Do not distribute course material

Topic 4 cont
Neural Networks

Outline

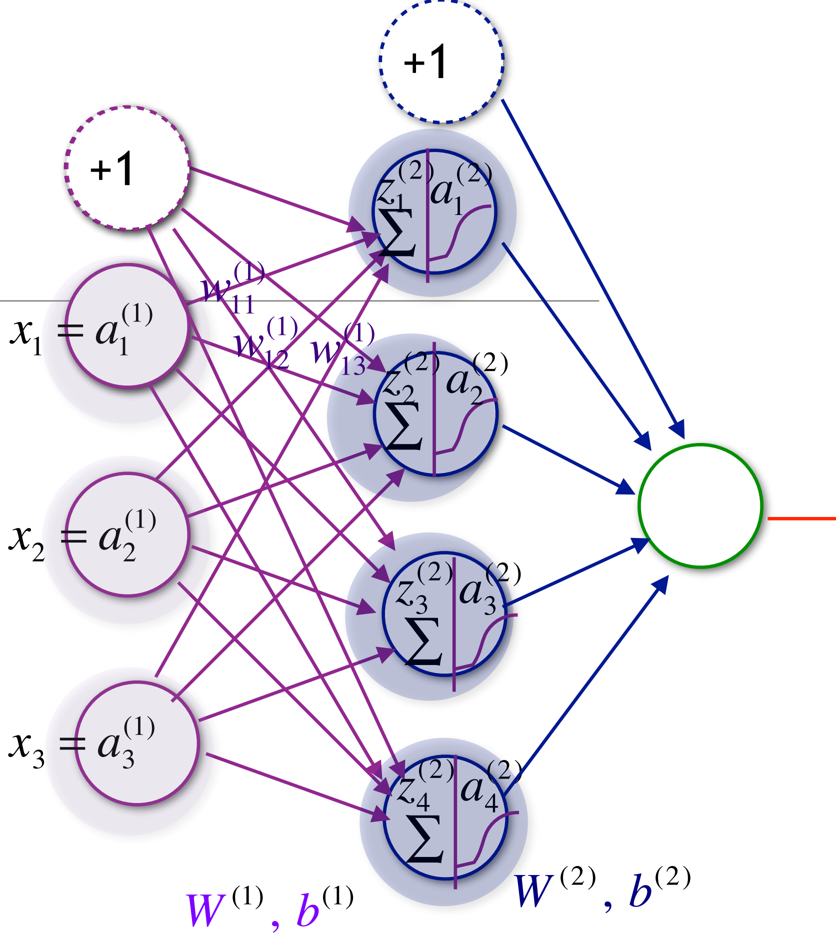
- ❑ Introduction to neurons
- ❑ Nonlinear classifiers from linear features
- ❑ Neural networks notation
- ❑ Pseudocode for prediction
- ❑ Training a neural network
- ❑ Implementing gradient descent for neural networks
  - Vectorization
  - Pseudocode
- ❑ Preprocessing
- ❑ Initialization
- ❑ Activations

$f(z)=\frac{1}{1+\exp(-z)}$

Forward Propagation

$$z^{(l+1)} = W^{(l)}a^{(l)} + b^{(l)} \qquad a^{(l+1)} = f(z^{(l+1)}) = f(W^{(l)}a^{(l)} + b^{(l)})$$

Generalizing for an arbitrary neural network. Forward propagation:



$\mathbf{a}^{(1)} = \mathbf{x}$

for ℓ = 1 to n_ℓ-1 do

$$\mathbf{z}^{(\ell+1)} = W^{(\ell)}\mathbf{a}^{(\ell)} + \mathbf{b}^{(\ell)} = z^{(2)} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \\ W_{31}^{(1)} & W_{32}^{(1)} & W_{33}^{(1)} \\ W_{41}^{(1)} & W_{42}^{(1)} & W_{43}^{(1)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1^{(1)} \\ b_2^{(1)} \\ b_3^{(1)} \\ b_4^{(1)} \end{bmatrix} = \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ z_3^{(2)} \\ z_4^{(2)} \end{bmatrix}$$

$$\mathbf{a}^{(\ell+1)} = f(\mathbf{z}^{(\ell+1)})$$

$$\hat{\mathbf{y}} = \mathbf{a}^{(n_\ell)}$$

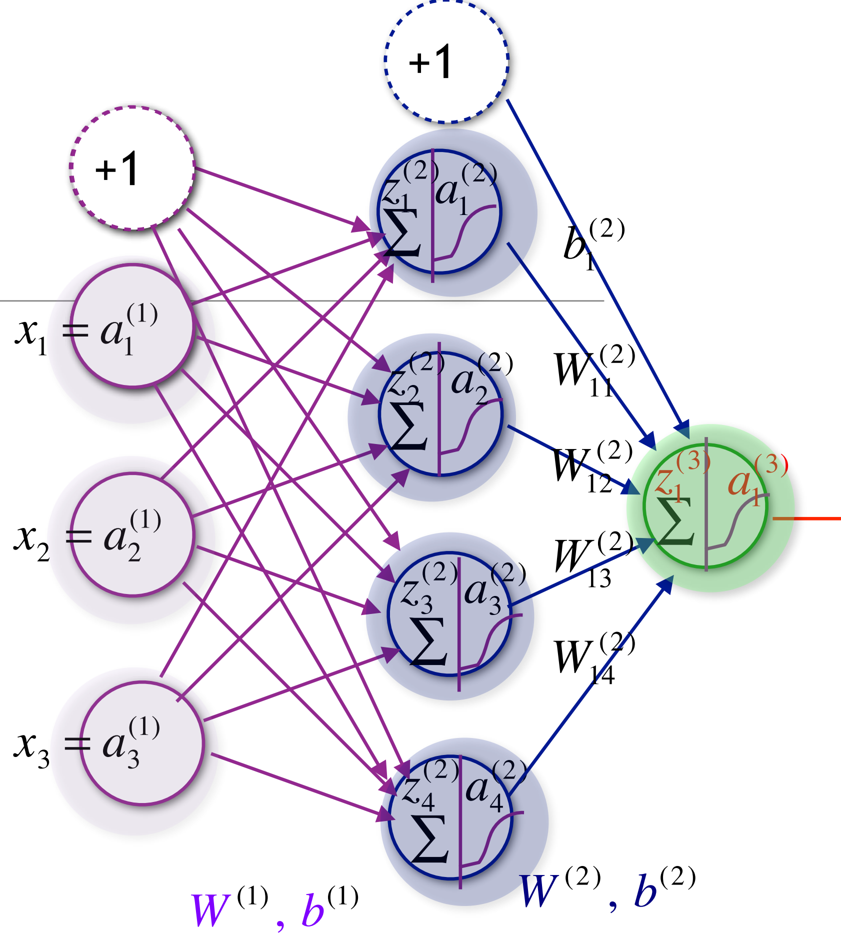
$$a^{(2)} = \begin{bmatrix} f(z_1^{(2)}) \\ f(z_2^{(2)}) \\ f(z_3^{(2)}) \\ f(z_4^{(2)}) \end{bmatrix} = \begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \\ a_3^{(2)} \\ a_4^{(2)} \end{bmatrix}$$

$f(z) = \frac{1}{1+\exp(-z)}$

Forward Propagation

$\mathbf{z}^{(\ell+1)} = W^{(\ell)}\mathbf{a}^{(\ell)} + \mathbf{b}^{(\ell)}$ $\quad$ $\mathbf{a}^{(\ell+1)} = f(\mathbf{z}^{(\ell+1)}) = f(W^{(\ell)}\mathbf{a}^{(\ell)} + \mathbf{b}^{(\ell)})$

Generalizing for an arbitrary neural network. Forward propagation:



