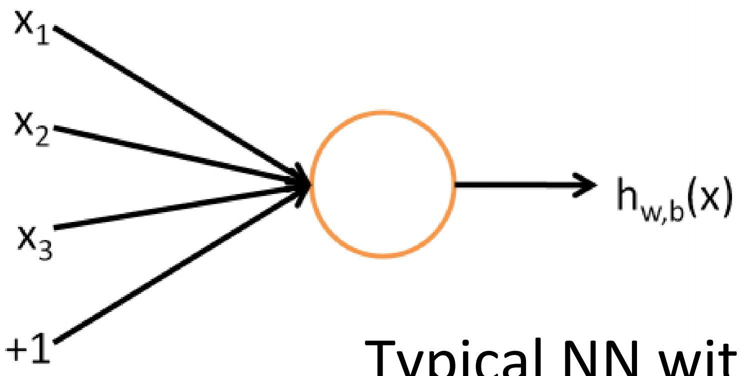
Back Propagation Algorithm for MLP

EE 5179

Instructor: Kaushik Mitra

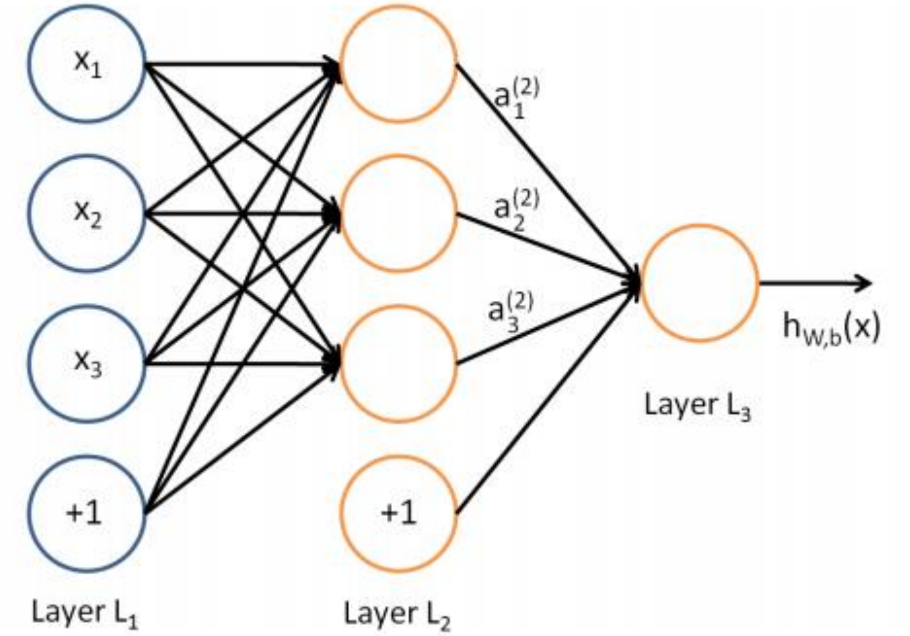
How to learn the parameters?



