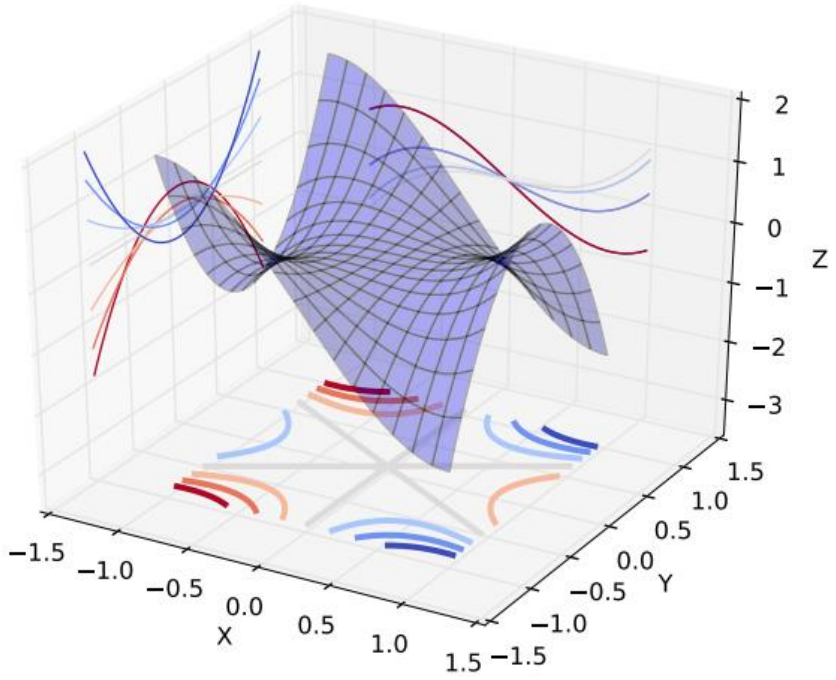
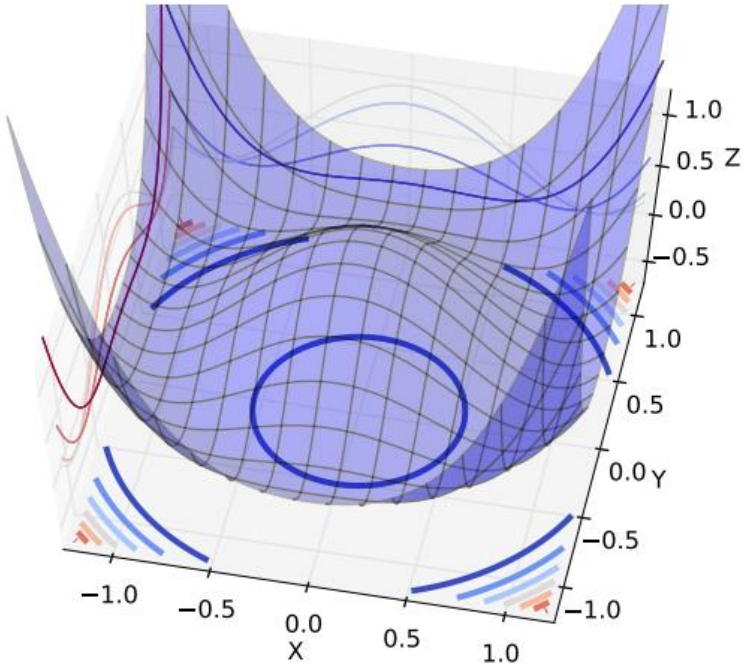
Learning Deep Neural Networks with Backpropagation





