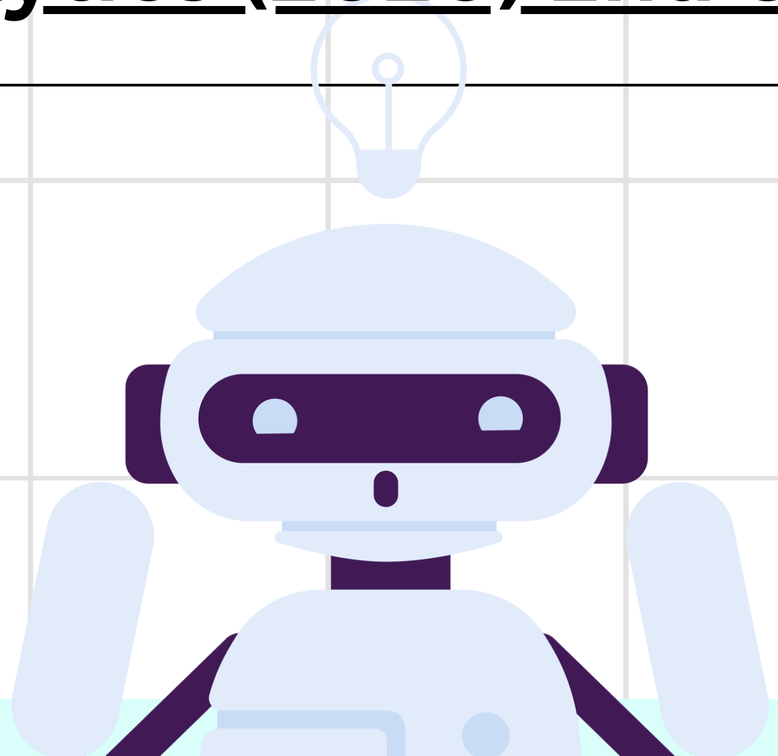
101

Deep Learning 101

Selma MANI

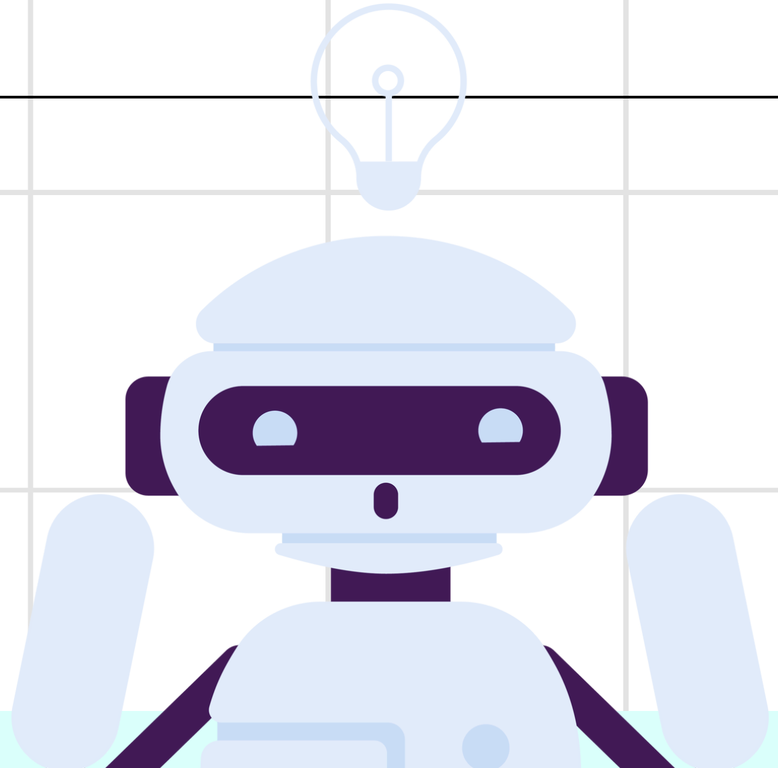
For this session we will be using illustrations from the book