Bias

$$h_{W,b}(x) = f(W^T x) = f\left(\sum_{i=1}^{3} W_i x_i + b\right)$$

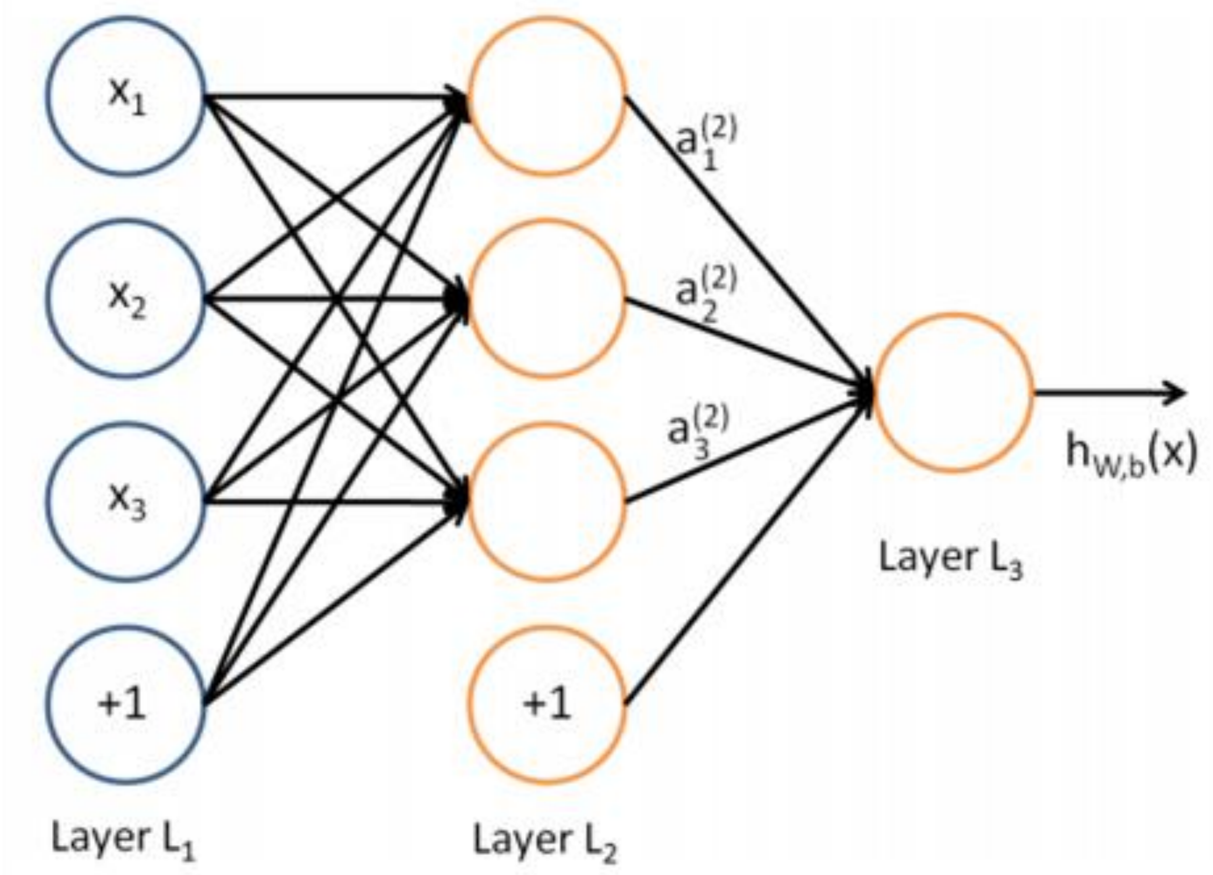
Typical NN with many such units



- One hidden layer
  - 3 neuron units
- One output

*Slide courtesy, [sparse autoencoder by Andrew Ng]

How to learn the parameters?



L_l – Layer l

$a_i^{(l)}$ – activation of unit i in layer l

$W_{ij}^{(l)}$ – Weight from j^{th} unit in l to i^{th} unit in $l+1$

$b_i^{(l)}$ – bias to unit i in layer $l+1$

Parameters:

$$(W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)})$$

$$W^{(1)} \in \mathbb{R}^{3\times3}, W^{(2)} \in \mathbb{R}^{1\times3}$$

*Slide courtesy, [sparse autoencoder by Andrew Ng]

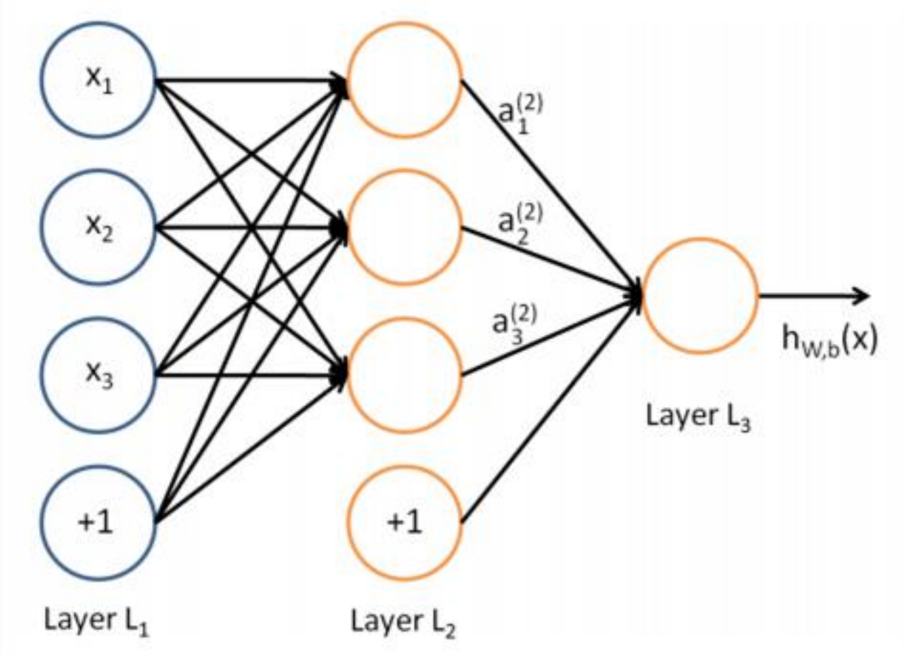
How to learn the parameters?

