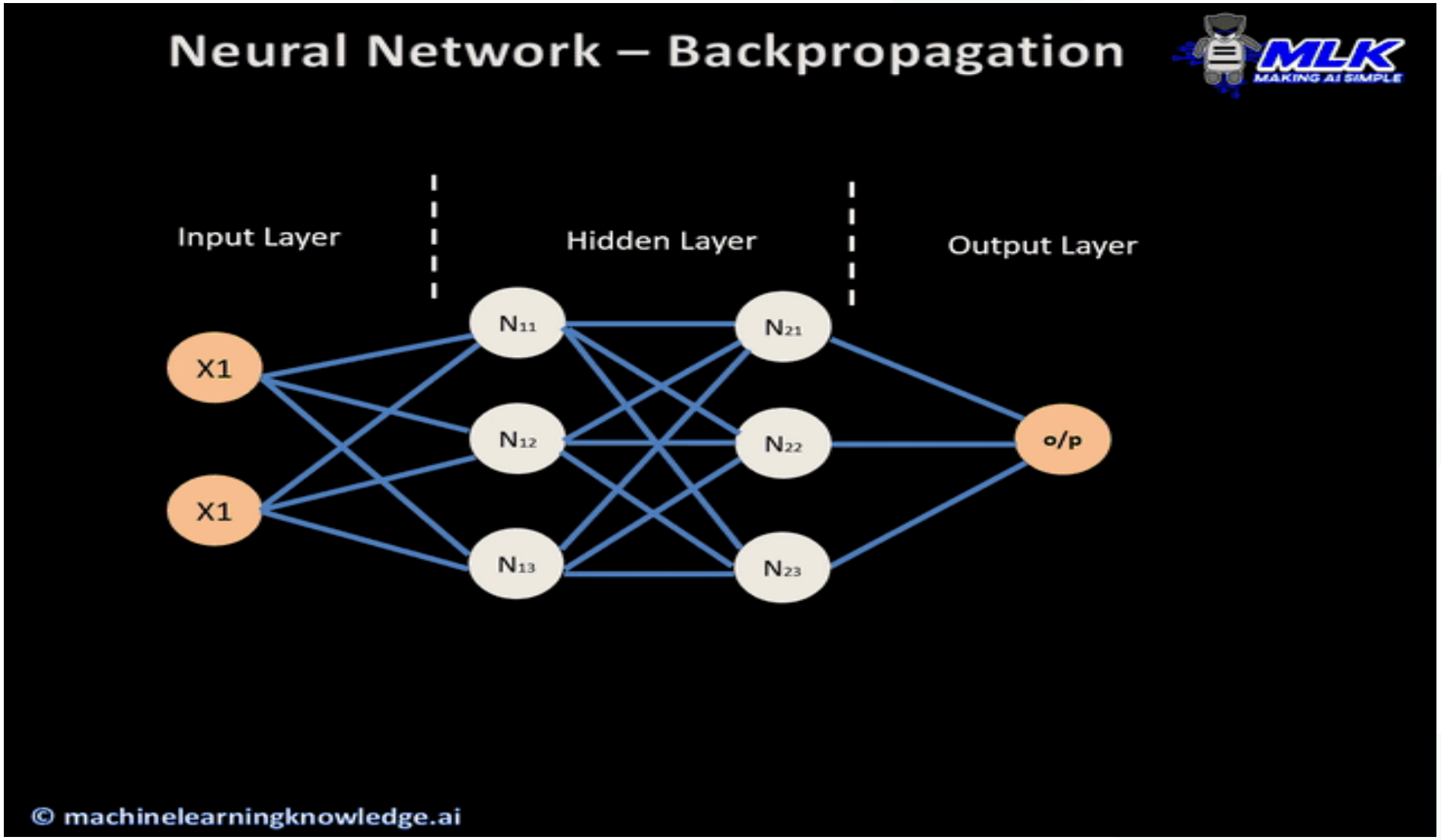
BACKPROPAGATION

Backward Propagation of Errors

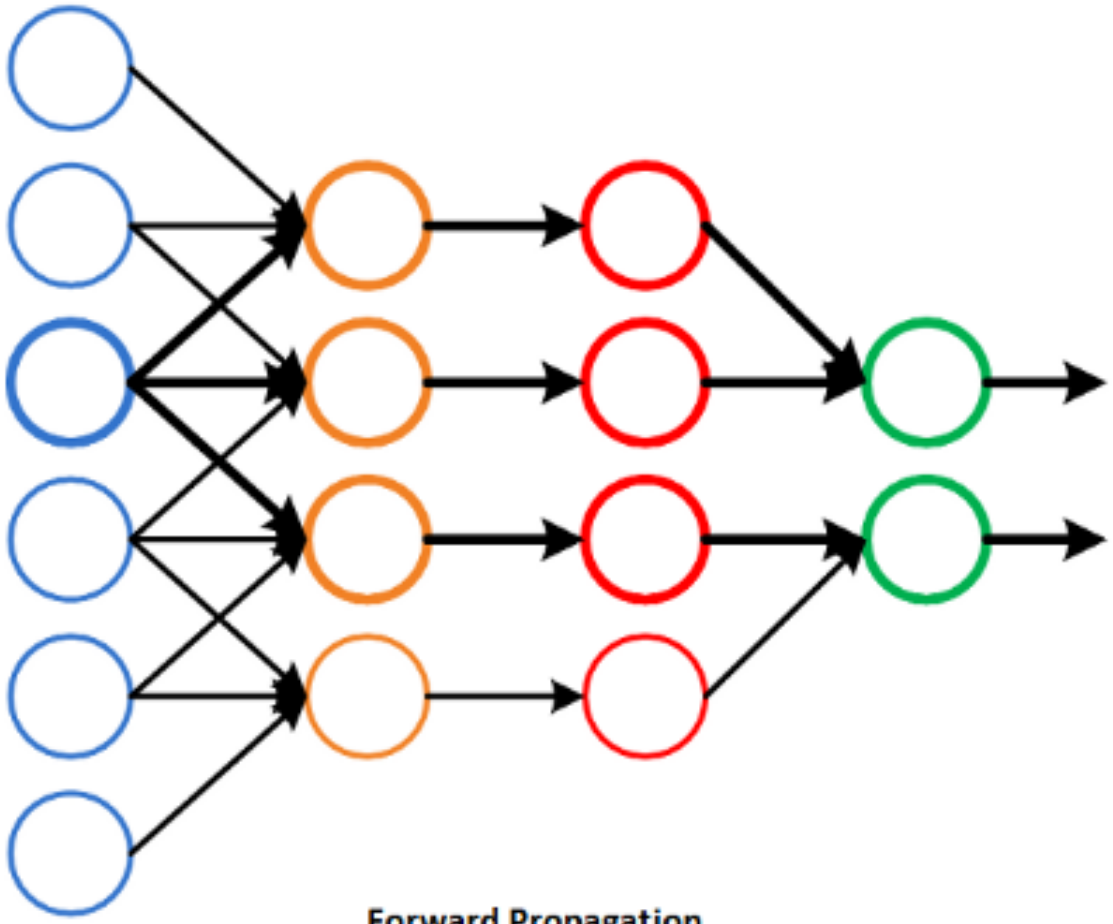
Backpropagation, short for "backward propagation of errors", is a mechanism used to update the **weights** using gradient descent. It calculates the gradient of the error function with respect to the neural network's weights. The calculation proceeds backwards through the network.

> **Gradient descent** is an iterative optimization algorithm for finding the minimum of a function; in our case we want to minimize th error function. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient of the function at the current point.

