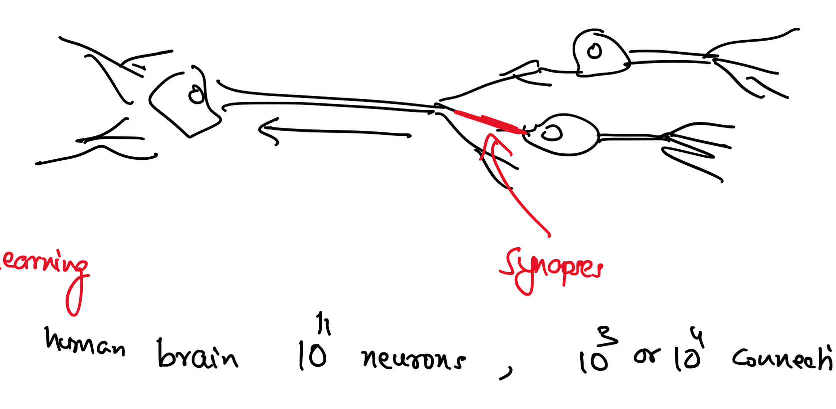
Lecture 19

Thursday, November 14, 2024 10:03 AM

Neural Networks

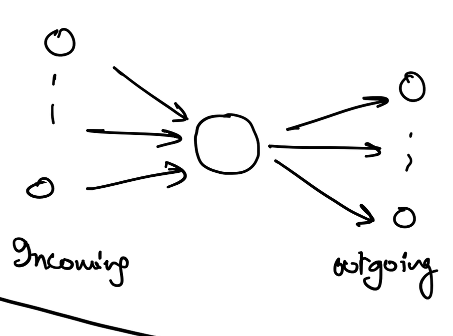


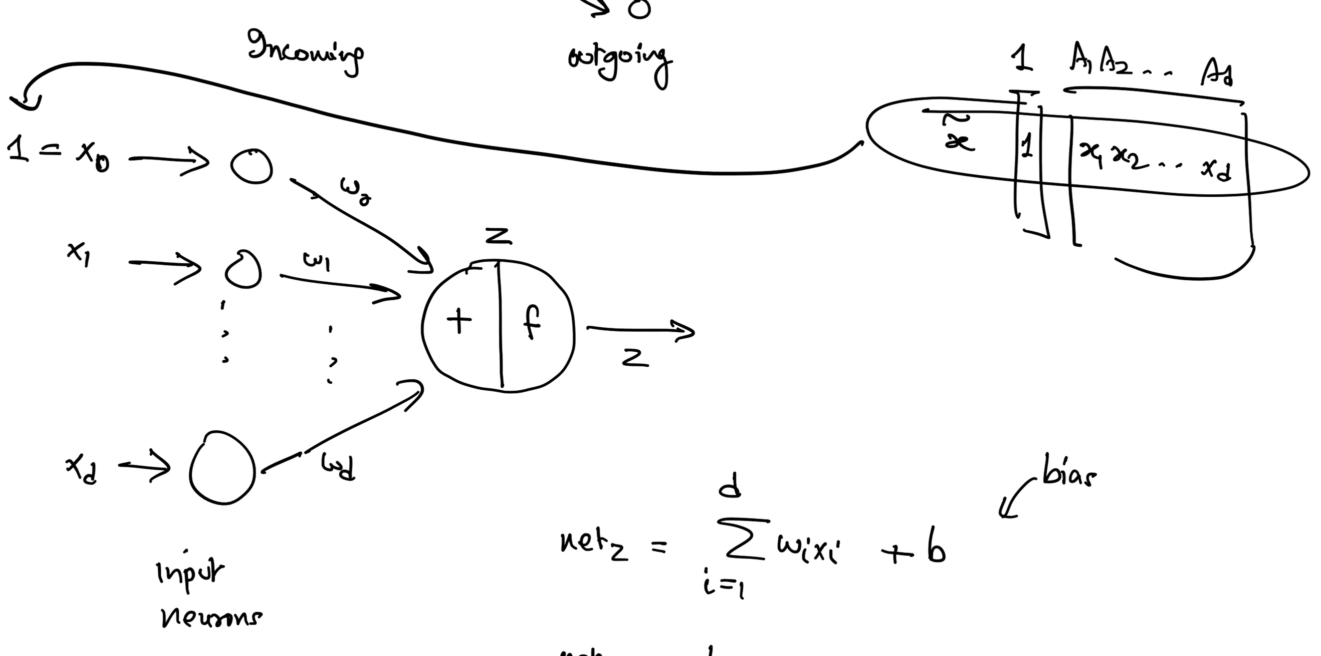
human brain 10^{11} neurons, 10^3 or 10^4 connections per neuron

