KHOA CNTT & TRUYỀN THÔNG
BM KHOA HỌC MÁY TÍNH

MẠNG NƠ-RON NHÂN TẠO

ARTIFICIAL NEURAL NETWORK

Giới thiệu

Mạng nơ-ron nhân tạo (Artificial neural network - ANN)

- Mô hình hoá hoạt động của hệ thần kinh con người
- Được nghiên cứu lần đầu vào năm 1943 (McCulloch và Pitts, 1943)
- Perceptron: thế hệ đầu tiên của mạng nơ-ron (Rosenblatt, 1958)
  - Mô phỏng quá trình hoạt động của thị giác con người

Giới thiệu

Lịch sử

- 1943, McCulloch & Pitts đưa ra mô hình nơ-ron đầu tiên
- 1982, Mô hình mạng nơ-ron hồi tiếp của Hopfield
- 1984, Mạng nơ-ron Kohonen hay còn gọi là Bản đồ tự tổ chức (SOM)
- 1985, Mạng nơ-ron đa tầng (MLP)

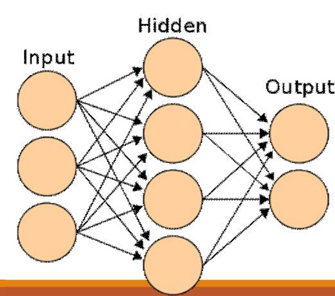
Mô hình mạng nơ-ron khác
– Mạng nơ-ron tích chập (Convolutional NN) của Yan LeCun.

3

Giới thiệu

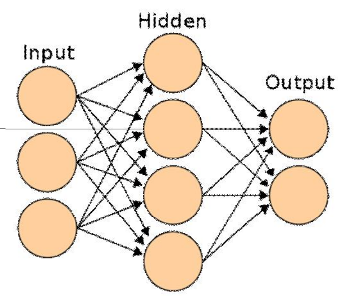
Mạng nơ-ron nhân tạo hay thường gọi ngắn gọn là **mạng nơ-ron** là một mô hình toán học hay mô hình tính toán được xây dựng dựa trên các mạng nơ-ron sinh học. Nó gồm có một nhóm các nơ-ron nhân tạo (nút) nối với nhau, và xử lý thông tin bằng cách truyền theo các kết nối và tính giá trị mới tại các nút (cách tiếp cận connectionism đối với tính toán).

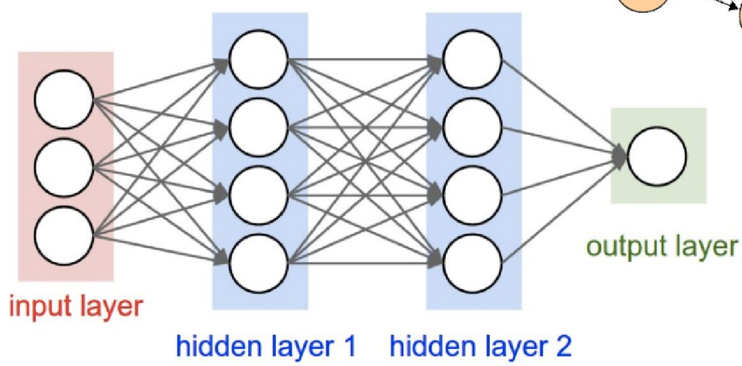
Trong nhiều trường hợp, mạng nơ-ron nhân tạo là một hệ thống thích ứng (*adaptive system*) tự thay đổi cấu trúc của mình dựa trên các thông tin bên ngoài hay bên trong chảy qua mạng trong quá trình học.



4

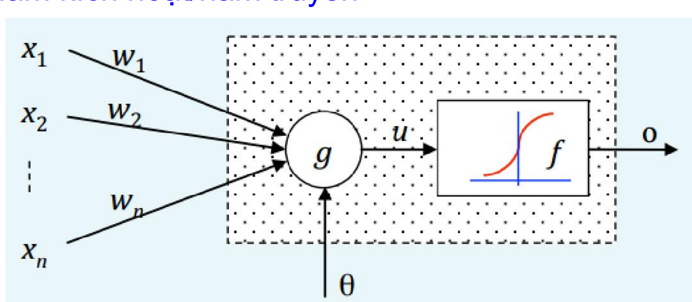
Giới thiệu





Mô hình nơ-ron McCulloch & Pitts

Nơ-ron: đơn vị tính toán cơ bản/đơn vị của tất cả các mạng nơ-ron:
– n đầu vào, 1 tham số, 1 đầu ra
– Hàm mạng/hàm kết hợp
– Hàm kích hoạt/hàm truyền



w_i: trọng số

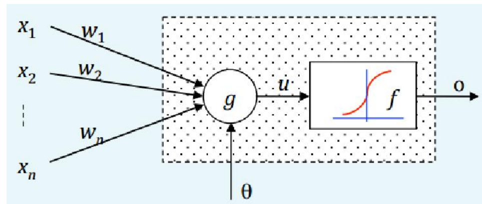
θ: ngưỡng (threshold), độ lệch (bias)

Mô hình nơ-ron McCulloch & Pitts

Nơ-ron: đơn vị tính toán cơ bản/đơn vị của tất cả các mạng nơ-ron:
– n đầu vào, 1 tham số, 1 đầu ra
– Hàm mạng/hàm kết hợp

$$u = g(x) = \theta + \sum_{i=1}^{n} w_i x_i$$

– Hàm kích hoạt/hàm truyền

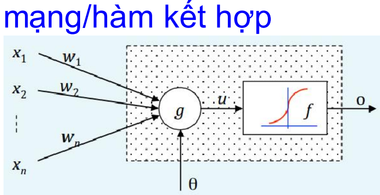


w_i: trọng số

θ: ngưỡng (threshold), độ lệch (bias)

Mô hình nơ-ron McCulloch & Pitts

Nơ-ron: đơn vị tính toán cơ bản/đơn vị của tất cả các mạng nơ-ron:
– n đầu vào, 1 tham số, 1 đầu ra
– Hàm mạng/hàm kết hợp



$$u = g(x) = \theta + \sum_{i=1}^{n} w_i x_i$$

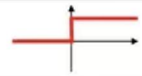
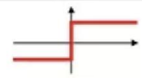
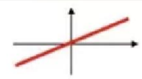
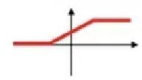
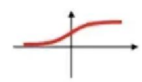
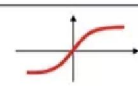
– Hàm kích hoạt/hàm truyền

Đầu ra

$$o = f(u) = f(g(x))$$

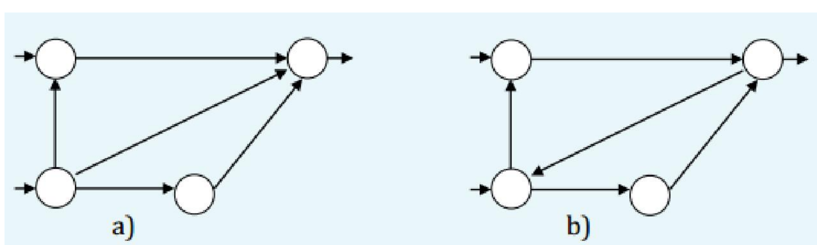
$$f(u) = \frac{1}{1 + e^{-u}} \quad f'(u) = f(u)[1 - f(u)]$$

Hàm kích hoạt/ hàm truyền

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$	Multi-layer NN	

Kiến trúc mạng

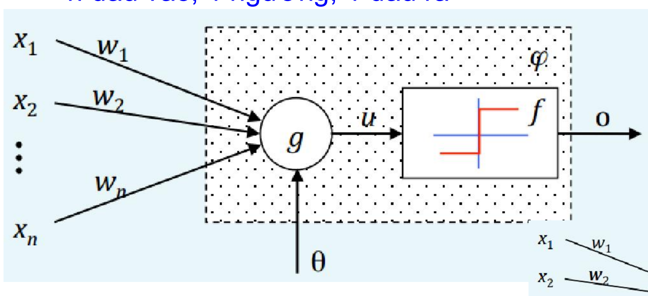
- Mạng truyền thẳng (forward)
- Mạng hồi tiếp (feedback)