Layer 2,

$$
\begin{aligned}
a_1^{(2)} &= f(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_1^{(1)}) \\
a_2^{(2)} &= f(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_2^{(1)}) \\
a_3^{(2)} &= f(W_{31}^{(1)}x_1 + W_{32}^{(1)}x_2 + W_{33}^{(1)}x_3 + b_3^{(1)})
\end{aligned}
$$

Layer 3,

$$h_{W,b}(x) = a_1^{(3)} = f(W_{11}^{(2)}a_1^{(2)} + W_{12}^{(2)}a_2^{(2)} + W_{13}^{(2)}a_3^{(2)} + b_1^{(2)})$$



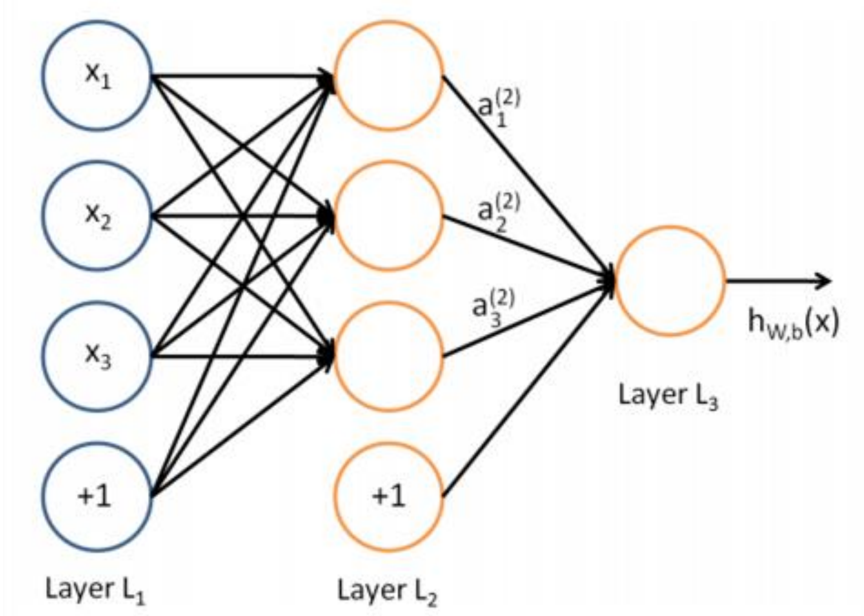
Let, $z_i^{(l)}$ denote weighted sum for the $a_i^{(l)}$ activation

$$
\begin{aligned}
z^{(2)} &= W^{(1)}x + b^{(1)} \\
a^{(2)} &= f(z^{(2)}) \\
z^{(3)} &= W^{(2)}a^{(2)} + b^{(2)} \\
h_{W,b}(x) &= a^{(3)} = f(z^{(3)})
\end{aligned}
$$

$$
\begin{aligned}
z^{(l+1)} &= W^{(l)}a^{(l)} + b^{(l)} \\
a^{(l+1)} &= f(z^{(l+1)})
\end{aligned}
$$

*Slide courtesy, [sparse autoencoder by Andrew Ng]

How to learn the parameters?



Given *m* training examples

$$\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$$

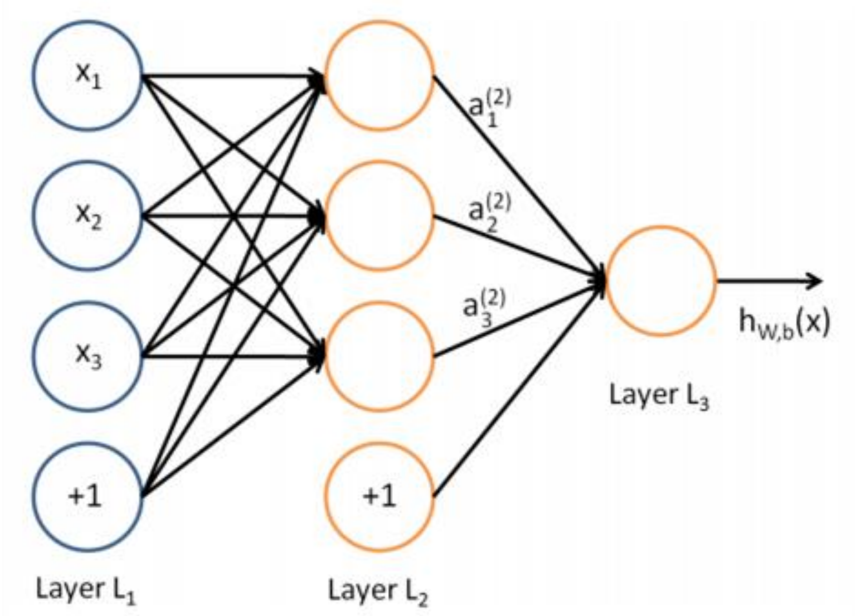
Minimize:

$$J(W,b;x,y) = \frac{1}{2}\left\|h_{W,b}(x) - y\right\|^2$$

$$\begin{aligned} J(W,b) &= \left[\frac{1}{m}\sum_{i=1}^{m} J(W,b;x^{(i)},y^{(i)})\right] \\ &= \left[\frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\left\|h_{W,b}(x^{(i)}) - y^{(i)}\right\|^2\right)\right] \end{aligned}$$

*Slide courtesy, [sparse autoencoder by Andrew Ng]

How to learn the parameters?



Minimize:

$$J(W,b;x,y) = \frac{1}{2} \left\| h_{W,b}(x) - y \right\|^2$$

Gradient descent:

$$W_{ij}^{(l)} := W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W,b)$$

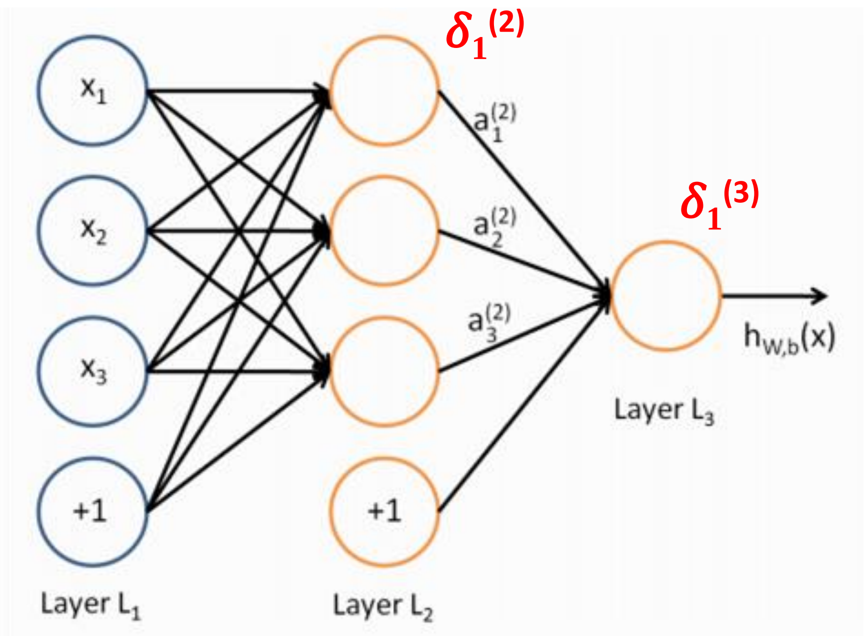
$$b_i^{(l)} := b_i^{(l)} - \alpha \frac{\partial}{\partial b_i^{(l)}} J(W,b)$$

How to evaluate these partial derivatives?

Error back-propagation

*Slide courtesy, [sparse autoencoder by Andrew Ng]

Back-propagation algorithm



Gradient descent:

$$W_{ij}^{(l)} := W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W, b)$$

Idea:

First, forward pass the data to calc. all responses

In backward pass, for each unit i in layer l calculate **error** term $\delta_i^{(l)}$ - measures how much unit i is responsible for output error

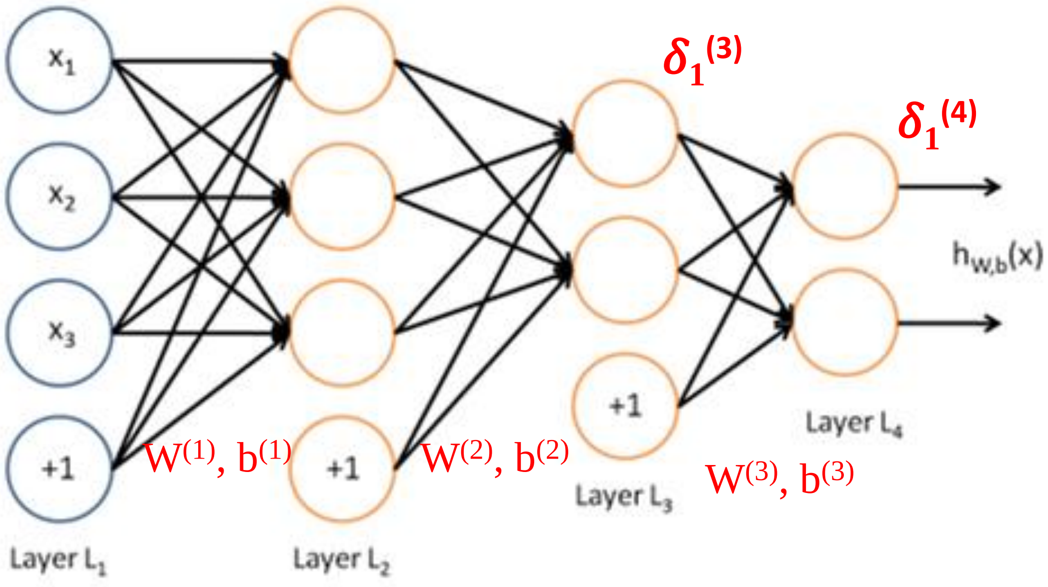
- For output unit in last layer (n_l), this is easy

