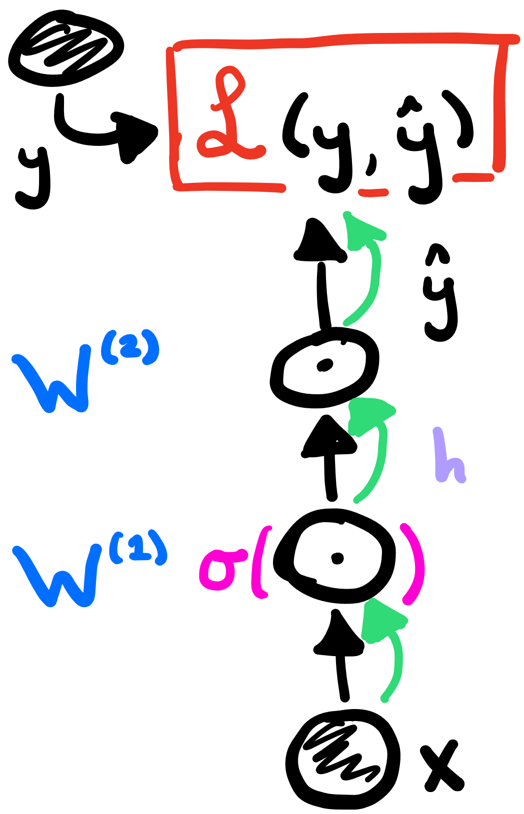
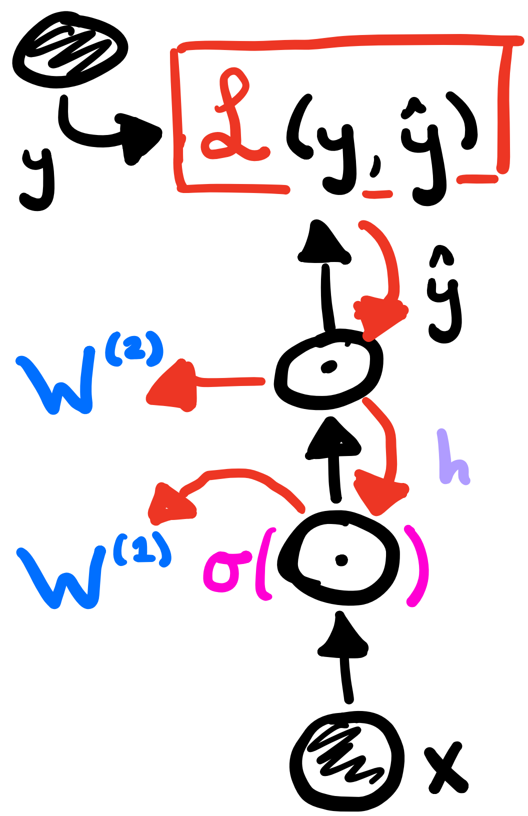
DS 4440 Backpropagation (2)

Last time: Gradients on Computation graphs via backprop.



.forward()



.backward()

For implementation, each node/layer must know how to go forward and backward.

