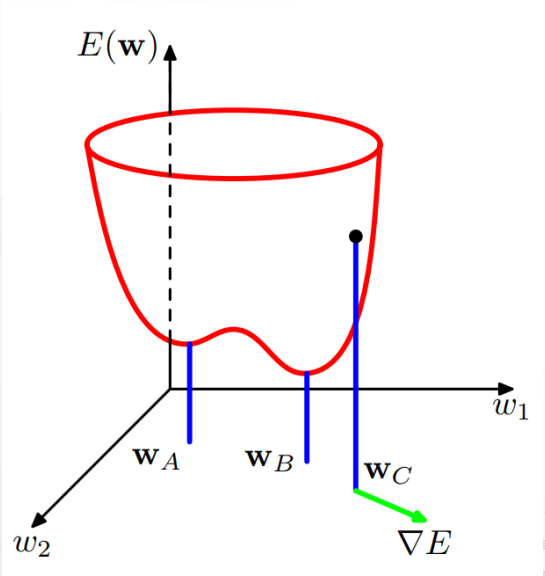
Mục 1: Tối ưu tham số

Giới thiệu

- Mục tiêu: tìm vector trọng số w để **minimize** hàm lỗi $E(w)$.
- Phương pháp được sử dụng ở đây sẽ là cập nhập các trọng số để hội tụ về giá trị cực trị:

$$w \to w + \delta w, \quad \delta E \simeq \delta w^T \nabla E(w) \tag{21}$$

với vector $\nabla E(w)$ chỉ hướng tăng nhanh nhất của $E(w)$.



Hình: Điểm w_A là một điểm cực tiểu cục bộ và w_B là điểm cực tiểu toàn cục. Điểm w_C và vectơ $\nabla E(w)$ ứng với điểm này.

Điều kiện cực trị của hàm lỗi

- Các điểm mà $\nabla E(w) = 0$ gọi là **stationary points**.
- Phân loại:
- Trong mạng nơ-ron, $E(w)$ thường **phi tuyến mạnh** $\rightarrow$ có nhiều điểm dừng.
- $E(w)$ có thể có nhiều cực tiểu không tương đương:

$$E(w_{\text{local}}) > E(w_{\text{global}}) \tag{22}$$

- Trong thực tế, ta không cần tìm chính xác global minimum $\rightarrow$ tìm nghiệm tốt (local minimum "đủ tốt") có khả năng tổng quát hóa tốt.

Thuật toán tối ưu lặp

- Trong giải tích, việc tìm nghiệm chính xác của phương trình $\nabla E(w) = 0$ là không thể $\rightarrow$ dùng **thuật toán lặp**.
- Công thức cập nhật tổng quát:

$$w^{(\tau+1)} = w^{(\tau)} + \Delta w^{(\tau)} \tag{23}$$

- Sau mỗi bước, tính lại gradient tại $w^{(\tau+1)}$.
- Các thuật toán khác nhau ở cách chọn $\Delta w^{(\tau)}$.

Phần 2

Huấn luyện mô hình

Mục 2: Xấp xỉ bậc hai cục bộ

Local Quadratic Approximation

- Gần một điểm w_b, khai triển Taylor bậc hai:

$$E(w) \simeq E(w_b) + (w - w_b)^T b + \frac{1}{2}(w - w_b)^T H(w - w_b) \quad (24)$$

với:

$$b = \nabla E|_{w=w_b}, \quad H = \nabla\nabla E|_{w=w_b} = \frac{\partial E}{\partial w_i \partial w_j}|_{w=w_b}$$

- Đây là xấp xỉ bậc hai của $E(w)$ quanh w_b. Khi đó gradient tương ứng:

$$\nabla E \simeq b + H(w - w_b) \quad (25)$$

- Nếu w_b là điểm cực tiểu $\rightarrow b = 0$:

$$E(w) = E(w_b) + \frac{1}{2}(w - w_b)^T H(w - w_b) \quad (26)$$

Giá trị riêng và hình dạng bề mặt lỗi

- Giải phương trình riêng của ma trận Hessian H:

$$Hu_i = \lambda_i u_i, \quad u_i^T u_j = \delta_{ij} \tag{27}$$

- Bây giờ chúng ta viết lại $(w - w_b)$ thành một tổ hợp tuyến tính của các vectơ riêng dưới dạng: $w - w_b = \sum_i \alpha_i u_i$.

Câu hỏi

Thay vào chứng minh $E(w)$ có dạng:

$$E(w) = E(w_b) + \frac{1}{2} \sum_i \lambda_i \alpha_i^2 \tag{28}$$

Câu hỏi

Khi đó các $\lambda_i > 0$ ta sẽ có ma trận Hessian H là **positive definite**. Chứng minh w_b là cực tiểu.

Phần 2

Huấn luyện mô hình

Mục 3: Tối ưu gradient descent

Batch Version

- Quy tắc cập nhật cơ bản:

$$w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E(w^{(\tau)})$$

- $\eta > 0$: learning rate (bước nhảy).
- Mỗi bước cần duyệt qua toàn bộ training set.
- Gọi là batch gradient descent.
- Nhược điểm của Gradient Descent: Dễ dao động khi hàm lỗi có độ cong khác nhau. Dễ mắc kẹt trong local minima (hoặc plateau).
- Cần chọn η phù hợp:
  - η quá nhỏ $\rightarrow$ hội tụ chậm.
  - η quá lớn $\rightarrow$ dao động, không hội tụ.
- Các phương pháp hiệu quả hơn: **Conjugate Gradient**, **Quasi-Newton**.

On-line (Stochastic) Gradient Descent[1]

- Hàm lỗi toàn bộ:

$$E(w) = \sum_{n=1}^{N} E_n(w) \tag{29}$$

- Cập nhật theo từng mẫu:

$$w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E_n(w^{(\tau)}) \tag{30}$$

- Mỗi bước chỉ dùng một điểm dữ liệu.
- Còn gọi là **stochastic gradient descent (SGD)**.
- Ưu điểm của On-line Learning
  1. Xử lý tập dữ liệu lớn hiệu quả hơn.
  2. Có khả năng thoát khỏi local minima do nhiễu trong cập nhật.
  3. Có thể cập nhật liên tục khi có dữ liệu mới.

[1] Le Cun, et al. (1989). Backpropagation applied to handwritten zip code recognition. Neural Computation 1(4), 541–551.

Good luck!