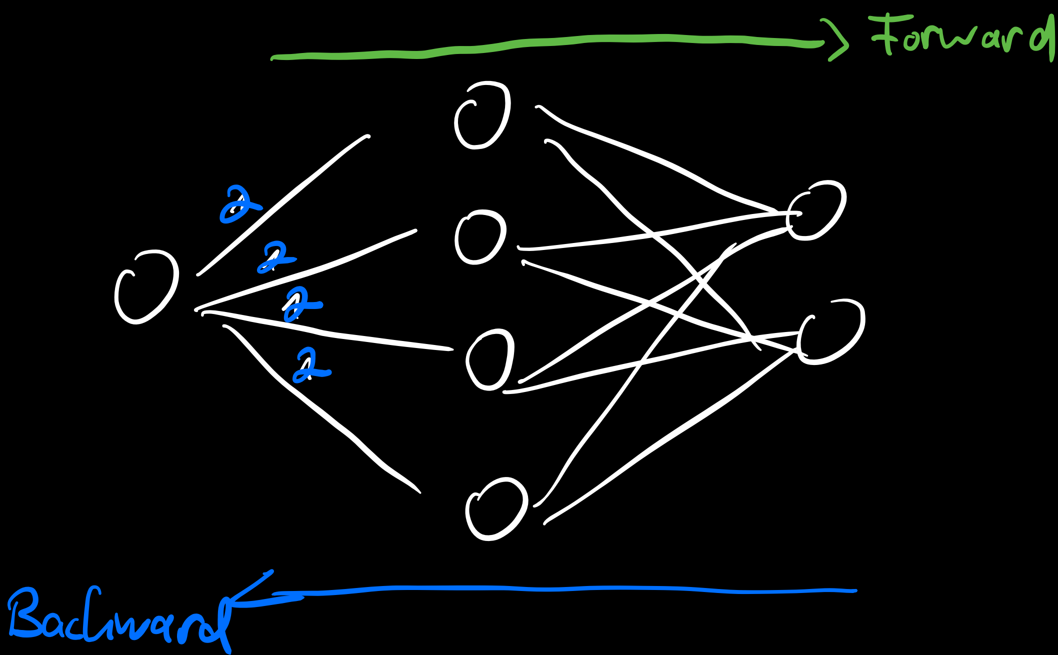
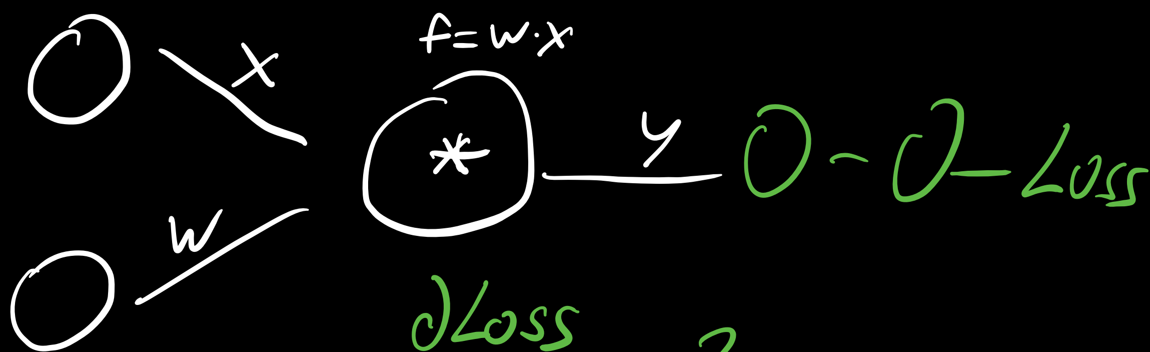
Backpropagation computes the gradients of a loss function with respect to the weights in a neural network.

