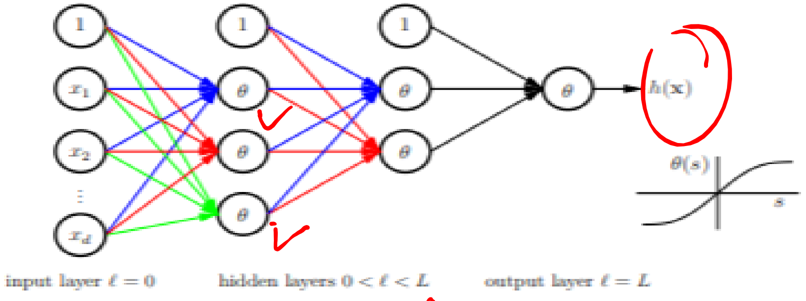
Machine Learning from Data

Lecture 21: Spring 2021

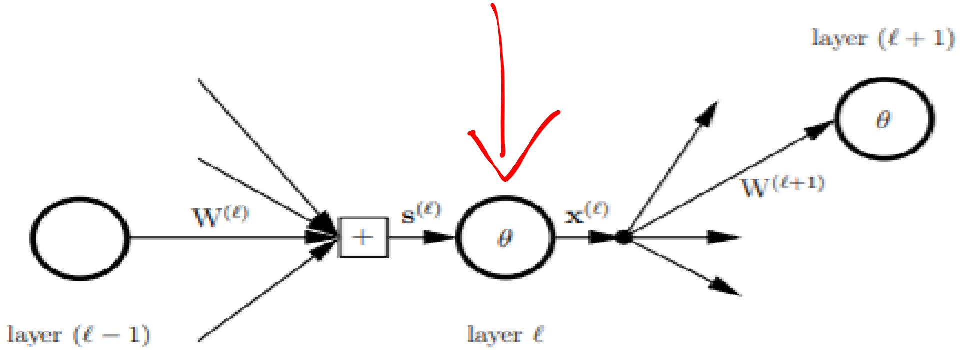
Today's Lecture

- Neural Networks
  - Forward Propagation ✓
  - Backward Propagation
  - Overfitting

Recap



NN



layer ℓ parameters

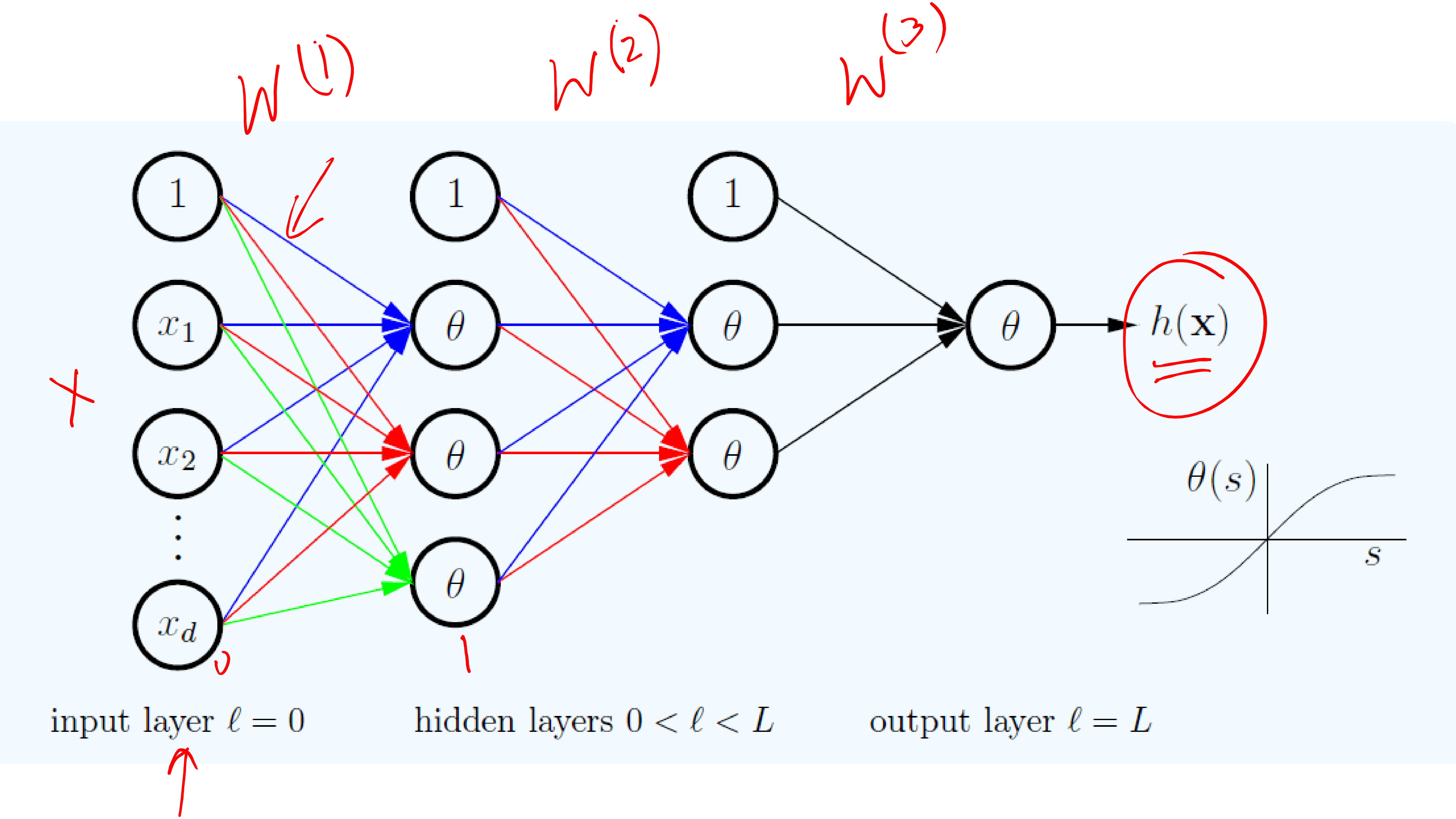
signals in	$\mathbf{s}^{(\ell)}$	$d^{(\ell)}$ dimensional input vector
outputs	$\mathbf{x}^{(\ell)}$	$d^{(\ell)} + 1$ dimensional output vector
weights in	$\mathrm{W}^{(\ell)}$	$(d^{(\ell-1)} + 1) \times d^{(\ell)}$ dimensional matrix
weights out	$\mathrm{W}^{(\ell+1)}$	$(d^{(\ell)} + 1) \times d^{(\ell+1)}$ dimensional matrix