$\mathbf{a}^{(1)} = \mathbf{x}$

for ℓ = 1 to n_ℓ-1 do

$\mathbf{z}^{(\ell+1)} = W^{(\ell)}\mathbf{a}^{(\ell)} + \mathbf{b}^{(\ell)}$

$\mathbf{a}^{(\ell+1)} = f(\mathbf{z}^{(\ell+1)})$

$\hat{\mathbf{y}} = \mathbf{a}^{(n_\ell)}$

$$\mathbf{z}^{(3)} = \begin{bmatrix} W_{11}^{(2)} & W_{12}^{(2)} & W_{13}^{(2)} & W_{14}^{(2)} \end{bmatrix} \begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \\ a_3^{(2)} \\ a_4^{(2)} \end{bmatrix} + \begin{bmatrix} b_1^{(2)} \end{bmatrix} = \begin{bmatrix} z_1^{(3)} \end{bmatrix}$$

$$\mathbf{a}^{(3)} = [f(\mathbf{z}^{(3)})] = \begin{bmatrix} a_1^{(3)} \end{bmatrix}$$

$$a^{(3)} = f(W^{(2)} f(W^{(1)} a^{(1)} + b^{(1)}) + b^{(2)})$$

Outline

- ❑ Introduction to neurons
- ❑ Nonlinear classifiers from linear features
- ❑ Neural networks notation
- ❑ Pseudocode for prediction
- ❑ Training a neural network
- ❑ Implementing gradient descent for neural networks
  - Vectorization
  - Pseudocode
- ❑ Preprocessing
- ❑ Initialization
- ❑ Activations

Training a neural network

- The algorithm
- The problem
- The clever strategy
- Carrying out the strategy step by step
  - Last layer
  - Non-last layer
  - Putting the pieces together
- Partial derivative with regularization and different activation functions

Stochastic gradient descent algorithm

Randomly initialize the biases and weights for each layer: $b^{(\ell)}$ $W^{(\ell)}$

While iterations < iteration limit:

Randomly choose a training example: $(\mathbf{x}, y)$

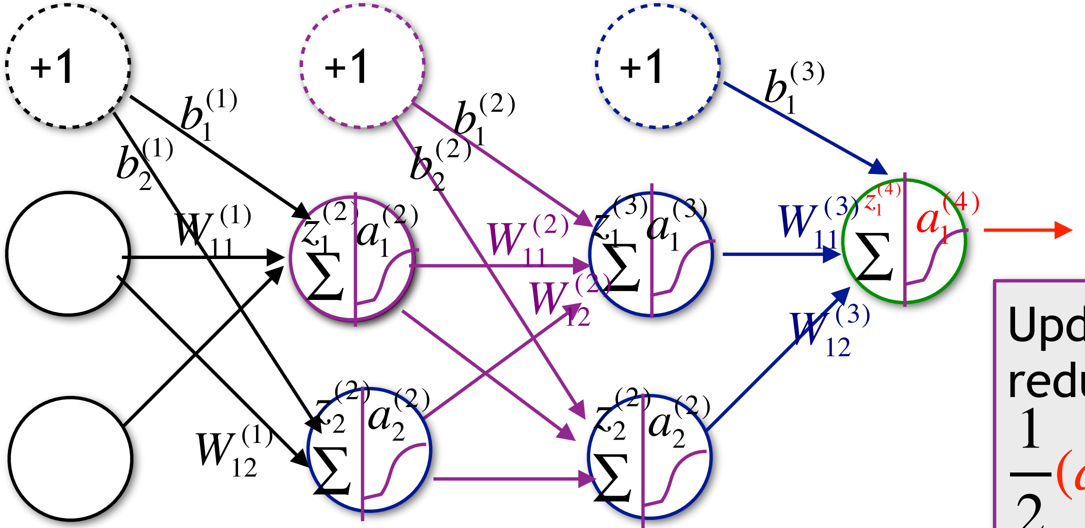
Run forward propagation and save for each level, $a^{(\ell)}$ and $z^{(\ell)}$

Run back propagation to compute the partial derivatives: $\nabla_{b^{(\ell)}} J(W, b, \mathbf{x}, y)$, $\nabla_{W^{(\ell)}} J(W, b, \mathbf{x}, y)$

Perform a gradient descent step for $\ell \in \{1 \ldots n_\ell\}$:

$$W^{(\ell)} = W^{(\ell)} - \alpha \nabla_{W^{(\ell)}} J(W, b, \mathbf{x}, y)$$

$$b^{(\ell)} = b^{(\ell)} - \alpha \nabla_{b^{(\ell)}} J(W, b, \mathbf{x}, y)$$



Update weight to reduce error:
$$\frac{1}{2}(a_1^{(4)} - y)^2$$

How do we compute $\nabla_{W^{(\ell)}} J(W, b, \mathbf{x}, y)$?

The gradient descent algorithm depends on the error function and the structure of the network.

$$J(W, \mathbf{b}, \mathbf{x}, y) = \frac{1}{2}(a^{(4)} - y)^2 = \frac{1}{2}((f(W^{(3)}f(W^{(2)}f(W^{(1)}a^{(1)} + b^{(1)}) + b^{(2)}) + b^{(3)}) - y)^2$$

Cost function:

$$J(W, \mathbf{b}, \mathbf{x}, y) = \frac{1}{2}(a^{(4)} - y)^2 = \frac{1}{2}((f(W^{(3)}f(W^{(2)}f(W^{(1)}a^{(1)} + b^{(1)}) + b^{(2)}) + b^{(3)}) - y)^2$$

Update rule (local optimization):

$$W^{(\ell)} = W^{(\ell)} - \alpha \nabla_{W^{(\ell)}} J(W, b, \mathbf{x}, y)$$

$$b^{(\ell)} = b^{(\ell)} - \alpha \nabla_{b^{(\ell)}} J(W, b, \mathbf{x}, y)$$

$$W_{i,j}^{(\ell)} = W_{i,j}^{(\ell)} - \alpha \frac{\partial J(W, b, \mathbf{x}, y)}{\partial W_{i,j}^{(\ell)}}$$

$$b_i^{(\ell)} = b_i^{(\ell)} - \alpha \frac{J(W, b, \mathbf{x}, y)}{b_i^{(\ell)}}$$

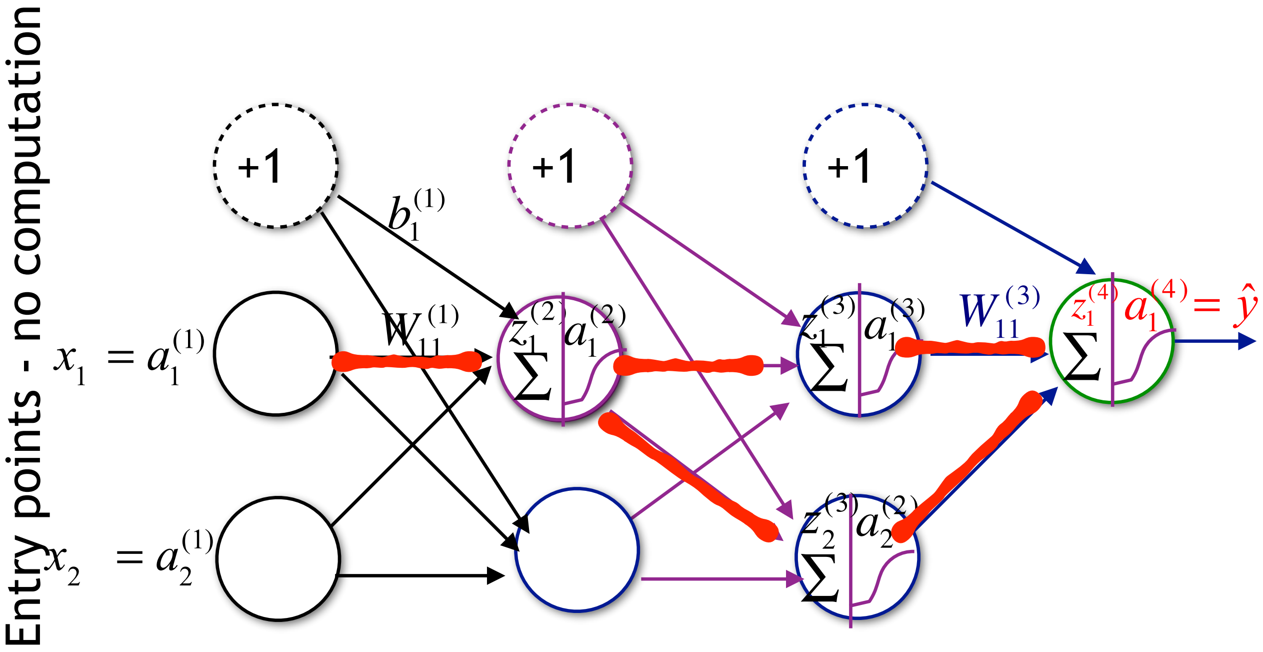
Pair share: compute $\frac{\partial J}{\partial W_{11}^{(1)}}$