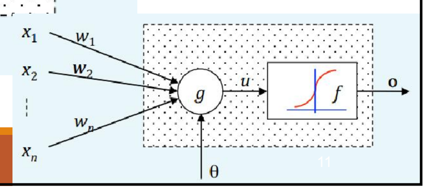


Mô hình perceptron

Mô hình perceptron

- Do Rosenblatt đề xuất năm 1958
- Tương tự như mô hình nơ-ron của McCulloch&Pitts
- Perceptron tuyến tính có ngưỡng
  - n đầu vào, 1 ngưỡng, 1 đầu ra



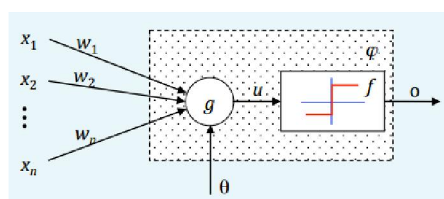


Mô hình perceptron

Perceptron tuyến tính có ngưỡng

- n đầu vào, **1 ngưỡng**, 1 đầu ra
- Hàm mạng tuyến tính

$$u = g(x) = \theta + \sum_{i=1}^{n} w_i x_i$$



- Hàm kích hoạt là hàm ngưỡng

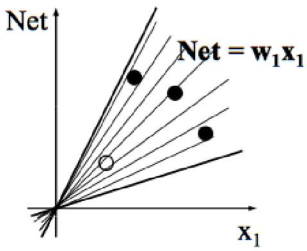
$$o = f(u) = f(g(x)) = \begin{cases} 1 & g(x) \geq 0 \\ 0 & g(x) < 0 \end{cases}$$

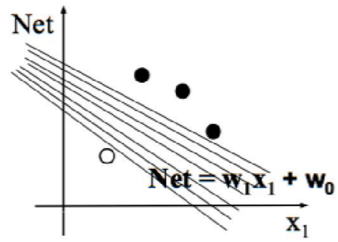
$$o = f(u) = f(g(x)) = \begin{cases} 1 & g(x) \geq 0 \\ -1 & g(x) < 0 \end{cases}$$

12

Ý nghĩa của giá trị theta

$$u = g(x) = \theta + \sum_{i=1}^{n} w_i x_i$$



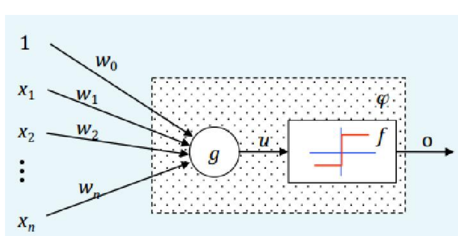


Mô hình perceptron

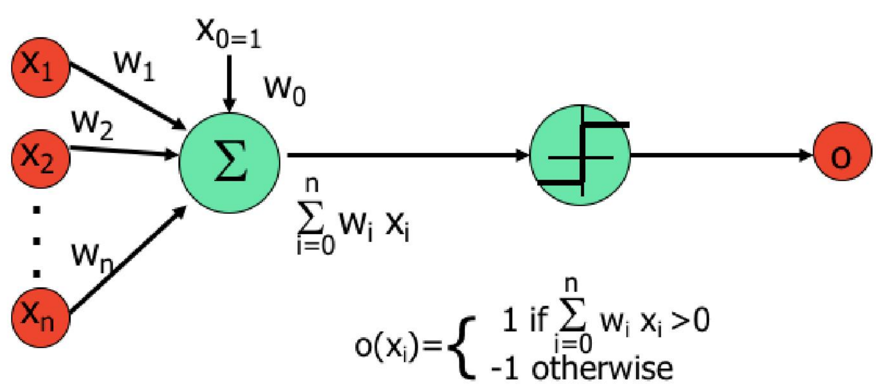
Perceptron tuyến tính không ngưỡng

- $n+1$ đầu vào, 1 đầu ra
- Đầu vào giả x_0 luôn có giá trị 1, $w_0 = \theta$
- Hàm mạng tuyến tính

$$u = g(x) = \sum_{i=0}^{n} w_i x_i$$



Mô hình perceptron



15

Huấn luyện perceptron

- Huấn luyện - Dạy cho perceptron
  - **Tìm kiếm n tham số: $w_0, w_1, w_2, \ldots, w_n$** sao cho đầu ra của nơ-ron phù hợp với giá trị mong muốn của tất cả dữ liệu học nhất.
- **Dữ liệu đầu vào**:
  - Tập các mẫu huấn luyện
  - Mỗi mẫu huấn luyện gồm: véc-tơ đặc trưng x và nhãn y.
- **Tham số:**
  - Tốc độ học: η (đọc là eta)
- **Về mặt hình học**:
  - Tìm siêu phẳng tách dữ liệu thành 2 lớp sao cho mỗi lớp về 1 phía của siêu phẳng này.

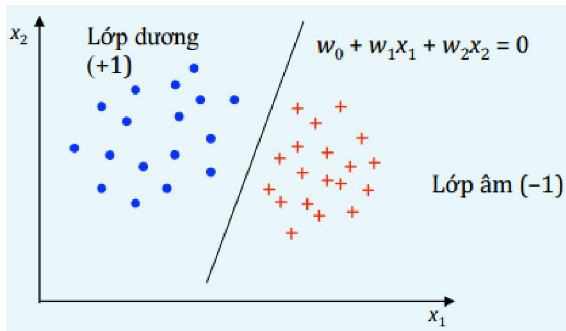
16

Mô hình perceptron

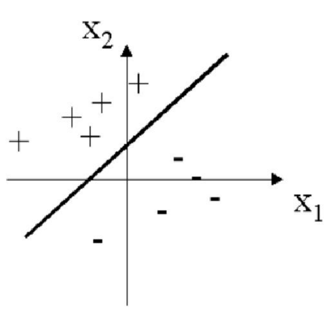
Ý nghĩa hình học

- Phương trình u = g(x) = 0 là phương trình của 1 siêu phẳng trong không gian n chiều.

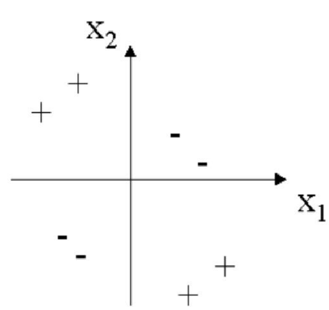
$$u = g(x) = \sum_{i=0}^{n} w_i x_i = 0$$



Dữ liệu khả tách tuyến tính?



Linearly Separable



Not Linearly Separable

Huấn luyện perceptron

Trường hợp dữ liệu khả tách tuyến tính

- Khởi tạo ngẫu nhiên các w
- Đưa từng mẫu học qua perceptron và quan sát giá trị đầu ra
- Nếu giá trị đầu ra khác với giá trị mong muốn, cập nhật lại các trọng số theo công thức:

$$w_j = w_j + \eta \cdot (y_i - o_i) \cdot x_{ij}, \forall j = 0..n$$

- Nếu giá trị **output bằng giá trị mong muốn:** trọng số không thay đổi do y-o =0
- Nếu giá trị **output nhỏ hơn giá trị mong muốn**: các trọng số sẽ được tăng một lượng tỉ lệ thuận với thành phần x_j của vector đặc trưng đang xét
- Nếu giá trị **output lớn hơn giá trị mong muốn:** các trọng số sẽ giảm đi một lượng tỉ lệ thuận với đầu vào