Fundamentals of ML for Predictive Data Analytics (2020) 2nd edition



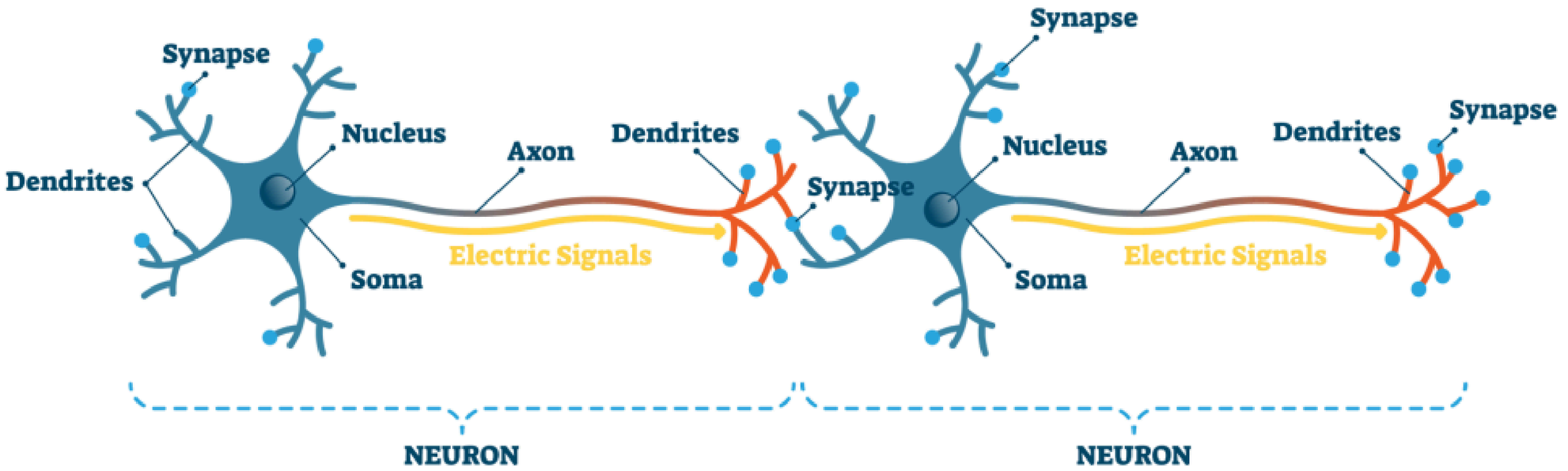
The notebook to this session

