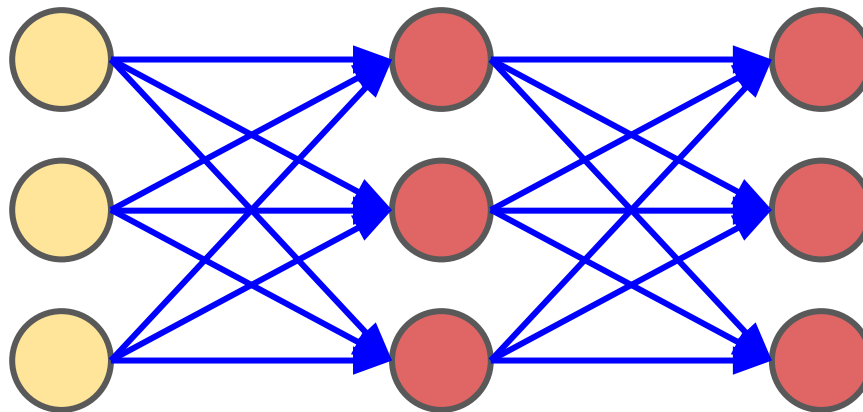


Neural Networks

1. Single Layer
2. Single Neuron
3. Multiple Layers
4. Input and Output



What does a single Neural Network Layer do ?

$$\begin{bmatrix} 2 & 4 & 2 \\ 1 & 3 & 7 \\ 0 & 7 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 18 \\ 24 \\ 27 \end{bmatrix}$$

What does a single Neural Network Layer do

$$\begin{bmatrix} 18 \\ 24 \\ 27 \end{bmatrix} + \begin{bmatrix} -3 \\ 0 \\ 2 \end{bmatrix}$$

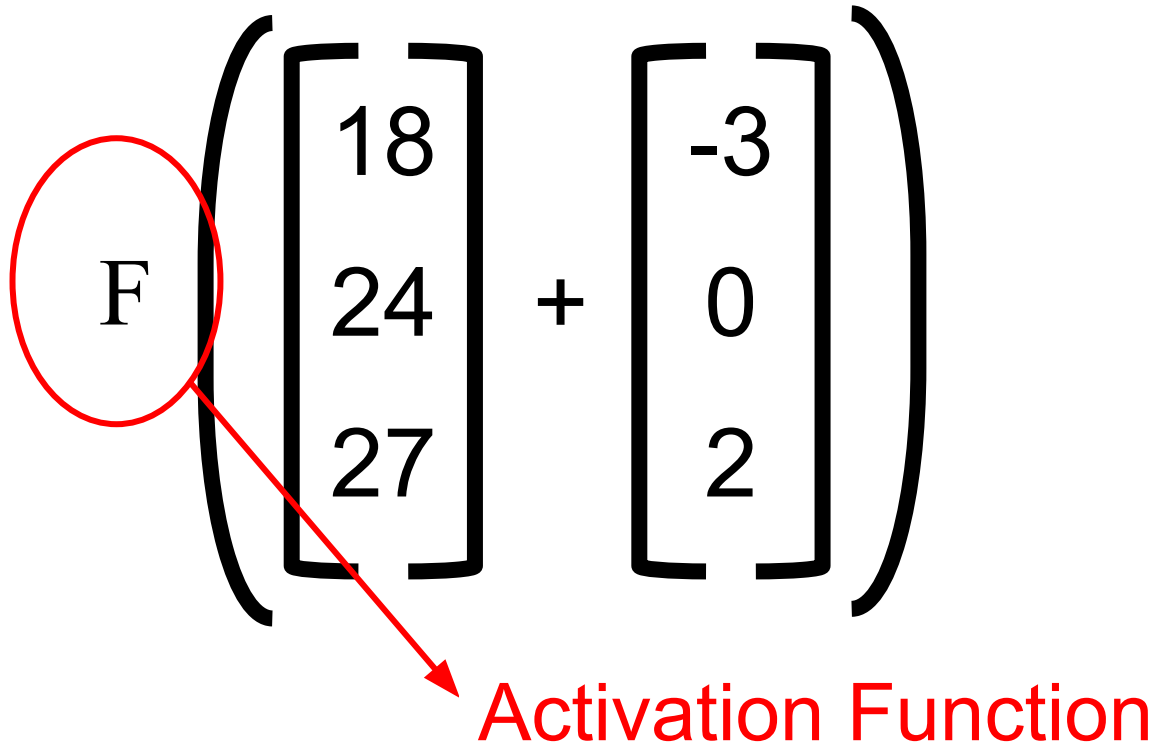
What does a single Neural Network Layer do

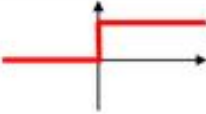
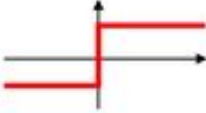
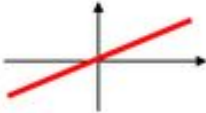

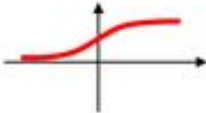

$$F \left(\begin{bmatrix} 18 \\ 24 \\ 27 \end{bmatrix} + \begin{bmatrix} -3 \\ 0 \\ 2 \end{bmatrix} \right)$$

What does a single Neural Network Layer do

$$F \left(\begin{bmatrix} 18 \\ 24 \\ 27 \end{bmatrix} + \begin{bmatrix} -3 \\ 0 \\ 2 \end{bmatrix} \right)$$

Activation Function

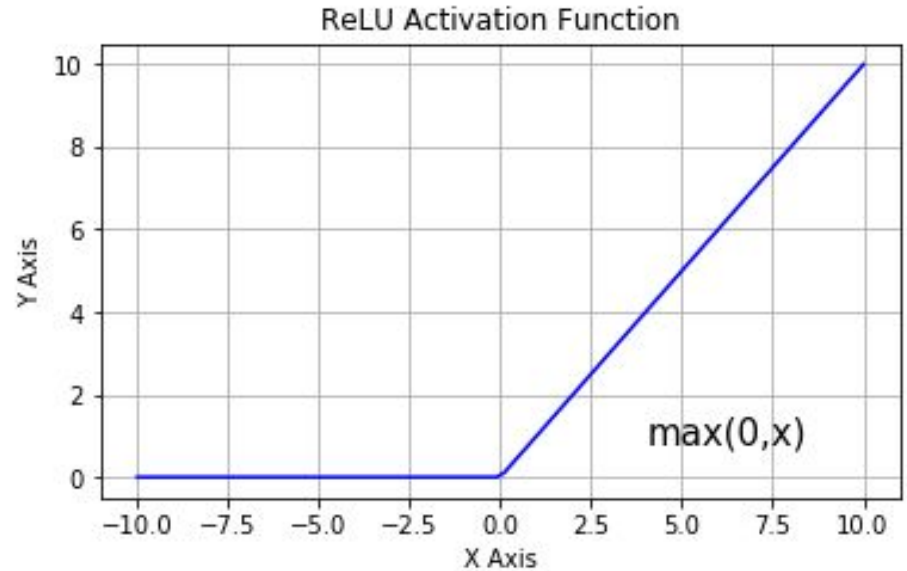
The diagram illustrates the operation of a single neural network layer. It shows a function F, which is circled in red, applied to the sum of two vectors. The first vector is [18, 24, 27] and the second vector is [-3, 0, 2]. A red arrow points from the circled 'F' to the text 'Activation Function' below the equation.