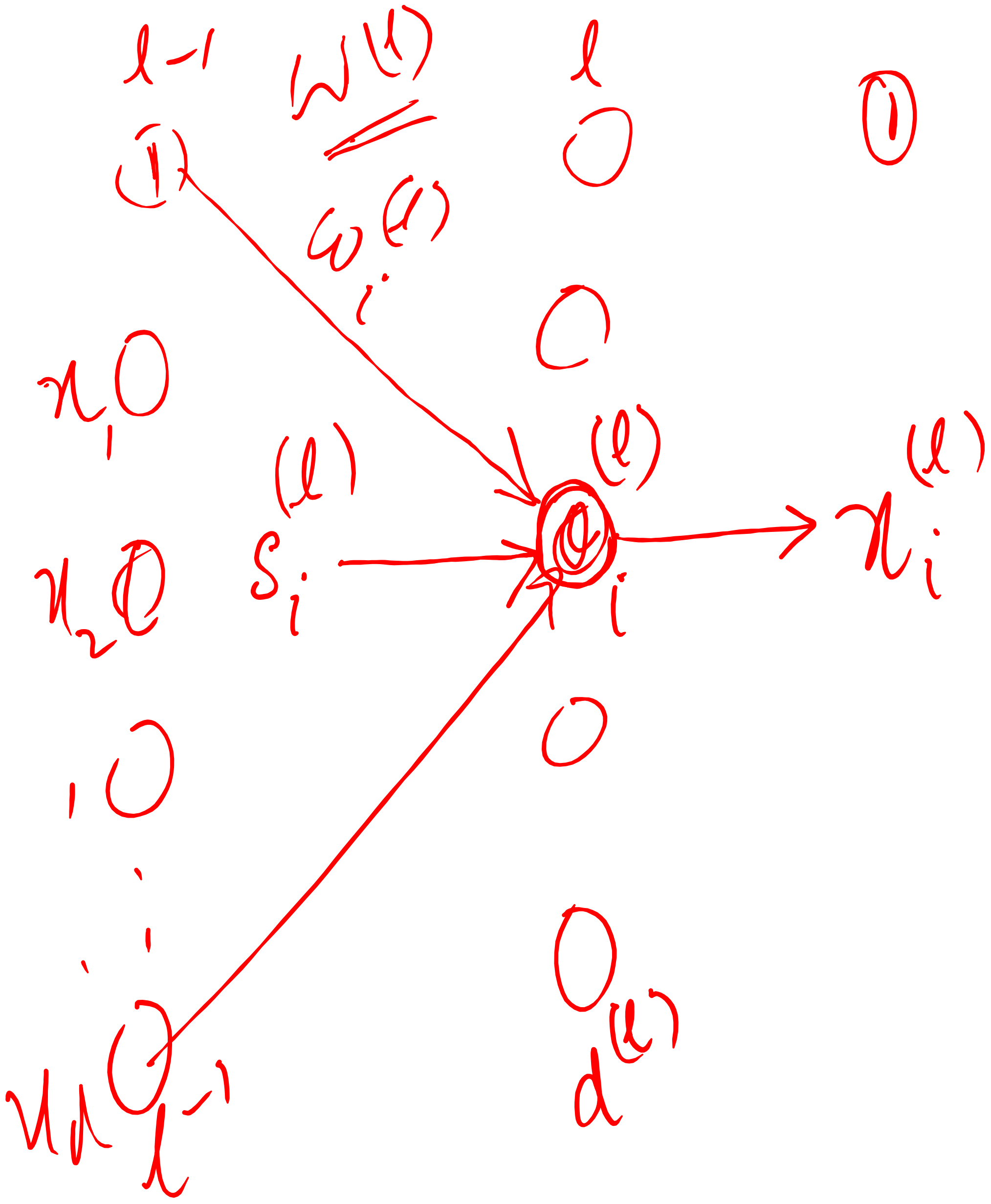
layers $\ell = 0, 1, 2, \ldots, L$

layer ℓ has "dimension" $d^{(\ell)} \implies d^{(\ell)} + 1$ nodes

$$\mathrm{W}^{(\ell)} = \begin{bmatrix} \mathbf{w}_1^{(\ell)} & \mathbf{w}_2^{(\ell)} & \cdots & \mathbf{w}_{d^{(\ell)}}^{(\ell)} \\ | & | & \vdots & | \end{bmatrix}$$

$W^{(1)} \quad W^{(2)} \quad \ldots \quad W^{(L)}$


W(1)
W(2)
W(3)
1
1
1
x1
x2
xd
θ
θ
θ
θ
θ
θ
h(x)
θ(s)
s
x
0
1
input layer ℓ = 0
hidden layers 0 < ℓ < L
output layer ℓ = L



$$\underline{S}^{(l)} = \begin{bmatrix} s_1^{(l)} \\ s_2^{(l)} \\ \vdots \\ s_{d^{(l)}}^{(l)} \end{bmatrix}$$

$$x^{(l)} = \begin{array}{cc} \begin{bmatrix} 1 \\ x_1^{(l)} \\ x_2^{(l)} \\ \vdots \\ x_d^{(l)} \end{bmatrix} & d^l+1 \end{array}$$

$$W^{(l)} = \underbrace{\begin{bmatrix} w_1^{(l)} & w_2^{(l)} & \cdots & w_d^{(l)} \\ | & | & & | \\ | & | & & | \end{bmatrix}}_{d\text{-dimensional}} \left.\vphantom{\begin{bmatrix} a \\ b \\ c \end{bmatrix}}\right\} d^{(l-1)}+1$$

perceptron

Forward Propogation Algorithm

$$W = \{ \underset{0,1}{W^{(1)}}, \underset{1,2}{W^{(2)}}, \underset{2,3}{W^{(3)}} \}$$

$\theta_i^l \longrightarrow s_i^{(l)} = \underbrace{(W_i^{(l)})^T x^{(l-1)}}_{\text{linear signal}}$

weight of perceptron