1. Neural Network



$$W^T \vec{x} + b$$

2. Computational Graph



$$\frac{\partial Loss}{\partial w} = ?$$

3. Chain Rule

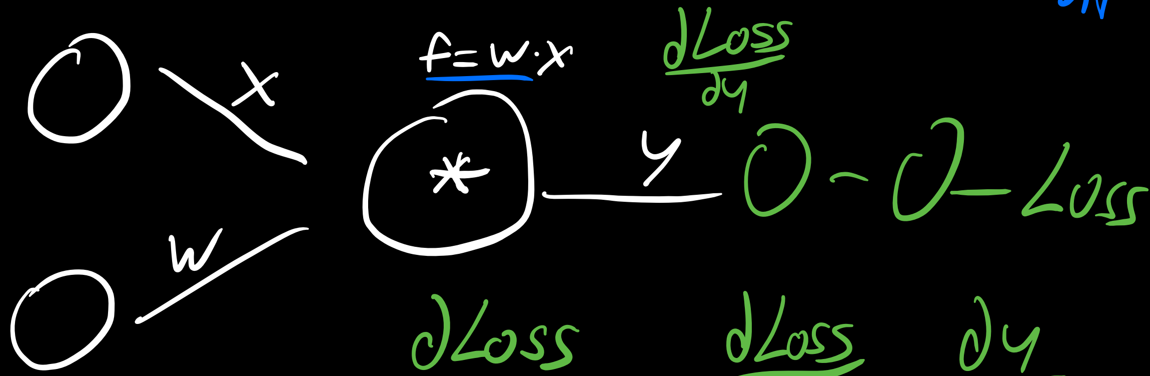
$$\xrightarrow{a} \; f_1(a) \; \xrightarrow{b} \; f_2(b) \; \xrightarrow{c}$$

$$\frac{\partial c}{\partial a} = \frac{\partial c}{\partial b} \cdot \frac{\partial b}{\partial a}$$

2. Computational Graph

$$\frac{\partial y}{\partial w} = \frac{\partial w \cdot x}{\partial w} = x$$

Local gradient



$$\frac{\partial Loss}{\partial w} = \frac{\partial Loss}{\partial y} \cdot \frac{\partial y}{\partial w}$$