10^{15} connections — quadrillion

Artificial neuron

↳ hyperplane!





$$net_z = \sum_{i=1}^{d} w_i x_i + b \quad \leftarrow \text{bias}$$

$$net_z = b + w_1x_1 + w_2x_2 + \dots + w_dx_d$$

$\equiv$ weighted sum

$\Theta = \{b, w_1, w_2, \dots, w_d\}$ ← Unknown parameters

$z = f(net_z)$ ← activation function

f = sigmoid

Biologically plausible activation

step activation

$$f(net_z) = \begin{cases} 1 & \text{if } net_z > 0 \\ 0 & \text{if } net_z \le 0 \end{cases}$$

← approx to the spiking neuron

not conducive to calculus based optimization

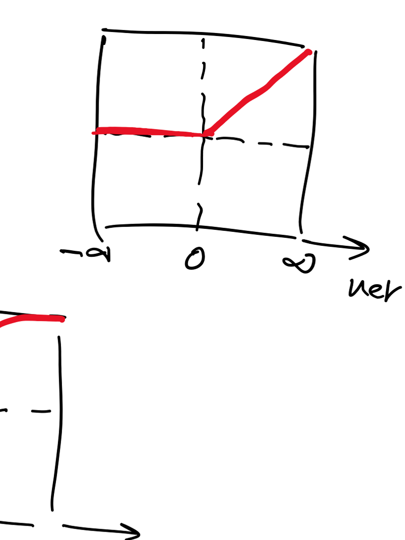
$$net_z = b + \sum_{i=1}^{d} w_i x_i > 0 \quad \text{then fire!}$$

$\sum w_i x_i > -b$ ← firing threshold

$net_z = b + \vec{w}^T\vec{x}$

Some common activation functions f:

1) step: $f(net_z) = \begin{cases} 1 & \text{if } net_z > 0 \\ 0 & \text{if } net_z \le 0 \end{cases}$



2) linear / Identity:

$$f(net_z) = net_z = b + \vec{w}^T\vec{x}$$

3) ReLU: rectified linear unit

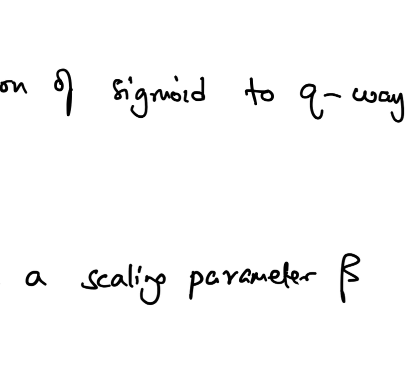
internal

$$f(net_z) = \begin{cases} net_z & \text{if } net_z > 0 \\ 0 & \text{otherwise} \end{cases}$$