Deep Learning 101 Notebook

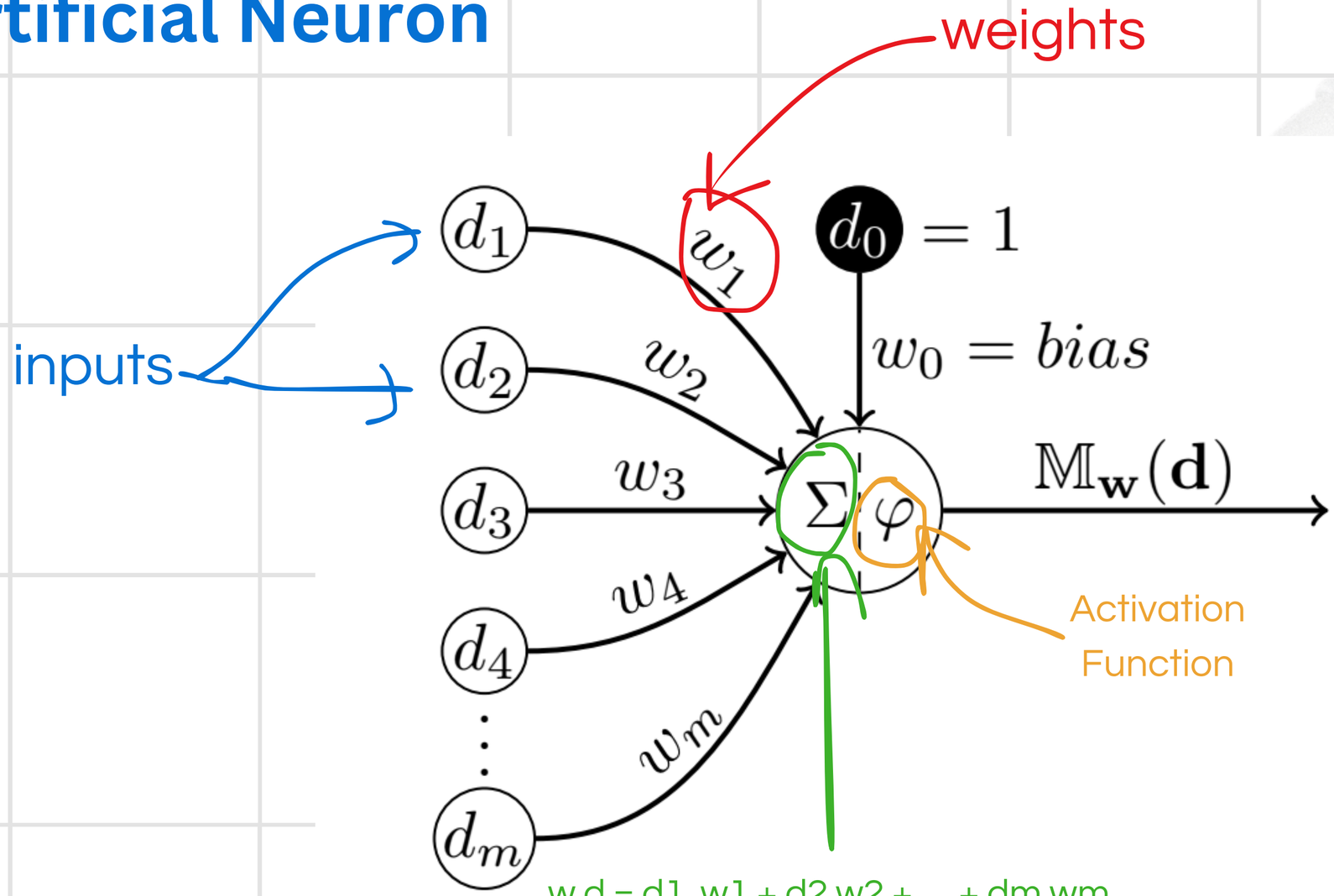


How does our brain work?

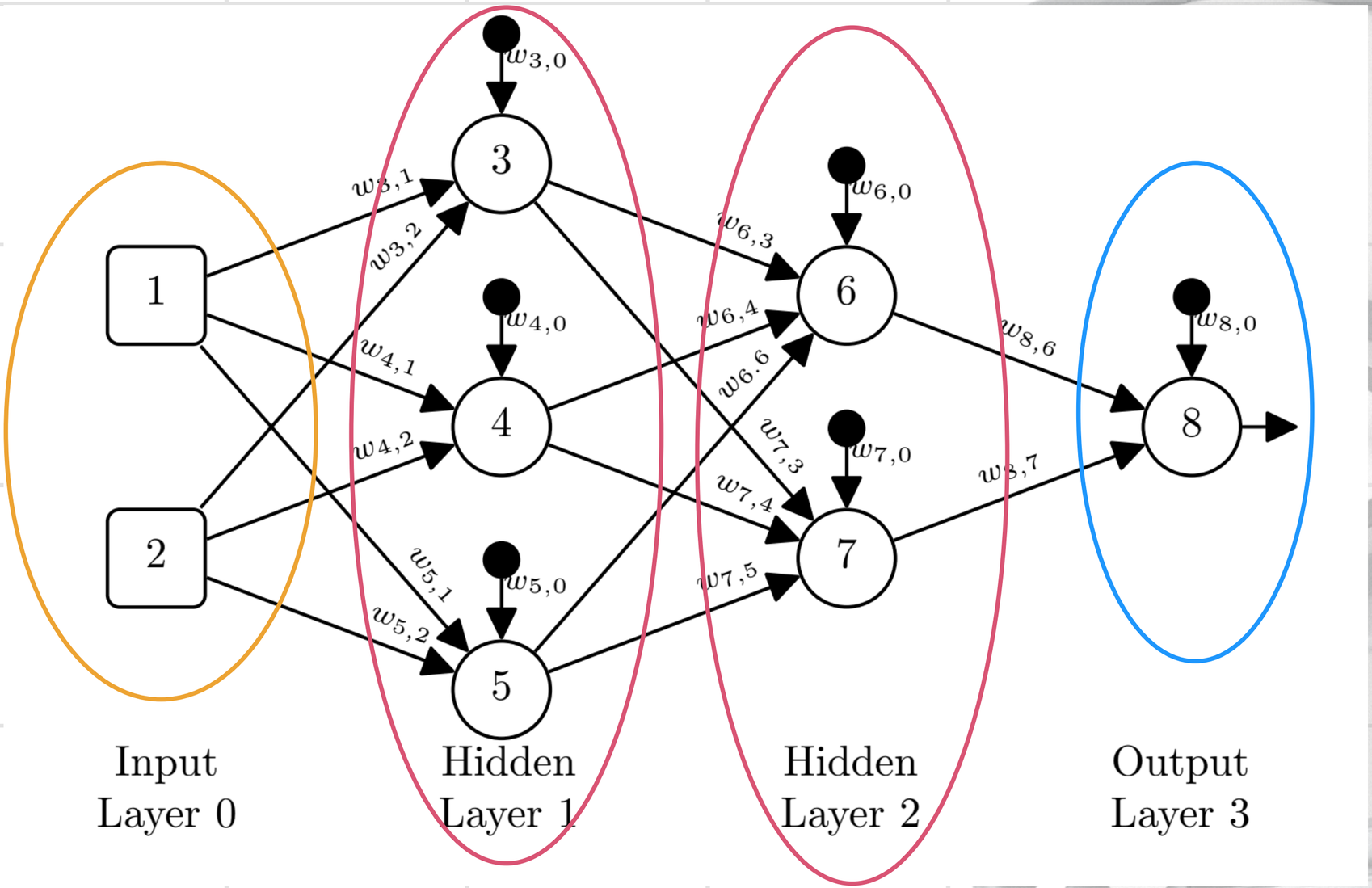
How does our brain work?