http://arxiv.org/pdf/1406.2572.pdf

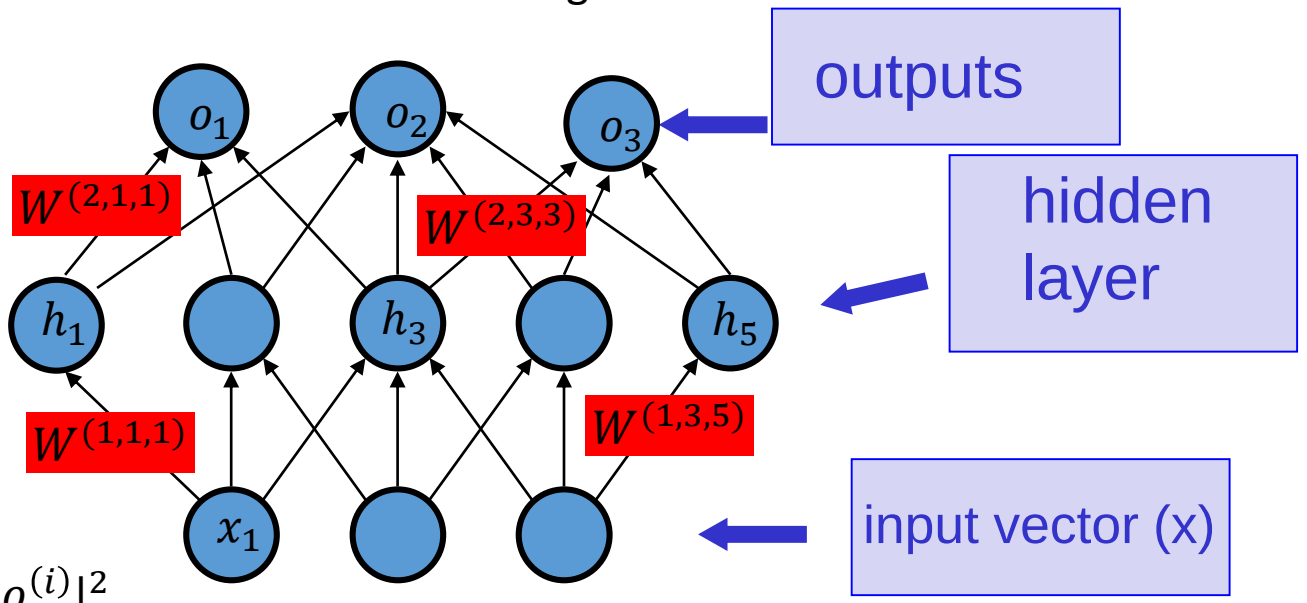
Slides from Geoffrey Hinton

CSC321: Intro to Machine Learning and Neural Networks, Winter 2016

Michael Guerzhoy

Multilayer Neural Network for Classification

o_i is large if the probability that the correct class is *i* is high



A possible cost function:

$$C(o,y) = \sum_{i=1}^{m} |y^{(i)} - o^{(i)}|^2$$

$y^{(i)}$'s and $o^{(i)}$'s encoded using one-hot encoding

On the board: Chain Rule

Partial Derivatives of the Cost Function

- We need the partial derivatives of the cost function $C(o, y)$ w.r.t all the W and b
- $o_i = g\left(\sum_j W^{(2,j,i)} h_j + b^{(2,j)}\right)$
- Partial derivative of $C(o, y)$ w.r.t $W^{(2,j,i)}$
-

$$\frac{\partial C}{\partial W^{(2,j,i)}}(x, y, W, b, h, o) = \frac{\partial o_i}{\partial W^{(2,j,i)}}(x, y, W, b, h, o) \frac{\partial C}{\partial o_i}(x, y, W, b, h, o)$$

$$= \frac{\partial (\sum_j W^{(2,j,i)} h_j)}{\partial W^{(2,j,i)}}(x, y, W, b, h, o) \frac{\partial g}{\partial (\sum_j W^{(2,j,i)} h_j)}(x, y, W, b, h, o) \frac{\partial C}{\partial o_i}(x, y, W, b, h, o)$$

$$= h_j \frac{\partial g}{\partial \sum_j W^{(2,j,i)} h_j}(x, y, W, b, h, o) \frac{\partial C}{\partial o_i}(x, y, W, b, h, o)$$

$$= h_j g' \left(\sum_j W^{(2,j,i)} h_j \right) \frac{\partial}{\partial o_i} C(o, y)$$

$$h_j g'\left(\sum_j W^{(2,j,i)} h_j\right)\frac{\partial C}{\partial o_i}(o,y)$$

- $g(t)=\frac{1}{1+\exp(-t)}$ 

$$g'(t)=\frac{\exp(-t)}{(1+\exp(-t))^2}=\frac{1}{(1+\exp(-t))}\frac{\exp(-x)}{(1+\exp(-t))}=g(t)\big(1-g(t)\big)$$