4) Sigmoid

output

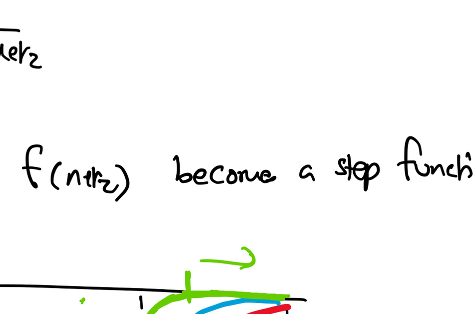
$$f(net_z) = \frac{1}{1+e^{-net_z}}$$



5) hyperbolic tangent: tanh

output

$$f(net_z) = \frac{e^{net_z} - e^{-net_z}}{e^{net_z} + e^{-net_z}}$$



6) softmax $(net_z \mid net_1, net_2, \dots, net_q)$ — output neurons, q of them

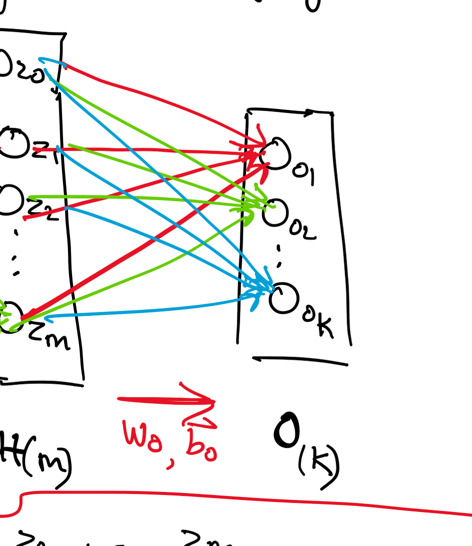
$$f(net_z) = \frac{e^{net_z}}{\sum_{i=1}^{q} e^{net_i}}$$

(generalization of sigmoid to q-way)

7) Variation of sigmoid — introduce a scaling parameter β

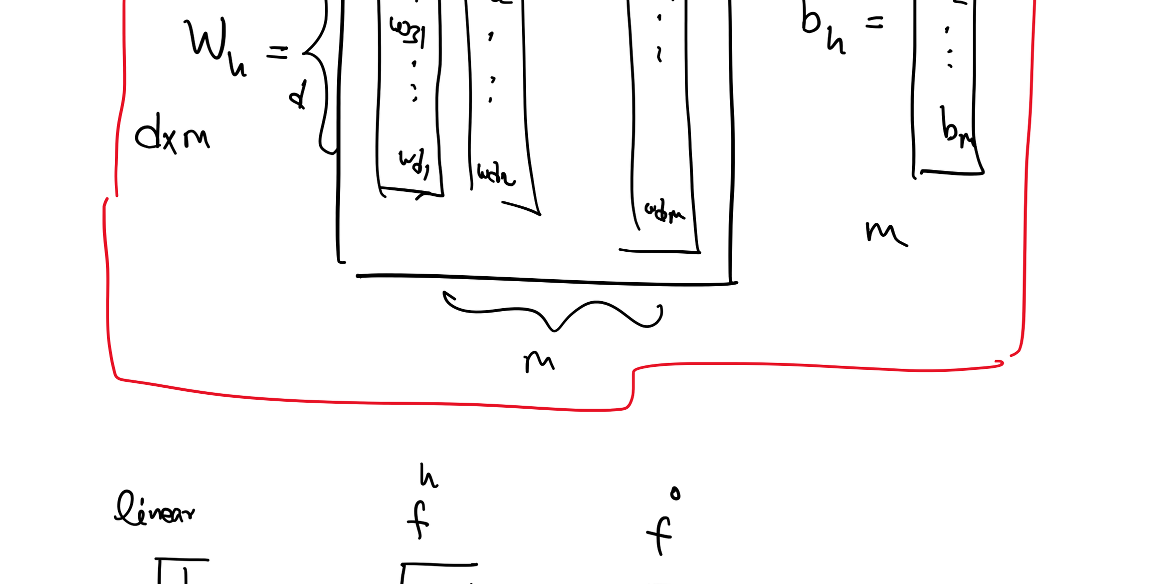
$$f(net_z) = \frac{1}{1+e^{-\beta \cdot net_z}}$$

as $\beta \to \infty$ $f(net_z)$ become a step function



MLP: Multilayer perceptron

feed forward layered NN (fully connected)