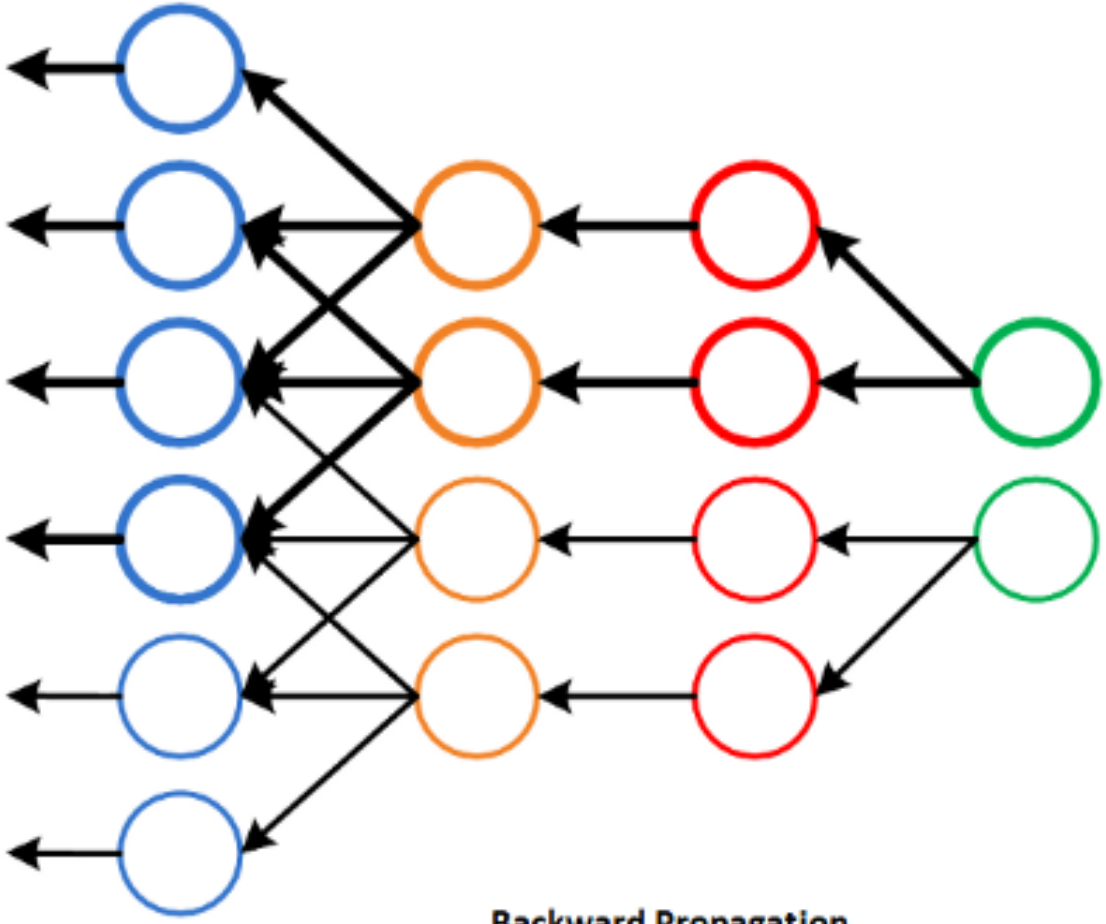
BACKPROPAGATION



BACKPROPAGATION

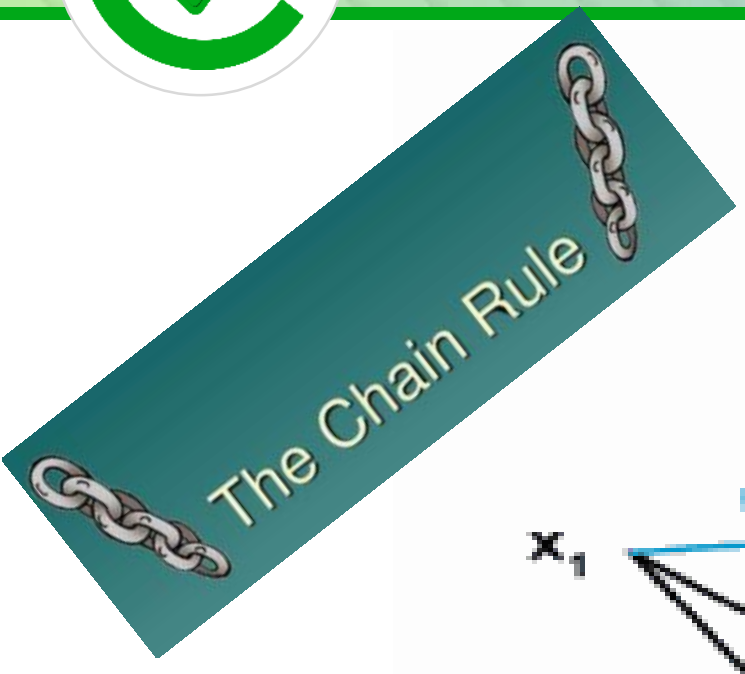


Forward Propagation

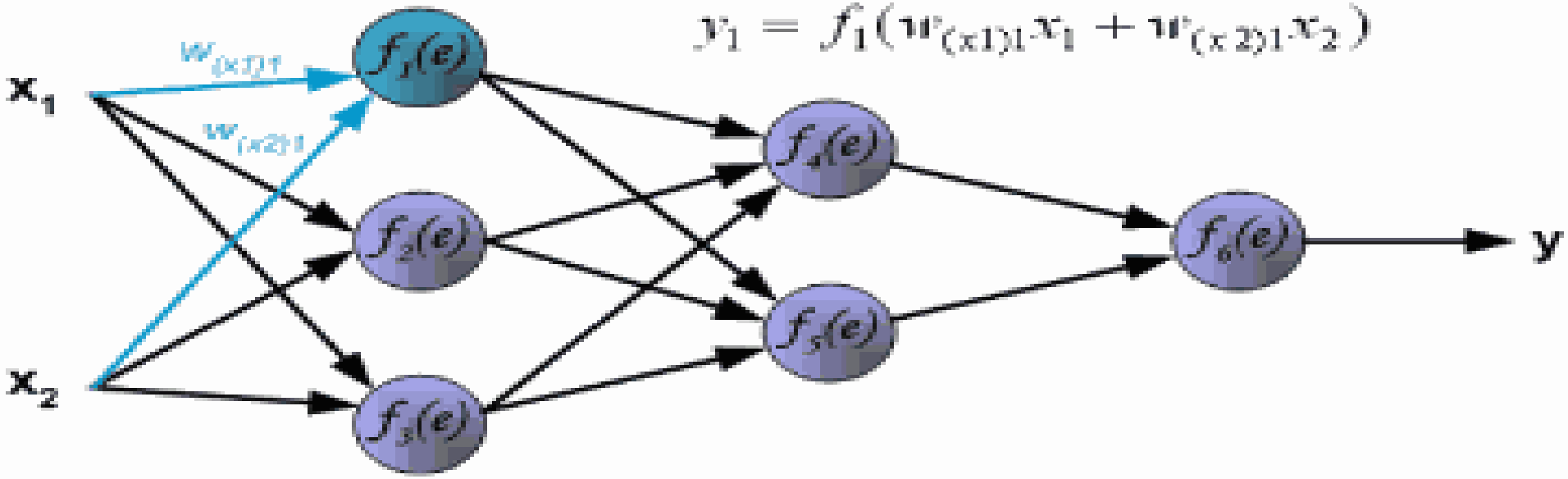


Backward Propagation

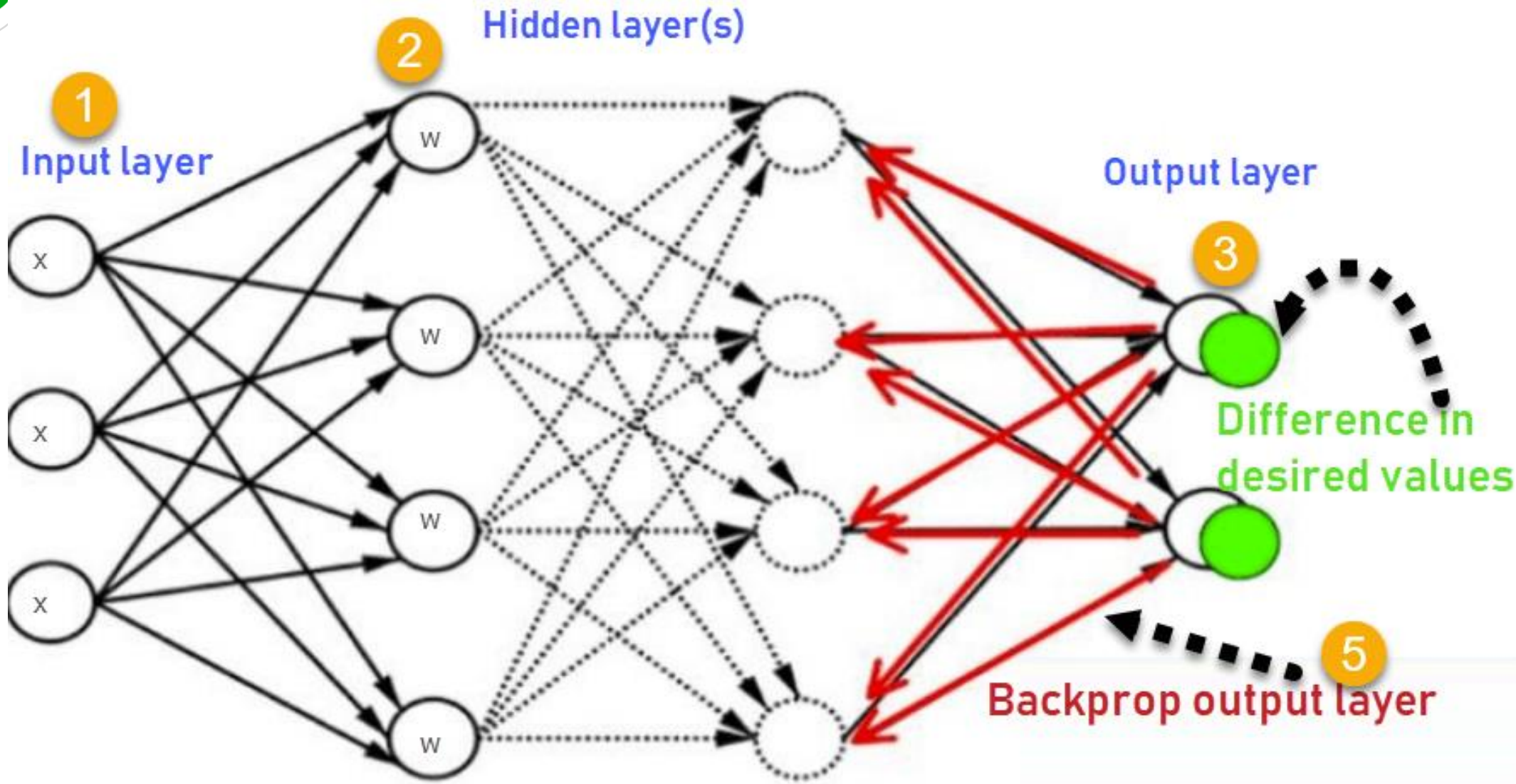