- $C(o,y)=\sum_{i=1}^{N}(o_i-y_i)^2$ 

$$\frac{\partial}{\partial o_i}\sum_{i=1}^{N}(o_i-y_i)^2=2(o_i-y_i)$$

$$\frac{\partial C}{\partial W^{(2,j,i)}}(x, y, W, b, h, o) = h_j g' \left(\sum_j W^{(2,j,i)} h_j \right) \frac{\partial C}{\partial o_i}(o, y)$$

$$= 2h_j \; g\left(\sum_j W^{(2,j,i)} h_j \right) \left(1 - g\left(\sum_j W^{(2,j,i)} h_j \right) \right) (o_i - y_i)$$

Vectorization

- $\frac{\partial C}{\partial W^{(2,j,i)}}(x, y, W, b, h, o) =$

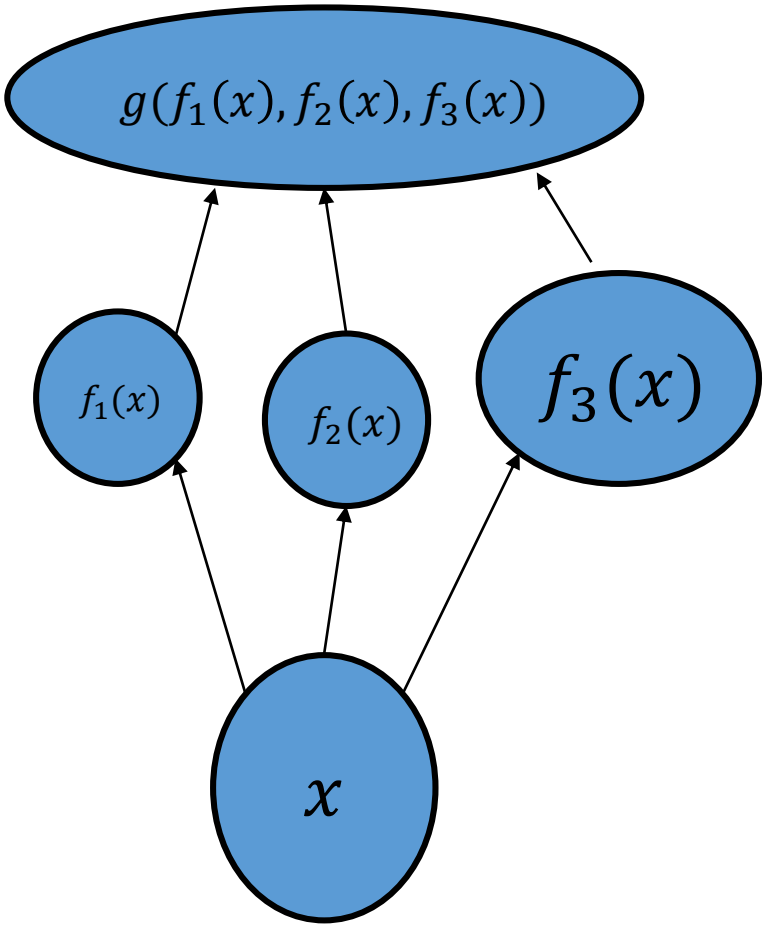
$$2h_j g\left(\sum_j W^{(2,j,i)} h_j\right)\left(1 - g\left(\sum_j W^{(2,j,i)} h_j\right)\right)(o_i - y_i)$$