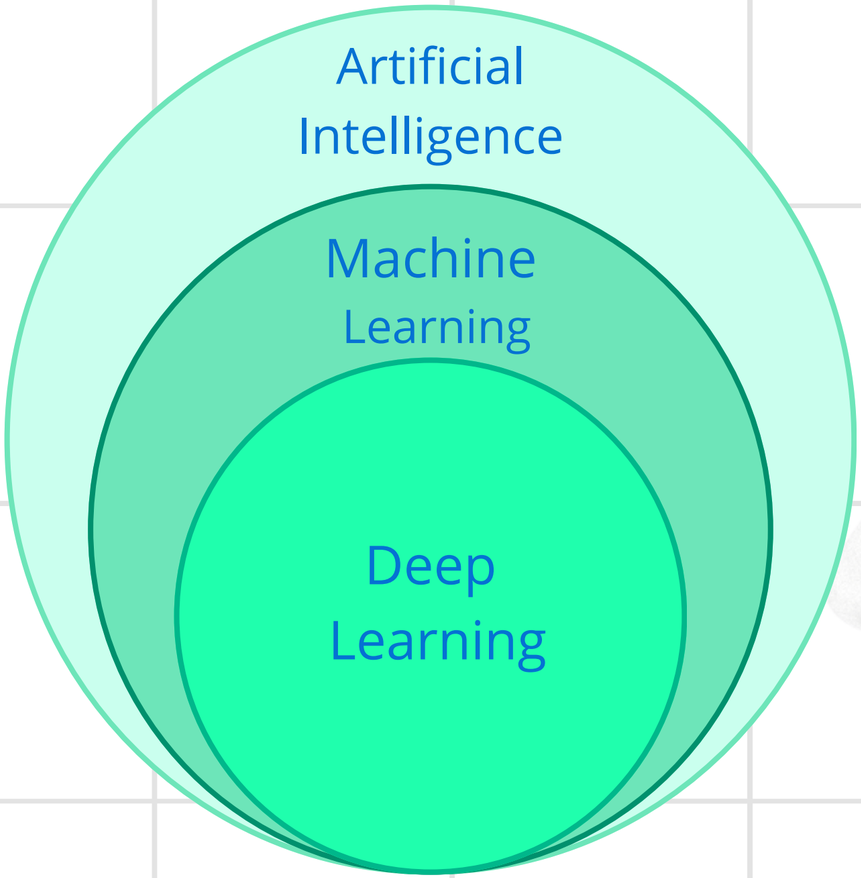
Artificial Neuron



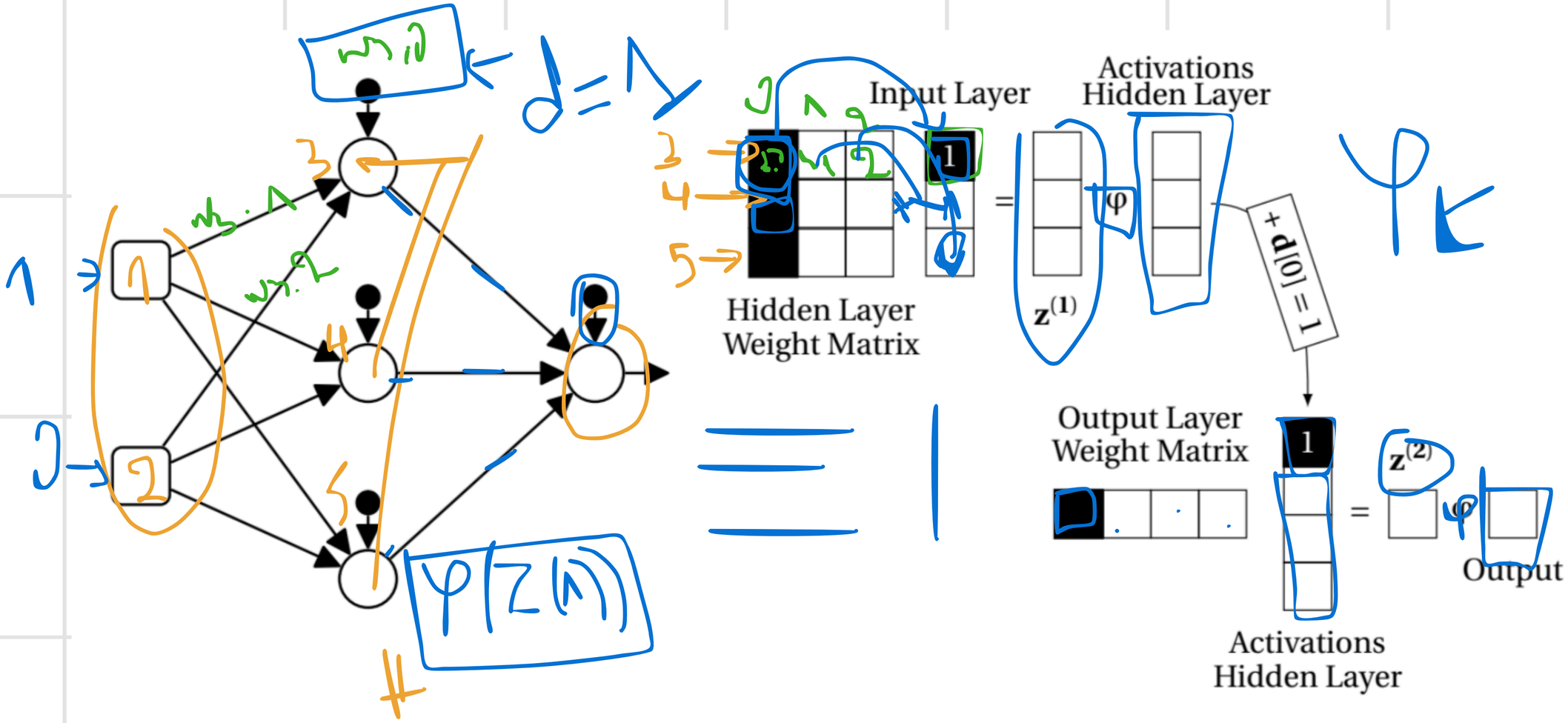
Artificial Neuron Network



Machine Learning VS Deep Learning

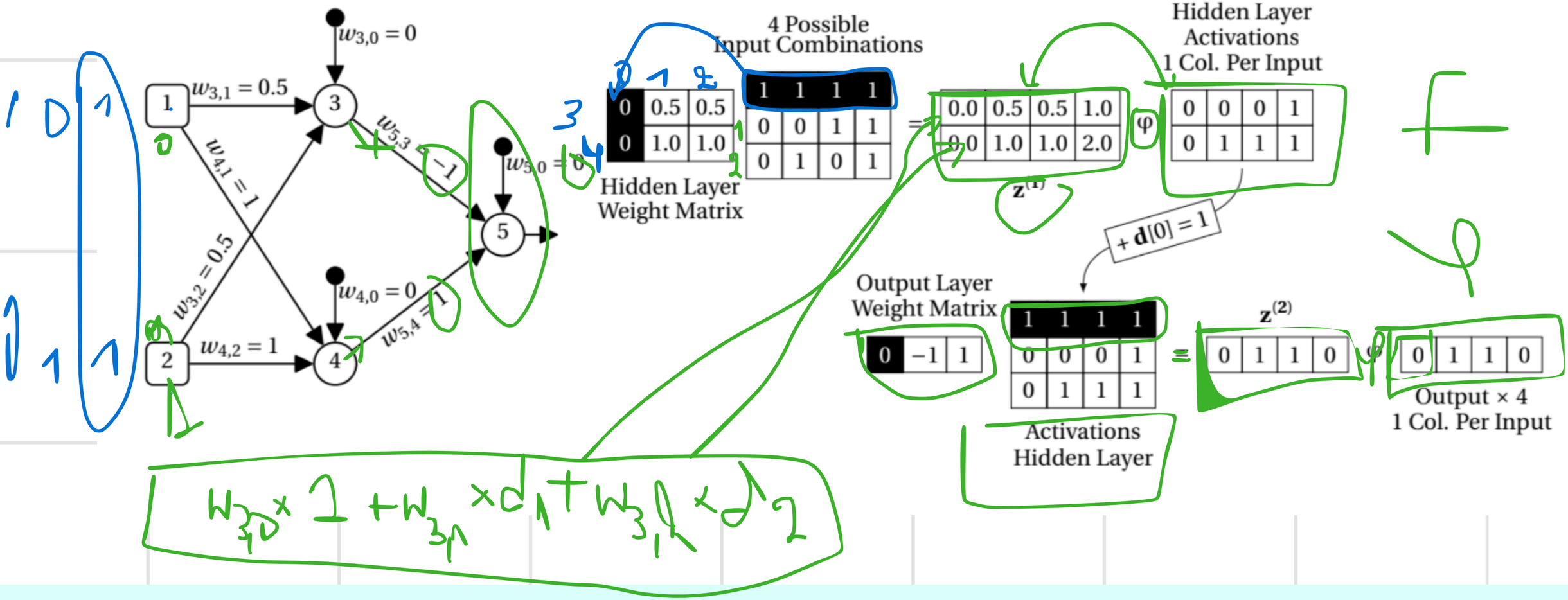


Matrices in Deep Learning



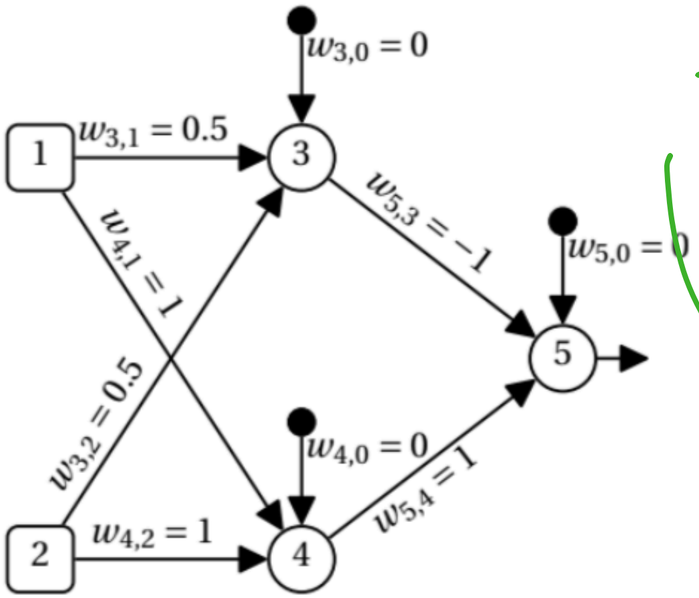
Activation Function

What is the activation function in this example?



Activation Function

M(w.d) = 0 if z < 1 else 1



4 Possible Input Combinations

0	0.5	0.5
0	1.0	1.0

Hidden Layer Weight Matrix

1	1	1	1
0	0	1	1
0	1	0	1

=

0.0	0.5	0.5	1.0
0.0	1.0	1.0	2.0

$\mathbf{z}^{(1)}$

φ

Hidden Layer Activations 1 Col. Per Input

0	0	0	1
0	1	1	1

+ $\mathbf{d}[0] = 1$

Output Layer Weight Matrix

0	−1	1

1	1	1	1
0	0	0	1
0	1	1	1

Activations Hidden Layer

=

$\mathbf{z}^{(2)}$

0	1	1	0

φ

0	1	1	0

Output × 4 1 Col. Per Input

Activation Function

$$\mathbb{M}_{\mathbf{w}}(\mathbf{d}) = \varphi\left(\mathbf{w}[0] \times \mathbf{d}[0] + \mathbf{w}[1] \times \mathbf{d}[1] + \cdots + \mathbf{w}[m] \times \mathbf{d}[m]\right)$$