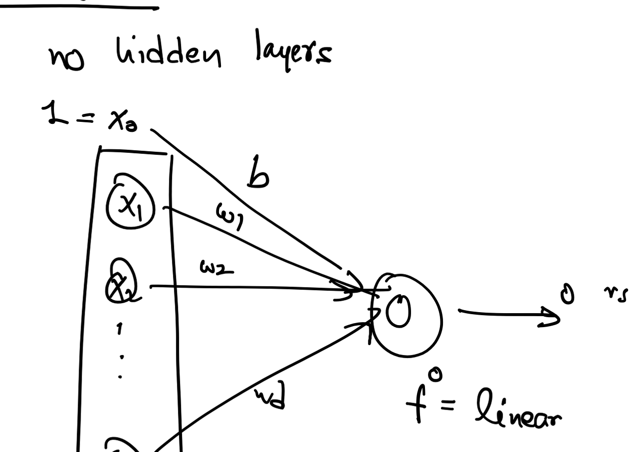


$W_o \in \mathbb{R}^{m \times k}$

$\vec{b}_o \in \mathbb{R}^k$

$$W_h = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1m} \\ w_{21} & w_{22} & & w_{2m} \\ w_{31} & \vdots & & \vdots \\ \vdots & & & \\ w_{d1} & w_{d2} & & w_{dm} \end{bmatrix} \quad (d \times m), \text{ columns } z_1, z_2, \dots, z_m$$

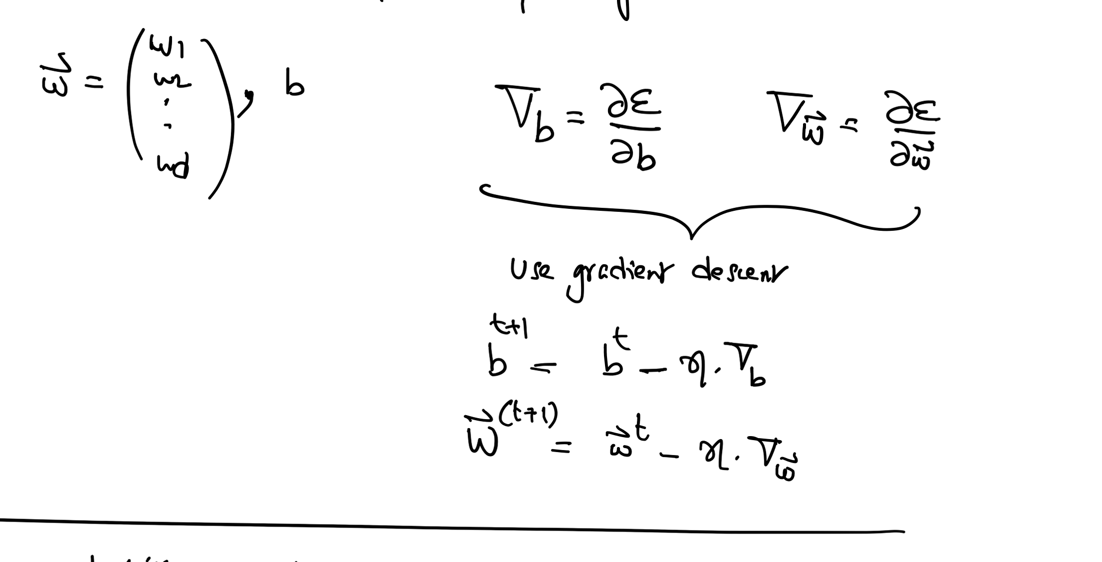
$$\vec{b}_h = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix} \quad m$$



Θ parameters $\begin{cases} W_h \equiv d \times m \\ \vec{b}_h \equiv m \end{cases}$ $\quad W_o \equiv m \times k$, $\vec{b}_o \equiv k$

Linear regression

no hidden layers



Error

$$\mathcal{E} = \frac{1}{2}(y - o)^2 \quad \text{(squared error)}$$

$$net_o = b + \sum_{i=1}^{d} w_i x_i$$

$$f^o(net_o) = net_{o_1} = \hat{y} \equiv o$$

$$\vec{w} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{pmatrix}, b$$

$$\nabla_b = \frac{\partial \mathcal{E}}{\partial b} \qquad \nabla_{\vec{w}} = \frac{\partial \mathcal{E}}{\partial \vec{w}}$$