$$s^l = \begin{bmatrix} s_1^{(l)} \\ s_2^{(l)} \\ \vdots \\ s_d^{(l)} \end{bmatrix} = \begin{bmatrix} \text{---} (W_1^{(l)})^T \text{---} \\ \text{---} (W_2^{(l)})^T \text{---} \\ \\ \text{---} (W_d^{(l)})^T \text{---} \end{bmatrix} x^{(l-1)}$$

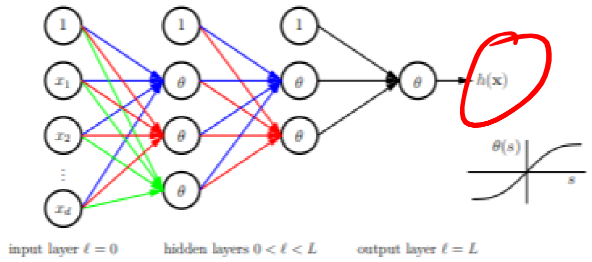
Signal

$$s^l = (W^{(l)})^T x^{(l-1)}$$

$$X^l = \begin{bmatrix} 1 \\ \theta(s^l) \end{bmatrix}$$

The Linear Signal

Input $\mathbf{s}^{(\ell)}$ is a linear combination (using weights) of the outputs of the previous layer $\mathbf{x}^{(\ell-1)}$.



$$\mathbf{s}^{(\ell)} = (\mathrm{W}^{(\ell)})^{\mathrm{T}}\mathbf{x}^{(\ell-1)}$$

$$\begin{bmatrix} s_1^{(\ell)} \\ s_2^{(\ell)} \\ \vdots \\ s_j^{(\ell)} \\ \vdots \\ s_{d^{(\ell)}}^{(\ell)} \end{bmatrix} = \begin{bmatrix} (\mathbf{w}_1^{(\ell)})^{\mathrm{T}} & \text{———} \\ (\mathbf{w}_2^{(\ell)})^{\mathrm{T}} & \text{———} \\ & \vdots \\ (\mathbf{w}_j^{(\ell)})^{\mathrm{T}} & \text{———} \\ & \vdots \\ (\mathbf{w}_{d^{(\ell)}}^{(\ell)})^{\mathrm{T}} & \text{———} \end{bmatrix} \mathbf{x}^{(\ell-1)}$$

$$s_j^{(\ell)} = (\mathbf{w}_j^{(\ell)})^{\mathrm{T}}\mathbf{x}^{(\ell-1)}$$

(recall the linear signal $s = \mathbf{w}^{\mathrm{T}}\mathbf{x}$)

$$\mathbf{s}^{(\ell)} \xrightarrow{\theta} \mathbf{x}^{(\ell)}$$

Forward prop. algo.