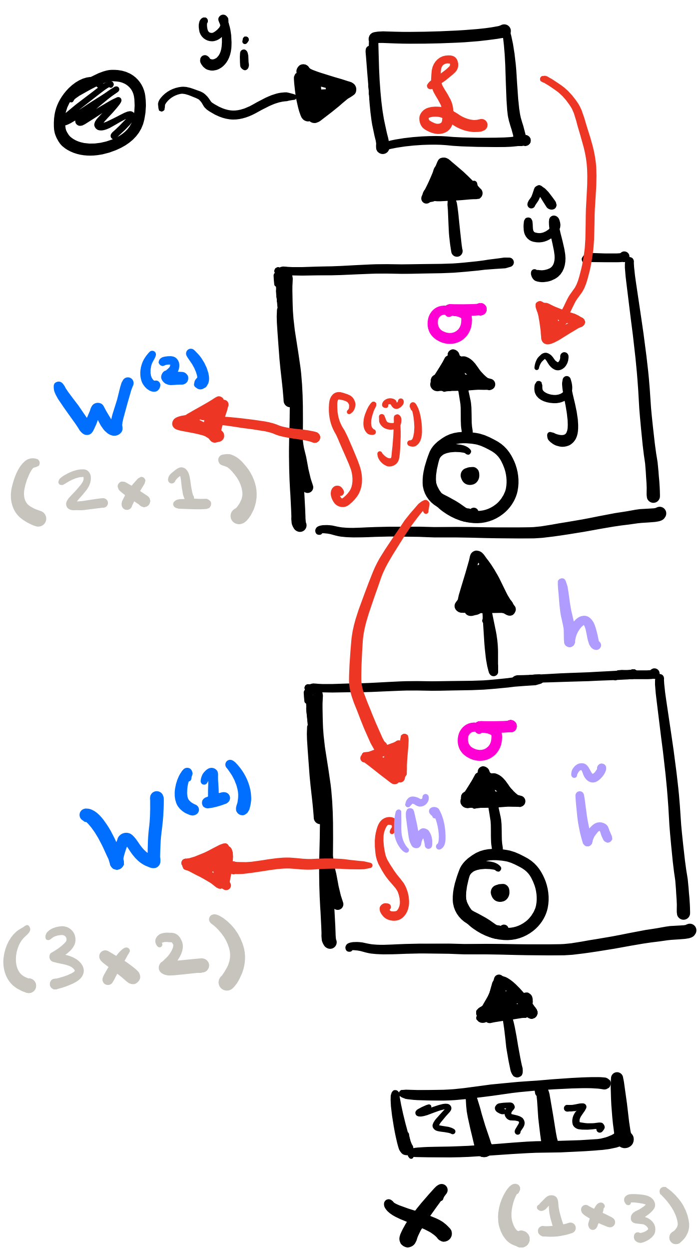
Let's consider an example

$$\mathcal{L} = \text{BCELoss}(y, \hat{y})$$

Probabilities

$$= -y \lg \hat{y} - (1-y) \lg (1-\hat{y})$$

0/1



$$\frac{\partial \mathcal{L}}{\partial \tilde{y}} = \underbrace{\frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \tilde{y}}}_{\text{"local error"}} \quad \boxed{\delta^{(\tilde{y})}}$$

$$\nabla_{W^{(2)}} \mathcal{L} = \nabla_{W^{(2)}} \tilde{y} \cdot \frac{\partial \mathcal{L}}{\partial \tilde{y}}$$

$$= \underbrace{\nabla_{W^{(2)}} \tilde{y}}_{h} \boxed{\delta^{(\tilde{y})}}$$

$$\nabla_{\tilde{h}} \mathcal{L} = \nabla_{\tilde{h}} \tilde{y} \boxed{\delta^{(\tilde{y})}}$$

local error at $\tilde{h}$ $\quad \delta^{(\tilde{h})}$

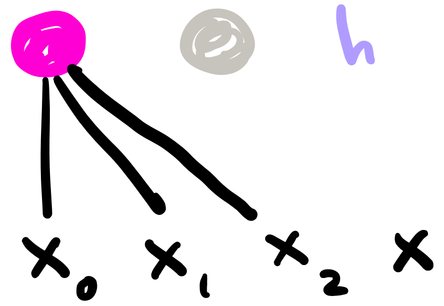
$$\nabla_{W^{(1)}} \mathcal{L} = \underbrace{\nabla_{W^{(1)}} \tilde{h}}_{} \cdot \delta^{(\tilde{h})}$$