Training a neural network

- The algorithm
- The problem
- The clever strategy
- Carrying out the strategy step by step
  - Last layer
  - Non-last layer
  - Putting the pieces together
- Partial derivative with regularization and different activation functions

It is a mess

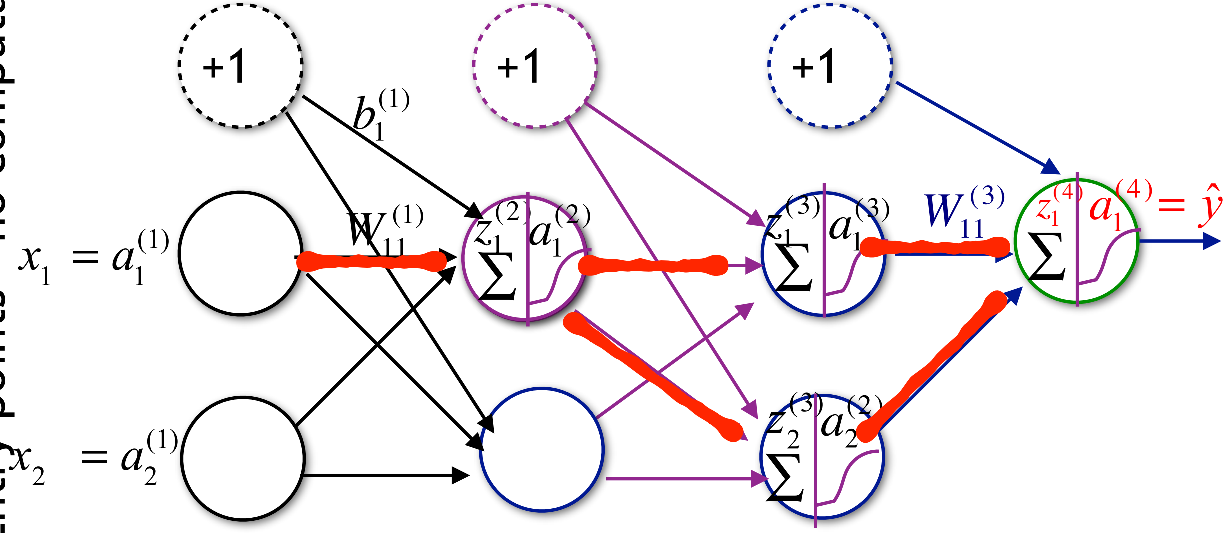
$$J(W, \mathbf{b}, \mathbf{x}, y) = \frac{1}{2}(a^{(4)} - y)^2 = \frac{1}{2}((f(W^{(3)} f(W^{(2)} f(W^{(1)} a^{(1)} + b^{(1)}) + b^{(2)}) + b^{(3)}) - y)^2$$



Pair share: compute $\dfrac{\partial J}{\partial W_{11}^{(1)}}$

The chain/sum rule

Entry points - no computation



Chain Rule

$$\frac{\partial J}{\partial W_{11}^{(1)}} = \frac{\partial J}{\partial z_1^{(2)}}\frac{\partial z_1^{(2)}}{\partial W_{11}^{(1)}} = \frac{\partial J}{\partial z_1^{(3)}}\frac{\partial z_1^{(3)}}{\partial z_1^{(2)}}\frac{\partial z_1^{(2)}}{\partial W_{11}^{(1)}} + \frac{\partial J}{\partial z_2^{(3)}}\frac{\partial z_2^{(3)}}{\partial z_1^{(2)}}\frac{\partial z_1^{(2)}}{\partial W_{11}^{(1)}}$$

$$= \frac{\partial J}{\partial z_1^{(4)}}\frac{\partial z_1^{(4)}}{\partial z_1^{(3)}}\frac{\partial z_1^{(3)}}{\partial z_1^{(2)}}\frac{\partial z_1^{(2)}}{\partial W_{11}^{(1)}} + \frac{\partial J}{\partial z_1^{(4)}}\frac{\partial z_1^{(4)}}{\partial z_2^{(3)}}\frac{\partial z_2^{(3)}}{\partial z_1^{(2)}}\frac{\partial z_1^{(2)}}{\partial W_{11}^{(1)}} = -(y - f(z_1^{(4)}))f'(z_1^{(4)})\frac{\partial z_1^{(4)}}{\partial z_1^{(3)}}\frac{\partial z_1^{(3)}}{\partial z_1^{(2)}}\frac{\partial z_1^{(2)}}{\partial W_{11}^{(1)}} + -(y - f(z_1^{(4)}))f'(z_1^{(4)})\frac{\partial z_1^{(4)}}{\partial z_2^{(3)}}\frac{\partial z_2^{(3)}}{\partial z_1^{(2)}}\frac{\partial z_1^{(2)}}{\partial W_{11}^{(1)}}$$

Its complicated and we will end up recomputing the same values again and again

Training a neural network

- The algorithm
- The problem
- The clever strategy
- Carrying out the strategy step by step
  - Last layer
  - Non-last layer
  - Putting the pieces together
- Partial derivative with regularization and different activation functions

Can we find a pattern to make this much easier?

How to update the weights…

Our update rule:

$W_{\text{new}} = W_{\text{old}} - \alpha \nabla \text{error}$

$$W_{ij}^{(\ell)} = W_{ij}^{(\ell)} - \alpha \frac{\partial J(W, b, \mathbf{x}, y)}{\partial W_{ij}^{(\ell)}}$$

$$b_i^{(\ell)} = b_i^{(\ell)} - \alpha \frac{\partial J(W, b, \mathbf{x}, y)}{\partial b_i^{(\ell)}}$$

$$\frac{\partial J}{\partial W_{11}^{(5)}} = \boxed{\frac{\partial J}{\partial z_1^{(6)}}} \frac{\partial z_1^{(6)}}{\partial W_{11}^{(5)}}$$

$$\frac{\partial J}{\partial W_{11}^{(4)}} = \boxed{\frac{\partial J}{\partial z_1^{(5)}}} \frac{\partial z_1^{(5)}}{\partial W_{11}^{(4)}}$$

$$\frac{\partial J}{\partial W_{11}^{(3)}} = \boxed{\frac{\partial J}{\partial z_1^{(4)}}} \frac{\partial z_1^{(4)}}{\partial W_{11}^{(3)}}$$