Bài tập

x1	x2	Y
0	0	0
0	1	0
1	0	0
1	1	1

Thiết kế perceptron cho dữ liệu trong bảng

Với các trọng số w0 = -0.2, w1 =0.5, w2=0.5

Tốc độ học: eta = 0.15

Bài tập

x1	x2	Y
0	0	0
0	1	0
1	0	0
1	1	1

Đầu vào: x1, x2
Đầu ra: y – có 2 giá trị 0-1,
sử dụng hàm kích hoạt ngưỡng {0,1}
Hàm mạng

$$\varphi(x_i) = g(x_i) = \sum_{j=0}^{n} w_j . x_{ij}$$

Hàm kích hoạt hay hàm ngưỡng

$$o = f(u) = f(g(x)) = \begin{cases} 1 & g(x) \geq 0 \\ 0 & g(x) < 0 \end{cases}$$

Thiết kế perceptron cho dữ liệu trong bảng
Với các trọng số w0 = -0.2, w1 =0.5, w2=0.5
Tốc độ học: eta = 0.15

Bài tập

x1	x2	Y
0	0	0
0	1	0
1	0	0
1	1	1

Đầu vào: x1, x2
Đầu ra: y – có 2 giá trị 0-1,
sử dụng hàm kích hoạt ngưỡng {0,1}

Xác định dữ liệu có khả tách tuyến tính?
Cập nhật lại các giá trị w khi giá trị đầu ra khác với giá trị mong muốn

$$w_j = w_j + \eta \cdot (y_i - o_i) \cdot x_{ij}, \forall j = 0..n$$

Bài tập

x1	x2	Y
0	0	0
0	1	0
1	0	0
1	1	1

Lần lặp 1:
Xáo trộn thứ tự tập huấn luyện: 1,3,4,2
Xét phần tử 1 có vector đặc trưng (0,0) – y1=0
Hàm mạng $\varphi(x_i) = g(x_i) = \sum_{j=0}^{n} w_j . x_{ij}$

$U_1 = w_0 + w_1 * x_1 + w_2 * x_2 = -0.2 + 0.5*0 + 0.5*0 = -0.2$
Do $U_1 < 0 \Rightarrow o_1 = 0$
Giống với đầu ra thực tế => giữ nguyên

Bài tập

	x1	x2	Y
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Lần lặp 1:
Xáo trộn thứ tự tập huấn luyện: 1,3,4,2

Xét phần tử 3 có vector đặc trưng (1,0) – y3=0
Hàm mạng $\varphi(x_i) = g(x_i) = \sum_{j=0}^{n} w_j . x_{ij}$

$U_3 = w_0 + w_1 * x_1 + w_2 * x_2 = -0.2 + 0.5*1 + 0.5*0 = 0.3$
Do $U_3 > 0 \Rightarrow o_3 = 1$
Khác với đầu ra thực tế ($y_3=0$)=> cập nhật lại trọng số
$w_0 = w_0 + 0.15 * (0-1)*1 = -0.2 - 0.15 = -0.35$
$w_1 = w_1 + 0.15 * (0-1)*1 = 0.5 - 0.15 = 0.35$
$w_2 = w_2 + 0.15 * (0-1)*0 = 0.5 + 0 = 0.5$

$$w_j = w_j + \eta \cdot (y_i - o_i) \cdot x_{ij}, \forall j = 0..n$$

Bài tập

x1	x2	Y
0	0	0
0	1	0
1	0	0
1	1	1

Lần lặp 1:
Xáo trộn thứ tự tập huấn luyện: 1,3,4,2

Xét phần tử 4 có vector đặc trưng (1,1) – y_4=1
Hàm mạng $\varphi(x_i) = g(x_i) = \sum_{j=0}^{n} w_j.x_{ij}$

$U_4 = w_0 + w_1 * x_1 + w_2 * x_2$ = -0.35 + 0.35*1 + 0.5*1 = 0.5
Do $U_4 > 0 \Rightarrow o_4 = 1$, giữ nguyên trọng số

$$w_j = w_j + \eta \cdot (y_i - o_i) \cdot x_{ij}, \forall j = 0..n$$

25

Bài tập

x1	x2	Y
0	0	0
0	1	0
1	0	0
1	1	1

Lần lặp 1:
Xáo trộn thứ tự tập huấn luyện: 1,3,4,2

Xét phần tử 2 có vector đặc trưng (0,1) – y2=0
Hàm mạng $\varphi(x_i) = g(x_i) = \sum_{j=0}^{n} w_j.x_{ij}$

$U_2 = w_0 + w_1 * x_1 + w_2 * x_2$ = 0.15
Do $U_2 > 0 \Rightarrow o_2 = 1$
Khác với đầu ra thực tế (y2=0)=> cập nhật lại trọng số
$w_0 = w_0$ + 0.15 * (0-1)*1=-0.35 - 0.15 = -0.5
$w_1 = w_1$ + 0.15 * (0-1)*0= 0.35
$w_2 = w_2$ + 0.15 * (0-1)*1= 0.35

$$w_j = w_j + \eta \cdot (y_i - o_i) \cdot x_{ij}, \forall j = 0..n$$

26

Bài tập

Lần lặp 2:
Xáo trộn thứ tự tập huấn luyện: 3,1,2,4
----

x1	x2	Y
0	0	0
0	1	0
1	0	0
1	1	1

$$w_j = w_j + \eta \cdot (y_i - o_i) \cdot x_{ij}, \forall j = 0..n$$

Bài tập

Lần lặp 2:
Xáo trộn thứ tự tập huấn luyện: 3,1,2,4

Lỗi trên tập học E = 0
w_0 = -0.5
w_1 = 0.35
w_2 = 0.35

x1	x2	Y
0	0	0
0	1	0
1	0	0
1	1	1

$$w_j = w_j + \eta \cdot (y_i - o_i) \cdot x_{ij}, \forall j = 0..n$$

Huấn luyện perceptron

$$E(w) \equiv E(w_0, w_1, ..., w_n) = \frac{1}{2}\sum_{i=1}^{m}\left(y^{(i)} - g(x^{(i)})\right)^2$$

Trường hợp dữ liệu không khả tách tuyến tính

- Cố gắng tìm một siêu phẳng "tốt" nhất
  - Tốt = lỗi (trên tập học) nhỏ nhất có thể
- Định nghĩa hàm lỗi E(w) theo các trọng số w trên tất cả các phần tử của tập học:

$$E(w) \equiv E(w_0, w_1, ..., w_n) = \frac{1}{2}\sum_{i=1}^{m}\left(y_i - \varphi(x_i)\right)^2$$

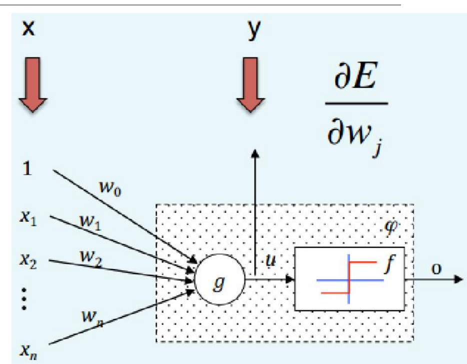
- **Bài toán huấn luyện trở thành tìm w sao cho E(w) nhỏ nhất**

Huấn luyện perceptron

Chú ý:

- Hàm kích hoạt trong trường hợp này được thay bằng hàm tuyến tính f(u) = u hay:

$$\varphi(x_i) = g(x_i) = \sum_{j=0}^{n} w_j . x_{ij}$$



Ta tối ưu E(w) bằng phương pháp gradient descent

Huấn luyện perceptron

Giải thuật:

- Khởi động ngẫu nhiên w
- Cập nhật lại w sao cho E càng lúc càng nhỏ (lặp cho đến khi điều kiện dừng thoả mãn)
  - **w = w –ηΔE(w)**

    = w + Δw (Với Δw = –ηΔE(w))

$$\nabla E(w) = \begin{pmatrix} \frac{\partial E}{\partial w_0} \\ \frac{\partial E}{\partial w_1} \\ \vdots \\ \frac{\partial E}{\partial w_n} \end{pmatrix}$$

Gradient của E(w):

$$\frac{\partial E}{\partial w_j} = -\sum_{i=1}^{m} \left(y_i - g(x_i) \right) . x_{ij}$$