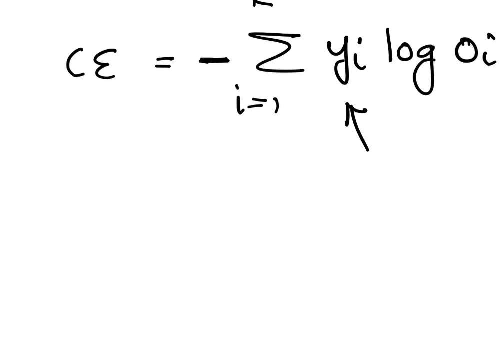
Use gradient descent

$$b^{t+1} = b^t - \eta \cdot \nabla_b$$

$$\vec{w}^{(t+1)} = \vec{w}^t - \eta \cdot \nabla_{\vec{w}}$$

---

k-way logistic regression



$$f^o(net_i) = \frac{e^{net_i}}{\sum_{j=1}^{k} e^{net_j}}$$

$$W_o = \begin{bmatrix} | & | & & | \\ w_1 & w_2 & \dots & w_k \\ | & | & & | \end{bmatrix} \quad d \times k \qquad b = (b_1, b_2 \dots b_k)^T \quad k$$

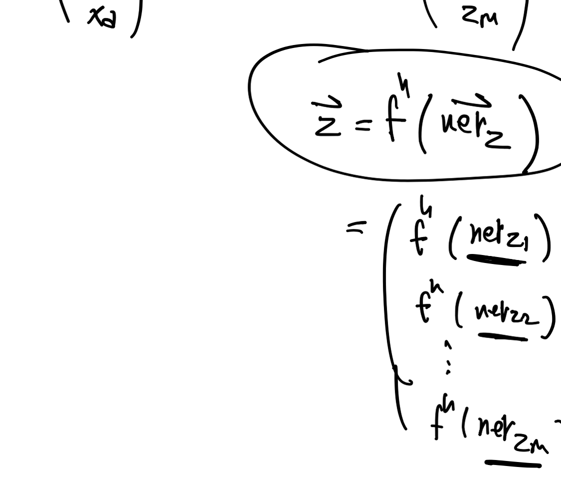
$$\vec{o} = \begin{pmatrix} f^o(net_1) \\ f^o(net_2) \\ \vdots \\ f^o(net_k) \end{pmatrix} \text{ vs } \vec{y} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ \vdots \end{pmatrix} \leftarrow \text{one-hot target}$$

CE error/loss

$$CE = -\sum_{i=1}^{k} y_i \log o_i$$

$$\vec{o} = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.5 \\ 0 \\ 0.2 \end{pmatrix} \quad \vec{y} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}$$

$$CE = -1 \cdot \log(0.5) = -\log(0.5)$$



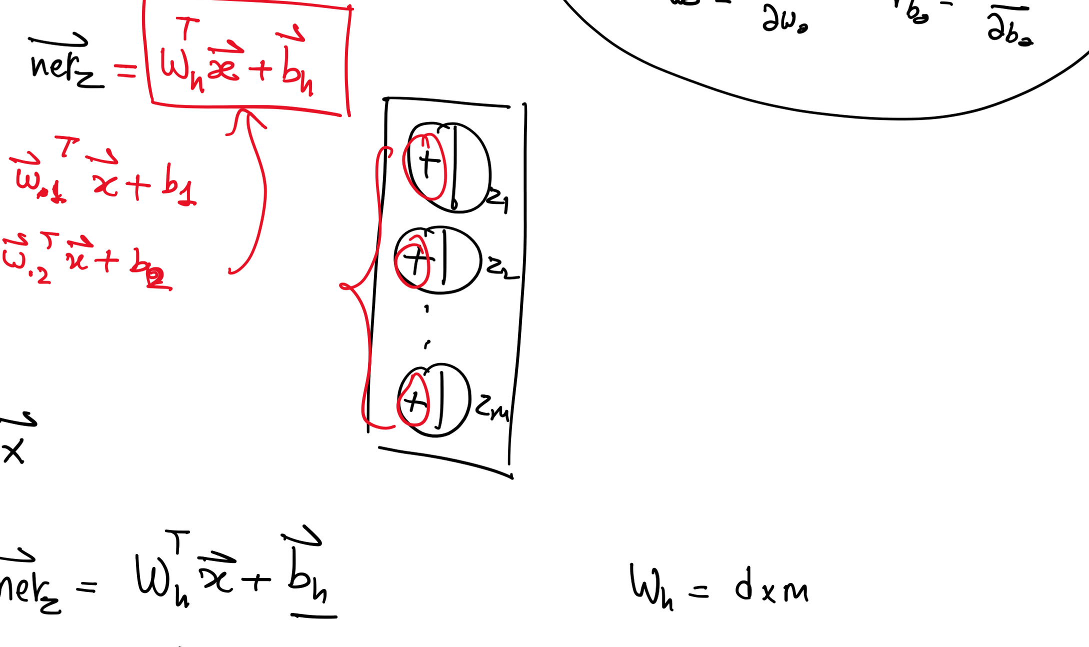
$\vec{x}$ = input vector $= \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix}$