$W = \{ W^{(1)}_{*}, W^{(2)}_{*}, \ldots, W^{(L)} \}$

1) Initialize $x^{(0)} = x$, $l = 0$ $h(x) \rightarrow$ goal $x = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$

2) Propogate, set $l = l+1$

$$\underline{\underline{s^l}} = \left(W^{(l)}\right)^T x^{(l-1)}$$

$$x^l = \begin{bmatrix} 1 \\ \theta(s^l) \end{bmatrix} \qquad \text{Repeat}$$

3) $x^{(L)} \longrightarrow h(x) = \underline{\underline{x^{(L)}_1}}$

Forward Propagation: Computing $h(\mathbf{x})$

$$\mathbf{x} = \mathbf{x}^{(0)} \xrightarrow{\mathrm{W}^{(1)}} \mathbf{s}^{(1)} \xrightarrow{\theta} \mathbf{x}^{(1)} \xrightarrow{\mathrm{W}^{(2)}} \mathbf{s}^{(2)} \xrightarrow{\theta} \mathbf{x}^{(2)} \cdots \xrightarrow{\mathrm{W}^{(L)}} \mathbf{s}^{(L)} \xrightarrow{\theta} \mathbf{x}^{(L)} = h(\mathbf{x}).$$

Forward propagation to compute $h(\mathbf{x})$:

1: $\mathbf{x}^{(0)} \leftarrow \mathbf{x}$ **[Initialization]**

2: **for** $\ell = 1$ to L **do** **[Forward Propagation]**

3: $\quad \mathbf{s}^{(\ell)} \leftarrow (\mathrm{W}^{(\ell)})^{\mathrm{T}} \mathbf{x}^{(\ell-1)}$

4: $\quad \mathbf{x}^{(\ell)} \leftarrow \begin{bmatrix} 1 \\ \theta(\mathbf{s}^{(\ell)}) \end{bmatrix}$

5: **end for**

6: $h(\mathbf{x}) = \mathbf{x}^{(L)}$ **[Output]**

Data

$$x_1 \quad y_1 \xrightarrow{NN} h(x_1)$$

$$x_2 \quad y_2 \xrightarrow{NN} h(x_2)$$

$$\vdots$$

$$x_n \quad y_n \xrightarrow{NN} h(x_n)$$

$$W^{(1)} \quad W^{(2)} \quad W^{(3)} \cdots W^{(L)}$$

$$E_{in}(h) = \frac{1}{N}\sum_{i=1}^{N}\left(h(x_n) - y_n\right)^2$$

$$E_{in}(h) = \frac{1}{N}\sum_{i=1}^{N} e\left(h(x_n), y_n\right)$$

Know $\longrightarrow$ W's $\longrightarrow$ find weights that minimize E_{in}

$$W^{(\ell)} \leftarrow W^{(\ell)} - \alpha \frac{\partial E_{in}}{\partial W^{(\ell)}}$$

learning rate

$$\frac{\partial E_{in}}{\partial W^{(\ell)}} = \frac{1}{N} \sum_{n=1}^{N} \frac{\partial e(h(x_n), y_n)}{\partial W^{(\ell)}}$$

Summary → Compute Output ×
Getting weights $\Longleftrightarrow$ fitting data.

In practice

$|W| = W^1 + W^2 + \dots W^L$

$|V| = $ No. of nodes $(\theta) \rightarrow$ tanh

$E_{in} \rightarrow$ how many computations?

$N|W| + N|V|$

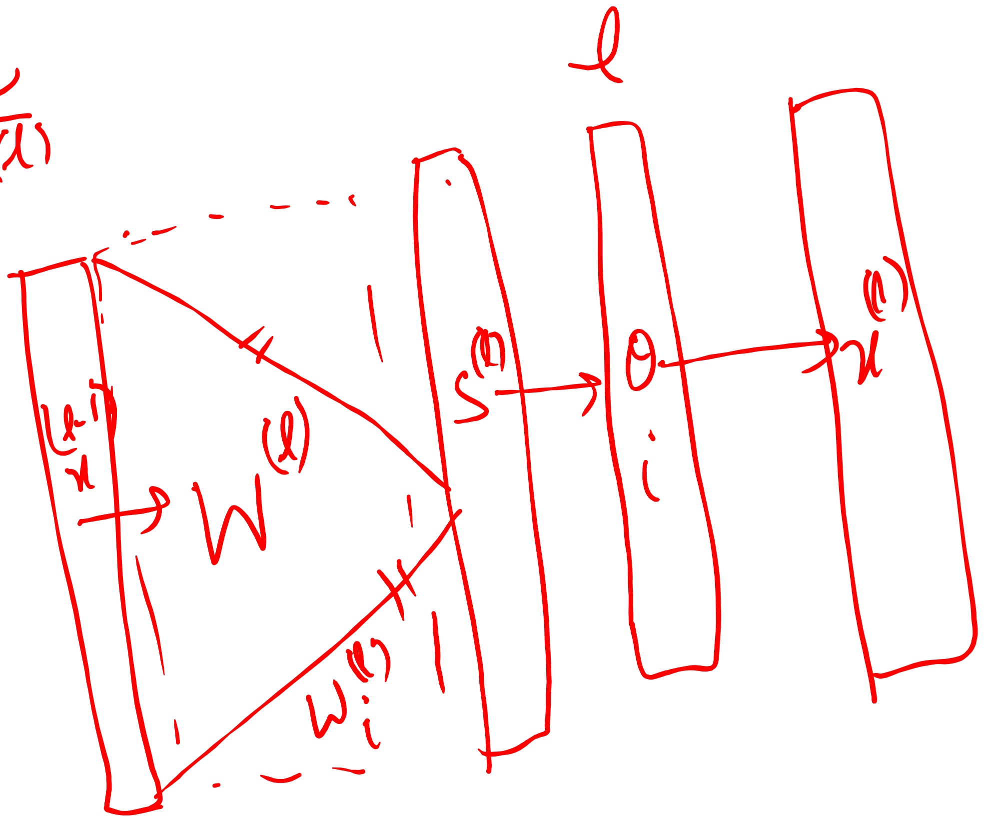
$w_{ij}^{(l)} \rightarrow \dfrac{\partial E_{in}}{\partial w_{ij}^{(l)}} \simeq \dfrac{E_{in}(w_{ij}^{(l)} + \Delta) - E(w_{ij}^{(l)} - \Delta)}{2\Delta}$