$$\delta_i^{(n_l)} = \frac{\partial}{\partial z_i^{(n_l)}} \frac{1}{2} \|y - h_{W,b}(x)\|^2 = -(y_i - a_i^{(n_l)}) \cdot f'(z_i^{(n_l)})$$

- How to measure $\delta_i^{(l)}$ for hidden units?

*Slide courtesy, sparse autoencoder by Andrew Ng

Back-propagation algorithm



Gradient descent:

$$W_{ij}^{(l)} := W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W,b)$$

$$J(W,b;x,y) = \frac{1}{2} \left\| h_{W,b}(x) - y \right\|^2$$

$$h_{W,b}(x) = a^{(4)} = f(z^{(4)}); \; z^{(4)} = W^{(3)} a^{(3)} + b^{(3)}$$

For last layer:

$$\frac{\partial J}{\partial W_{ij}^{(3)}} = \boxed{\frac{\partial J}{\partial z_i^4}} \boxed{\frac{\partial z_i^4}{\partial W_{ij}^{(3)}}}$$

$$\boxed{\frac{\partial J}{\partial z_i{}^4} = -(y_i - a_i^{(4)}) \;.f'(z_i^4)} \qquad \boxed{\frac{\partial z_i^4}{\partial W_{ij}^3} = a_j^{(3)}}$$

$\delta_i^{(4)}$ error term

$$\frac{\partial J}{\partial W_{ij}^{(l)}} = \delta_i^{(l+1)} a_j^{(l)} \qquad \frac{\partial J}{\partial b_i^{(l)}} = \delta_i^{(l+1)}$$

*Slide courtesy, [sparse autoencoder by Andrew Ng]

Back-propagation algorithm

Gradient descent:

$$W_{ij}^{(l)} \quad := \quad W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W,b)$$

$$J(W,b;x,y) = \frac{1}{2}\left\| h_{W,b}(x) - y \right\|^2$$

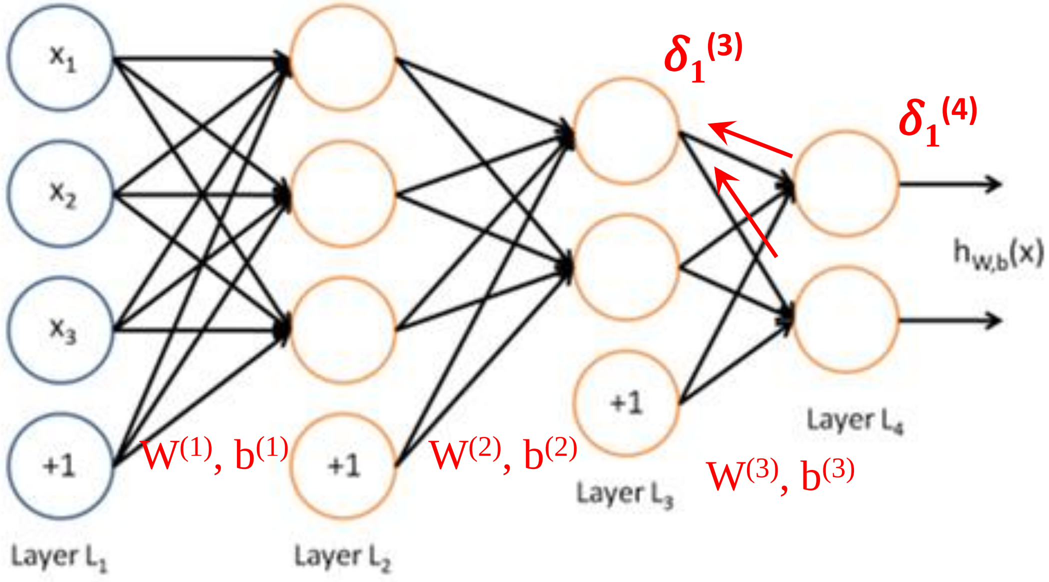
For layers other than last:

$$\frac{\partial J}{\partial W_{ij}^{(2)}} = \boxed{\frac{\partial J}{\partial z_i^{(3)}}}\ \boxed{\frac{\partial z_i^{(3)}}{\partial W_{ij}^{(2)}}}$$