$\vec{z}$ = hidden vector $= \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{pmatrix}$

$$\vec{z} = f^h(\vec{net_z}) = \begin{pmatrix} f^h(net_{z_1}) \\ f^h(net_{z_2}) \\ \vdots \\ f^h(net_{z_m}) \end{pmatrix}$$

$\vec{o}$ = output vector $= \begin{pmatrix} o_1 \\ o_2 \\ \vdots \\ o_k \end{pmatrix} = \begin{pmatrix} f^o(net_{o_1}) \\ f^o(net_{o_2}) \\ \vdots \\ f^o(net_{o_k}) \end{pmatrix}$

$$\vec{o} = f(\vec{net_o})$$

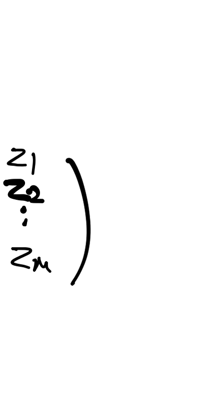


$$\nabla_{W_h} = \frac{\partial \mathcal{E}}{\partial W_h} \qquad \nabla_{\vec{b}_h} = \frac{\partial \mathcal{E}}{\partial b_h}$$

$$\nabla_{W_o} = \frac{\partial \mathcal{E}}{\partial W_o} \qquad \nabla_{\vec{b}_o} = \frac{\partial \mathcal{E}}{\partial b_o}$$

$$\vec{x} \longrightarrow \vec{net_z} = W_h^T\vec{x} + \vec{b}_h$$

$$\begin{cases} net_{z_1} = \vec{w}_{\cdot 1}^T\vec{x} + b_1 \\ net_{z_2} = \vec{w}_{\cdot 2}^T\vec{x} + b_2 \\ \vdots \end{cases}$$



0) $\vec{x}$

1) $\vec{net_z} = W_h^T\vec{x} + \vec{b}_h$ $\qquad W_h = d \times m$

$(m \times d)(d \times 1)$ → $m \times 1$; $\vec{b}_h$: $m \times 1$ → $m \times 1$

2) $f^h(\vec{net_z}) = \vec{z} = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{pmatrix}$ $\qquad \vec{z} = f^h(\vec{net_z}) = f^h(W_h^T\vec{x} + \vec{b}_h)$

3) $\vec{net_o} = W_o^T\vec{z} + \vec{b}_o$

4) $\vec{o} = f^o(\vec{net_o}) = f^o(W_o^T\vec{z} + \vec{b}_o)$

$$\vec{o} = f^o\left(W_o^T\left(f^h\left(W_h^T\vec{x} + \vec{b}_h\right)\right) + \vec{b}_o\right)$$

$$\mathcal{E} = \|\vec{y} - \vec{o}\|^2 = \mathcal{L}(\vec{y}, \vec{o})$$

$$\frac{\partial \mathcal{E}}{\partial W_o} = \frac{\partial \mathcal{L}}{\partial \_} \cdot \frac{\partial f^o}{\_}$$

$$\nabla_{W_h} = \frac{\partial \mathcal{E}}{\partial W_h} =$$