BACKPROPAGATION



$$y_1 = f_1(w_{(x1)1}x_1 + w_{(x2)1}x_2)$$



BACKPROPAGATION



BACKPROPAGATION

$$W_x = W_x - a\left(\frac{\partial Error}{\partial W_x}\right)$$

Old weight

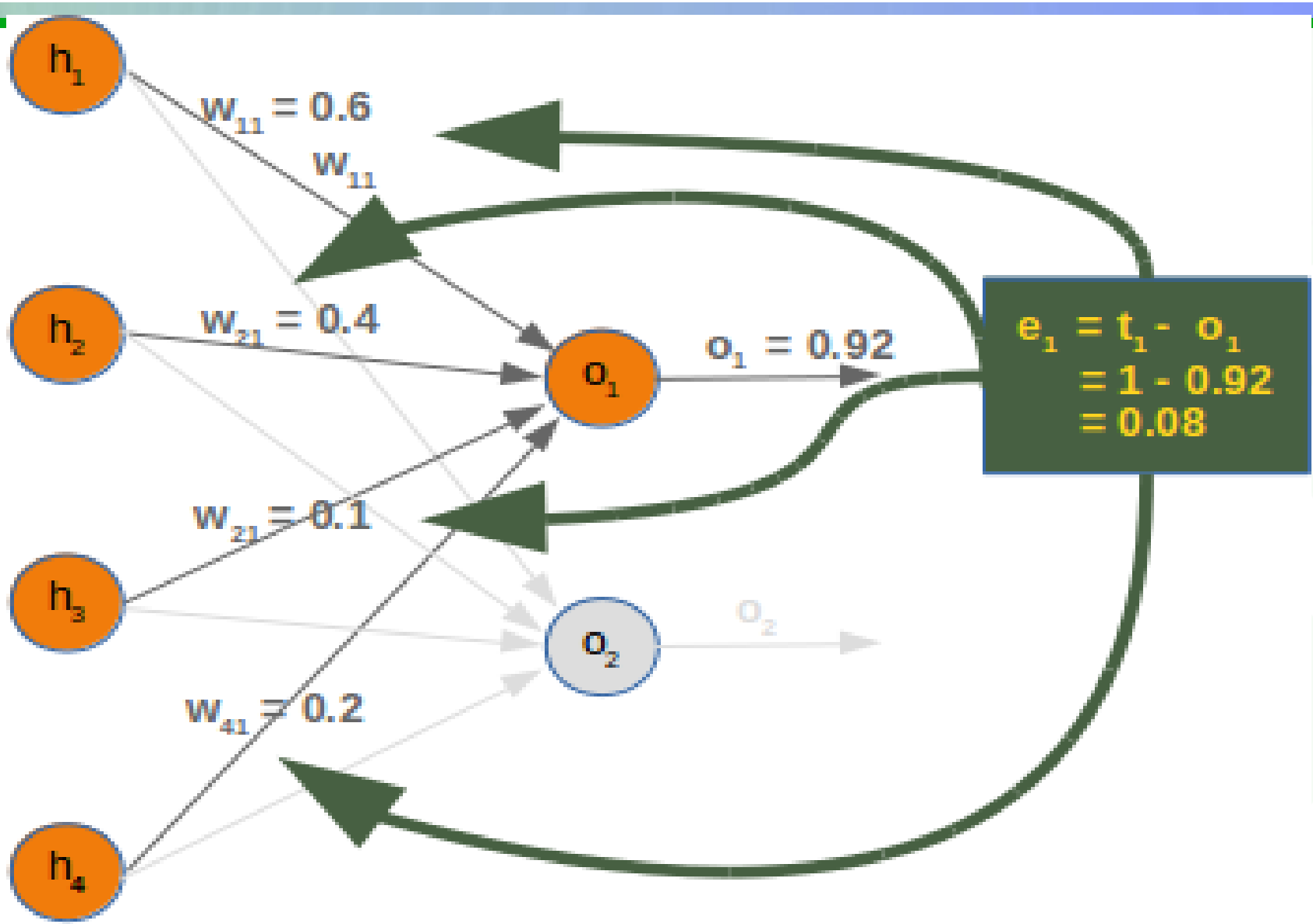
Derivative of Error with respect to weight