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer NN	

ReLU Activation function

It stands for Rectified Linear Unit

$$\text{ReLU}(x) = \max(x, 0)$$



What does a single Neural Network Layer do

$$F \left(\begin{bmatrix} 18 \\ 24 \\ 27 \end{bmatrix} + \begin{bmatrix} -3 \\ 0 \\ 2 \end{bmatrix} \right) = \begin{bmatrix} F(15) \\ F(24) \\ F(29) \end{bmatrix}$$

What does a single Neural Network Layer do ?

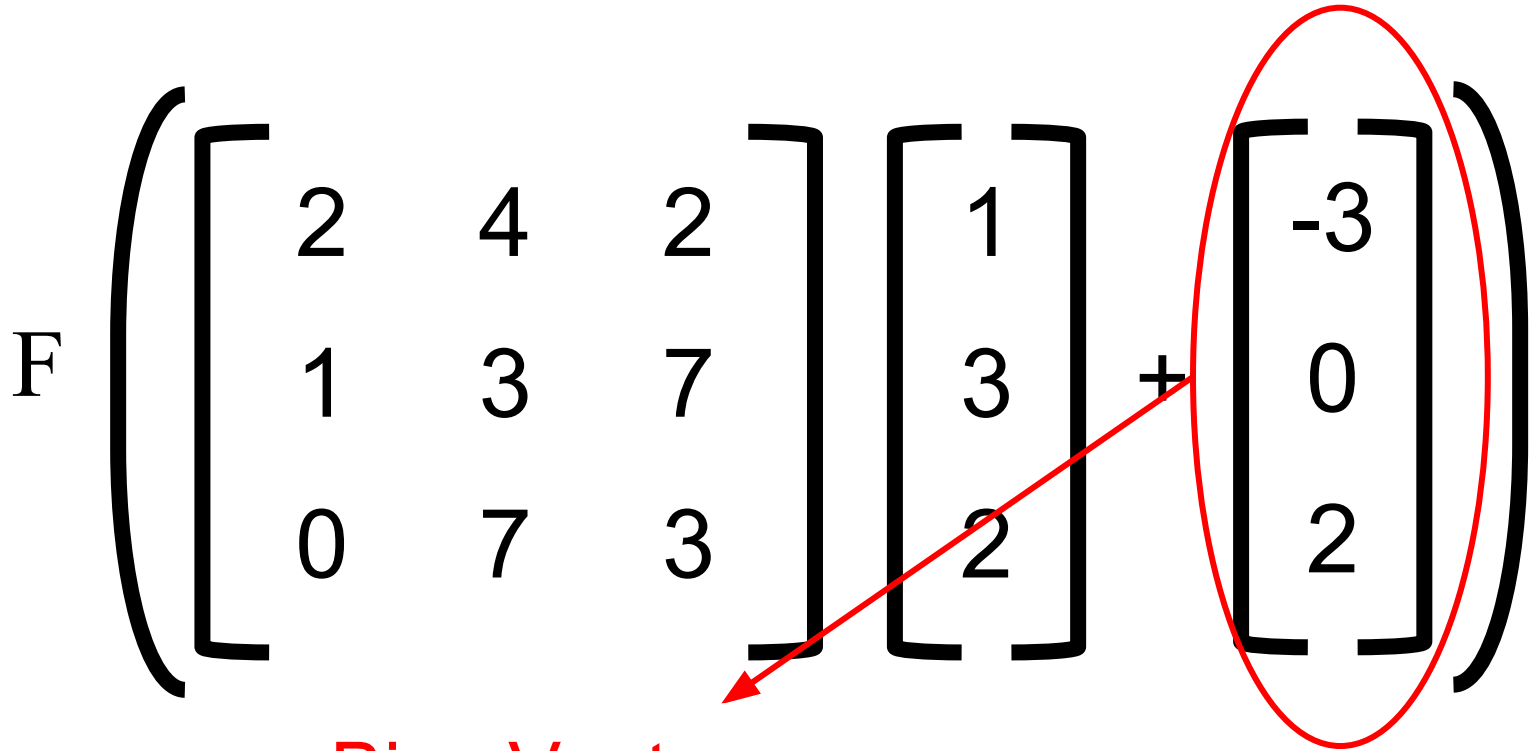
$$F \left(\begin{bmatrix} 2 & 4 & 2 \\ 1 & 3 & 7 \\ 0 & 7 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} + \begin{bmatrix} -3 \\ 0 \\ 2 \end{bmatrix} \right)$$

What does a single Neural Network Layer do ?

$$F \left(\begin{bmatrix} 2 & 4 & 2 \\ 1 & 3 & 7 \\ 0 & 7 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} + \begin{bmatrix} -3 \\ 0 \\ 2 \end{bmatrix} \right)$$