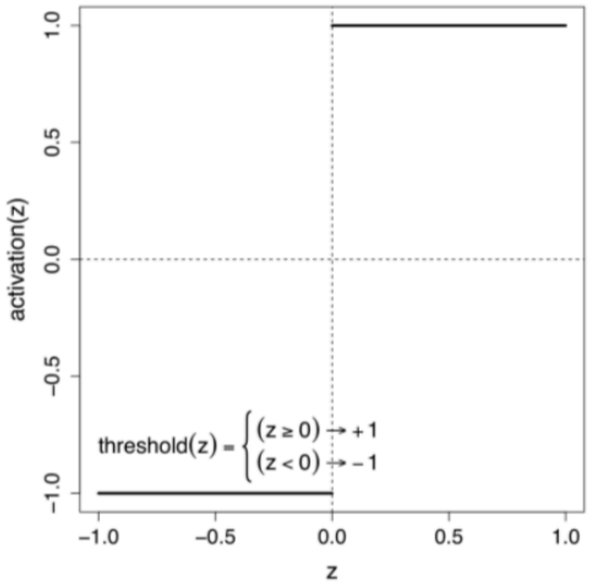
$$= \varphi\left(\sum_{i=0}^{m} w_i \times d_i\right) = \varphi\left(\underbrace{\mathbf{w} \cdot \mathbf{d}}_{dot\ product}\right)$$

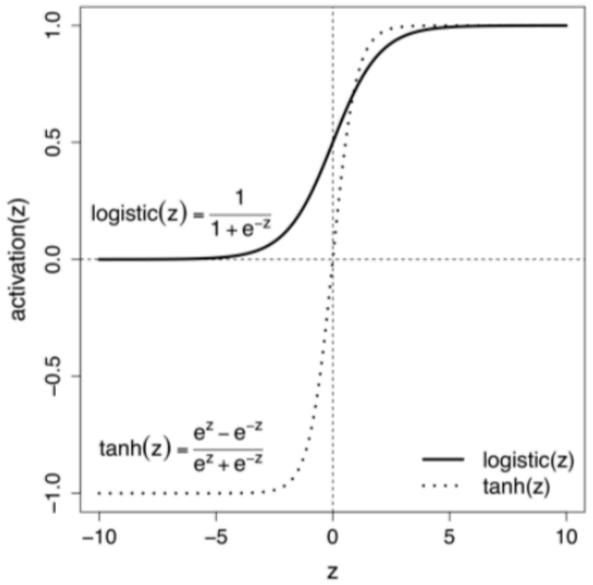
$$= \varphi\left(\underbrace{\mathbf{w}^T \mathbf{d}}_{matrix\ product}\right) = \varphi\left([w_0, w_1, \ldots, w_m] \begin{bmatrix} d_0 \\ d_2 \\ \vdots \\ d_m \end{bmatrix}\right)$$

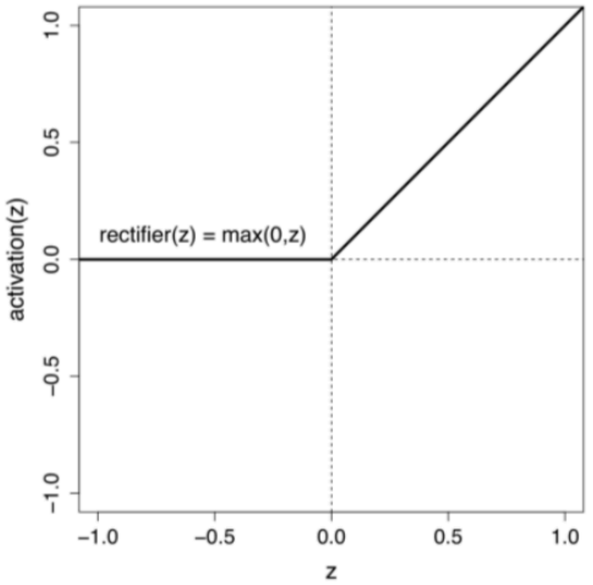
Activation Function

$$\mathbb{M}_{\mathbf{w}}(\mathbf{d}) = \begin{cases} 1 & \text{if } z \geq \theta \\ 0 & otherwise \end{cases}$$