Chương 2. Mạng nơ-ron nhân tạo *Đỗ Thanh Nghị - Phạm Nguyên Khang*

$$\frac{\partial E}{\partial w_j} = \frac{\partial}{\partial w_j}\left(\frac{1}{2}\sum_{i=1}^{m} (y_i - \varphi(x_i))^2 \right)$$

$$\frac{\partial E}{\partial w_j} = \frac{1}{2}\sum_{i=1}^{m} \left\{ \frac{\partial}{\partial w_j} (y_i - \varphi(x_i))^2 \right\}$$

$$\frac{\partial E}{\partial w_j} = \frac{1}{2}\sum_{i=1}^{m} \left\{ 2(y_i - \varphi(x_i)) \cdot \frac{\partial}{\partial w_j} (y_i - \varphi(x_i)) \right\}$$

$$\frac{\partial E}{\partial w_j} = \frac{1}{2}\sum_{i=1}^{m} \left\{ 2(y_i - \varphi(x_i)) \cdot \frac{\partial}{\partial w_j} (-\varphi(x_i)) \right\}$$

$$\frac{\partial E}{\partial w_j} = \sum_{i=1}^{m} \left\{ (y_i - \varphi(x_i)) \cdot \frac{\partial}{\partial w_j} (\varphi(x_i)) \right\}$$

$$\frac{\partial E}{\partial w_j} = \sum_{i=1}^{m} \left\{ (y_i - \varphi(x_i)) \cdot \frac{\partial}{\partial w_j} (f(g(x_i))) \right\}$$

$$\frac{\partial E}{\partial w_j} = -\sum_{i=1}^{m} \left\{ (y_i - \varphi(x_i)) \cdot \frac{\partial f(g(x_i))}{\partial g(x_i)} \cdot \frac{\partial g(x_i)}{\partial w_j} \right\}$$

$$\frac{\partial E}{\partial w_j} = -\sum_{i=1}^{m} \left\{ (y_i - g(x_i)) \cdot f'(g(x_i)) \cdot \frac{\partial}{\partial w_j} \left(\sum_{k=0}^{n} w_k \cdot x_{ik} \right) \right\} \quad (2.14$$

$$\frac{\partial E}{\partial w_j} = -\sum_{i=1}^{m} \left\{ (y_i - g(x_i)) \cdot f'(g(x_i)) \cdot x_{ij} \right\}$$

ong đó, f' là đạo hàm cấp 1 của f theo g.

Huấn luyện perceptron

$$\frac{\partial E}{\partial w_j} = -\sum_{i=1}^{m} \{(y_i - g(x_i)) \cdot f'(g(x_i)) \cdot x_{ij}\}$$

trong đó, f' là đạo hàm cấp 1 của f theo g.

Từ đây ta có thể suy ra:

$$\Delta w_j = \eta \cdot \sum_{i=1}^{m} \{(y_i - g(x_i)) \cdot f'(x_i) \cdot x_{ij}\} \quad (2.15)$$

Vì hàm kích hoạt f là hàm tuyến tính[1] $f(u) = u$, ta có $f'(u) = df/du = 1$. Lượng cập nhật cho trọng số w_j là:

$$\Delta w_j = \eta \cdot \sum_{i=1}^{m} \{(y_i - g(x_i)) \cdot x_{ij}\} \quad (2.16)$$

Huấn luyện perceptron

Luật cập nhật w:

- Luật gradient chuẩn: cập nhật sau khi xem xét tất cả các phần tử của tập học, cộng dồn Δw
- Luật Delta: cập nhật sau khi xét mỗi phần tử của tập học

Huấn luyện perceptron

Luật cập nhật w:

- Luật gradient chuẩn: cập nhật sau khi xem xét tất cả các phần tử của tập học, cộng dồn Δw

Đầu vào:

Tập mẫu huấn luyện $X = \{(x_i, y_i)\}_{i=1,m}$ với $x_i \in R^n$ và $y_i \in \{1, -1\}$

Tốc độ học: η

Dung sai: ε (lỗi lớn nhất có thể chấp nhận được)

Giải thuật:

Khởi tạo ngẫu nhiên các trọng số w_i, $\forall i = 0..n$
repeat
    $E = 0$
    for $j = 0$ **to** n **do**
        $\Delta w_j = 0$
    end for
    Xáo trộn ngẫu nhiên tập mẫu huấn luyện X
    for $i = 1$ **to** m **do**
        $E = E + (y_i - \varphi(x_i))^2$
        // Tích lũy lượng cần cập nhật Δw_j cho w_j
        for $j = 0$ **to** n **do**
            $\Delta w_j = \Delta w_j + \eta \,.\, (y_i - \varphi(x_i)) \,.\, x_i$
        end for
    end for
    // Cập nhật trọng số w_j
    for $j = 0$ **to** n **do**
        $w_j = w_j + \Delta w_j$
    end for
until $E < \varepsilon$

Huấn luyện perceptron

Luật cập nhật w:

- Luật Delta: cập nhật sau khi xét mỗi phần tử của tập học

Đầu vào:

Tập mẫu huấn luyện $X = \{(x_i, y_i)\}_{i=1,m}$ với $x_i \in R^n$ và $y_i \in \{1, -1\}$

Tốc độ học: η

Dung sai: ε (lỗi lớn nhất có thể chấp nhận được)

Giải thuật:

Khởi tạo ngẫu nhiên các trọng số w_i, $\forall i = 0..n$
repeat
    $E = 0$
    Xáo trộn ngẫu nhiên tập mẫu huấn luyện X
    for $i = 1$ **to** m **do**
        $E = E + (y_i - \varphi(x_i))^2$
        // Cập nhật w_j
        for $j = 0$ **to** n **do**
            $w_j = w_j + \eta \,.\, (y_i - \varphi(x_i)) \,.\, x_i$
        end for
    end for
until $E < \varepsilon$

Bài tập

x1	x2	Y
0	0	-1
0	1	1
1	0	+1
1	1	1

Cho perceptron với hàm kích hoạt

$$o = f(u) = f(g(x)) = \begin{cases} 1 & g(x) \geq 0 \\ -1 & g(x) < 0 \end{cases}$$

Thiết kế perceptron cho dữ liệu trong bảng
Với các trọng số khởi tạo w_0 = -0.5, w_1 =0.4, w_2=0.5
Tốc độ học: eta = 0.2

Bài tập

Thiết kế perceptron cho dữ liệu trong bảng
Với các trọng số khởi tạo w_0 = -0.5, w_1 =0.4, w_2=0.5
Tốc độ học: eta = 0.2

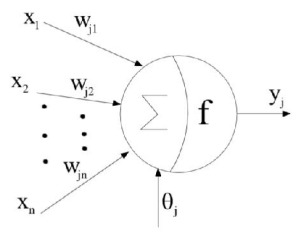
x1	x2	Y
0	0	-1
0	1	1
1	0	+1
1	1	1

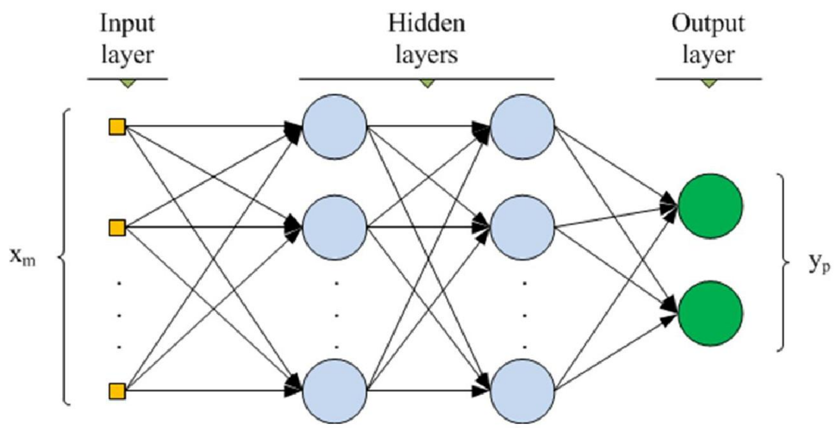
Cho perceptron với hàm kích hoạt

$$o = f(u) = f(g(x)) = \begin{cases} 1 & g(x) \geq 0 \\ -1 & g(x) < 0 \end{cases}$$