$$\hat{y} = \sigma(\underbrace{W^{(2)} \cdot h}_{\tilde{y}})$$

$$h = \sigma(W^{(2)} \cdot x)$$

$\tilde{h}$

$\delta_0^{(\tilde{h})}$

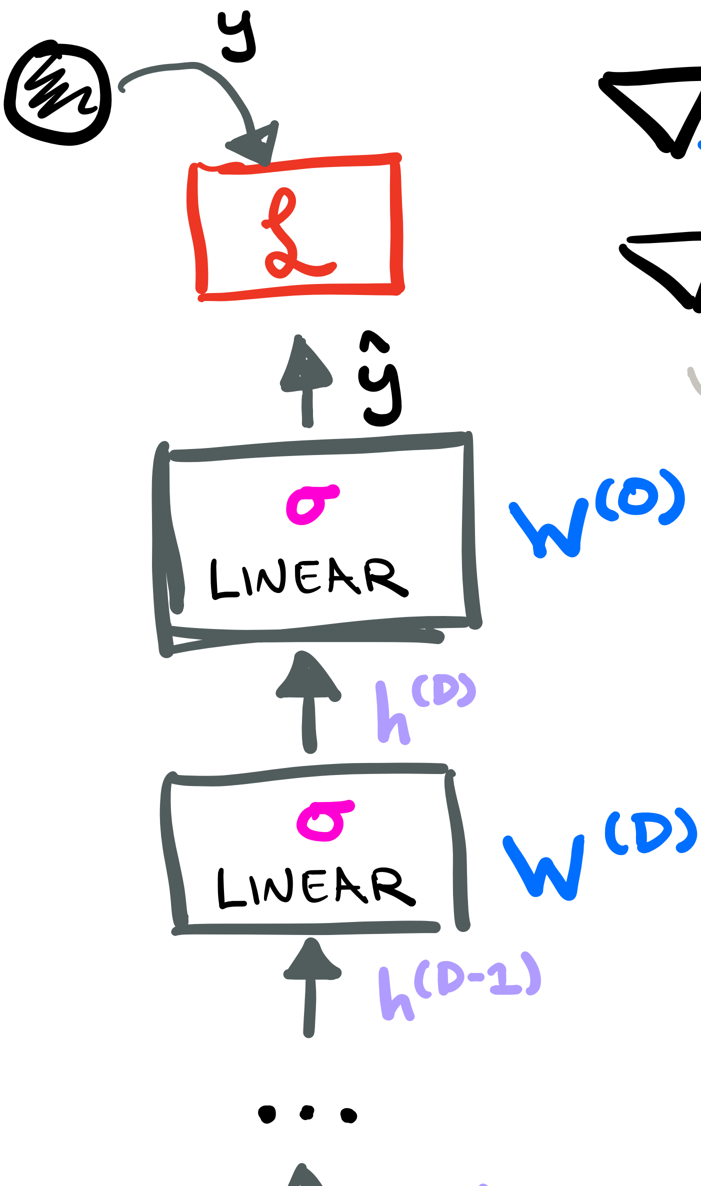


$$\nabla_{W^{(1)}} W^{(1)}_{:0} \cdot x = x$$

(1x3)

$$\to x \cdot \delta_0^{(\tilde{h})}$$

Generalizing: Consider a feed forward network with D layers.

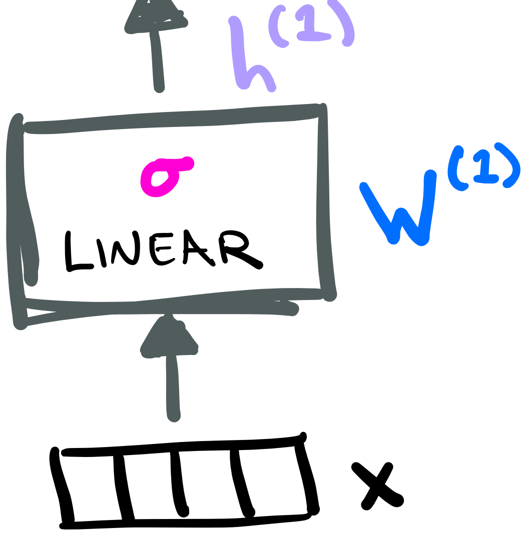


$$\nabla_{W^{(k)}} \mathcal{L} = \nabla_{W^{(k)}} \cdot \delta^{h^{(k)}}$$

$$\underbrace{\nabla_{h^{(k)}} \mathcal{L}}_{\text{Matrices!}} = \nabla_{h^{(k)}} h^{(k+1)} \, \delta^{h^{(k+1)}}$$

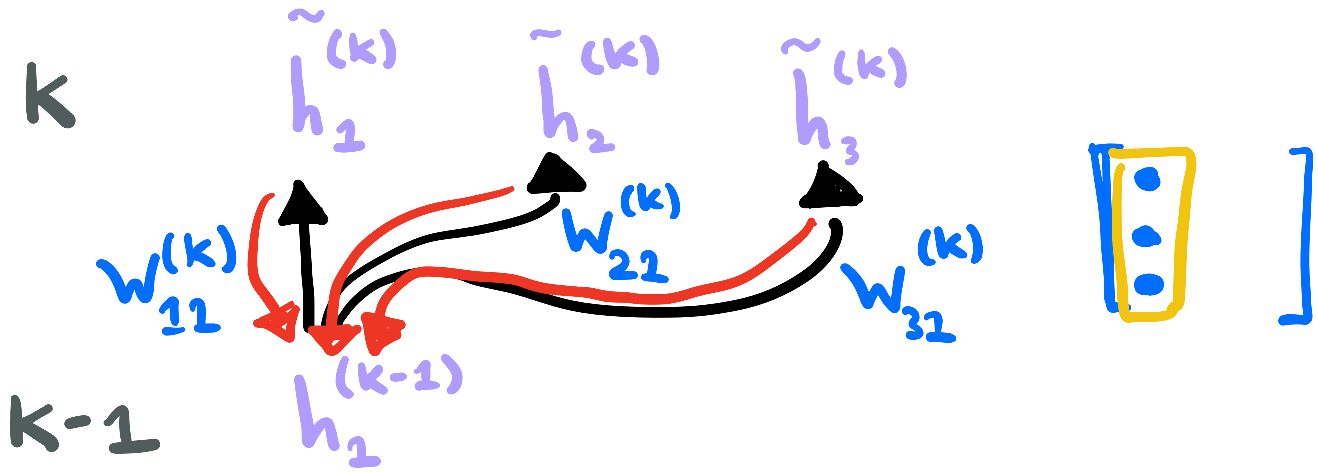
Jacobian

$$\begin{bmatrix} \frac{\partial h_1^{(k+1)}}{\partial h_1^{(k)}} & \cdots & \frac{\partial h_d^{(k+1)}}{\partial h_1^{(k)}} \\ & \ddots & \\ & & \frac{\partial h_d^{(k+1)}}{\partial h_d^{(k)}} \end{bmatrix}$$