tolerance

$O((N|W| + N|V|)|W|)$

BACKPROPOGATION ALGORITHM

$$\frac{\partial e}{\partial W^{(l)}}$$



$$\boxed{\frac{\partial e}{\partial w_i^{(l)}}}$$

Chain Rule

$$\frac{\partial e}{\partial w_i^{(l)}} = \frac{\partial s_i^{(l)}}{\partial w_i^{(l)}} \times \frac{\partial e}{\partial s_i^{(l)}} \quad , \qquad s_i^{(l)} = \left(w_i^{(l)}\right)^T x^{(l-1)}$$

$$\frac{\partial s_i^{(l)}}{\partial w_i^{(l)}} = x^{(l-1)}$$

$$\therefore \frac{\partial e}{\partial w_i^{(l)}} = x^{(l-1)} \times \frac{\partial e}{\partial s_i^{(l)}}$$

$$\frac{\partial e}{\partial w^{(l)}} = \left[x^{(l-1)} \frac{\partial e}{\partial s_1^{(l)}} \quad x^{(l-1)} \frac{\partial e}{\partial s_2^{(l)}} \quad \cdots \quad x^{(l-1)} \frac{\partial e}{\partial s_{d^{(l)}}^{(l)}} \right]$$

$$\frac{\partial e}{\partial w^{(l)}} = \underset{\text{previous layer.}}{x^{(l-1)}} \left(\frac{\partial e}{\partial s^{(l)}} \right)^T \qquad \left(\delta^{(l)}\right)^T \rightarrow \text{sensitivity}$$

vector of derivatives

$$\frac{\partial e}{\partial s^{(l)}}$$

$$\because \frac{\partial e}{\partial s_i^{(l)}} = \underbrace{\frac{\partial e}{\partial x_i^{(l)}} \times \frac{\partial x_i^{(l)}}{\partial s_i^{(l)}}}_{} \qquad x_i^{(l)} = \theta\left(s_i^{(l)}\right)$$

$$\therefore \frac{\partial e}{\partial s_i^{(l)}} = \theta'\left(s_i^{(l)}\right) \cdot \frac{\partial e}{\partial x_i^{(l)}}$$

$$\frac{\partial e}{\partial s^{(l)}} = \theta'\left(s^{(l)}\right) \otimes \left[\frac{\partial e}{\partial x^{(l)}}\right]_1^{d^{(l)}}$$

$$\frac{\partial e}{\partial x^{(l)}} = \sum_{k=1}^{d^{(l+1)}} \frac{\partial e}{\partial s_k^{(l+1)}} \cdot \frac{\partial s_k}{\partial x^{(l)}}$$

$$\because s_k^{(l+1)} = \left(W_k^{l+1}\right)^T x^{(l)}$$

$$\frac{\partial e}{\partial x^{(l)}} = \sum_{k=1}^{d^{(l+1)}} W_k^{(l+1)} \frac{\partial e}{\partial s^{(l+1)}}$$

$$\frac{\partial e}{\partial x^{(l)}} = \underline{\underline{W^{(l+1)} \, \delta^{(l+1)}}} \leftarrow \text{sensitivity}$$

$$\frac{\partial e}{\partial s^{(l)}} = \underline{\theta'(s^{(l)}) \otimes \left[W^{(l+1)} \delta^{(l+1)} \right]_1^{d^{(l)}}}$$

Established a relation between sensitivities across layers.

sensitivity $l \to \delta^{(l)}$ $\quad l+1 \longrightarrow l+2 \longrightarrow \ldots \ldots \delta^L$