1. Vẽ siêu phẳng 2 chiều tìm được trên hệ trục tọa độ Descartes
2. Huấn luyện perceptron theo luật gradient chuẩn
3. Huấn luyện perceptron theo luật delta

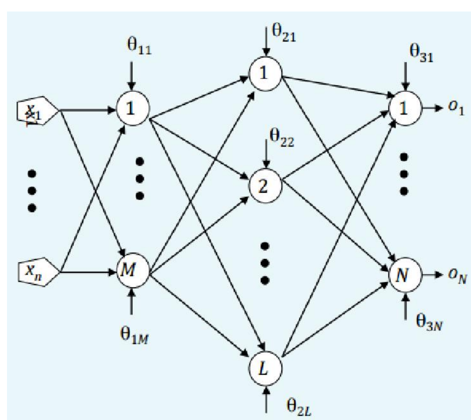
Mạng Nơron đa tầng





Mạng nơ-ron đa tầng

- Mạng nơ-ron truyền thẳng
- Nơ-ron:
  - hàm mạng tuyến tính
  - hàm kích hoạt phi tuyến, liên tục và khả vi (có thể lấy vi phân được)
    - Sigmoid
    - Hyperpolic tangent



Mạng nơ-ron đa tầng (MLP)

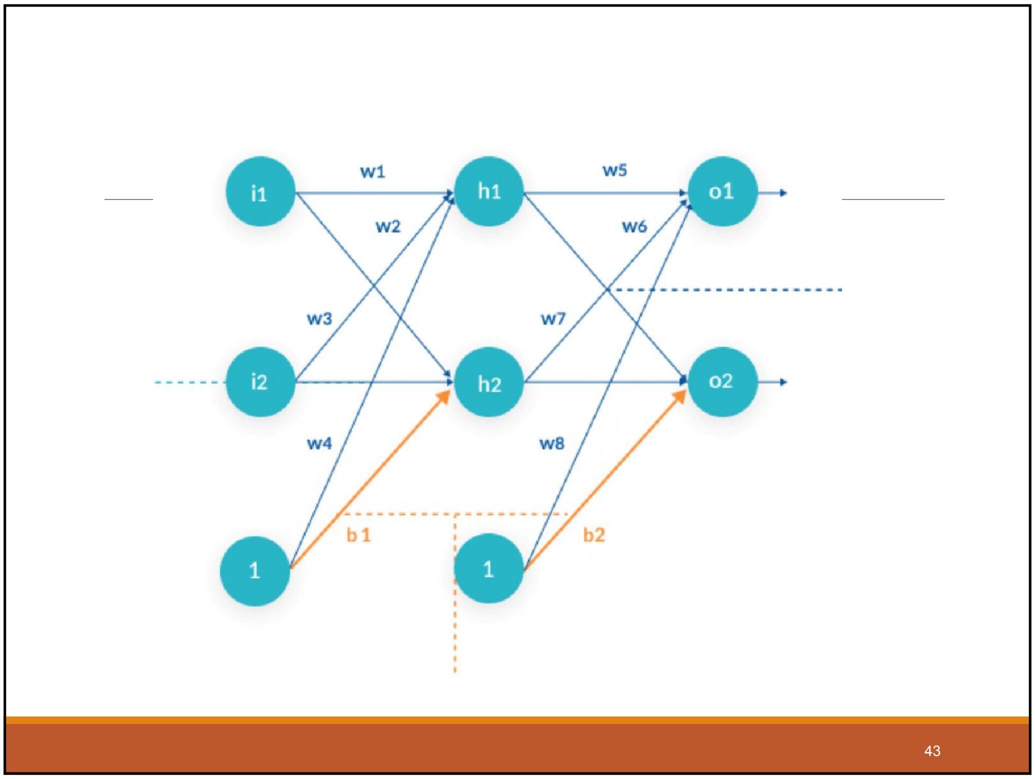
Huấn luyện mạng MLP:

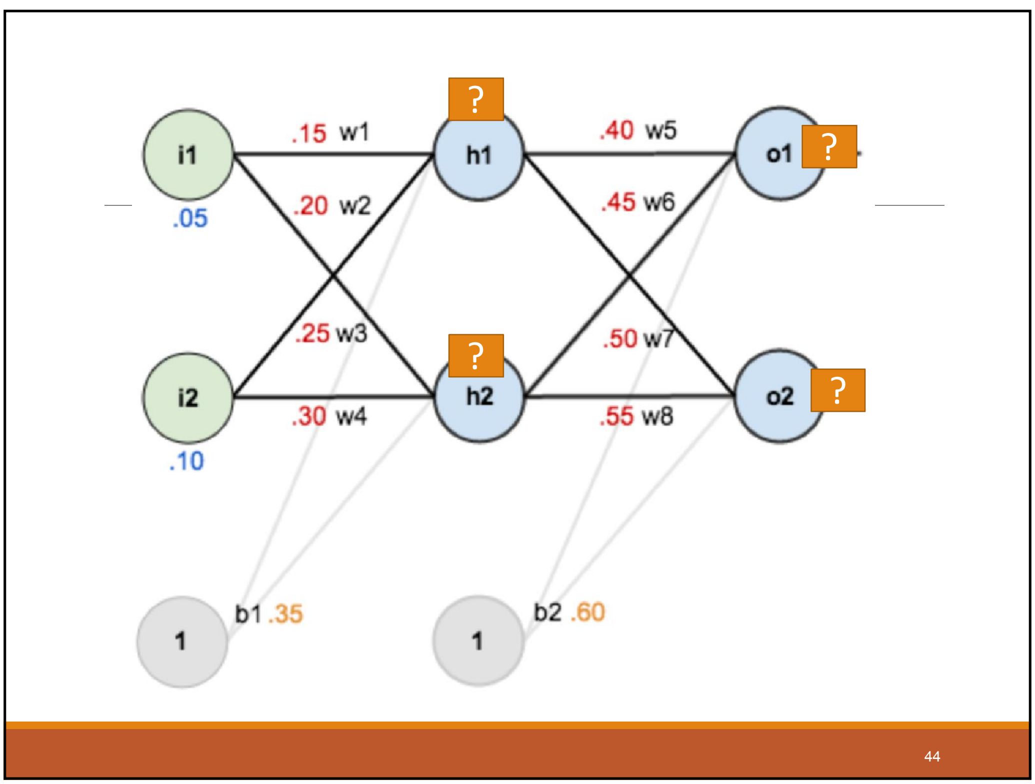
- **Giải thuật lan truyền ngược (back propagation)**

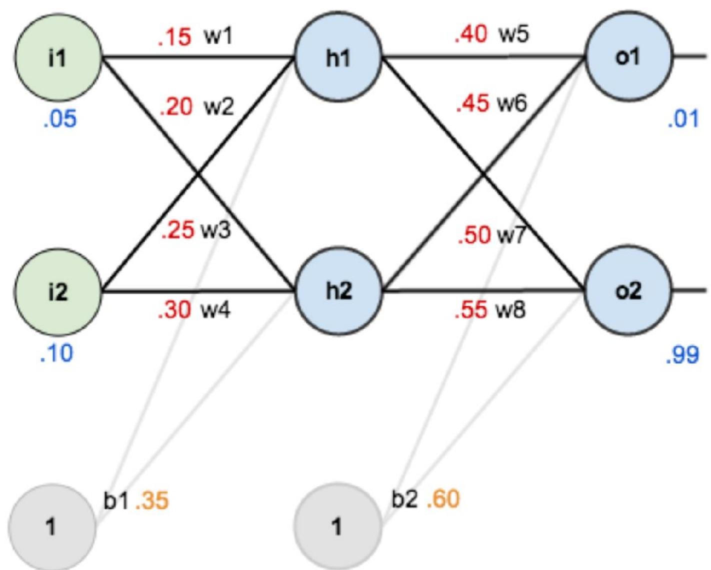
Yan Le Cun đề xuất năm 1986 - tổng quát hoá luật Delta

- Định nghĩa hàm lỗi: bình phương sai khác giữa đầu ra và đầu ra mong muốn
- Tính toán lỗi
- Lan truyền lỗi từ đầu ra ngược trở về đầu vào để cập nhật các trọng số w. Trọng số được cập nhật dựa trên gradient của hàm lỗi.