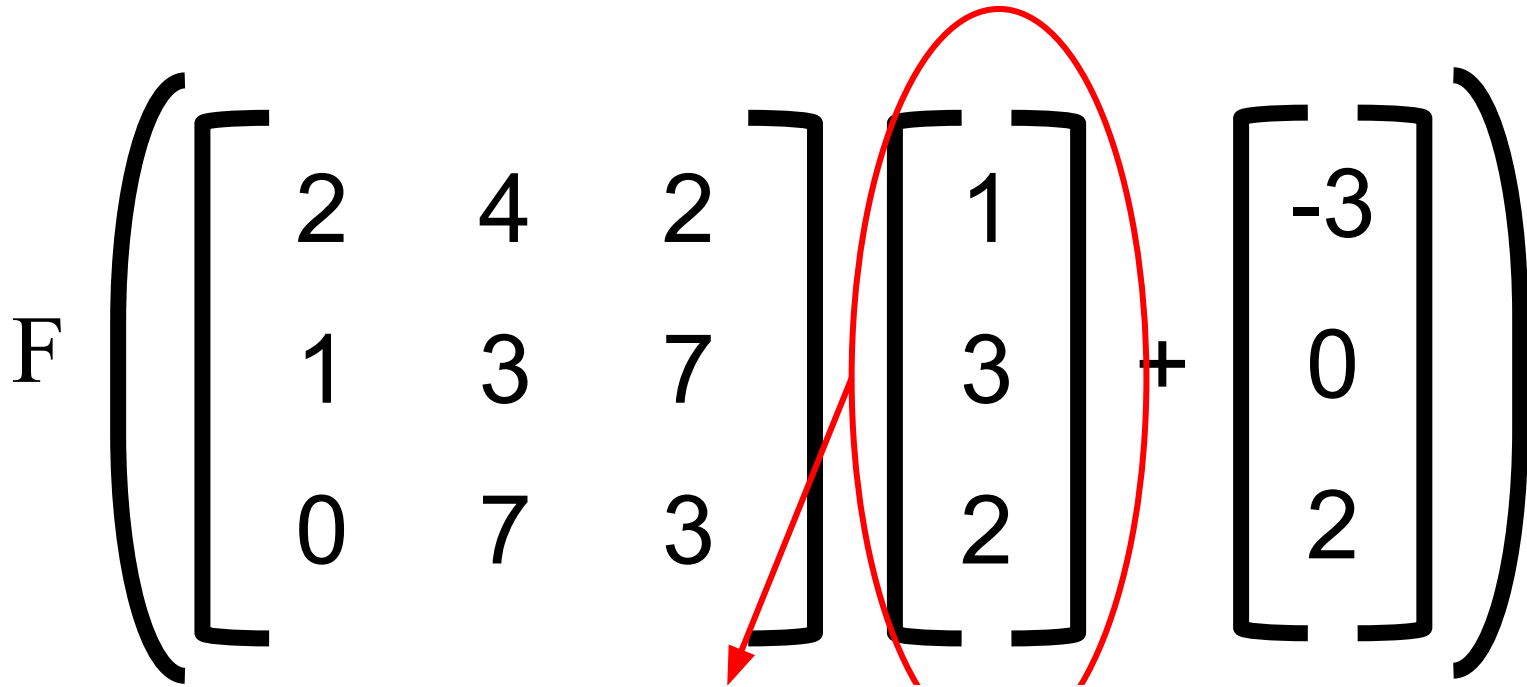
Weight Matrix

What does a single Neural Network Layer do ?

$$F \left(\begin{bmatrix} 2 & 4 & 2 \\ 1 & 3 & 7 \\ 0 & 7 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} + \begin{bmatrix} -3 \\ 0 \\ 2 \end{bmatrix} \right)$$


Bias Vector

What does a single Neural Network Layer do ?

$$F \left(\begin{bmatrix} 2 & 4 & 2 \\ 1 & 3 & 7 \\ 0 & 7 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} + \begin{bmatrix} -3 \\ 0 \\ 2 \end{bmatrix} \right)$$


Input Vector

What does a single Neural Network Layer do ?

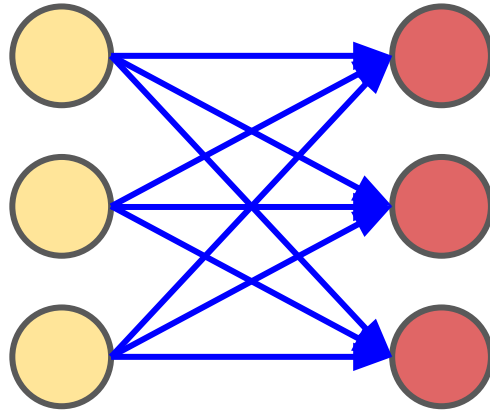
$$F \left(\begin{bmatrix} 2 & 4 & 2 \\ 1 & 3 & 7 \\ 0 & 7 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} + \begin{bmatrix} -3 \\ 0 \\ 2 \end{bmatrix} \right)$$

Trainable Parameters

What does a single Neural Network Layer do ?

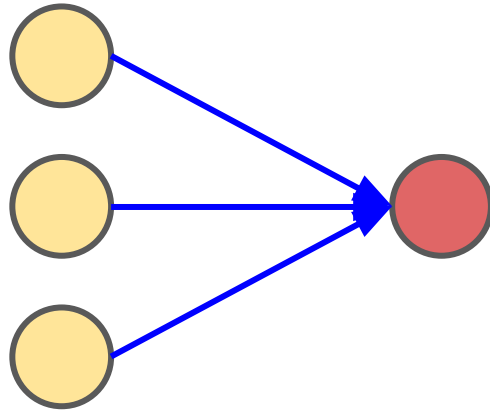
$$F(\mathbf{W}\mathbf{x} + \mathbf{b})$$

What does a single Neural Network Layer do ?



$$F(\mathbf{W}\mathbf{x} + \mathbf{b})$$

What does a single Neuron do ?

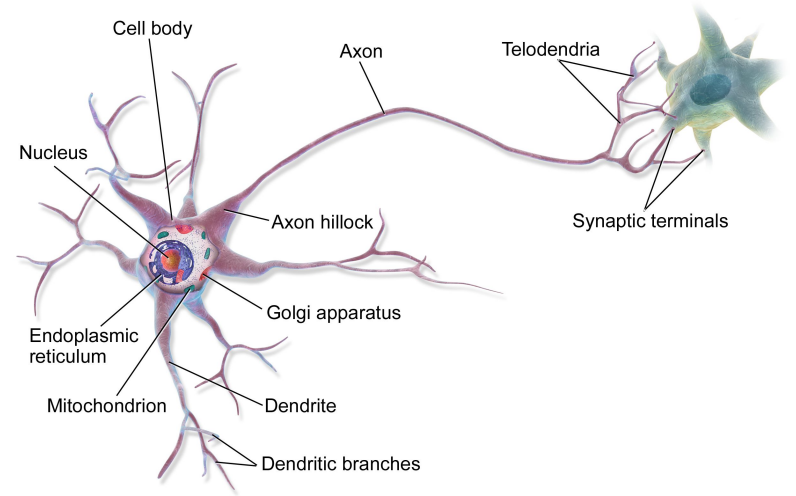
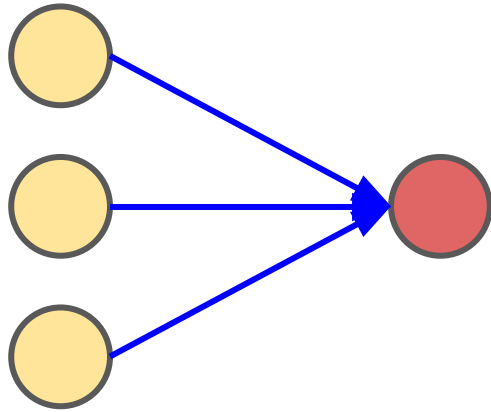


$$F(\mathbf{w}_1^T \mathbf{x} + b_1)$$

Weights for each neuron

$$W = \begin{bmatrix} 2 & 4 & 2 \\ 1 & 3 & 7 \\ 0 & 7 & 3 \end{bmatrix} = \begin{bmatrix} w_1^T \\ w_2^T \\ w_3^T \end{bmatrix}$$