$\uparrow$ compute

$\downarrow$

δ^1

1) $\dfrac{\partial e}{\partial W^{(l)}} = x^{(l-1)} (\delta^{l})^{T}$ ✓

$(1+d^{(l-1)}) \times d^{l}$

2) $\delta^{(l)} = \theta'(s^{(l)}) \otimes [W^{(l+1)} \delta^{(l+1)}]_{1}^{d^{(l)}}$

STOCHASTIC GRADIENT DESCENT

1) Pick x_n

2) $x_n = x^{(0)} \xrightarrow{W^{(1)}} s^{(1)} \xrightarrow{\theta} x^{(1)} \longrightarrow s^{(2)} \xrightarrow{\theta} x^{(2)} \cdots\cdots \longrightarrow x^{(L)} = h(x_n)$

$e(h(x_n), y_n)$

3) Run BACKPROPOGATION

$$\delta^{(L)} = \frac{\partial e}{\partial s^{(L)}}, \quad e = (x^{L} - y_n)^2 \qquad x_n \, y_n$$

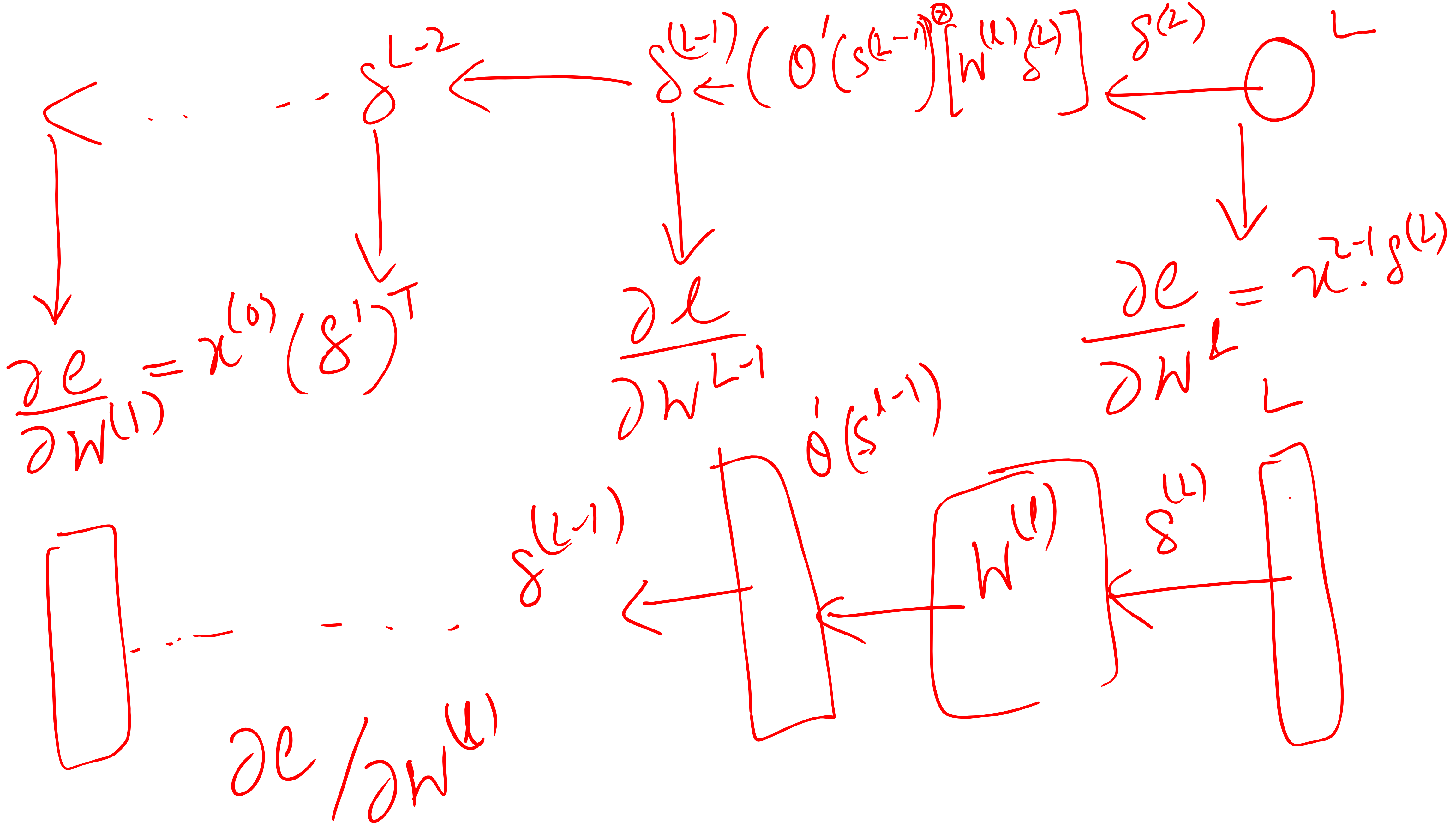
$$\delta^{(L)} = \frac{\partial (x^{(L)} - y_n)^2}{\partial s^{(L)}} = 2(x^{(L)} - y_n) \cdot \frac{\partial x^{(L)}}{\partial s^{(L)}}$$