$a_j^{(2)}$

$$\delta_i^{(l)} = \left(\sum_{j=1}^{s_{l+1}} W_{ji}^{(l)} \delta_j^{(l+1)} \right) f'(z_i^{(l)})$$

$$\frac{\partial J}{\partial W_{ij}^{(l)}} = \delta_i^{(l+1)} a_j^{(l)} \qquad \frac{\partial J}{\partial b_i^{(l)}} = \delta_i^{(l+1)}$$



$$h_{W,b}(x) = a^{(4)} = f(z^{(4)}); \; z^{(4)} = W^{(3)}a^{(3)} + b^{(3)}$$

$$a^{(3)} = f(z^{(3)}); \; z^{(3)} = W^{(2)}a^{(2)} + b^{(2)}$$

$\delta_i^{(3)}$
error term

$$\boxed{\begin{aligned} \frac{\partial J}{\partial z_i^{(3)}} &= \frac{\partial J}{\partial a_i^{(3)}} \frac{\partial a_i^{(3)}}{\partial z_i^{(3)}} \\ &= \left(\sum_j \frac{\partial J}{\partial z_j^{(4)}} \frac{\partial z_j^{(4)}}{\partial a_i^{(3)}} \right) f'(z_i^{(3)}) \end{aligned}}$$

$\delta_j^{(4)}$
Layer - $(l+1)$

$W_{ji}^{(3)}$

*Slide courtesy, sparse autoencoder by Andrew Ng

Back-propagation algorithm

1. Perform a feedforward pass
   - Computing activations L_1, L_2 and so on ...

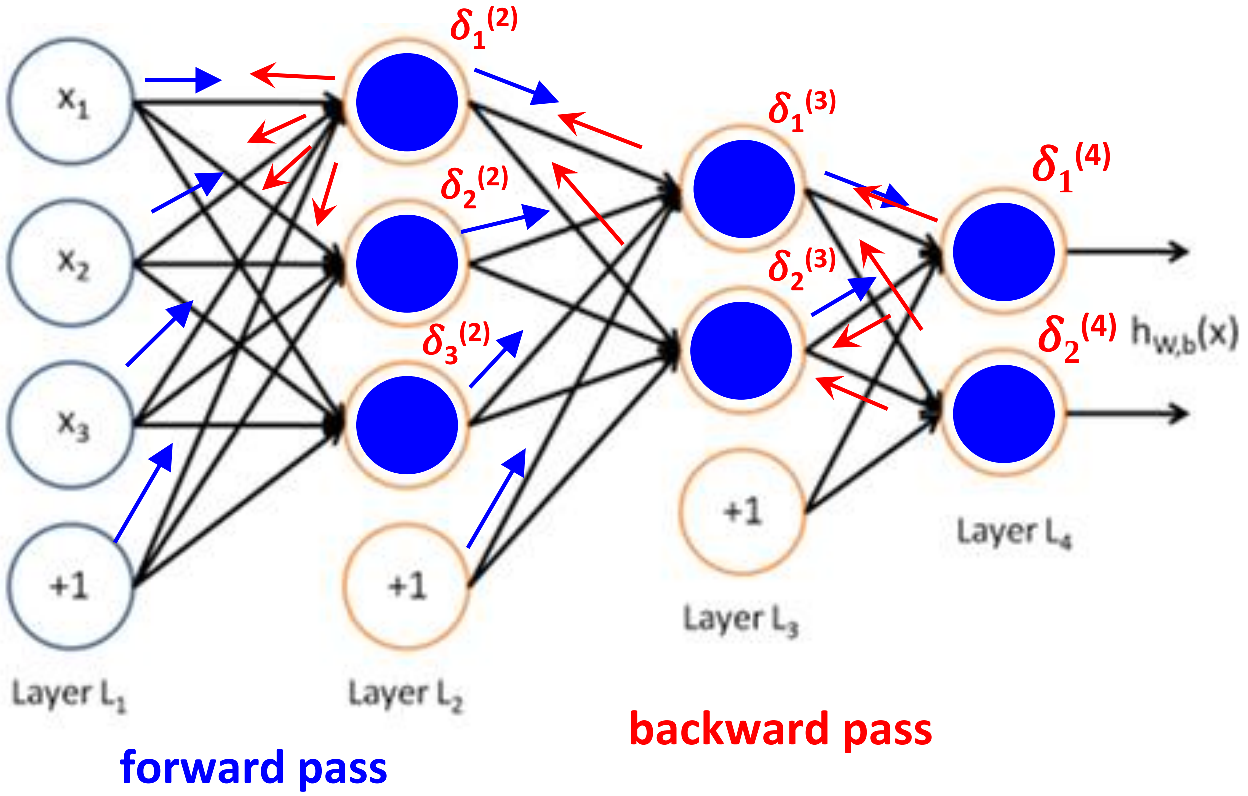
2. For each output unit i in layer L_4 (output layer), set