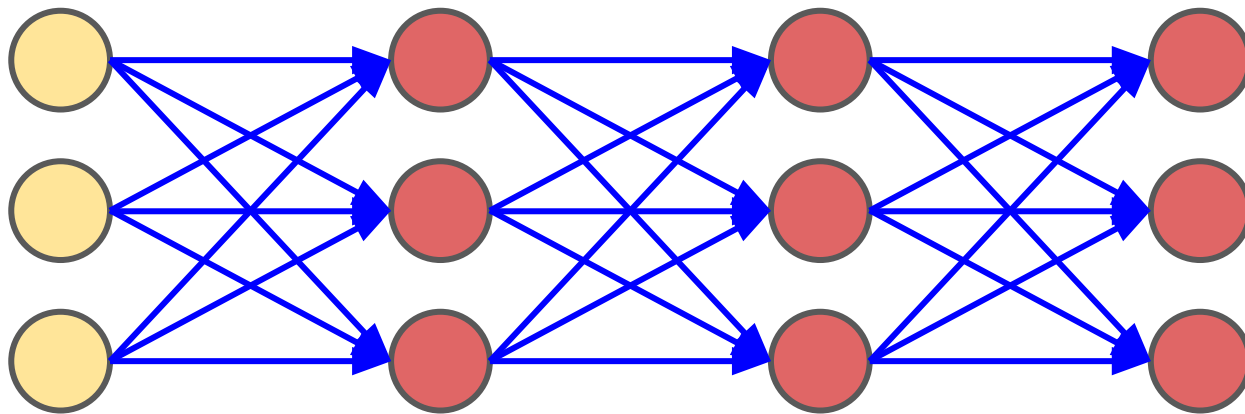
What does a single Neuron do ?



$$F\left(\mathbf{w}_1^T \mathbf{x} + b_1\right)$$

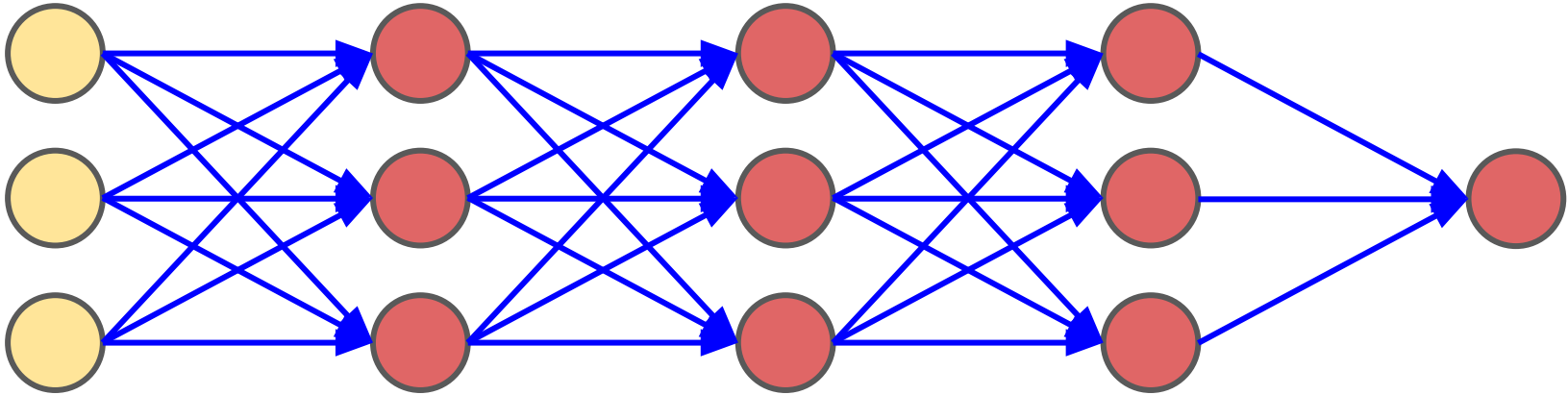
[Playground](#)

Stacking multiple layers



It's called a *deep* neural network when you use multiple layers

What are the inputs and outputs?



Gradient Descent

1. The Idea is to go downhill according to the linear approximation of your function
2. But not too much as your approximation will not be true
3. This means updating each weight in proportional to the gradient

$$w' = w - \alpha g$$

Gradient Descent

1. The Idea is to go downhill according to the linear approximation of your function
2. But not too much as your approximation will not be true
3. This means updating each weight in proportional to the gradient

$$w' = w - \alpha g$$

Learning Rate

How to compute gradients - Backpropagation

Numerically approximating gradients is too slow and computationally expensive

The neural network is a well defined mathematical function, we should be able to differentiate it and calculate the derivatives

This is also too tedious so we instead use an algorithm called backpropagation

Computational Graph

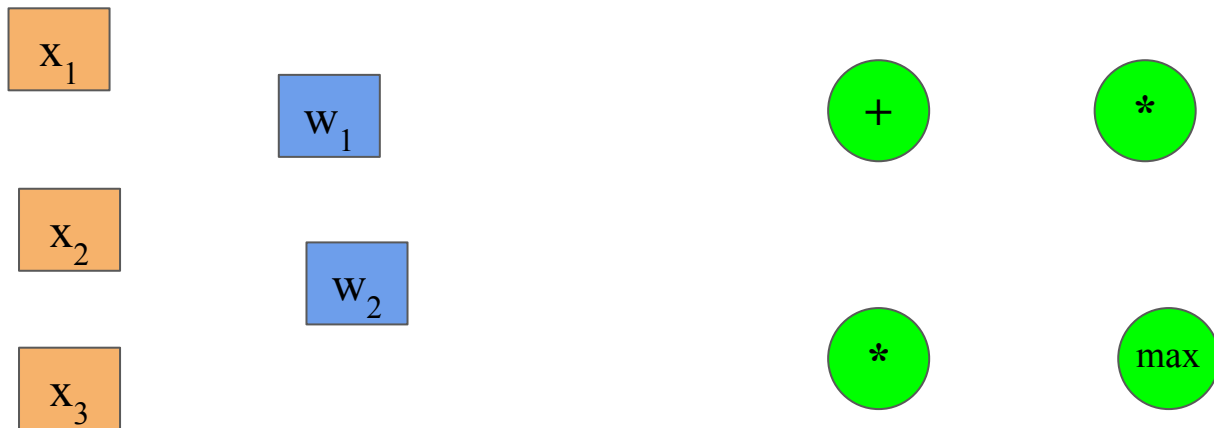
$$F(x_1, x_2, x_3) = \max(w_1x_1 + w_2x_2, x_3)$$

Consider this simple function, imagine that this is the cost function and we want to find the gradients with respect to the two weights