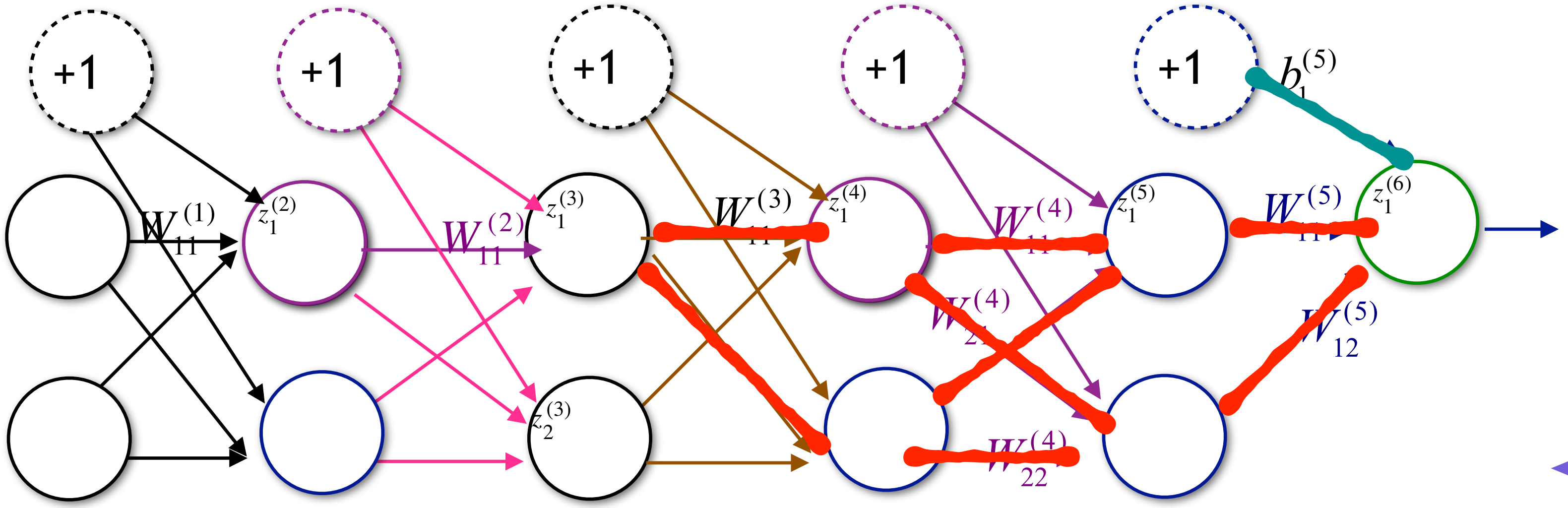
$$\frac{\partial J}{\partial W_{11}^{(2)}} = \boxed{\frac{\partial J}{\partial z_1^{(3)}}} \frac{\partial z_1^{(3)}}{\partial W_{11}^{(2)}}$$

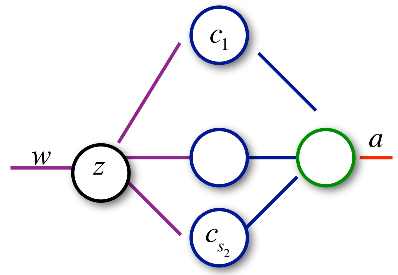
This rule works for all weights!

$$\frac{\partial J}{\partial W_{ij}^{(\ell)}} = \boxed{\frac{\partial J}{\partial z_i^{(\ell+1)}}} \overbrace{\frac{\partial z_i^{(\ell+1)}}{\partial W_{ij}^{(\ell)}}}^{a_j^{(\ell)}} = \delta_i^{(\ell+1)} a_j^{(\ell)}$$

$$\frac{\partial J}{\partial b_i^{(\ell)}} = \frac{\partial J}{\partial z_i^{(l+1)}} \overbrace{\frac{\partial z_i^{(l+1)}}{\partial b_i^{(\ell)}}}^{1} = \delta_i^{(\ell+1)}$$

$$\boxed{\delta_j^{(\ell)} = \frac{\partial J}{\partial z_j^{(\ell)}}}$$





$$\frac{da}{dw} = \frac{da}{dz}\frac{dz}{dw}$$

values propagate this way

$$z_i^{(\ell+1)} = W_{i1}^{(\ell)} a_1^{(\ell)} + \cdots + W_{ij}^{(\ell)} a_j^{(\ell)} + \cdots W_{is_\ell}^{(\ell)} a_{s_\ell}^{(\ell)} + b_i^{(\ell)}$$

How to update the weights…

Our update rule:

$W_{\text{new}} = W_{\text{old}} - \alpha \nabla \text{error}$

$$W_{ij}^{(\ell)} = W_{ij}^{(\ell)} - \alpha \frac{\partial J(W,b,\mathbf{x},y)}{\partial W_{ij}^{(\ell)}}$$

$$b_{i}^{(\ell)} = b_{i}^{(\ell)} - \alpha \frac{\partial J(W,b,\mathbf{x},y)}{\partial b_{i}^{(\ell)}}$$

$$\frac{\partial J}{\partial W_{11}^{(5)}} = \boxed{\delta_1^{(6)}} \frac{\partial z_1^{(6)}}{\partial W_{11}^{(5)}}$$

$$\frac{\partial J}{\partial W_{11}^{(4)}} = \boxed{\delta_1^{(5)}} \frac{\partial z_1^{(5)}}{\partial W_{11}^{(4)}}$$

$$\frac{\partial J}{\partial W_{11}^{(3)}} = \boxed{\delta_1^{(4)}} \frac{\partial z_1^{(4)}}{\partial W_{11}^{(3)}}$$

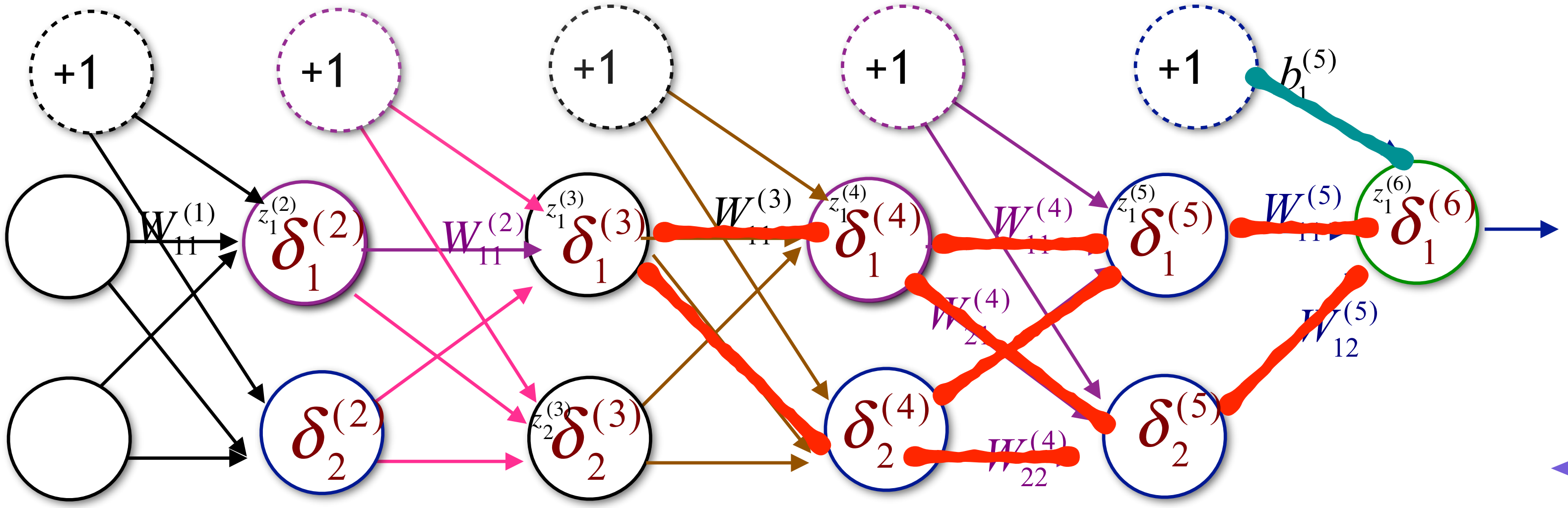
$$\frac{\partial J}{\partial W_{11}^{(2)}} = \boxed{\delta_1^{(3)}} \frac{\partial z_1^{(3)}}{\partial W_{11}^{(2)}}$$