$$= 2(x^{(L)} - y_n) \cdot \theta'(s^{(L)})$$

$$\delta^{(L)} = 2(x^{(L)} - y_n)(1 - \tanh^2(s^{(L)}))$$

$$= 2(x^{(L)} - y_n)(1 - (x^{(L)})^2)$$

$$\frac{\partial e}{\partial (w^{(l)})} = x^{(L-1)} \cdot \delta^{(L)}$$



δ^{L-2}
$\delta^{(L-1)} \leftarrow (\theta'(s^{(L-1)}) \otimes [w^{(L)}\delta^{(L)}]$
$\delta^{(L)}$
L
$\frac{\partial e}{\partial w^{(1)}} = x^{(0)}(\delta^{1})^{T}$
$\frac{\partial e}{\partial w^{L-1}}$
$\frac{\partial e}{\partial w^{L}} = x^{L-1} \cdot \delta^{(L)}$
L
$\theta'(s^{L-1})$
$\delta^{(L-1)}$
$w^{(l)}$
$\delta^{(L)}$
$\partial e / \partial w^{(l)}$

Update Step

$$W^{(l)} \leftarrow W^{(l)} - \eta \frac{\partial e}{\partial W^{(l)}}$$ ✓

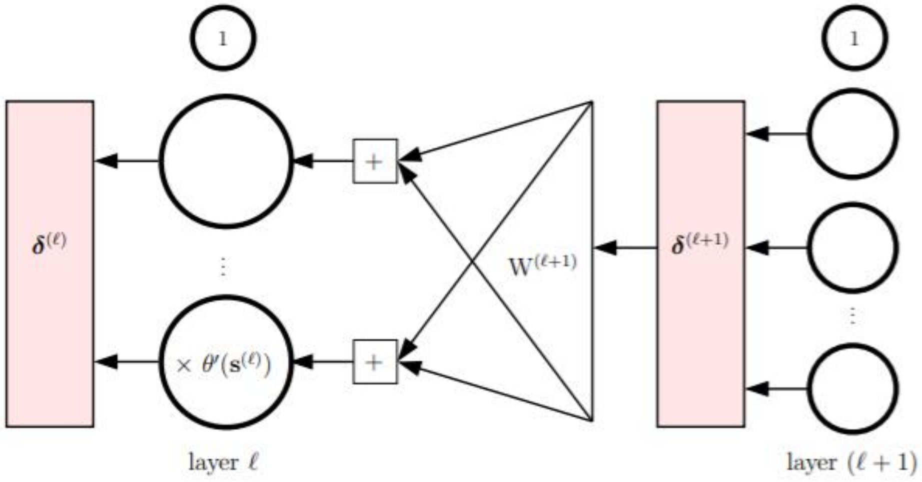
SGD, Batch grad. descent.

Computing $\boldsymbol{\delta}^{(\ell)}$ Using the Chain Rule

$$\boldsymbol{\delta}^{(1)} \longleftarrow \boldsymbol{\delta}^{(2)} \cdots \longleftarrow \boldsymbol{\delta}^{(L-1)} \longleftarrow \boldsymbol{\delta}^{(L)}$$

Multiple applications of the chain rule:

$$\Delta \mathbf{s}^{(\ell)} \xrightarrow{\theta} \Delta \mathbf{x}^{(\ell)} \xrightarrow{\mathrm{W}^{(\ell+1)}} \Delta \mathbf{s}^{(\ell+1)} \cdots \longrightarrow \Delta \mathrm{e}(\mathbf{x})$$



don't use 0th component (bias)

$$\boldsymbol{\delta}^{(\ell)} = \theta'(\mathbf{s}^{(\ell)}) \otimes [\mathrm{W}^{(\ell+1)}\boldsymbol{\delta}^{(\ell+1)}]_1^{d^{(\ell)}}$$

componentwise multiplication

The Backpropagation Algorithm

$$\boldsymbol{\delta}^{(1)} \longleftarrow \boldsymbol{\delta}^{(2)} \cdots \longleftarrow \boldsymbol{\delta}^{(L-1)} \longleftarrow \boldsymbol{\delta}^{(L)}$$

Backpropagation to compute sensitivities $\boldsymbol{\delta}^{(\ell)}$:
(Assume $\mathbf{s}^{(\ell)}$ and $\mathbf{x}^{(\ell)}$ have been computed for all ℓ)

$\delta^{(L)} \leftarrow 2(x^{(L)} - y) \cdot \theta'(s^{(L)})$ [Initialization]

for $\ell = L-1$ to 1 **do** [Back-Propagation]

Compute (for tanh hidden node):

$$\theta'(\mathbf{s}^{(\ell)}) = \left[1 - \mathbf{x}^{(\ell)} \otimes \mathbf{x}^{(\ell)}\right]_1^{d^{(\ell)}}$$

$$\boldsymbol{\delta}^{(\ell)} \leftarrow \theta'(\mathbf{s}^{(\ell)}) \otimes \left[\mathrm{W}^{(\ell+1)}\boldsymbol{\delta}^{(\ell+1)}\right]_1^{d^{(\ell)}}$$

$\longleftarrow$ componentwise multiplication

end for

Algorithm for Gradient Descent on E_{in}

Algorithm to Compute $E_{\text{in}}(\mathbf{w})$ and $\mathbf{g} = \nabla E_{\text{in}}(\mathbf{w})$:
Input: weights $\mathbf{w} = \{\mathrm{W}^{(1)}, \ldots, \mathrm{W}^{(L)}\}$; data $\mathcal{D}$.
Output: error $E_{\text{in}}(\mathbf{w})$ and gradient $\mathbf{g} = \{\mathrm{G}^{(1)}, \ldots, \mathrm{G}^{(L)}\}$.