We will first build the computational graph that this function represents

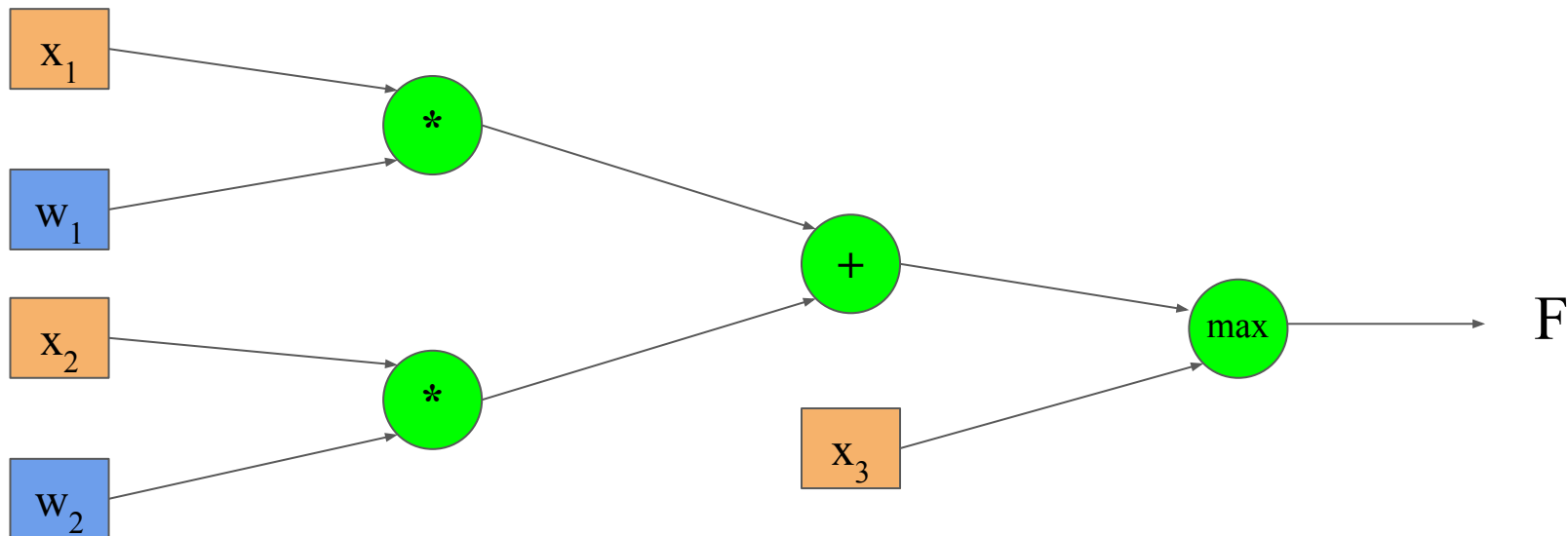
Computational Graph

$$F(x_1, x_2, x_3) = \max(w_1x_1 + w_2x_2, x_3)$$

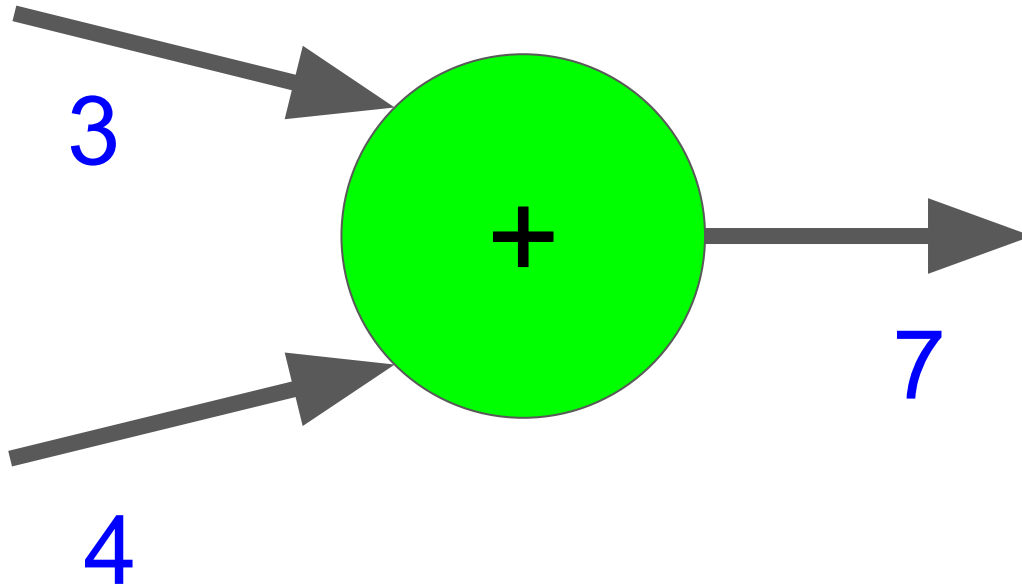


Computational Graph

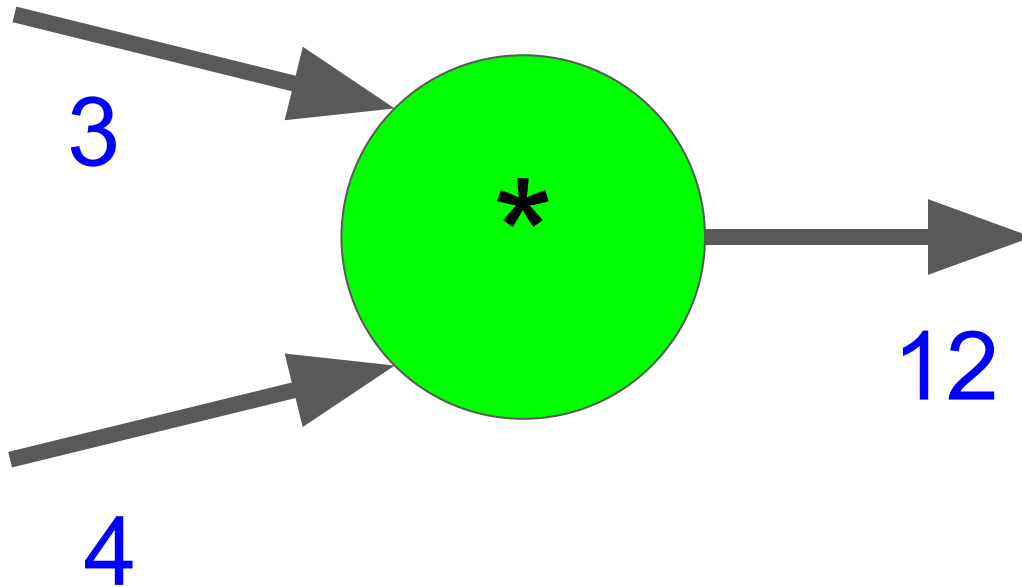
$$F(x_1, x_2, x_3) = \max(w_1x_1 + w_2x_2, x_3)$$