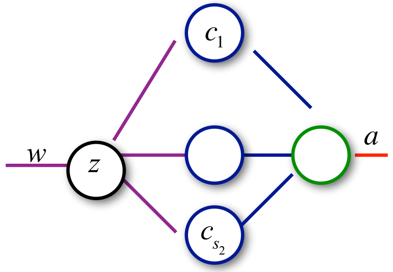
This rule works for all weights!

$$\frac{\partial J}{\partial W_{ij}^{(\ell)}} = \boxed{\delta_i^{(\ell+1)}} \overbrace{\frac{\partial z_i^{(\ell+1)}}{\partial W_{ij}^{(\ell)}}}^{a_j^{(\ell)}} = \delta_i^{(\ell+1)} a_j^{(\ell)}$$

$$\frac{\partial J}{\partial b_i^{(\ell)}} = \frac{\partial J}{\partial z_i^{(l+1)}} \overbrace{\frac{\partial z_i^{(l+1)}}{\partial b_i^{(\ell)}}}^{1} = \delta_i^{(\ell+1)}$$

$$\boxed{\delta_j^{(\ell)} = \frac{\partial J}{\partial z_j^{(\ell)}}}$$





$$\frac{da}{dw} = \frac{da}{dz}\frac{dz}{dw}$$

values propagate this way

$$z_i^{(\ell+1)} = \cancel{W_{i1}^{(\ell)}} a_1^{(\ell)} + \cdots + \cancel{W_{ij}^{(\ell)}} a_j^{(\ell)} + \cdots \cancel{W_{is_\ell}^{(\ell)}} a_{s_\ell}^{(\ell)} + b_i^{(\ell)}$$

$$z_i^{(\ell+1)} = \cancel{W_{i1}^{(\ell)}} a_1^{(\ell)} + \cdots + W_{ij}^{(\ell)} a_j^{(\ell)} + \cdots \cancel{W_{is_\ell}^{(\ell)}} a_{s_\ell}^{(\ell)} + \cancel{b_i^{(\ell)}}$$

The big idea!

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

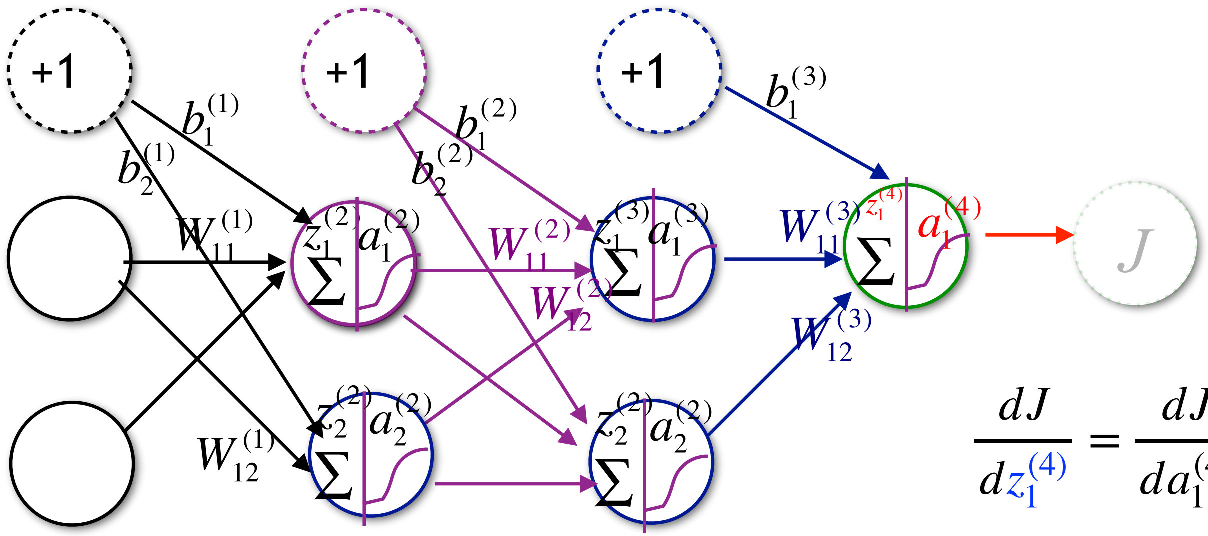
Computing the partial derivatives is easy if we can compute

$$\delta_j^{(\ell)} = \frac{\partial J}{\partial z_j^{(\ell)}}$$

Training a neural network

- The algorithm
- The problem
- The clever strategy
- Carrying out the strategy step by step
  - Last layer
  - Non-last layer
  - Putting the pieces together
- Partial derivative with regularization and different activation functions

Computing δ for the last layer



Goal: update weights to reduce error:

$$J(W, \mathbf{b}, \mathbf{x}, y) = \frac{1}{2}(\hat{y}_1 - y_1)^2 = \frac{1}{2}(a_1^{(4)} - y_1)^2$$

$$\frac{dJ}{dz_1^{(4)}} = \frac{dJ}{da_1^{(4)}}\frac{da_1^{(4)}}{dz_1^{(4)}}$$

$$= (a_1^{(4)} - y_1)\frac{\partial a^{(4)}}{\partial z_1^{(4)}}$$

$$= (a_1^{(4)} - y_1)\frac{\partial f(z^{(4)})}{\partial z_1^{(4)}}$$

$$= \underbrace{(a_1^{(4)} - y_1)f'(z^{(4)})}_{\frac{\partial J}{\partial z_1^{(4)}}}$$

Training a neural network

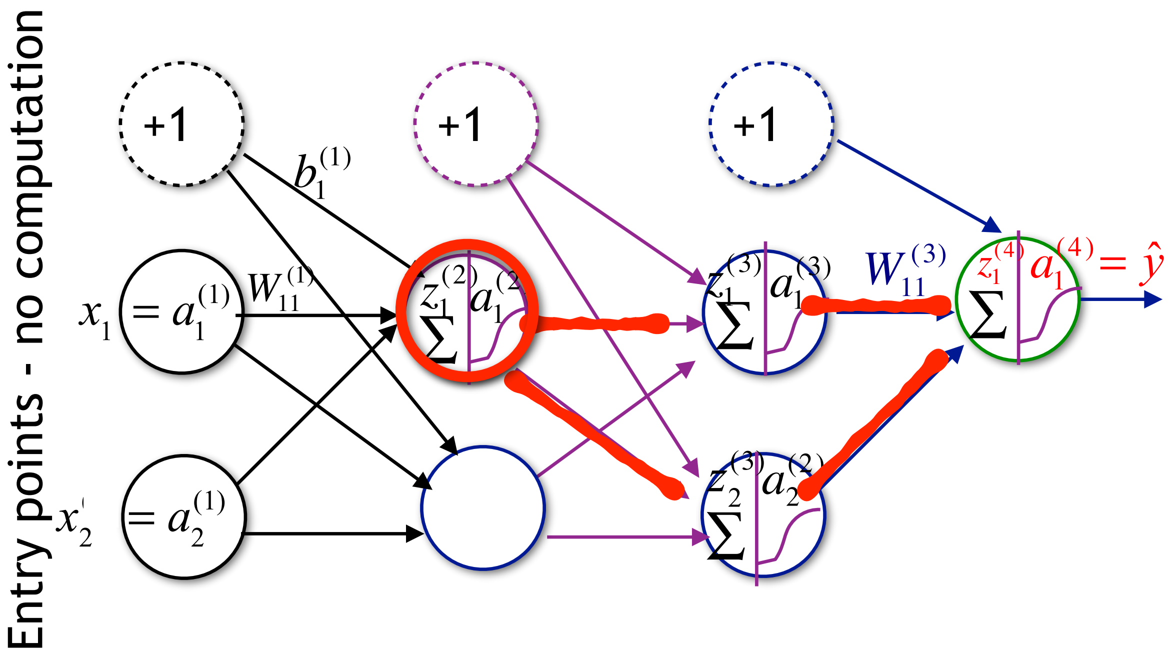
- The algorithm
- The problem
- The clever strategy
- Carrying out the strategy step by step
  - Last layer
  - Non-last layer
  - Putting the pieces together
- Partial derivative with regularization and different activation functions