New weight

Learning rate

Step Size = slope x learning rate

New Parameter = Old Parameter - Step Size



$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - a\,\Delta \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - \begin{bmatrix} a h_1 \Delta \\ a h_2 \Delta \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - a\,\Delta \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} . [w_5 \quad w_6] = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \begin{bmatrix} a\, i_1 \Delta w_5 & a\, i_1 \Delta w_6 \\ a\, i_2 \Delta w_5 & a\, i_2 \Delta w_6 \end{bmatrix}$$

Old weight — Derivative of Error with respect to weight

$$^*W_x = W_x - a\left(\frac{\partial Error}{\partial W_x}\right)$$

it — Learning rate

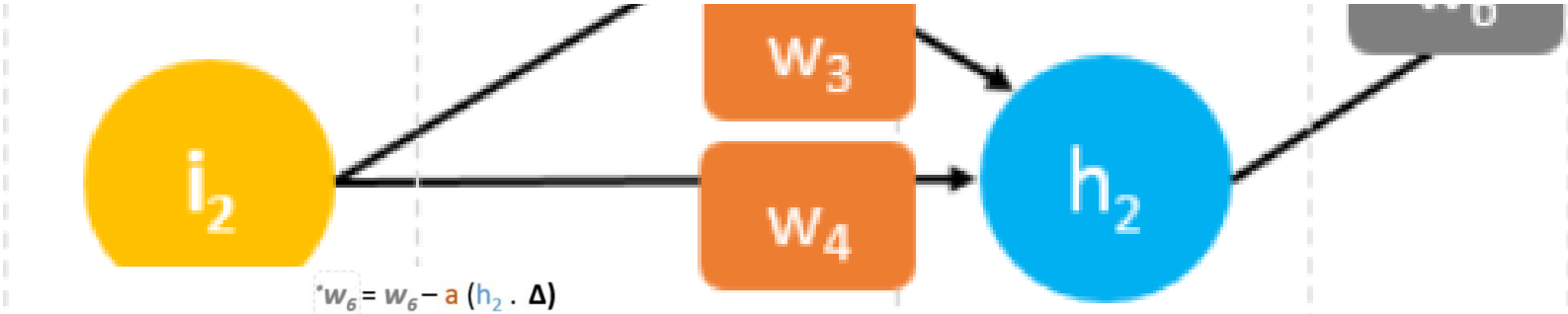
$\Delta = 0.191 - 1 = -0.809$ ← **Delta** = prediction - actual

$a = 0.05$ ← **Learning rate,** we smartly guess this number

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix} - 0.05(-0.809) \begin{bmatrix} 0.85 \\ 0.48 \end{bmatrix} = \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix} - \begin{bmatrix} -0.034 \\ -0.019 \end{bmatrix} = \begin{bmatrix} 0.17 \\ 0.17 \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} .11 & .12 \\ .21 & .08 \end{bmatrix} - 0.05(-0.809) \begin{bmatrix} 2 \\ 3 \end{bmatrix} . [0.14 \quad 0.15] = \begin{bmatrix} .11 & .12 \\ .21 & .08 \end{bmatrix} - \begin{bmatrix} -0.011 & -0.012 \\ -0.017 & -0.018 \end{bmatrix} = \begin{bmatrix} .12 & .13 \\ .23 & .10 \end{bmatrix}$$

prediction

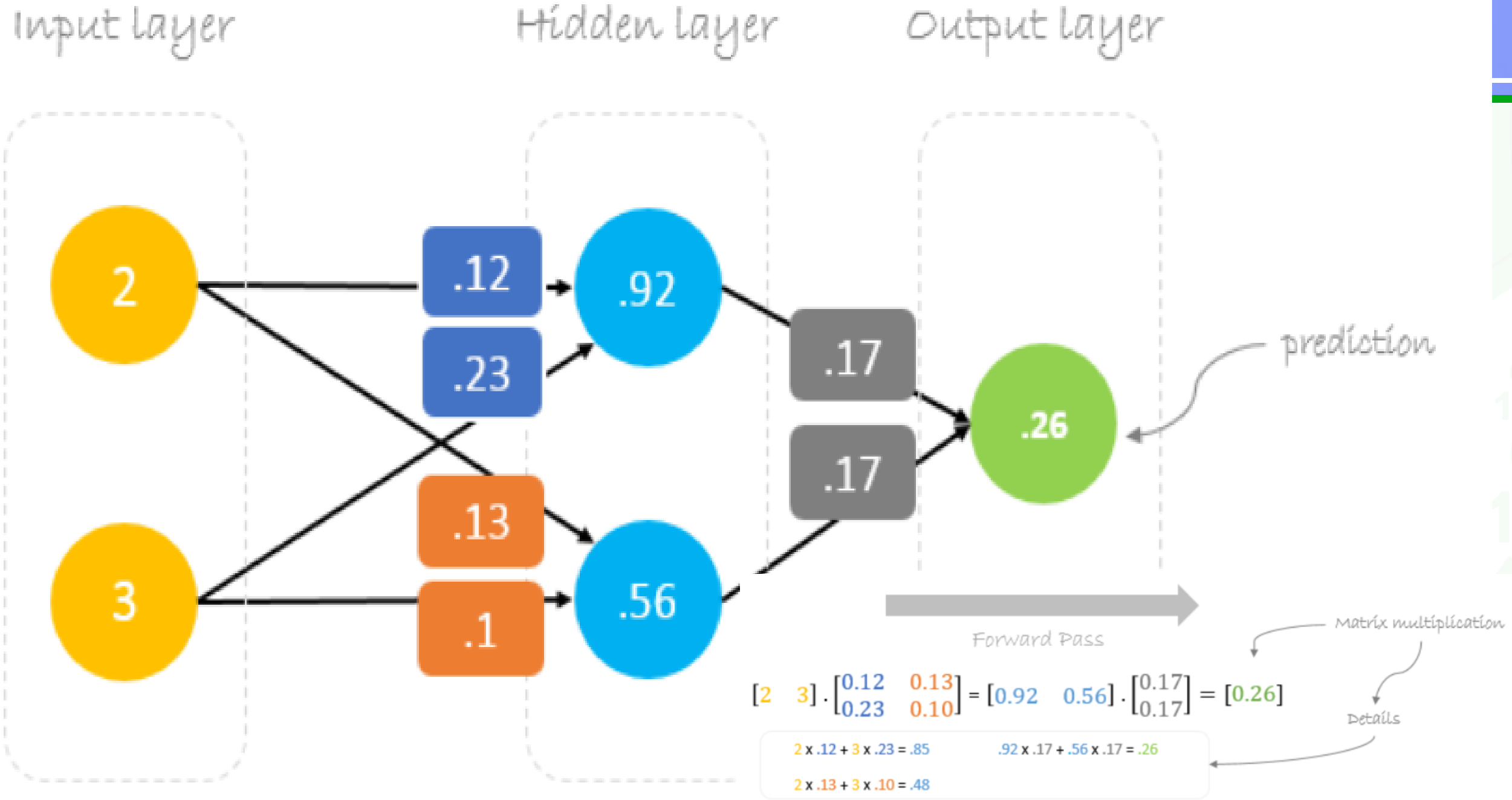


Updated weights

$^*w_6 = w_6 - a\,(h_2 . \Delta)$
$^*w_5 = w_5 - a\,(h_1 . \Delta)$
$^*w_4 = w_4 - a\,(i_2 . \Delta w_6)$
$^*w_3 = w_3 - a\,(i_1 . \Delta w_6)$
$^*w_2 = w_2 - a\,(i_2 . \Delta w_5)$
$^*w_1 = w_1 - a\,(i_1 . \Delta w_5)$