Forward propagation

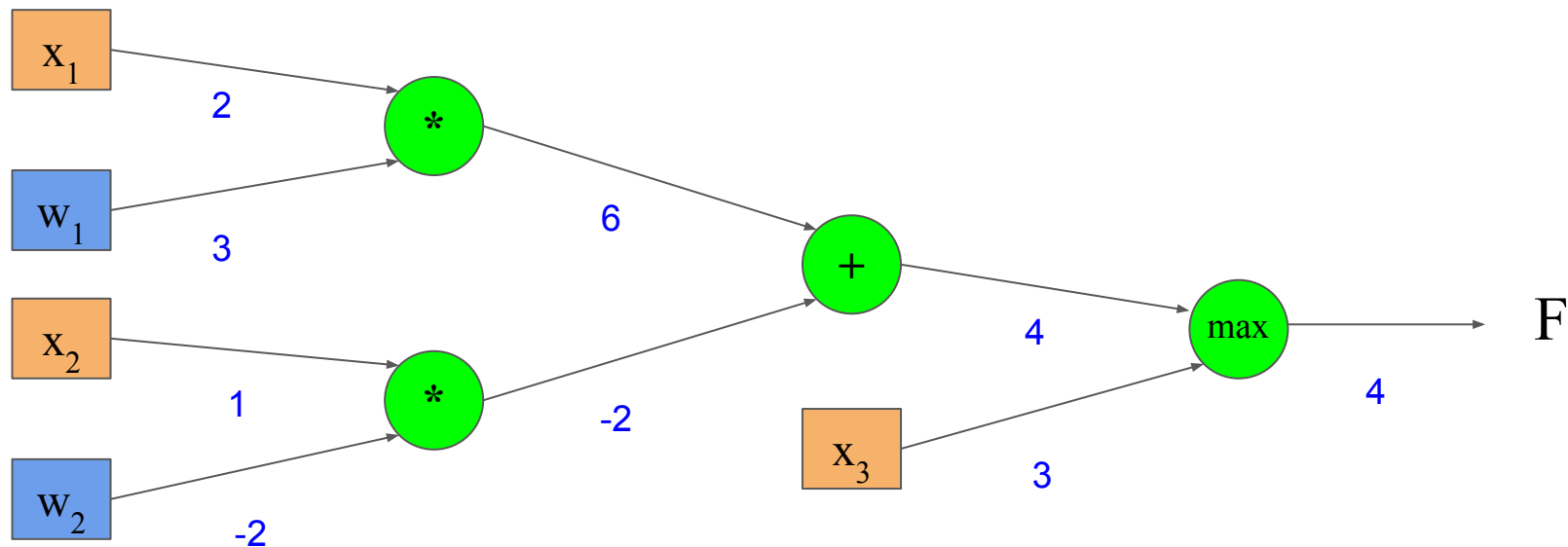


Forward propagation



Forward propagation

$$F(x_1, x_2, x_3) = \max(w_1x_1 + w_2x_2, x_3)$$



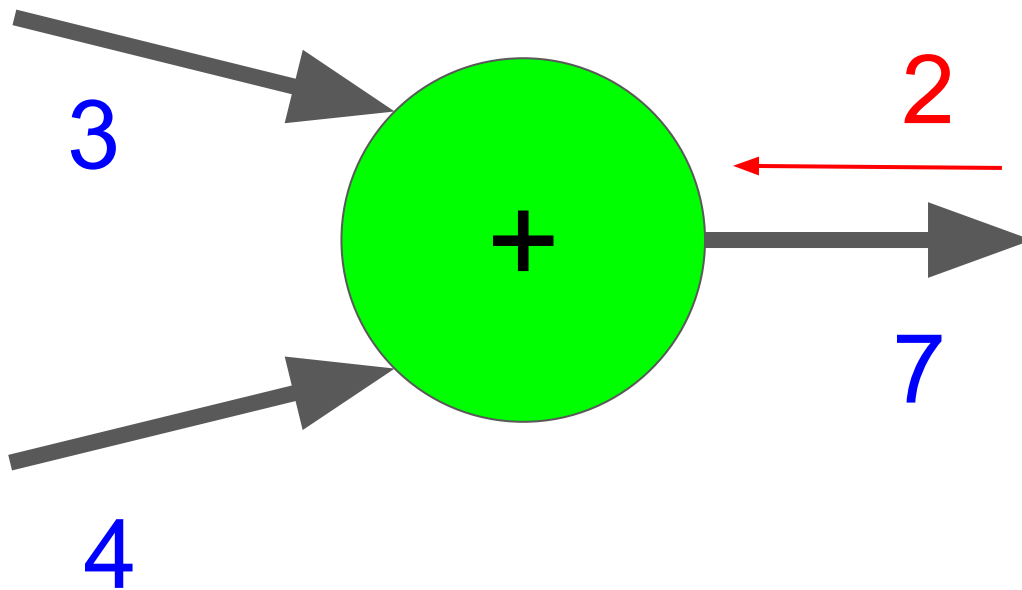
Forward propagation

1. Each **Circle** represents a computational operation
2. Each **arrow** is an intermediate quantity that is computed
3. Each box contains some value
4. **Orange** boxes are inputs
5. **Blue** boxes are parameters
6. Compute the inputs to an operation first then apply the operation to compute its output