Small example to build intuition: computing $\delta_1^{(2)}$ for the graph below

$J(W,b,x,y)=\frac{1}{2}(y-\hat{y})^2$

$$\delta_1^{(2)} = \frac{\partial J(W,b,\mathbf{x},y)}{\partial z_1^{(2)}} = \frac{\partial J}{\partial z_1^{(3)}}\frac{\partial z_1^{(3)}}{\partial z_1^{(2)}} + \frac{\partial J}{\partial z_2^{(3)}}\frac{\partial z_2^{(3)}}{\partial z_1^{(2)}}$$

$$= \sum_{i=1}^{2} \frac{\partial J}{\partial z_i^{(3)}}\frac{\partial z_i^{(3)}}{\partial z_1^{(2)}} = \sum_{i=1}^{s_{\ell+1}} \frac{\partial J}{\partial z_i^{(3)}} W_{i1} f'(z_i^{(2)})$$



$$\frac{\partial z_1^{(3)}}{\partial z_1^{(2)}} = W_{11}^{(2)}\frac{\partial f(z_1^{(2)})}{\partial z_1^{(2)}} = W_{11}^{(2)} f'(z_1^{(2)}) \qquad \frac{\partial z_2^{(3)}}{\partial z_1^{(2)}} = W_{21}^{(2)}\frac{\partial f(z_1^{(2)})}{\partial z_1^{(2)}} = W_{21}^{(2)} f'(z_1^{(2)})$$

$$z_1^{(3)} = W_{11}^{(2)} a_1^{(2)} + W_{12}^{(2)} a_2^{(2)} + b_1^{(2)} \qquad z_2^{(3)} = W_{21}^{(2)} a_1^{(2)} + W_{22}^{(2)} a_2^{(2)} + b_2^{(2)}$$

$$z_1^{(3)} = W_{11}^{(2)} f(z_1^{(2)}) + W_{12}^{(2)} f(z_2^{(2)}) + b_1^{(2)} \qquad z_2^{(3)} = W_{21}^{(2)} f(z_1^{(2)}) + W_{22}^{(2)} f(z_2^{(2)}) + b_2^{(2)}$$

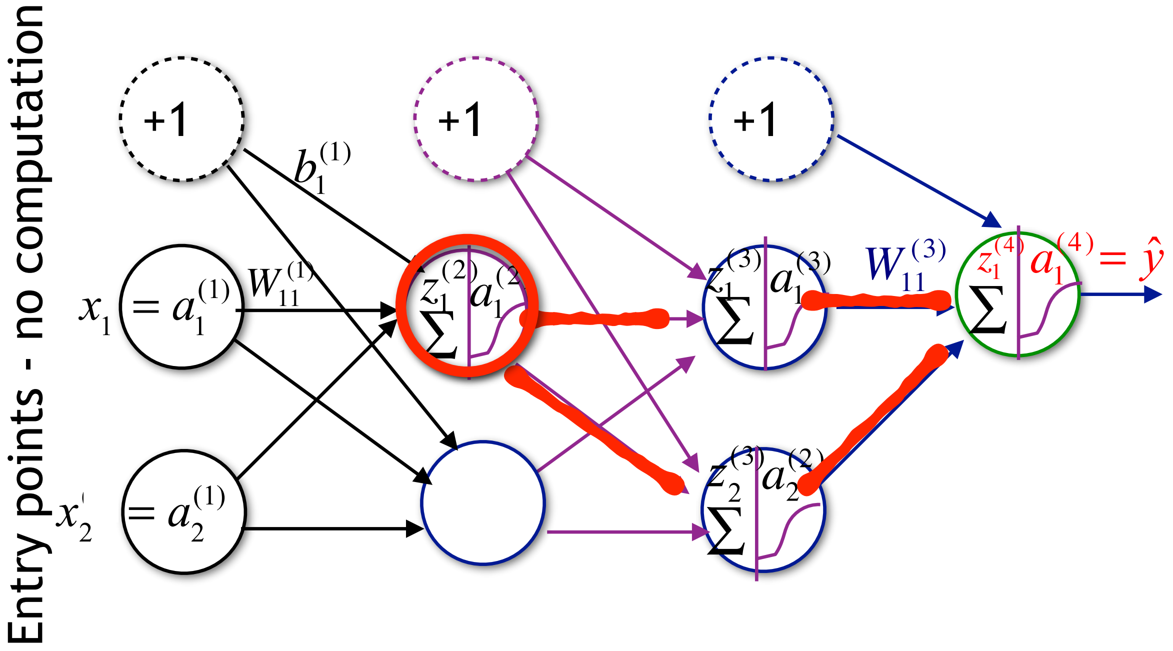
Note to 11:00 class: Sergey was right!

Small example to build intuition: computing $\delta_1^{(2)}$ for the graph below

$$J(W,b,x,y)=\frac{1}{2}(y-\hat{y})^2$$

$$\delta_1^{(2)} = \frac{\partial J(W,b,\mathbf{x},y)}{\partial z_1^{(2)}} = \boxed{\delta_1^{(3)}}\frac{\partial z_1^{(3)}}{\partial z_1^{(2)}} + \boxed{\delta_2^{(3)}}\frac{\partial z_2^{(3)}}{\partial z_1^{(2)}}$$

$$= \sum_{i=1}^{2}\boxed{\delta_i^{(3)}}\frac{\partial z_i^{(3)}}{\partial z_1^{(2)}} = \sum_{i=1}^{s_{\ell+1}}\boxed{\delta_i^{(3)}}W_{i1}f'(z_i^{(2)})$$



$$\frac{\partial z_1^{(3)}}{\partial z_1^{(2)}} = W_{11}^{(2)}\frac{\partial f(z_1^{(2)})}{\partial z_1^{(2)}} = W_{11}^{(2)}f'(z_1^{(2)})$$

$$\frac{\partial z_2^{(3)}}{\partial z_1^{(2)}} = W_{21}^{(2)}\frac{\partial f(z_1^{(2)})}{\partial z_1^{(2)}} = W_{21}^{(2)}f'(z_1^{(2)})$$

$$z_1^{(3)} = W_{11}^{(2)}a_1^{(2)} + W_{12}^{(2)}a_2^{(2)} + b_1^{(2)}$$

$$z_2^{(3)} = W_{21}^{(2)}a_1^{(2)} + W_{22}^{(2)}a_2^{(2)} + b_2^{(2)}$$

$$z_1^{(3)} = W_{11}^{(2)}f(z_1^{(2)}) + \cancel{W_{12}^{(2)}f(z_2^{(2)})} + \cancel{b_1^{(2)}}$$

$$z_2^{(3)} = W_{21}^{(2)}f(z_1^{(2)}) + \cancel{W_{22}^{(2)}f(z_2^{(2)})} + \cancel{b_2^{(2)}}$$

Note to 11:00 class:
Sergey was right!