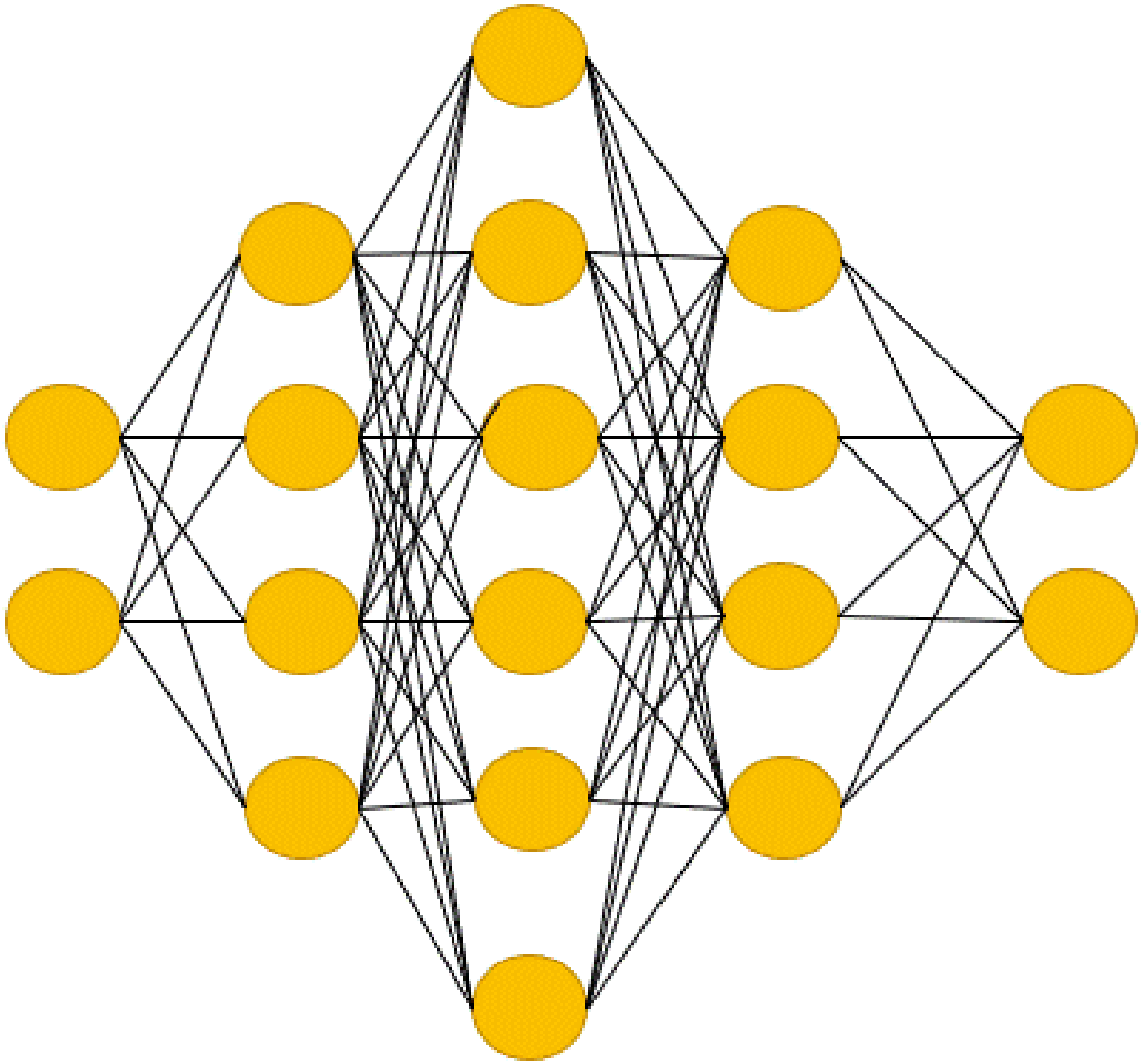
$$^*W_6 = W_6 - a\left(\frac{\partial Error}{\partial W_6}\right)$$

$$^*W_6 = W_6 - a\,\Delta h_2$$


Input layer
Hidden layer
Output layer
2
3
.12
.23
.13
.1
.92
.56
.17
.17
.26
prediction
Forward Pass
Matrix multiplication
$[2 \quad 3] \cdot \begin{bmatrix} 0.12 & 0.13 \\ 0.23 & 0.10 \end{bmatrix} = [0.92 \quad 0.56] \cdot \begin{bmatrix} 0.17 \\ 0.17 \end{bmatrix} = [0.26]$
Details
2 x .12 + 3 x .23 = .85
2 x .13 + 3 x .10 = .48
.92 x .17 + .56 x .17 = .26

