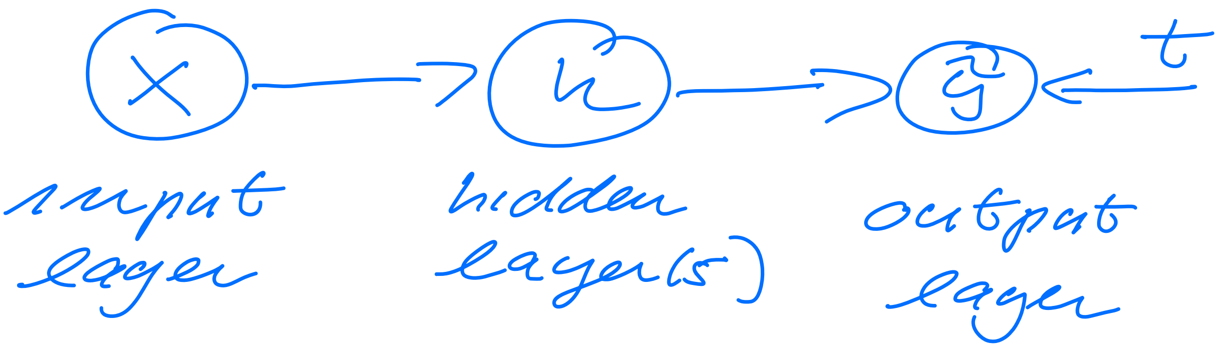
FYS-STK4155, OCT 7, 2022

Neural Networks

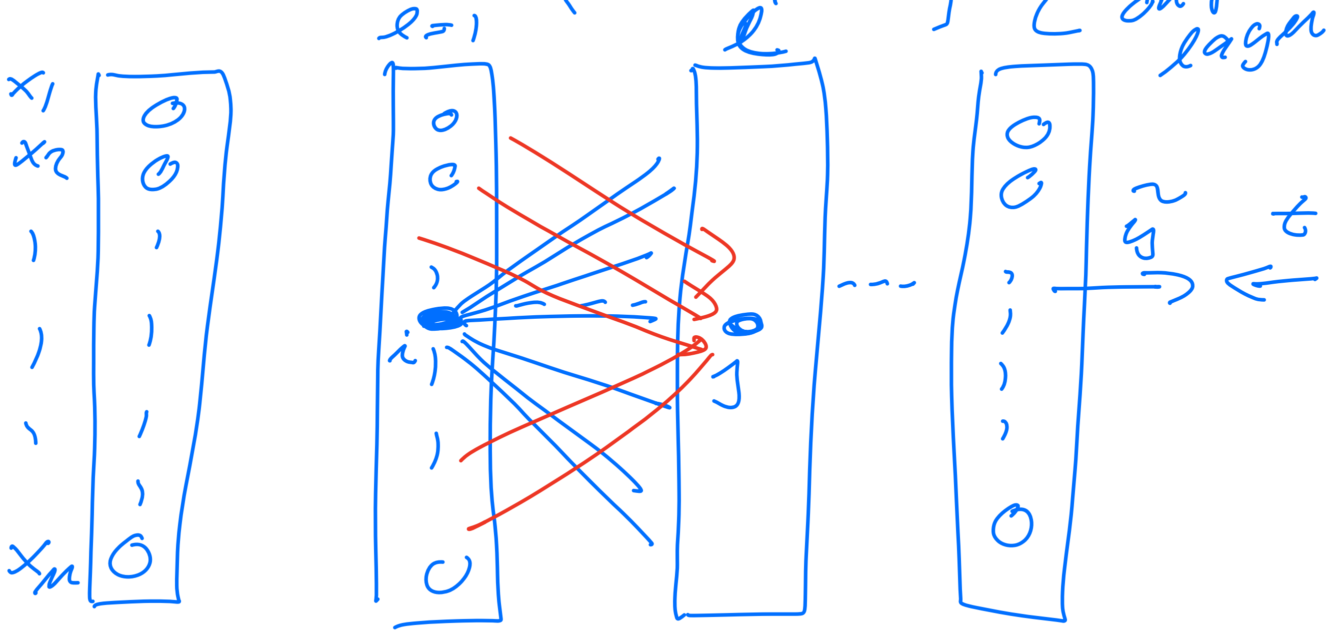


Multiperceptron (MLP)