- Giải thuật học lan truyền ngược tìm kiếm một vectơ các trọng số (weights vector) giúp **cực tiểu hóa lỗi tổng thể** của hệ thống đối với tập học
- Giải thuật BP bao gồm 2 giai đoạn (bước)
  - Giai đoạn **lan truyền tiến tín hiệu (Signal forward)**. Các tín hiệu đầu vào (vectơ các giá trị đầu vào) được lan truyền tiến từ tầng đầu vào đến tầng đầu ra (đi qua các tầng ẩn)
  - Giai đoạn **lan truyền ngược lỗi (Error backward)**
    - Căn cứ vào giá trị đầu ra mong muốn của vectơ đầu vào, hệ thống tính toán giá trị lỗi
    - Bắt đầu từ tầng đầu ra, giá trị lỗi được lan truyền ngược qua mạng, từ tầng này qua tầng khác (phía trước), cho đến tầng đầu vào
    - Việc lan truyền ngược lỗi (error back-propagation) được thực hiện thông qua việc tính toán (một cách truy hồi) giá trị gradient cục bộ của mỗi nơ-ron


i1
w1
h1
w5
o1
w2
w6
w3
w7
i2
h2
o2
w4
w8
1
b1
1
b2
43


i1
.05
.15 w1
.20 w2
?
h1
.40 w5
.45 w6
o1
?
i2
.10
.25 w3
.30 w4
?
h2
.50 w7
.55 w8
o2
?
1
b1 .35
1
b2 .60
44



Sai số ở tầng đầu ra

$$\frac{\partial E}{\partial w_{a\to b}} = -(y_b - o_b)\cdot$$

$$\frac{\partial o_k}{\partial o_b} = \frac{\partial o_k}{\partial u_k}\cdot\frac{\partial u_k}{\partial o_b} = f'(u_k)\cdot\frac{\partial u_k}{\partial o_b}$$

$$\frac{\partial u_k}{\partial o_b} = w_{b\to k}$$

$$\frac{\partial E}{\partial o_b} = \sum_k\left(\frac{\partial E}{\partial o_k}\cdot f'(u_k)\cdot w_{b\to k}\right)$$

Sai số ở tầng bất kỳ khác tang đầu ra

$$\frac{\partial E}{\partial w_{a\to b}} = \frac{\partial E}{\partial o_b} \cdot f'(u_b) \cdot o_a$$

$$\frac{\partial o_k}{\partial o_b} = \frac{\partial o_k}{\partial u_k} \cdot \frac{\partial u_k}{\partial o_b} = f'(u_k) \cdot \frac{\partial u_k}{\partial o_b}$$

$$\frac{\partial u_k}{\partial o_b} = w_{b\to k}$$

$$\frac{\partial E}{\partial o_b} = \sum_k \left(\frac{\partial E}{\partial o_k} \cdot f'(u_k) \cdot w_{b\to k} \right)$$

$$w_{a\to b} = w_{a\to b} + \Delta w_{a\to b}$$

$$\Delta w_{a\to b} = -\eta \cdot \frac{\partial E}{\partial w_{a\to b}}$$

Back Propagation Learning (Derivation)

The squared error on a single example is defined as

$$E = \frac{1}{2}\sum_i (y_i - a_i)^2 \ ,$$

where the sum is over the nodes in the output layer.

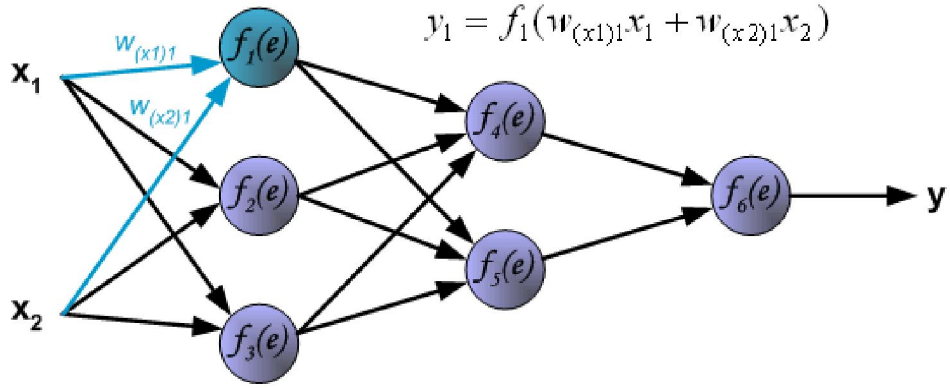
$$\begin{aligned}
\frac{\partial E}{\partial W_{j,i}} &= -(y_i - a_i)\frac{\partial a_i}{\partial W_{j,i}} = -(y_i - a_i)\frac{\partial g(in_i)}{\partial W_{j,i}} \\
&= -(y_i - a_i)g'(in_i)\frac{\partial in_i}{\partial W_{j,i}} = -(y_i - a_i)g'(in_i)\frac{\partial}{\partial W_{j,i}}\left(\sum_j W_{j,i}a_j\right) \\
&= -(y_i - a_i)g'(in_i)a_j = -a_j\Delta_i
\end{aligned}$$

Back Propagation Learning (Derivation)

$$\begin{aligned}
\frac{\partial E}{\partial W_{k,j}} &= -\sum_i (y_i - a_i)\frac{\partial a_i}{\partial W_{k,j}} = -\sum_i (y_i - a_i)\frac{\partial g(in_i)}{\partial W_{k,j}} \\
&= -\sum_i (y_i - a_i)g'(in_i)\frac{\partial in_i}{\partial W_{k,j}} = -\sum_i \Delta_i \frac{\partial}{\partial W_{k,j}}\left(\sum_j W_{j,i}a_j\right) \\
&= -\sum_i \Delta_i W_{j,i}\frac{\partial a_j}{\partial W_{k,j}} = -\sum_i \Delta_i W_{j,i}\frac{\partial g(in_j)}{\partial W_{k,j}} \\
&= -\sum_i \Delta_i W_{j,i} g'(in_j)\frac{\partial in_j}{\partial W_{k,j}} \\
&= -\sum_i \Delta_i W_{j,i} g'(in_j)\frac{\partial}{\partial W_{k,j}}\left(\sum_k W_{k,j}a_k\right) \\
&= -\sum_i \Delta_i W_{j,i} g'(in_j)a_k = -a_k\Delta_j
\end{aligned}$$