- $\frac{\partial C}{\partial W^{(2)}}(x, y, W, b, h, o) =$

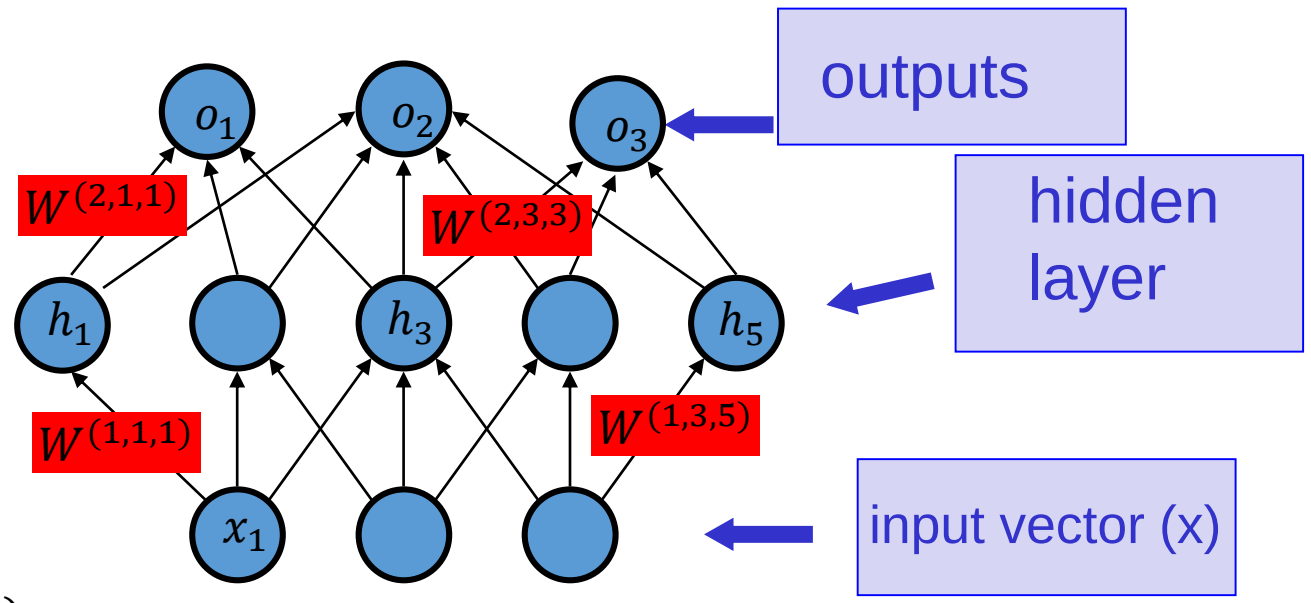
$$= h \cdot \left(g\left((W^{(2)})^T h\right).\left(1 - g\left((W^{(2)})^T h\right)\right).(o - y)\right)^T$$

$$= h \cdot \left(o.(1 - o).(o - y)\right)^T$$

More Chain Rule



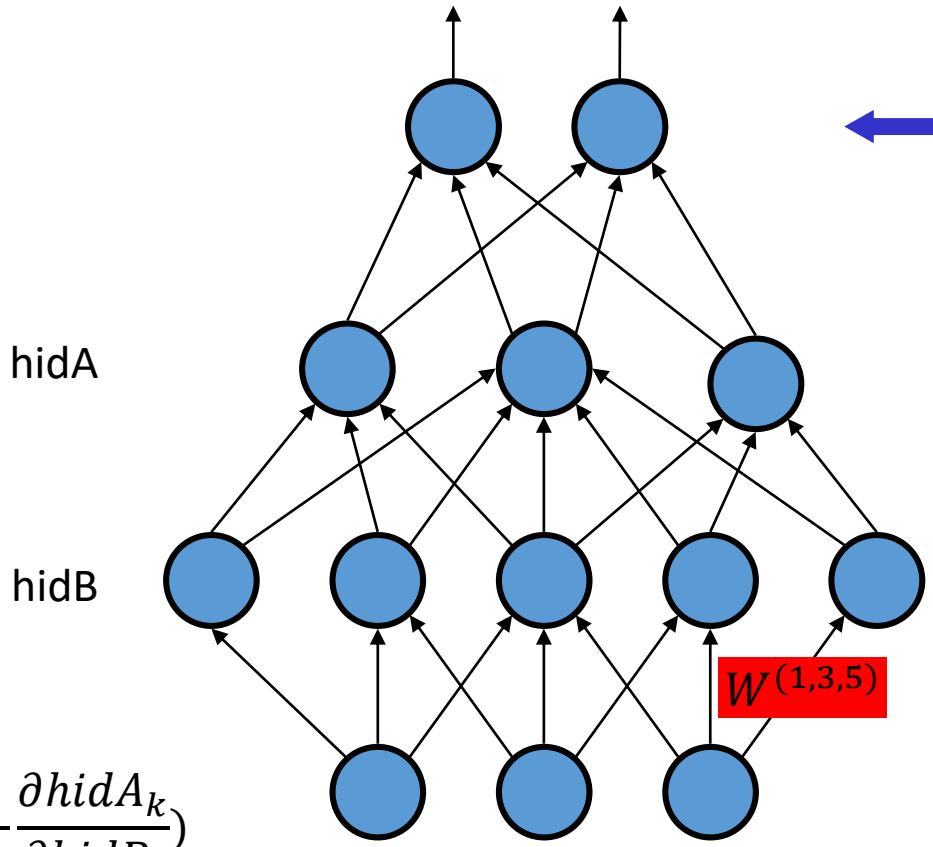
$$\frac{\partial g}{\partial x} = \sum \frac{\partial g}{\partial f_i} \frac{\partial f_i}{\partial x}$$