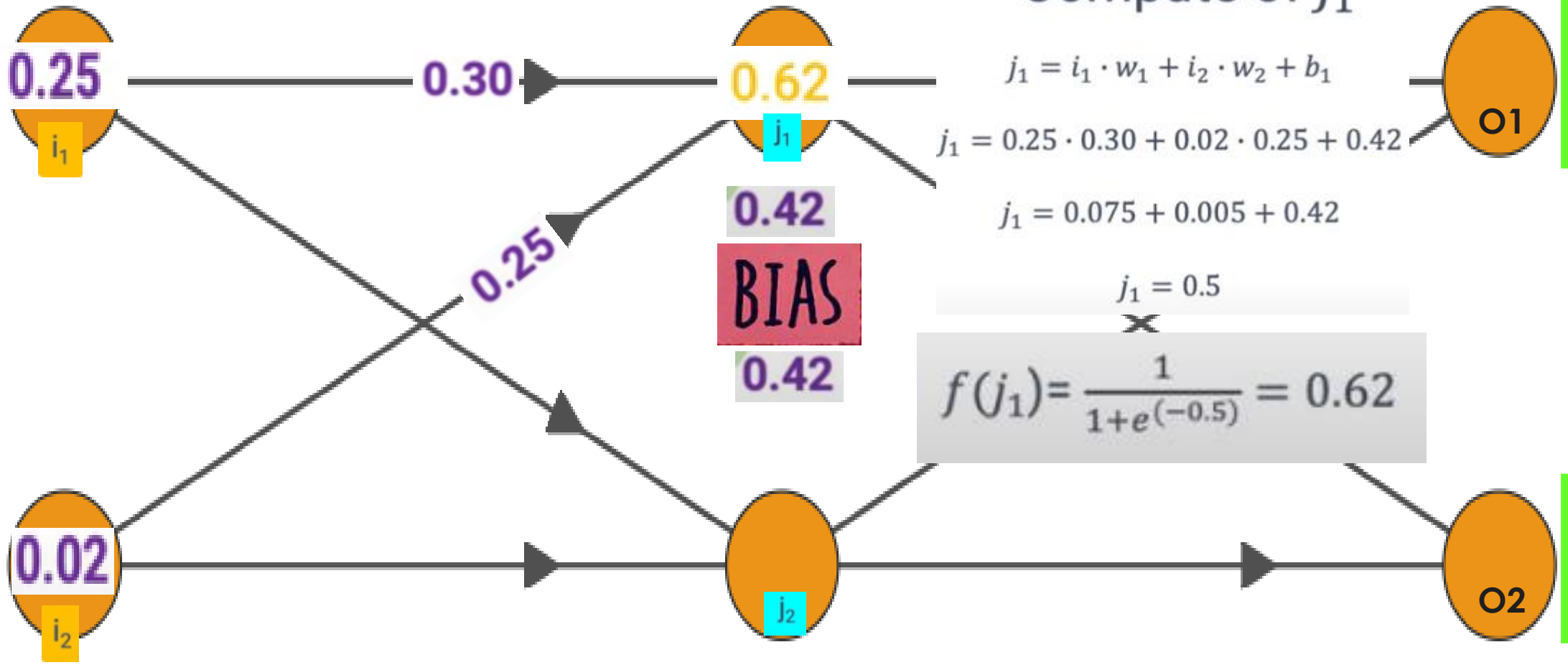
Let me ask, Gru

What is the Price of the Jet, Bob? It must be 50K



BACKPROPAGATION

Forward propagation



Compute of j_1

$$j_1 = i_1 \cdot w_1 + i_2 \cdot w_2 + b_1$$

$$j_1 = 0.25 \cdot 0.30 + 0.02 \cdot 0.25 + 0.42$$

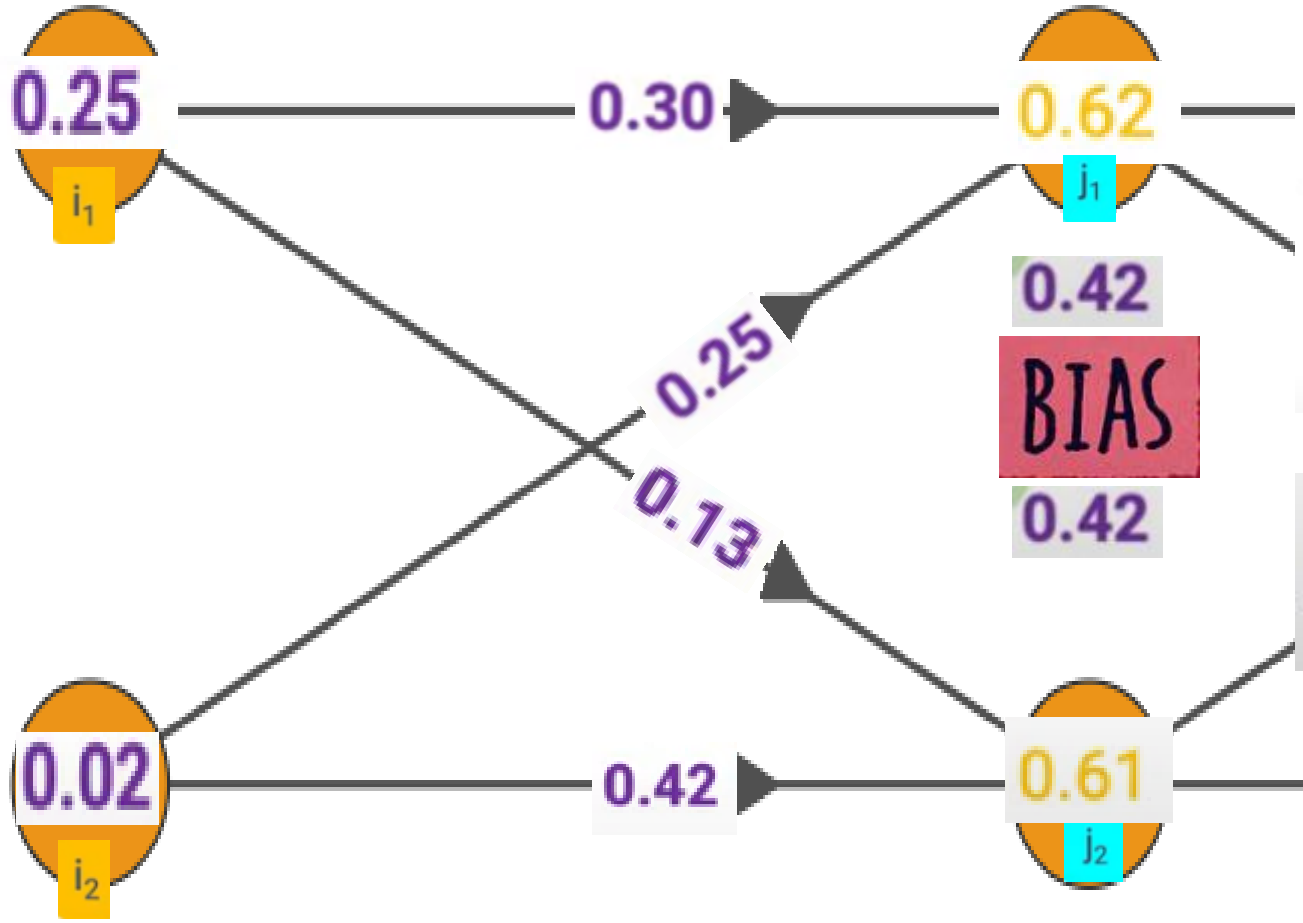
$$j_1 = 0.075 + 0.005 + 0.42$$

$$j_1 = 0.5$$

$$f(j_1) = \frac{1}{1+e^{(-0.5)}} = 0.62$$

BACKPROPAGATION

Forward propagation



Compute of j_2

$$j_2 = i_1 \cdot w_3 + i_2 \cdot w_4 + b_1$$

$$j_2 = 0.25 \cdot 0.13 + 0.02 \cdot 0.42 + 0.42$$

$$j_2 = 0.0325 + 0.0086 + 0.42$$

$$j_2 = 0.4609$$

$$f(j_2) = \frac{1}{1+e^{(-0.4609)}} = 0.61$$

BACKPROPAGATION

Forward propagation

Compute of o_1

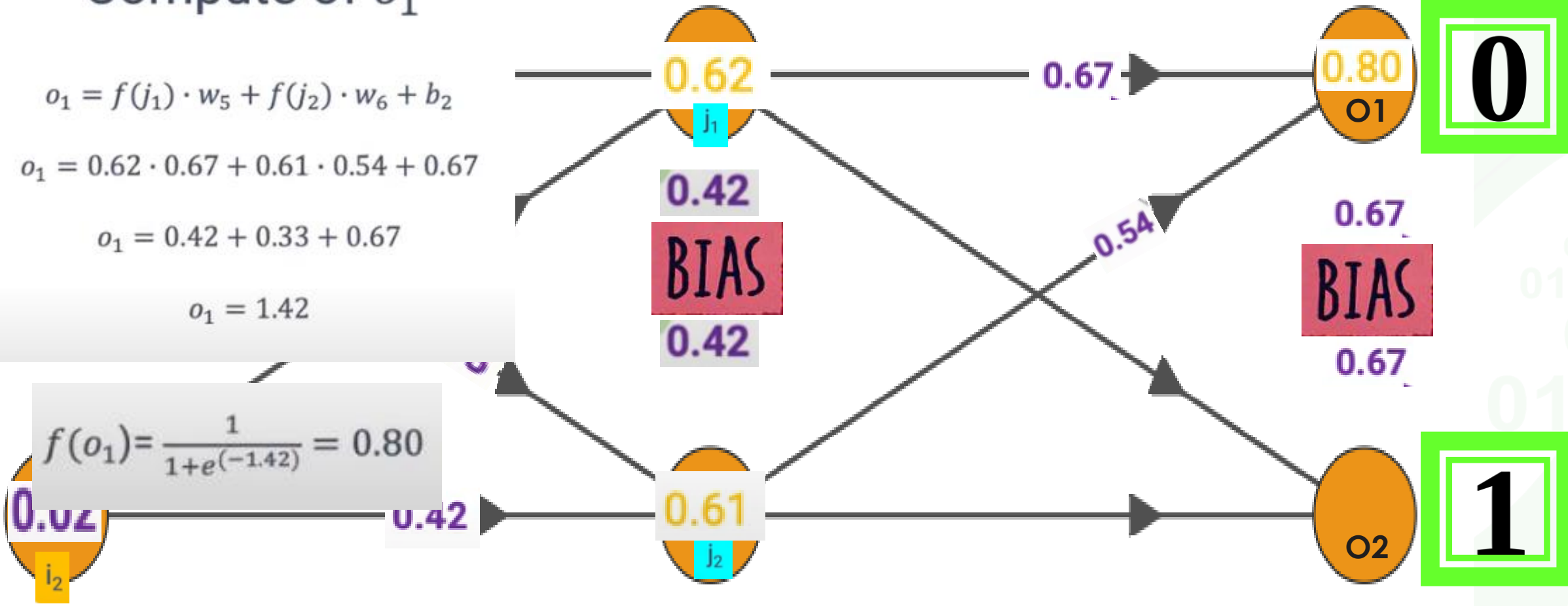
$$o_1 = f(j_1) \cdot w_5 + f(j_2) \cdot w_6 + b_2$$