Feed Forward NN (FFNN)

our parameters

$$\Theta = \{ W^l, b^l \}$$



N_1 $\qquad$ N_ℓ

input to node j in layer (hidden) $-\ell-$ from layer $-\ell-1-$

$$z_j^\ell = \sum_{i=1}^{N_{\ell-1}} W_{ij}^\ell a_i^{\ell-1} + b_j^\ell$$

weights $\qquad$ bias

$a_i^{\ell-1}$ = output from node i in layer $\ell-1$

Cost/Loss function (Regression)

$$C(\theta) = \frac{1}{2} \sum_{i=1}^{n} (a_i^L - t_i)^2$$

$\tilde{y}_i$

$$a_j^\ell = \sigma^\ell(z_j^\ell)$$

activation function

$$z^l = (W^l)^T a^{l-1} + b^l$$

Popular activation function

sigmoid

$$\sigma^l(z_j^l) = \frac{1}{1+e^{-z_j^l}}$$

$$= a_j^l$$

Basic ingredients in FFNN

- Feed forward stage
- Back propagation stage to train Θ
  - Expression for gradients of C as function of the various layers.
  - Gradient optimization (SGD, momentum

Adaptive ...)

- Repeat with Feed forward and back propagation to update Θ till our cost function reaches a converged value.

Back propagation algo

$$z_j^l = \sum_{i=1}^{N_{l-1}} w_{ij}^l a_i^{l-1} + b_j^l$$

$$a_i^{l-1} = \sigma(z_i^{l-1})$$

$$\frac{\partial z_j^l}{\partial w_{ij}^l} = a_i^{l-1}$$

$$\frac{\partial z_j^l}{\partial a_i^{l-1}} = w_{ij}^l$$

$$\frac{\partial a_j^l}{\partial z_j^l} = ?$$

$$a_j^l = \sigma^l(z_j^l) = \frac{1}{1+e^{-z_j^l}}$$

$$\frac{\partial a_j^l}{\partial z_j^l} = \underbrace{\sigma^l(z_j^l)(1-\sigma(z_j^l))}_{\text{changes wrt activation function}}$$

$$= a_j^l(1 - a_j^l)$$

$l = L$ (output layer)

$$C(\theta) = \frac{1}{2}\sum_i (a_i^L - t_i)^2$$

$$\frac{\partial C(\theta)}{\partial w_{jk}^L} = (a_i^L - t_i)\frac{\partial a_j^L}{\partial w_{jk}^L}$$

$$\frac{\partial a_j^L}{\partial w_{jk}^L} = \frac{\partial a_j^L}{\partial z_j^L}\,\frac{\partial z_j^L}{\partial w_{jk}^L}$$

$$= \underbrace{a_j^L\left(1-a_j^L\right)}_{\frac{\partial \sigma(z_j^L)}{\partial z_j^L}}\, a_k^{L-1}$$

$$\frac{\partial C}{\partial w_{jk}^L} = \left(a_j^L - t_j\right) a_j^L\left(1-a_j^L\right)$$

$$\times\ a_k^{L-1}$$

$$= \left(a_j^L - t_j\right) \sigma'\left(z_j^L\right) a_k^{L-1}$$

$$\delta_j^L = \sigma'\left(z_j^L\right)\left(a_j^L - t_j\right)$$

$$= \sigma'\left(z_j^L\right) \frac{\partial C}{\partial a_j^L}$$

in vector notation

$$\delta^L = \sigma'(z^L) \odot \frac{\partial C}{\partial a^L}$$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \odot \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 y_1 \\ x_2 y_2 \end{bmatrix}$$

$$\boxed{\frac{\partial C}{\partial w^L_{jk}} = \delta^L_j \, a^{L-1}_k}$$

$$\delta^L_j = \frac{\partial C}{\partial z^L_j} = \frac{\partial C}{\partial a^L_j} \frac{\partial a^L_j}{\partial z^L_j}$$

$$\delta^L_j = \frac{\partial C}{\partial b^L_j} \frac{\partial b^L_j}{\partial z^L_j} = \frac{\partial C}{\partial b^L_j}$$

$$\boxed{l = L}$$

$$\frac{\partial C}{\partial w^L_{jk}} = \delta^L_j \, a^{L-1}_k$$

$$\delta^L_j = \sigma'(z^L_j) \frac{\partial C}{\partial a^L_j}$$

$$\boxed{\frac{\partial C}{\partial b_j^L} = \delta_j^L}$$

$l \neq L$

$$\delta_j^l = \frac{\partial C}{\partial z_j^l} = \sum_k \underbrace{\frac{\partial C}{\partial z_k^{l+1}}}_{\delta_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l}$$

$$= \sum_k \delta_k^{l+1} \frac{\partial z_k^{l+1}}{\partial z_j^l}$$

$$z_j^{l+1} = \sum_{i=1}^{N_l} w_{ij}^{l+1} \underbrace{a_i^l}_{\sigma(z_j^l)} + b_j^{l+1}$$

$$\frac{\partial z_k^{l+1}}{\partial z_j^l} = w_{kj}^{l+1} \sigma'(z_j^l)$$

$$\boxed{\delta_j^{l} = \sum_k \delta_k^{l+1} w_{kj}^{l+1} \sigma'(z_j^{l})}$$

Algo

- Define Model
  (architecture of NN)
  - # hidden layers
  - # nodes in layers
- initialize $\{W, b\} = \Theta$
- activation function
- hyperparameters
  λ, l_2 or l_1 regularization
- learning rate +
  learning rate schedule
  $\rightarrow$ Adagrad
  RMS Prop ...