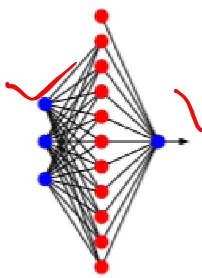
Initialize: $E_{\text{in}} = 0$; for $\ell = 1, \ldots, L$, $\mathrm{G}^{(\ell)} = 0 \cdot \mathrm{W}^{(\ell)}$.

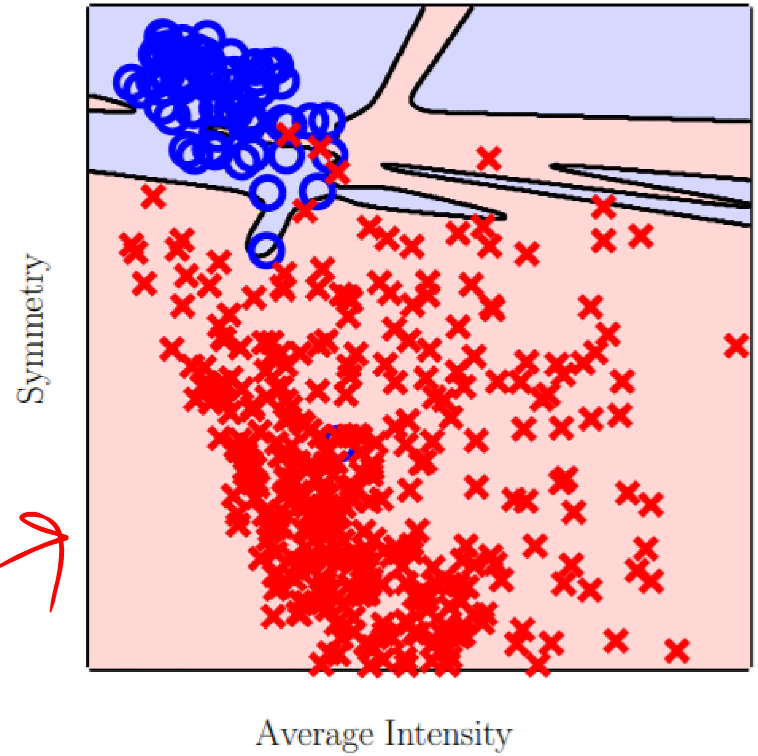
for Each data point $\mathbf{x}_n$ $(n = 1, \ldots, N)$ **do**
  Compute $\mathbf{x}^{(\ell)}$ for $\ell = 0, \ldots, L$. [forward propagation]
  Compute $\boldsymbol{\delta}^{(\ell)}$ for $\ell = 1, \ldots, L$. [backpropagation]
  $E_{\text{in}} \leftarrow E_{\text{in}} + \frac{1}{N}(\mathbf{x}_1^{(L)} - y_n)^2$.
  for $\ell = 1, \ldots, L$ **do**
    $\mathrm{G}^{(\ell)}(\mathbf{x}_n) = [\mathbf{x}^{(\ell-1)}(\boldsymbol{\delta}^{(\ell)})^{\mathrm{T}}]$
    $\mathrm{G}^{(\ell)} \leftarrow \mathrm{G}^{(\ell)} + \frac{1}{N}\mathrm{G}^{(\ell)}(\mathbf{x}_n)$.
  end for
end for

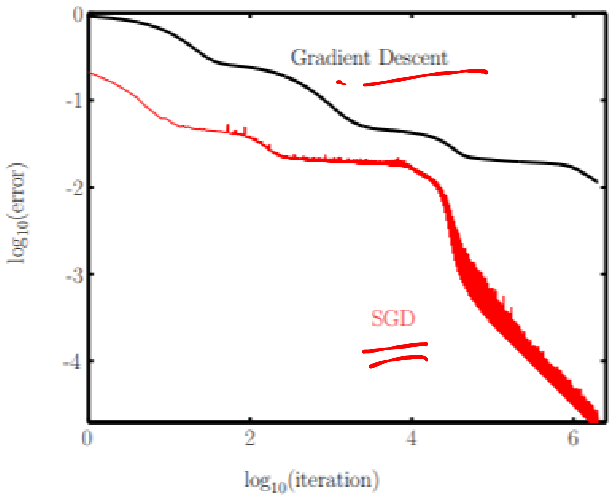
Can do batch version or sequential version (SGD).

Digits Data

Overfitting







Thanks!