Computing $\delta_j^{(\ell)}$ for $\ell < n_\ell$

$$\delta_j^{(\ell)} = \frac{\partial J(W,b,\mathbf{x},y)}{\partial z_j^{(\ell)}} = \sum_{i=1}^{s_{\ell+1}} \frac{\partial J}{\partial z_i^{(\ell+1)}} \frac{\partial z_i^{(\ell+1)}}{\partial z_j^{(\ell)}} = \sum_{i=1}^{s_{\ell+1}} \frac{\partial J}{\partial z_i^{(\ell+1)}} W_{ij} f'(z_j^{(\ell)})$$

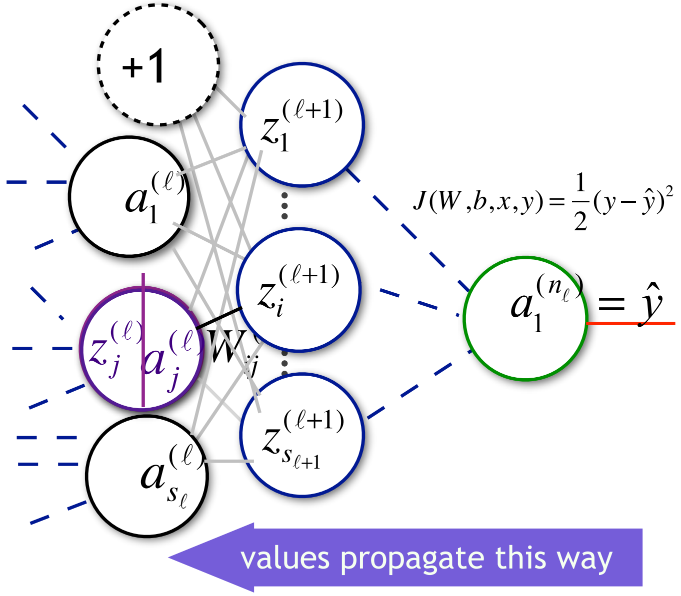
$$= \frac{\partial J}{\partial z_1^{(\ell+1)}} \frac{\partial z_1^{(\ell+1)}}{\partial z_j^{(\ell)}} + \frac{\partial J}{\partial z_2^{(\ell+1)}} \frac{\partial z_2^{(\ell+1)}}{\partial z_j^{(\ell)}} + \cdots + \frac{\partial J}{\partial z_{s_{\ell+1}}^{(\ell+1)}} \frac{\partial z_{s_{\ell+1}}^{(\ell+1)}}{\partial z_j^{(\ell)}}$$

$$\frac{\partial z_i^{(\ell+1)}}{\partial z_j^{(\ell)}} = W_{ij}^{(\ell)} \frac{\partial f(z_j^{(\ell)})}{\partial z_j^{(\ell)}} = W_{ij}^{\ell} f'(z_j^{(\ell)})$$

$$z_i^{(\ell+1)} = \sum_{k=1}^{s_\ell} W_{ik}^{(\ell)} a_k^{(\ell)} + b_i^{(\ell)}$$

$$= \sum_{k=1}^{s_\ell} W_{ik}^{(\ell)} f(z_k^{(\ell)}) + b_i^{(\ell)}$$

$$= W_{i1}^{(\ell)} f(z_1^{(\ell)}) + \cdots + W_{ij}^{(\ell)} f(z_j^{(\ell)}) + \cdots W_{is_\ell}^{(\ell)} f(z_{s_\ell}^{(\ell)}) + b_i^{(\ell)}$$



Computing $\delta_j^{(\ell)}$ for $\ell < n_\ell$

$$\delta_j^{(\ell)} = \frac{\partial J(W,b,\mathbf{x},y)}{\partial z_j^{(\ell)}} = \sum_{i=1}^{s_{\ell+1}} \boxed{\delta_i^{(\ell+1)}} \frac{\partial z_i^{(\ell+1)}}{\partial z_j^{(\ell)}} = \sum_{i=1}^{s_{\ell+1}} \boxed{\delta_i^{(\ell+1)}} W_{ij} f'(z_j^{(\ell)})$$

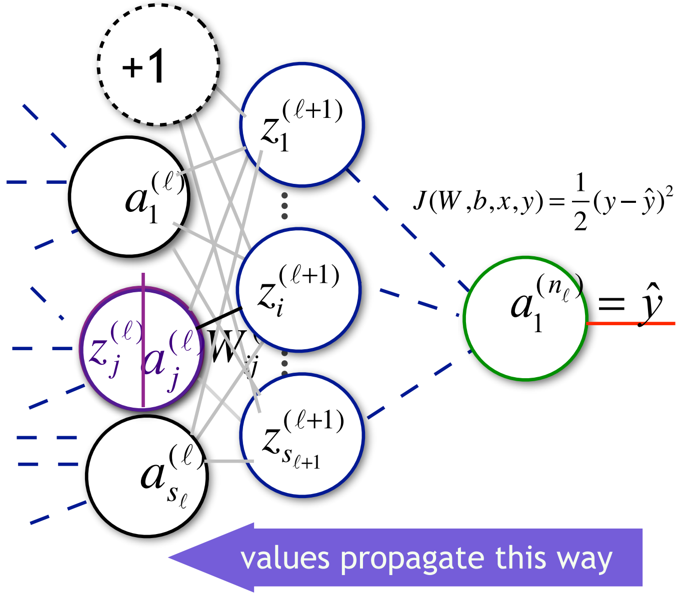
$$= \frac{\partial J}{\partial z_1^{(\ell+1)}} \frac{\partial z_1^{(\ell+1)}}{\partial z_j^{(\ell)}} + \frac{\partial J}{\partial z_2^{(\ell+1)}} \frac{\partial z_2^{(\ell+1)}}{\partial z_j^{(\ell)}} + \cdots + \frac{\partial J}{\partial z_{s_{\ell+1}}^{(\ell+1)}} \frac{\partial z_{s_{\ell+1}}^{(\ell+1)}}{\partial z_j^{(\ell)}}$$

$$\frac{\partial z_i^{(\ell+1)}}{\partial z_j^{(\ell)}} = W_{ij}^{(\ell)} \frac{\partial f(z_j^{(\ell)})}{\partial z_j^{(\ell)}} = W_{ij}^{\ell} f'(z_j^{(\ell)})$$

$$z_i^{(\ell+1)} = \sum_{k=1}^{s_\ell} W_{ik}^{(\ell)} a_k^{(\ell)} + b_i^{(\ell)}$$

$$= \sum_{k=1}^{s_\ell} W_{ik}^{(\ell)} f(z_k^{(\ell)}) + b_i^{(\ell)}$$

$$= \cancel{W_{i1}^{(\ell)} f(z_1^{(\ell)})} + \cdots + W_{ij}^{(\ell)} f(z_j^{(\ell)}) + \cdots \cancel{W_{is_\ell}^{(\ell)} f(z_{s_\ell}^{(\ell)})} + \cancel{b_i^{(\ell)}}$$



Training a neural network

- The algorithm
- The problem
- The clever strategy
- Carrying out the strategy step by step
  - Last layer
  - Non-last layer
  - Putting the pieces together
- Partial derivative with regularization and different activation functions