$$\frac{\partial C}{\partial h_i} = \sum_k \left(\frac{\partial C}{\partial o_k}\frac{\partial o_k}{h_i}\right)$$

$$\frac{\partial C}{\partial W^{(1,j,i)}} = \frac{\partial C}{\partial h_i}\frac{\partial h_i}{\partial W^{(1,j,i)}}$$

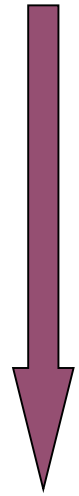
Backpropagation



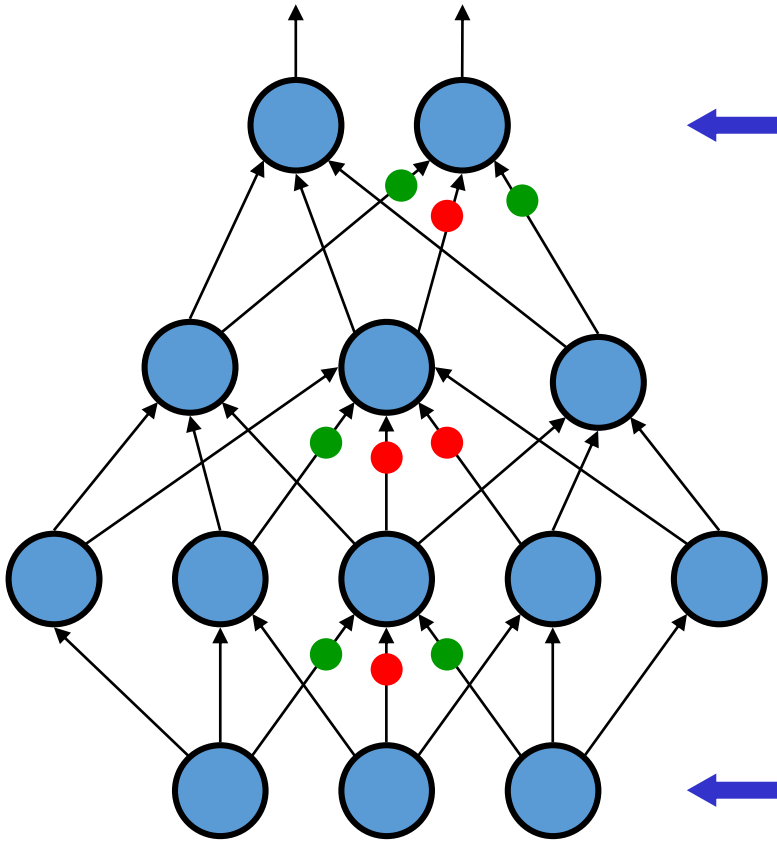
$$\frac{\partial C}{\partial hidB_i} = \sum_k \left(\frac{\partial C}{\partial hidA_k}\frac{\partial hidA_k}{\partial hidB_i}\right)$$

$$\frac{\partial C}{\partial W^{(1,j,i)}} = \frac{\partial C}{\partial hidB_i}\frac{\partial hidB_i}{\partial W^{(1,j,i)}}$$

Back-propagate error signal to get derivatives for learning



Compare outputs with correct answer to get error signal



outputs

hidden layers

input vector