$$\delta_i^{(n_l)} = \frac{\partial}{\partial z_i^{(n_l)}} \frac{1}{2} \|y - h_{W,b}(x)\|^2 = -(y_i - a_i^{(n_l)}) \cdot f'(z_i^{(n_l)})$$

3. Starting from last but one layer to 2nd layer;
   $l = n_l\text{-}1, n_l\text{-}2, \ldots\ldots, 2$
   - For each node i in layer l, set $\delta_i^{(l)} = \left(\sum_{j=1}^{s_{l+1}} W_{ji}^{(l)} \delta_j^{(l+1)} \right) f'(z_i^{(l)})$

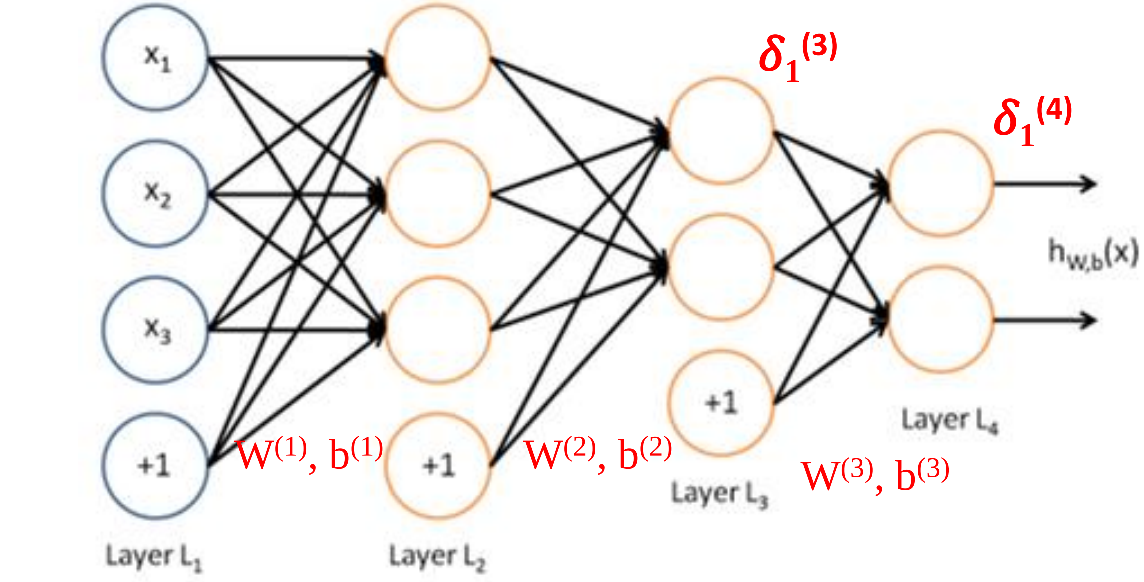
4. Compute the desired partial derivatives, as:

$$\frac{\partial}{\partial W_{ij}^{(l)}} J(W, b; x, y) = a_j^{(l)} \delta_i^{(l+1)} \qquad \frac{\partial}{\partial b_i^{(l)}} J(W, b; x, y) = \delta_i^{(l+1)}.$$



*Slide courtesy, sparse autoencoder by Andrew Ng

Back-propagation algorithm



$$h_{W,b}(x) = a^{(4)} = f(z^{(4)}); \; z^{(4)} = W^{(3)}a^{(3)} + b^{(3)}$$

Gradient descent:

$$W_{ij}^{(l)} := W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W,b)$$

$$J(W,b;x,y) = \frac{1}{2} \left\| h_{W,b}(x) - y \right\|^2$$

Partial derivatives:

$$\delta_i^{(l)} = \left(\sum_{j=1}^{s_{l+1}} W_{ji}^{(l)} \delta_j^{(l+1)} \right) f'(z_i^{(l)})$$

$$\frac{\partial J}{\partial W_{ij}^{(l)}} = \delta_i^{(l+1)} a_j^{(l)} \qquad \frac{\partial J}{\partial b_i^{(l)}} = \delta_i^{(l+1)}$$