— Gradient descent methods

— Need X and t (target)

while convergence not met

1. First Feed Forward

output $= a^L = \sigma^L(\sigma^{L-1}(\ldots \sigma^1(z^1)\ldots))\ldots)$

$z^1 = W^1 x + b$

2. First Back prop stage

$l = L-1, L-2 \ldots 1$

$$\delta_j^l = \sum_k \delta_k^{l+1} w_{kj}^{l+1} \sigma'(z_j^l)$$

$$w_{jk}^l \leftarrow w_{jk}^l - \underbrace{\eta}_{\text{learning rate}} \delta_j^l a_k^{l-1}$$

$$b_j^b \leftarrow b_j^l - \eta \delta_j^l$$

$\underline{3}$ back to $\underline{1}$ and $\underline{2}$

repeat $\underline{3}$ till convergence

Problems: Vanishing gradients.

Example $L=2$, simple case, x, w and b are scalars.

output $f(x;\theta) = \hat{y}$

$$= \sigma_2(w_2\sigma_1(w_1x+b_1)+b_2)$$

$$\partial_{w_1} f(x;\theta) = \sigma_2'(w_2\sigma_1(w_1x+b_1)+b_2) \otimes w_2\sigma_1'(w_1x+b_1)x]$$

with many layers L

$$\partial_{w_1} f(x;\theta) = \left[\prod_{l=2}^{2} w_2\right] \times$$

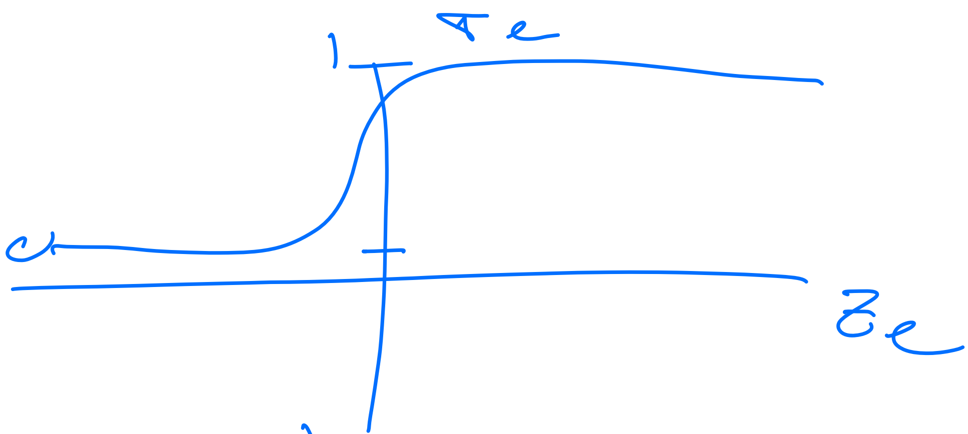
$$\left[\prod_{l=1}^{L} \sigma_z'(z_l)\right] X$$

$$z_l = A_l\left(\sigma_{l-1}\left(A_{l-1}\left(\cdots \sigma_1(A_1(x)) \cdots\right)\right)\right)$$

$\uparrow$

output
from $l=1$, $l-2$ . .

if σ_l is a sigmoid function, then if $|z_l| \gg 0$



then $\sigma_l'(z_l) \to 0$

List of $\sigma(z_l)$ $(\sigma(z))$

Rectified linear unit

RELU $\sigma(z) = \max(0, z)$

$\sigma'(z) = 1$ for $z > 0$

Leaky ReLU $\sigma(z) = \begin{cases} \max(\alpha, z) & z \geq 0 \\ (z \text{ if } z \geq 0) & \\ \alpha z & \text{for } z < 0 \end{cases}$

~ 0.01

$\sigma'(z) = \begin{cases} \alpha & z < 0 \\ 1 & z \geq 0 \end{cases}$

ELU : $\sigma(z) = \begin{cases} \alpha(e^{z} - 1) & z < 0 \\ z & z \geq 0 \end{cases}$

$\sigma'(z) = \begin{cases} \alpha e^{z} & z < 0 \\ 1 & z \geq 0 \end{cases}$

tanh : $\sigma(z) = \tanh(z)$

$\sigma'(z) = 1 - (\tanh z)^2$

Sigmoid: $\sigma(z) = \dfrac{1}{1+e^{-z}}$

$$\sigma'(z) = \sigma(z)(1-\sigma(z))$$