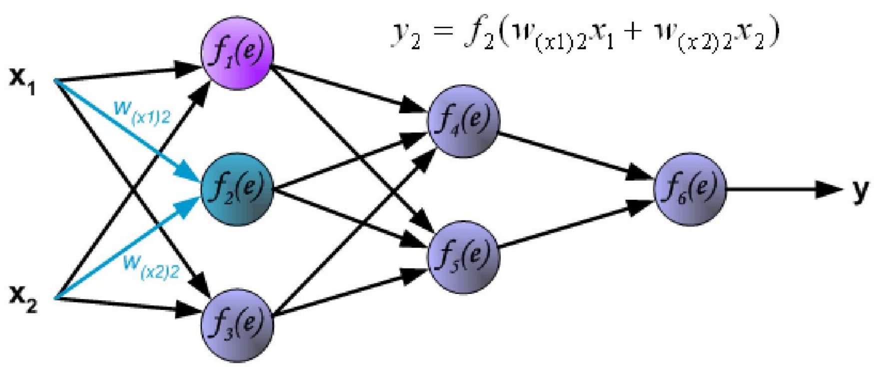
Mạng nơ-ron MLP

$$y_1 = f_1(w_{(x1)1}x_1 + w_{(x2)1}x_2)$$



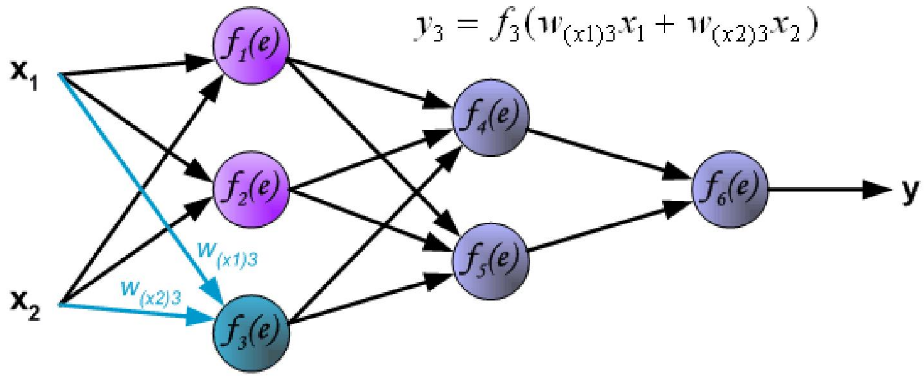
Mạng nơ-ron MLP

$$y_2 = f_2(w_{(x1)2}x_1 + w_{(x2)2}x_2)$$



Mạng nơ-ron MLP

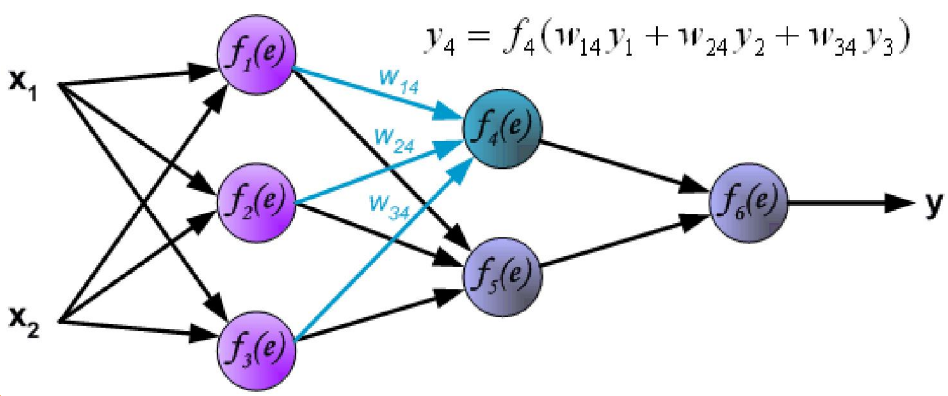
$$y_3 = f_3(w_{(x1)3}x_1 + w_{(x2)3}x_2)$$



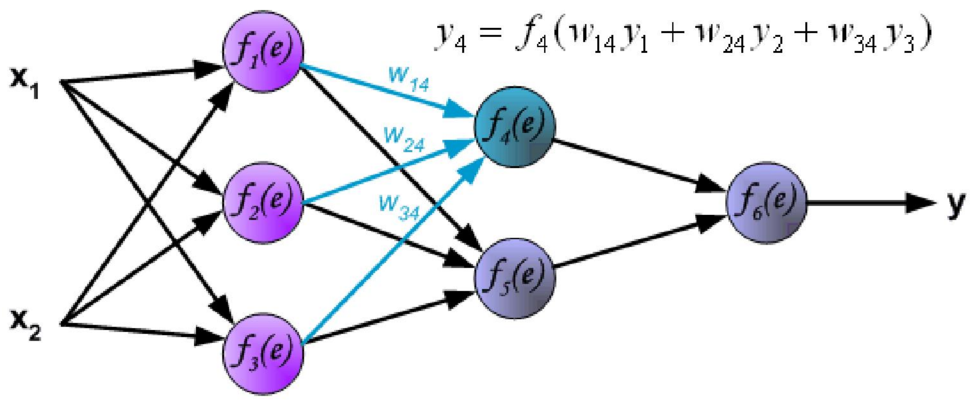
Mạng nơ-ron MLP

w_{mn} là trọng số nối kết giữa đầu ra của nơron m và đầu vào của nơron n ở tầng kế tiếp

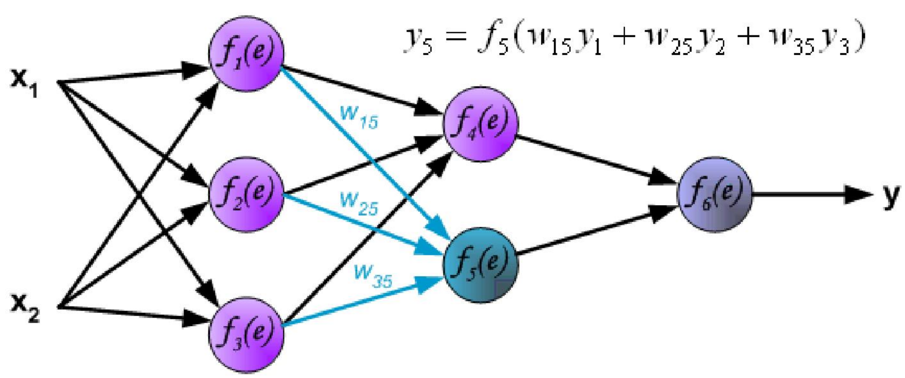
$$y_4 = f_4(w_{14}y_1 + w_{24}y_2 + w_{34}y_3)$$



Mạng nơ-ron MLP

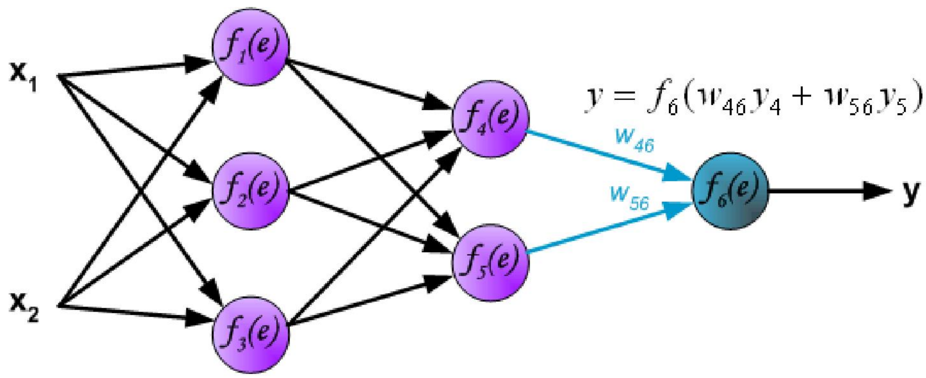


Mạng nơ-ron MLP



Mạng nơ-ron MLP

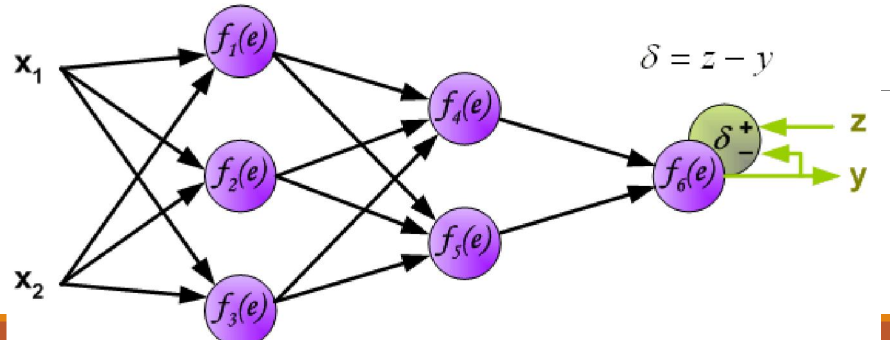
Quá trình lan truyền được thực hiện đến khi gặp đầu ra