Matrix notation:

$$\delta^{(l)} = \left((W^{(l)})^T \delta^{(l+1)} \right) \bullet f'(z^{(l)})$$

$$\frac{\partial J}{\partial W^{(l)}} = \delta^{(l+1)} (a^{(l)})^T \qquad \frac{\partial J}{\partial b^{(l)}} = \delta^{(l+1)}$$

*Slide courtesy, sparse autoencoder by Andrew Ng

Back-propagation algorithm

1. Perform a feedforward pass
   - Computing activations L_1, L_2 and so on ...

2. For each output unit i in layer L_4 (output layer), set

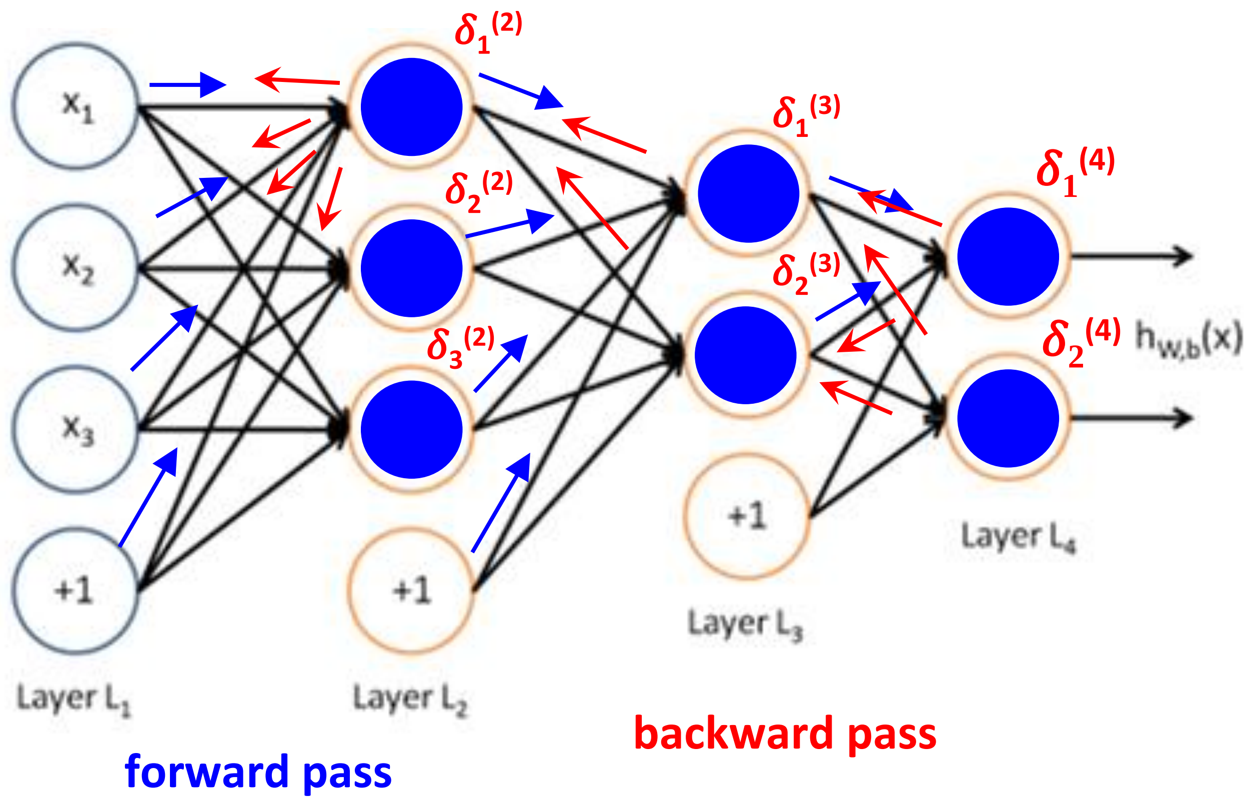
$$\delta^{(n_l)} = -(y - a^{(n_l)}) \bullet f'(z^{(n)})$$

3. Starting from last but one layer to 2nd layer;
   $l = n_l\text{-}1, n_l\text{-}2, \ldots.., 2$

$$\delta^{(l)} = \left((W^{(l)})^T \delta^{(l+1)}\right) \bullet f'(z^{(l)})$$

4. Compute the desired partial derivatives, as:

$$\begin{aligned} \nabla_{W^{(l)}} J(W,b;x,y) &= \delta^{(l+1)} (a^{(l)})^T, \\ \nabla_{b^{(l)}} J(W,b;x,y) &= \delta^{(l+1)}. \end{aligned}$$



*Slide courtesy, sparse autoencoder by Andrew Ng

END