In a simple fully connected layer, the local error is intuitive

$$h^{(k)} = \sigma(\underbrace{W^{(k)} \cdot h^{(k-1)}}_{\tilde{h}^{(k)}})$$



$$\frac{\partial \tilde{h}_2^{(k)}}{\partial h_1^{(k-1)}} = \frac{\partial W_{21}^{(k)} h_1^{(k-1)}}{\partial h_1^{(k-1)}} = W_{21}^{(k)}$$

So "blame" for error at h_1^k on $h_1^{(k-1)}$ scales with connection weight

$$\delta_1^{(k-1)} = \nabla_{h_1^{(k-1)}} \cdot \delta^k$$

$$= \sum^d \frac{\partial \tilde{h}_j^{(k)}}{\ldots} \quad \delta^k \Big\{ \text{local error at } j$$

$$\underbrace{{}_{j=1} \; \partial h_1^{(k-1)}}_{\text{Influence of } h_1^{k-1} \text{ on } h_j^k} \; \delta_j$$

$$\approx \sum_{j=1}^{d} W_{j1} \, \delta_j^{k}$$

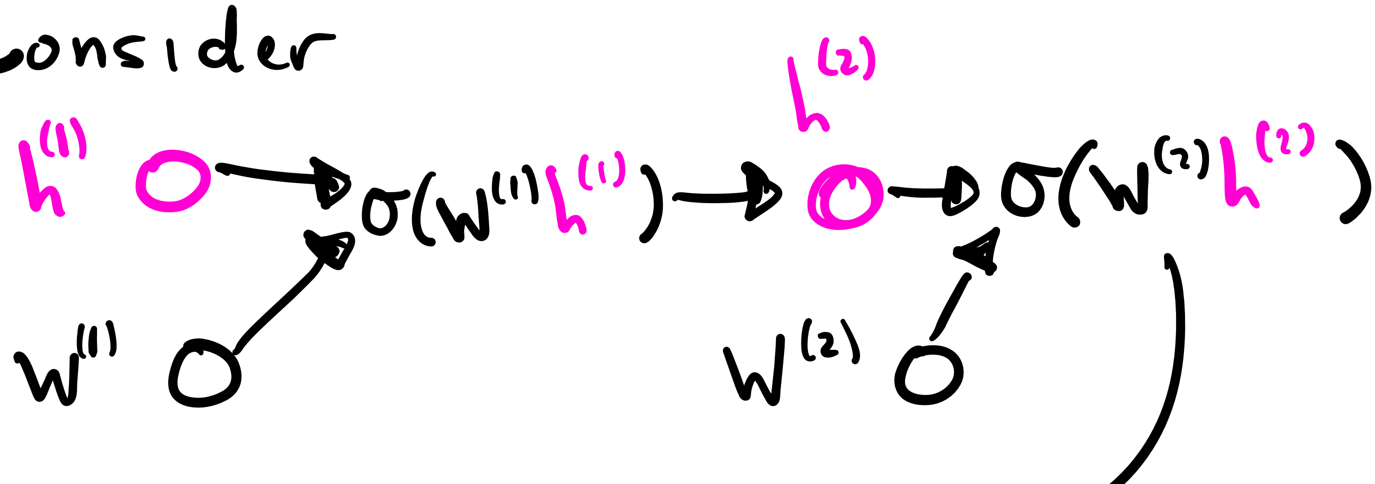
ignoring σ

Exercise! Let's implement a wacky custom layer (Colab)

Memory, Backprop & Detach

- What do we need to store for Backprop @ each layer?

- Consider



$$\underset{\hat{y}}{\circ} \rightarrow \mathcal{L}(\hat{y}, y) \leftarrow \underset{y}{\circledast}$$

$$\frac{\partial}{\partial W^{(1)}} =$$

$$\frac{\partial}{\partial \hat{y}} \mathcal{L} \cdot \underbrace{\frac{\partial}{\partial h^{(2)}} \hat{y} \cdot \frac{\partial}{\partial h^{(1)}} h^{(2)} \cdot \frac{\partial}{\partial W^{(1)}} h^{(1)}}$$

Note that if we freeze lower layers we can discard corresponding activations.