1. Forward Pass

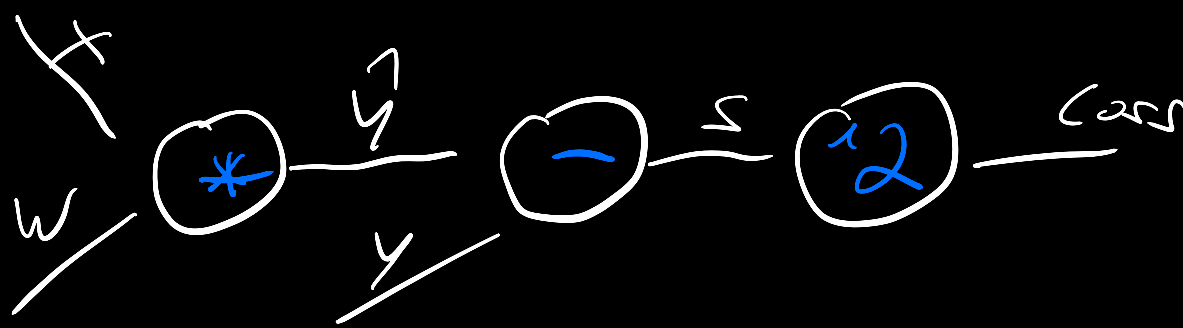
2. Compute Local Gradients

3. Backward Pass

-> Compute $\frac{\partial Loss}{\partial w}$

using Chain Rule

$$\hat{y} = w \cdot x \qquad \text{Loss} = (\hat{y} - y)^2 = (wx - y)^2$$



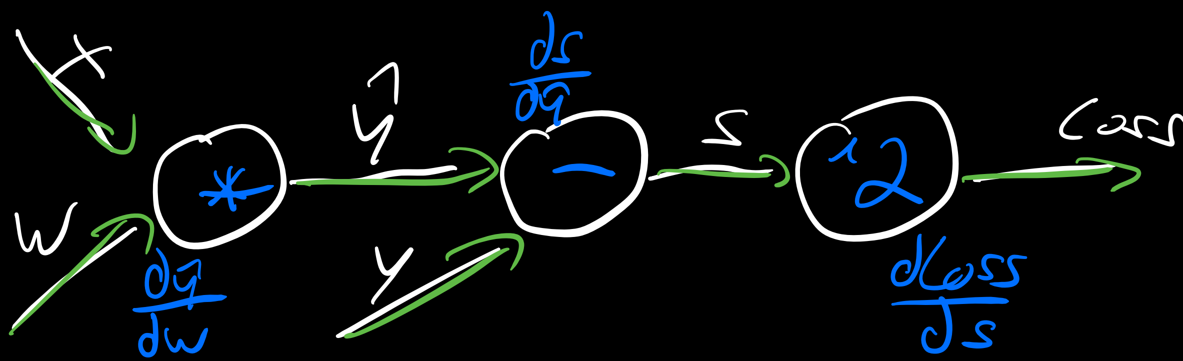
Minimize Loss

$$\Rightarrow \frac{\partial \text{Loss}}{\partial w} = ?$$

$\hat{y} = w \cdot x$

$loss = (\hat{y} - y)^2 = (wx - y)^2$

(1) Forward Pass



(2) Local Gradients

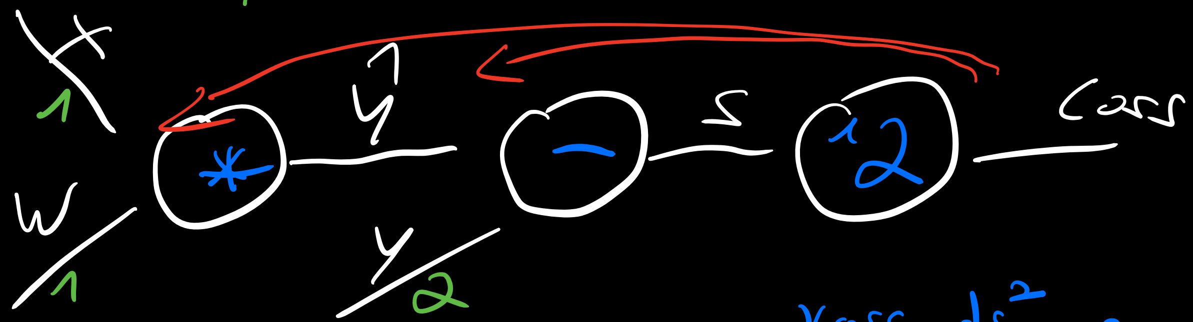
(3) Backward Pass

$$\boxed{\frac{d\,loss}{dw}} \leftarrow \frac{d\,loss}{d\hat{y}} \leftarrow \frac{d\,loss}{ds}$$

Chain Rule

$$\hat{y} = w \cdot x \qquad loss = (\hat{y} - y)^2 = (wx - y)^2$$

$$\hat{y} = 1 \cdot 1 = 1 \qquad s = 1 - 2 = -1 \qquad loss = (-1)^2 = 1$$



$$\frac{\partial \hat{y}}{\partial w} = \frac{\partial w \cdot x}{\partial w} = x \qquad \frac{\partial s}{\partial \hat{y}} = \frac{\partial \hat{y} - y}{\partial \hat{y}} = 1 \qquad \frac{\partial loss}{\partial s} = \frac{\partial s^2}{\partial s} = 2s$$

$$\frac{\partial loss}{\partial w} = \frac{\partial loss}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial w} = -2 \cdot x = -2 \qquad \frac{\partial loss}{\partial \hat{y}} = \frac{\partial loss}{\partial s} \cdot \frac{\partial s}{\partial \hat{y}} = 2 \cdot s \cdot 1 = -2$$

$$\boxed{\frac{\partial loss}{\partial w} = -2}$$