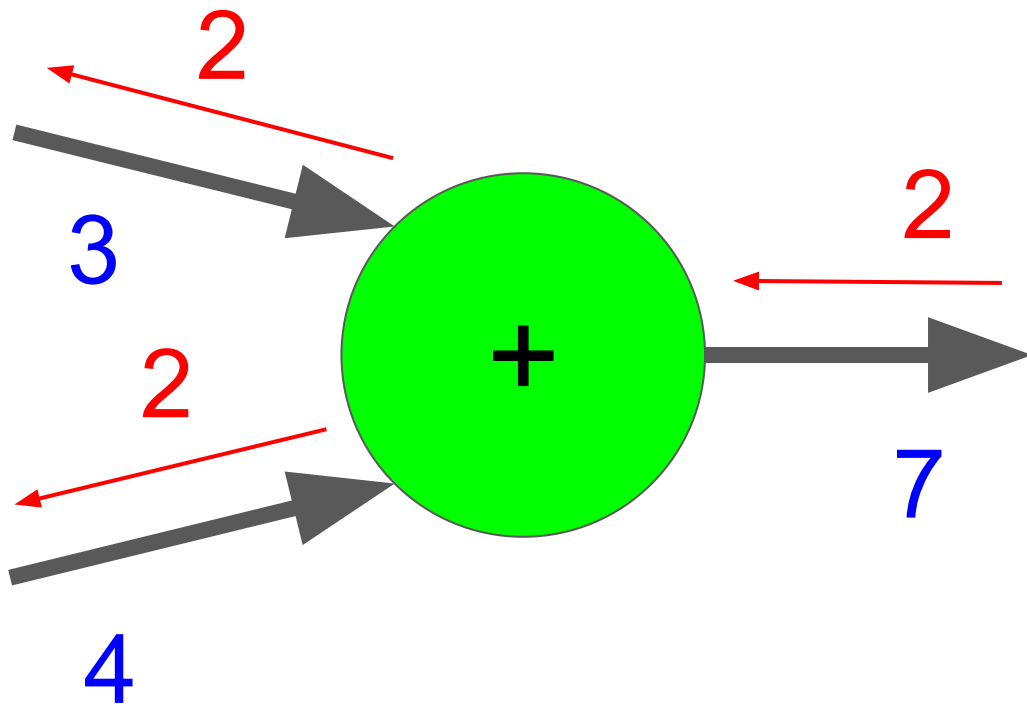
Backward propagation

1. The numbers flow backwards along the arrows
2. At each arrow the quantity represents the gradient of some final function with respect to the quantity at that arrow