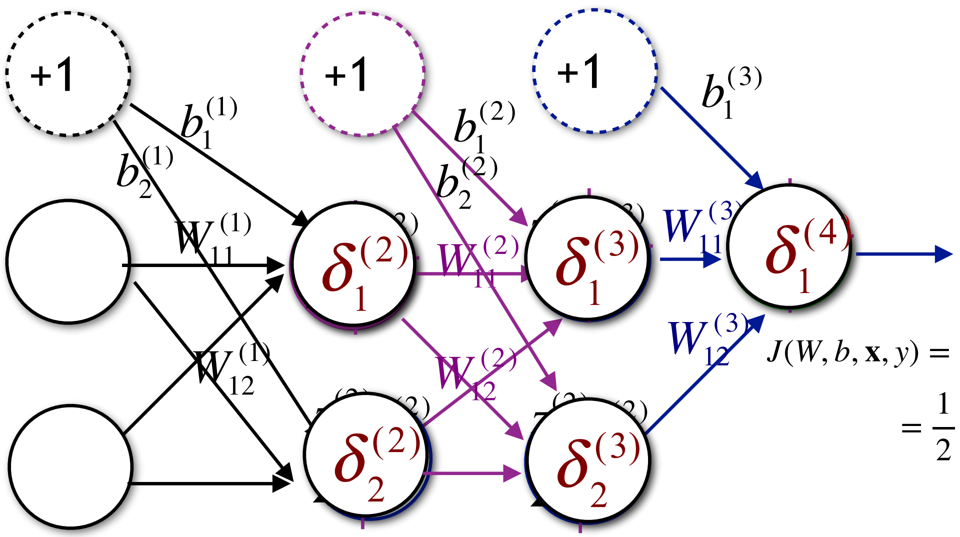
Putting the pieces together

$$\boxed{\delta_i^{(\ell)} = \frac{dJ}{dz_i^{(\ell)}}}$$

Chain rule from previous slide:

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

$$\frac{\partial J}{\partial b_i^{(\ell)}} = \frac{\partial J}{\partial z_i^{(l+1)}} \frac{\partial z_i^{(l+1)}}{\partial b_i^{(\ell)}} = \delta_i^{(\ell+1)}$$



$$J(W, b, \mathbf{x}, y) = \frac{1}{2}\left(y - a_1^{(n_\ell)}\right)^2 = \frac{1}{2}\left(y - f(z_1^{(n_\ell)})\right)^2$$

We first compute $\delta_1^{(4)}$, then we compute $\delta_1^{(3)}$ and $\delta_2^{(3)}$, then we compute $\delta_1^{(2)}$ and $\delta_2^{(2)}$.

What is $\delta_i^{(\ell)}$?

For the output layer:

$$\delta_j^{(n_\ell)} = \frac{\partial J}{\partial z_j^{(n_\ell)}} = -(y_j - f(z_j^{(n_\ell)}) f'(z_j^{(n_\ell)})$$

I added a subscript in case there was more than one output neuron

For the other layers:

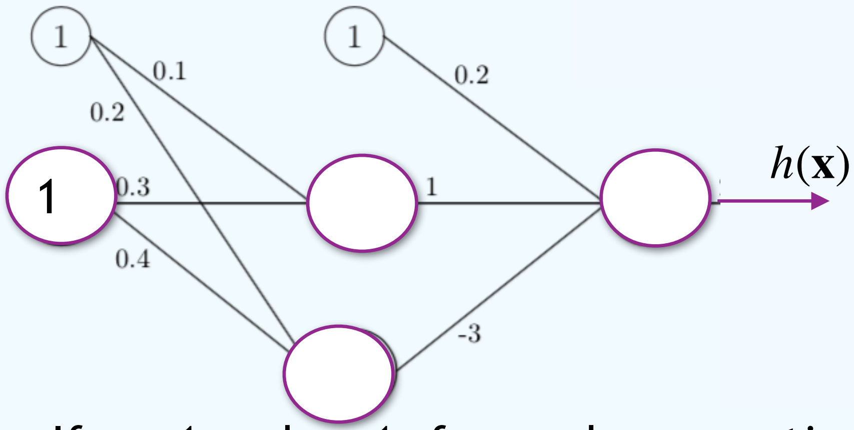
$$\delta_j^{(\ell)} = \frac{\partial J}{\partial z_j^{(\ell)}} = \sum_{i=1}^{s_{\ell+1}} \delta_i^{(\ell+1)} W_{ij}^{(\ell)} f'(z_j^{(\ell)})$$

$$\delta_j^{(n_\ell)} = \frac{dJ}{dz_j^{(n_\ell)}} = (f(z_j^{(n_\ell)}) - y)f'(z_j^{(n_\ell)}) \qquad \delta_j^{(\ell)} = \frac{dJ}{dz_j^{(\ell)}} = \sum_{i=1}^{s_{\ell+1}} \delta_i^{(\ell+1)} W_{ij}^{(\ell)} f'(z_j^{(\ell)}) \qquad \frac{df(z)}{dz} = f(z)(1-f(z))$$

Here we choose $f(z) = \sigma(z)$ to be the logistic function

$$\frac{dJ}{db_i^{(\ell)}} = \delta_i^{(\ell+1)}$$

$$W_{ij}^{(\ell)} = W_{ij}^{(\ell)} - \alpha \frac{\partial J(W, b, \mathbf{x}, y)}{\partial W_{ij}^{(\ell)}} \qquad \frac{\partial J(W, b, \mathbf{x}, y)}{\partial b_i^{(\ell)}}$$



$$W^{(1)} = \begin{bmatrix} 0.3 \\ 0.4 \end{bmatrix} \qquad W^{(2)} = \begin{bmatrix} 1 & -3 \end{bmatrix}$$

$$b^{(1)} = \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix} \qquad b^{(2)} = [0.2]$$

If x = 1 and y =1, forward propagation:

$$x = a^{(1)} = [1] \qquad z^{(2)} = \begin{bmatrix} 1 * 0.3 + 0.1 \\ 1 * 0.4 + 0.2 \end{bmatrix} \qquad a^{(2)} = \begin{bmatrix} 0.60 \\ 0.65 \end{bmatrix} \qquad z^{(3)} = [1 * 0.60 + (-3) * 0.65 + 0.2] \qquad a^{(3)} = [\sigma(-1.15)] = [0.24]$$

Backward propagation:

$$\delta^{(3)} = [(0.24 - 1)(.24)(1 - .24)] = [-0.14] \qquad \delta_1^{(2)} = \frac{\partial J}{\partial z_1^{(2)}} = \delta_1^{(3)} W_{11}^{(2)} f'(z_1^{(2)}) = [-0.14 * 1(0.60)(1 - 0.60)] = [-0.03]$$

$$\frac{\partial J}{\partial W_{11}^{(2)}} = \delta_1^{(3)} a_1^{(2)} = [-0.14 * 0.60] \qquad \frac{\partial J}{\partial W_{11}^{(1)}} = \delta_1^{(2)} a_1^{(1)} = [-0.03 * 1] = (f(z_j^{(n_\ell)}) - y)f'(z_j^{(n_\ell)})$$

Calculations done with rounded numbers...

Training a neural network

- The algorithm
- The problem
- The clever strategy
- Carrying out the strategy step by step
  - Last layer
  - Non-last layer
  - Putting the pieces together
- Partial derivative with regularization and different activation functions

Regularization

$$J(W,b,\mathbf{x},y) = \frac{1}{2}(y-\hat{y})^2 + \frac{\lambda}{2}\sum_{k=1}^{n_\ell-1}\sum_{j=1}^{s_\ell}\sum_{i=1}^{s_{\ell+1}}(W_{ij}^{(k)})^2$$

Adding L2 regularization

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

Activation functions

If we use a ***sigmoid*** activation: $f(z) = \dfrac{1}{1+e^{-z}}$ then $\dfrac{df(z)}{dz} = f(z)(1-f(z))$

If we use a ***relu*** activation: $f(z) = \max(0,z)$ then $\dfrac{df(z)}{dz} = \begin{cases} 0 & \text{if } z < 0 \\ 1 & \text{if } z > 0 \\ \text{undefined} & \text{if } z = 0 \end{cases}$

Batch Gradient Descent

$$J(W,b) = \frac{1}{2N}\sum_{i=1}^{N}(\hat{y}-y)^2 = \frac{1}{N}\sum_{i=1}^{N} J(W,b,\mathbf{x},y) + \frac{\lambda}{2}\sum_{k=1}^{n_\ell-1}\sum_{j=1}^{s_\ell}\sum_{i=1}^{s_{\ell+1}}(W_{ij}^{(k)})^2$$