57

Mạng nơ-ron MLP: lan truyền ngược

Tín hiệu đầu ra của mạng y được so sánh với giá trị đầu ra mong muốn (mục tiêu), được tìm thấy trong tập dữ liệu huấn luyện. Sự khác biệt được gọi là tín hiệu báo lỗi d của nơ-ron lớp đầu ra

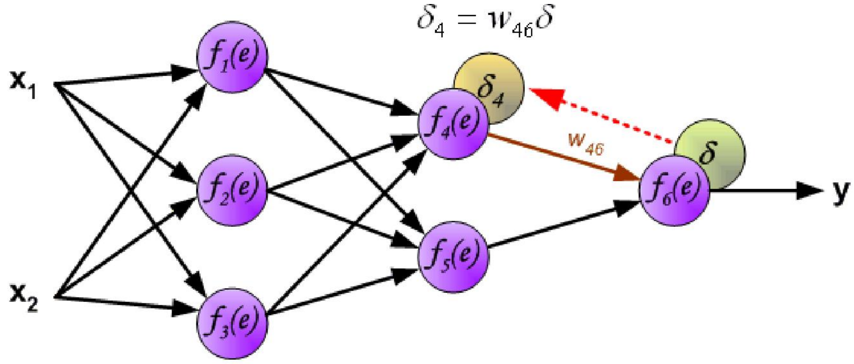


58

Mạng nơ-ron MLP: lan truyền ngược

Ý tưởng:
truyền tín hiệu báo lỗi d trở lại với tất cả các nơ-ron

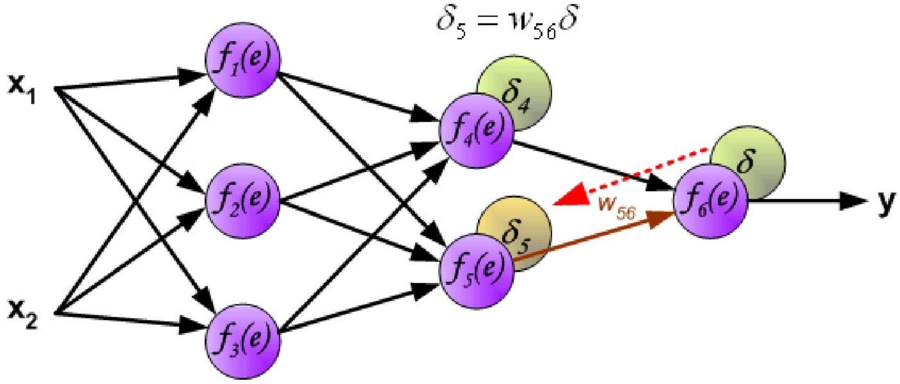
$$\delta_4 = w_{46}\delta$$



Mạng nơ-ron MLP: lan truyền ngược

Ý tưởng:
truyền tín hiệu báo lỗi d trở lại với tất cả các nơ-ron

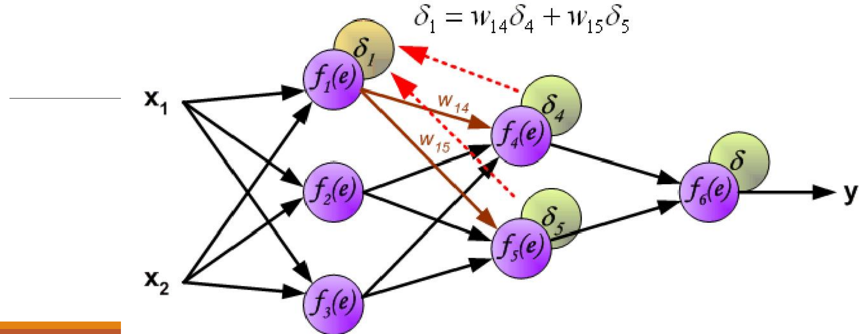
$$\delta_5 = w_{56}\delta$$



Mạng nơ-ron MLP: lan truyền ngược

Các hệ số trọng lượng của w_{mn} được sử dụng để truyền lỗi trở lại. Chỉ có hướng dòng chảy dữ liệu được thay đổi (tín hiệu được truyền từ đầu ra sang đầu vào). Kỹ thuật này được sử dụng cho tất cả các lớp mạng. Nếu các lỗi truyền xuất phát từ vài tế bào thần kinh, chúng được thêm vào. Hình minh họa dưới đây:

$$\delta_1 = w_{14}\delta_4 + w_{15}\delta_5$$

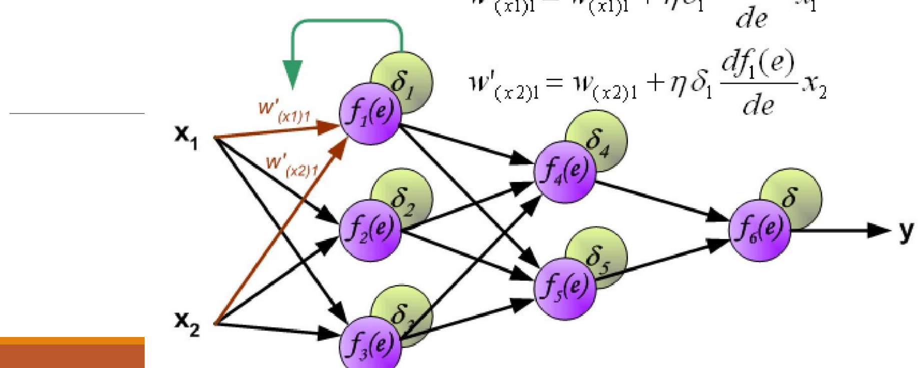


Mạng nơ-ron MLP: lan truyền ngược

Khi tín hiệu lỗi cho mỗi neuron được tính, các hệ số trọng lượng của mỗi nút đầu vào nơ-ron có thể được sửa đổi. Trong các công thức dưới df(e)/de biểu thị dẫn xuất của chức năng kích hoạt nơ-ron (trọng số được sửa đổi).

$$w'_{(x1)1} = w_{(x1)1} + \eta\delta_1 \frac{df_1(e)}{de} x_1$$

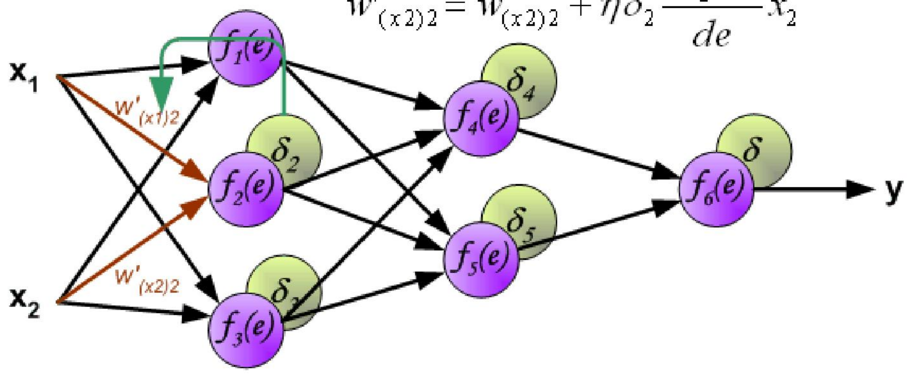
$$w'_{(x2)1} = w_{(x2)1} + \eta\delta_1 \frac{df_1(e)}{de} x_2$$



Mạng nơ-ron MLP: lan truyền ngược

$$w'_{(x1)2} = w_{(x1)2} + \eta \delta_2 \frac{df_2(e)}{de} x_1$$

$$w'_{(x2)2} = w_{(x2)2} + \eta \delta_2 \frac{df_2(e)}{de} x_2$$



Mạng nơ-ron MLP: lan truyền ngược

$$w'_{46} = w_{46} + \eta \delta \frac{df_6(e)}{de} y_4$$

$$w'_{56} = w_{56} + \eta \delta \frac{df_6(e)}{de} y_5$$