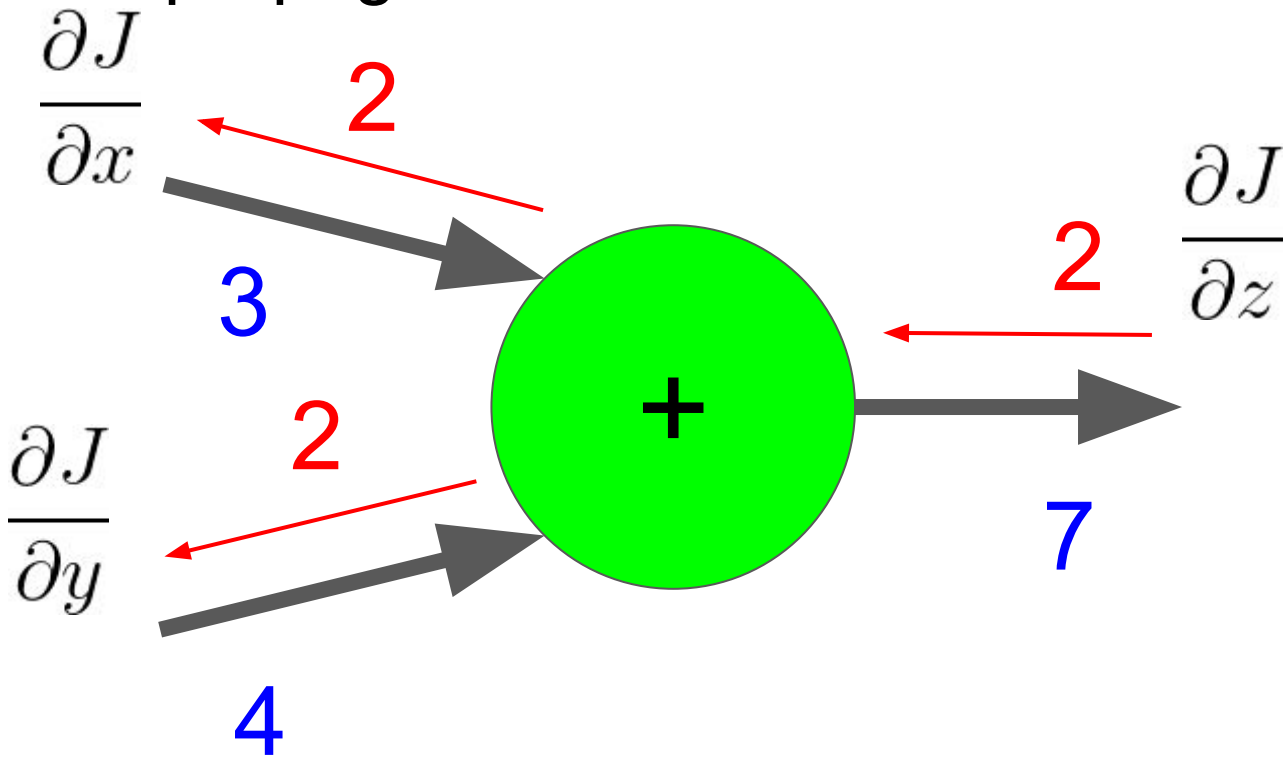
Backward propagation



Backward propagation

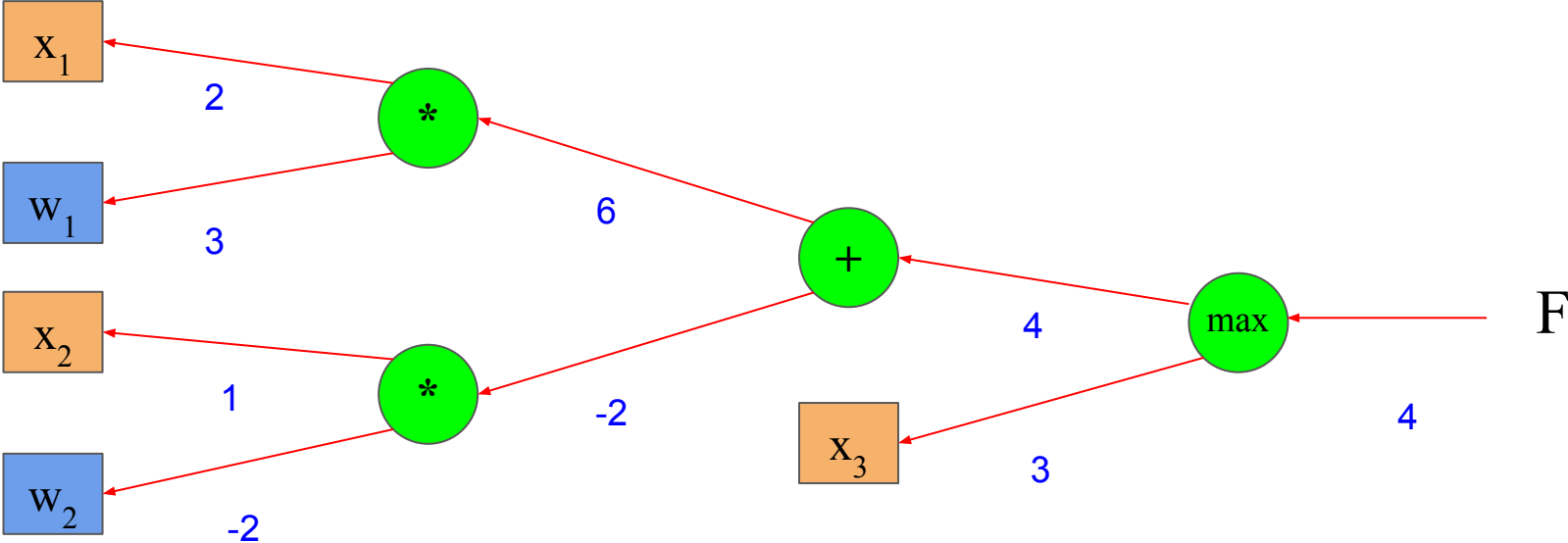


Backward propagation



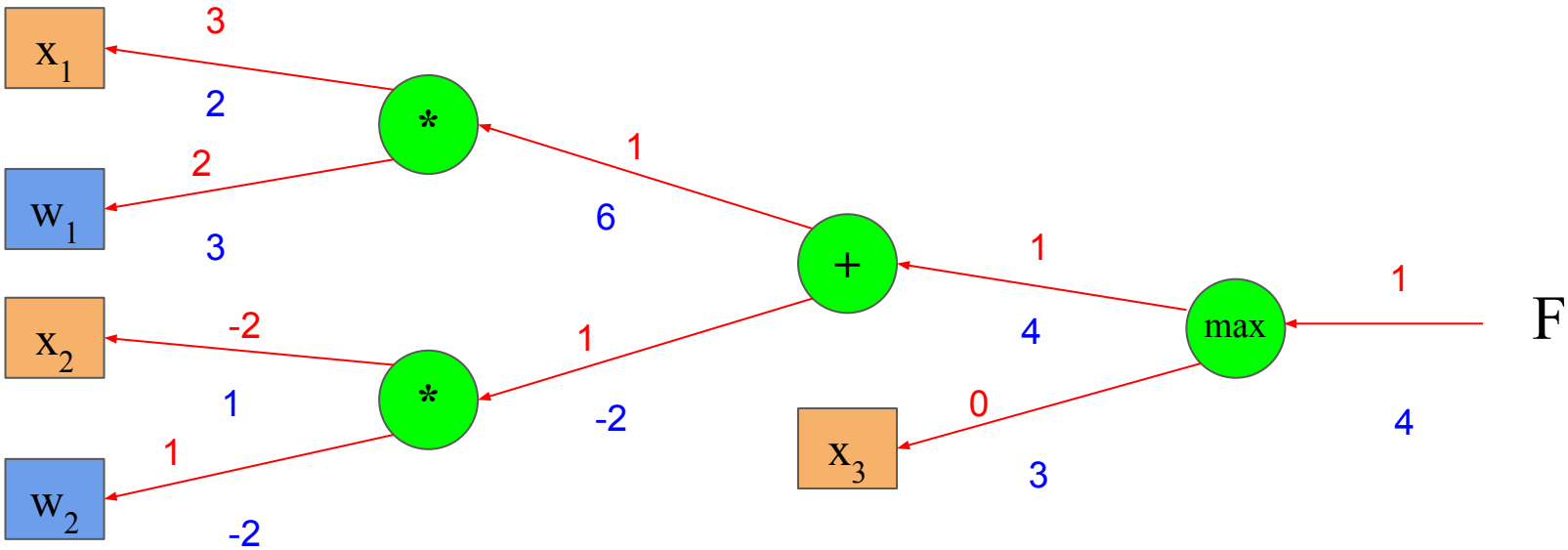
Backward propagation

$$F(x_1, x_2, x_3) = \max(w_1x_1 + w_2x_2, x_3)$$



Backward propagation

$$F(x_1, x_2, x_3) = \max(w_1x_1 + w_2x_2, x_3)$$

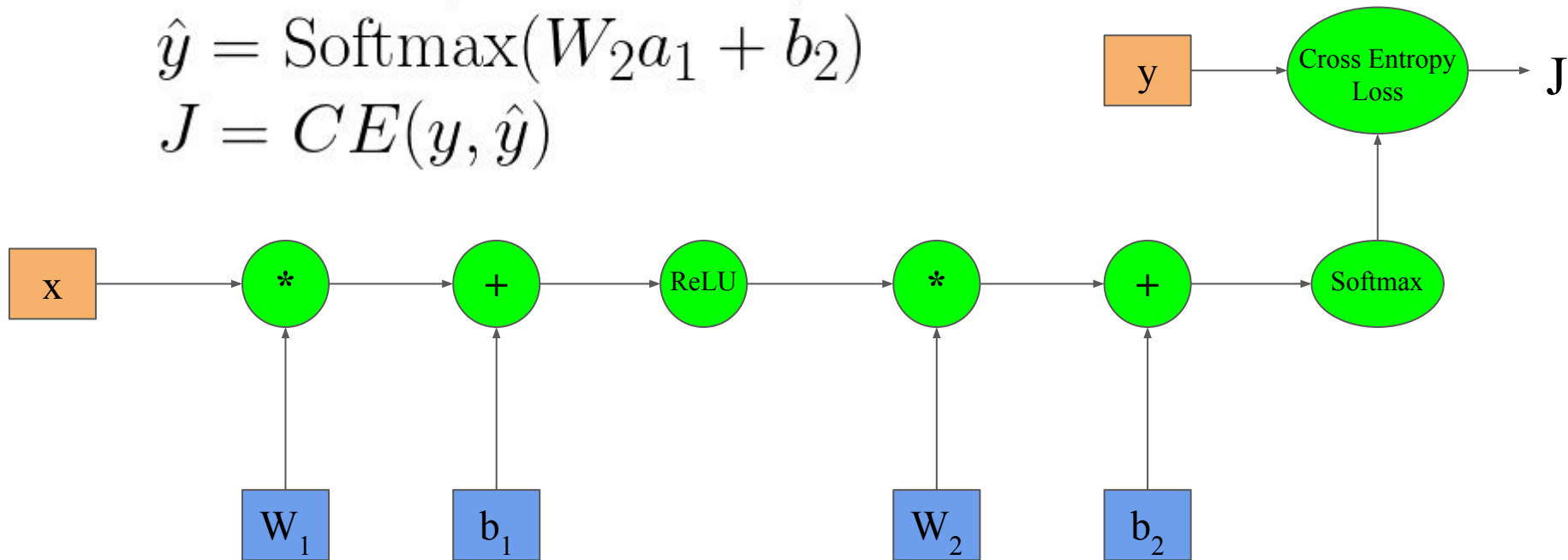


Two Layer Example

$$a_1 = \max(W_1 x + b_1, 0)$$

$$\hat{y} = \text{Softmax}(W_2 a_1 + b_2)$$

$$J = CE(y, \hat{y})$$



Two Layer Example

$$a_1 = \max(W_1 x + b_1, 0)$$

$$\hat{y} = \text{Softmax}(W_2 a_1 + b_2)$$

$$J = CE(y, \hat{y})$$