$$o_1 = 0.62 \cdot 0.67 + 0.61 \cdot 0.54 + 0.67$$

$$o_1 = 0.42 + 0.33 + 0.67$$

$$o_1 = 1.42$$

$$f(o_1) = \frac{1}{1+e^{(-1.42)}} = 0.80$$



BACKPROPAGATION

Forward propagation

Compute of o_2

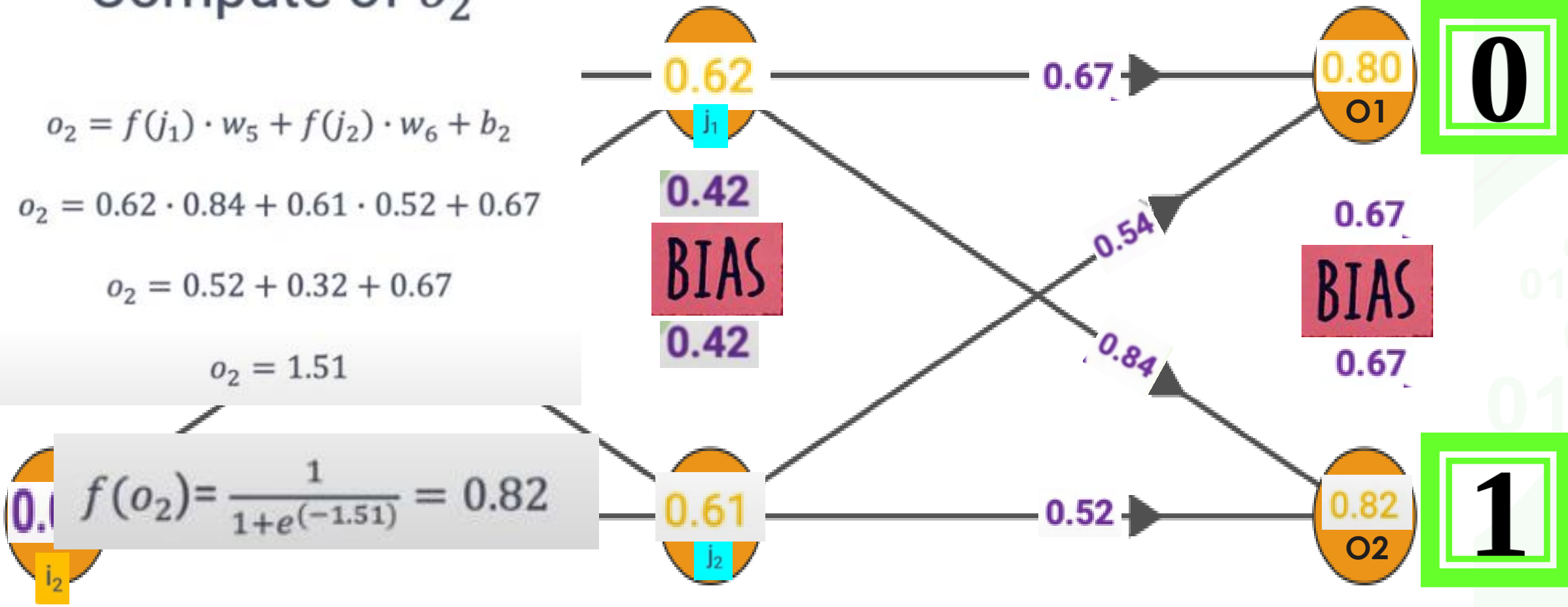
$$o_2 = f(j_1) \cdot w_5 + f(j_2) \cdot w_6 + b_2$$

$$o_2 = 0.62 \cdot 0.84 + 0.61 \cdot 0.52 + 0.67$$

$$o_2 = 0.52 + 0.32 + 0.67$$

$$o_2 = 1.51$$

$$f(o_2) = \frac{1}{1+e^{(-1.51)}} = 0.82$$



input layer

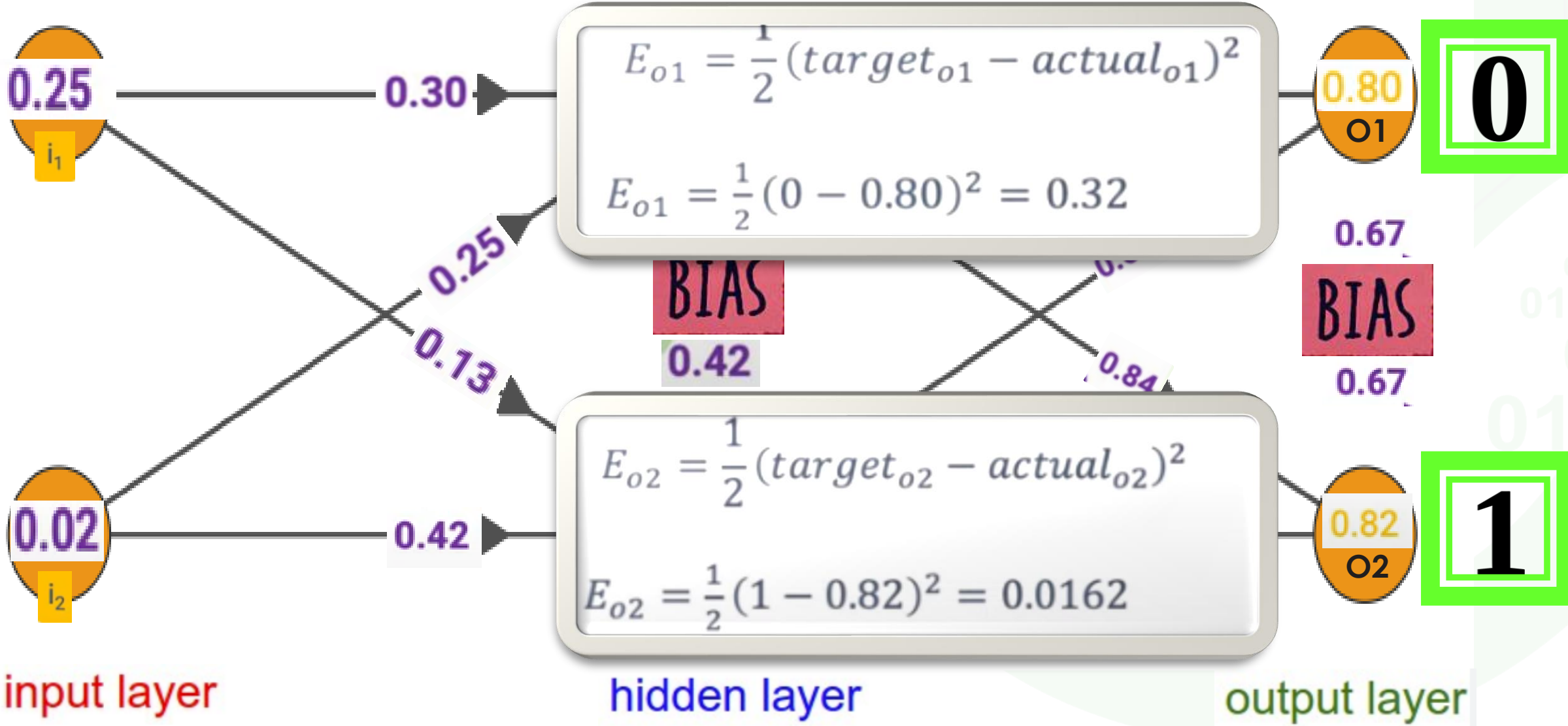
hidden layer

output layer

Calculate the error

For each neuron:

$$E_{total} = \sum \frac{1}{2}(target - actual)^2$$

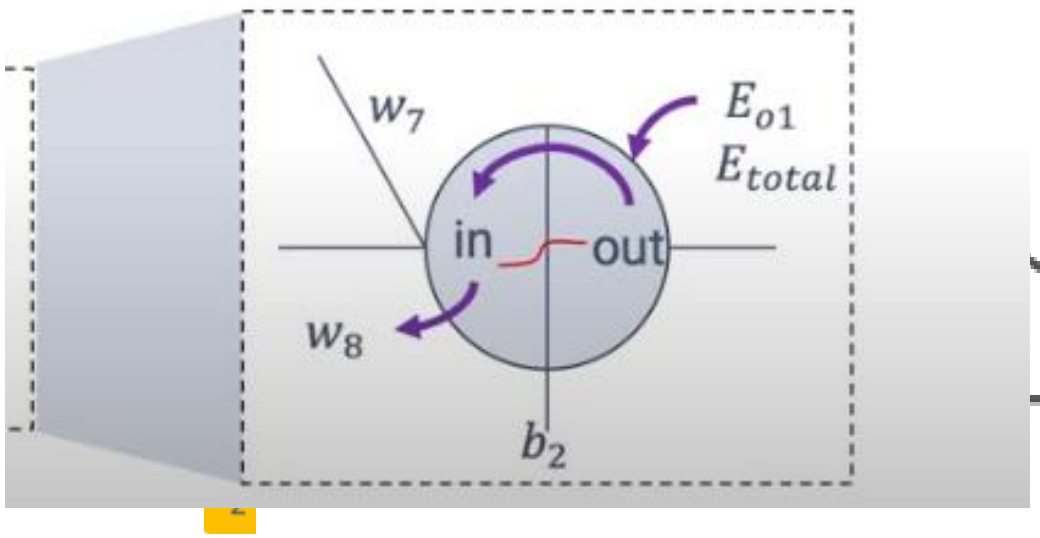


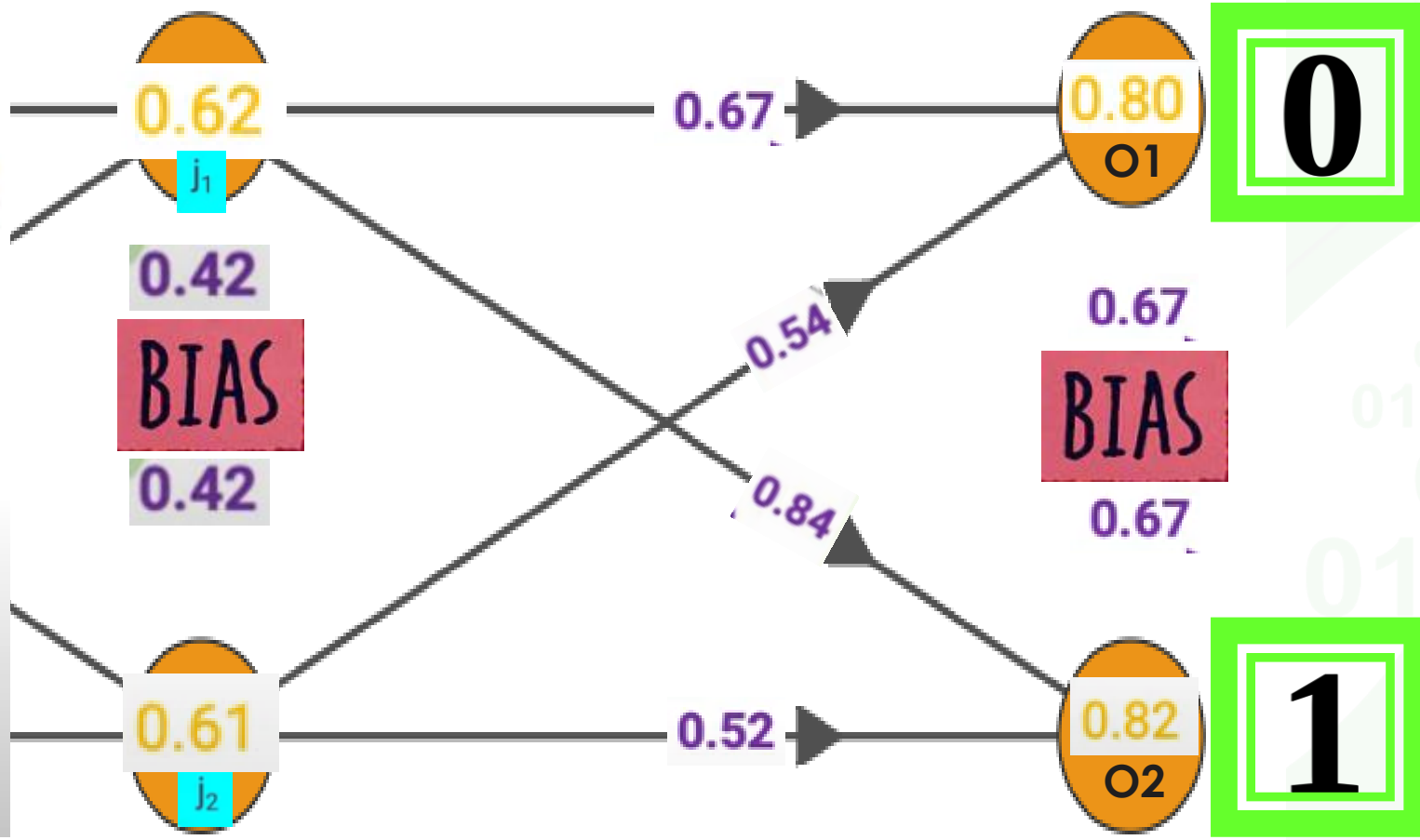
BACKPROPAGATION

Backward propagation

Adjust weights for output layer

$$\frac{\partial E_{total}}{\partial w_8} = \frac{\partial E_{total}}{\partial out_{o1}} \cdot \frac{\partial out_{o1}}{\partial in_{o1}} \cdot \frac{\partial in_{o1}}{\partial w_8}$$





input layer

hidden layer

output layer

0.84 - (0.1) * (-0.027812) * (0.62) işlemini hesaplarsak:

0.84 + 0.00171787 = 0.84171787

Backward propagation

δ

$\delta o2 = -(O2 - O2)\,çıkO2\,(1 - çıkO2)$

$-(1-0.82)0.82(1-0.82)$

$-0.18 * 0.82 * 0.18$

-0.027812

ESKİ AĞIRLIK (0.52)

Wyeni= Eski ağırlık- Learning rate δO2outputJ2

0.52-(0.1)(- 0.027812)(0.61)

0.52168517