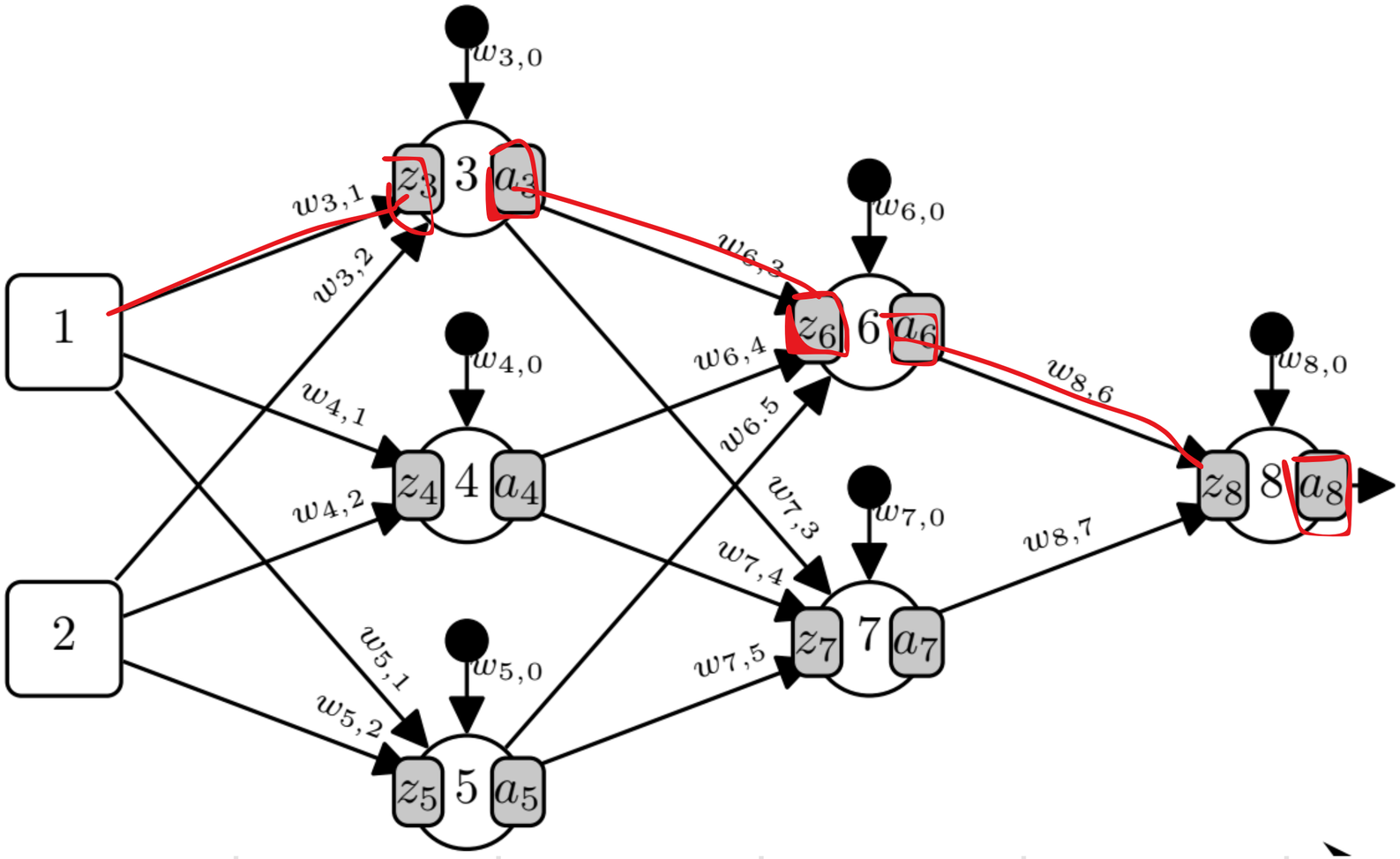
$$rectifier(z) = max(0, z)$$



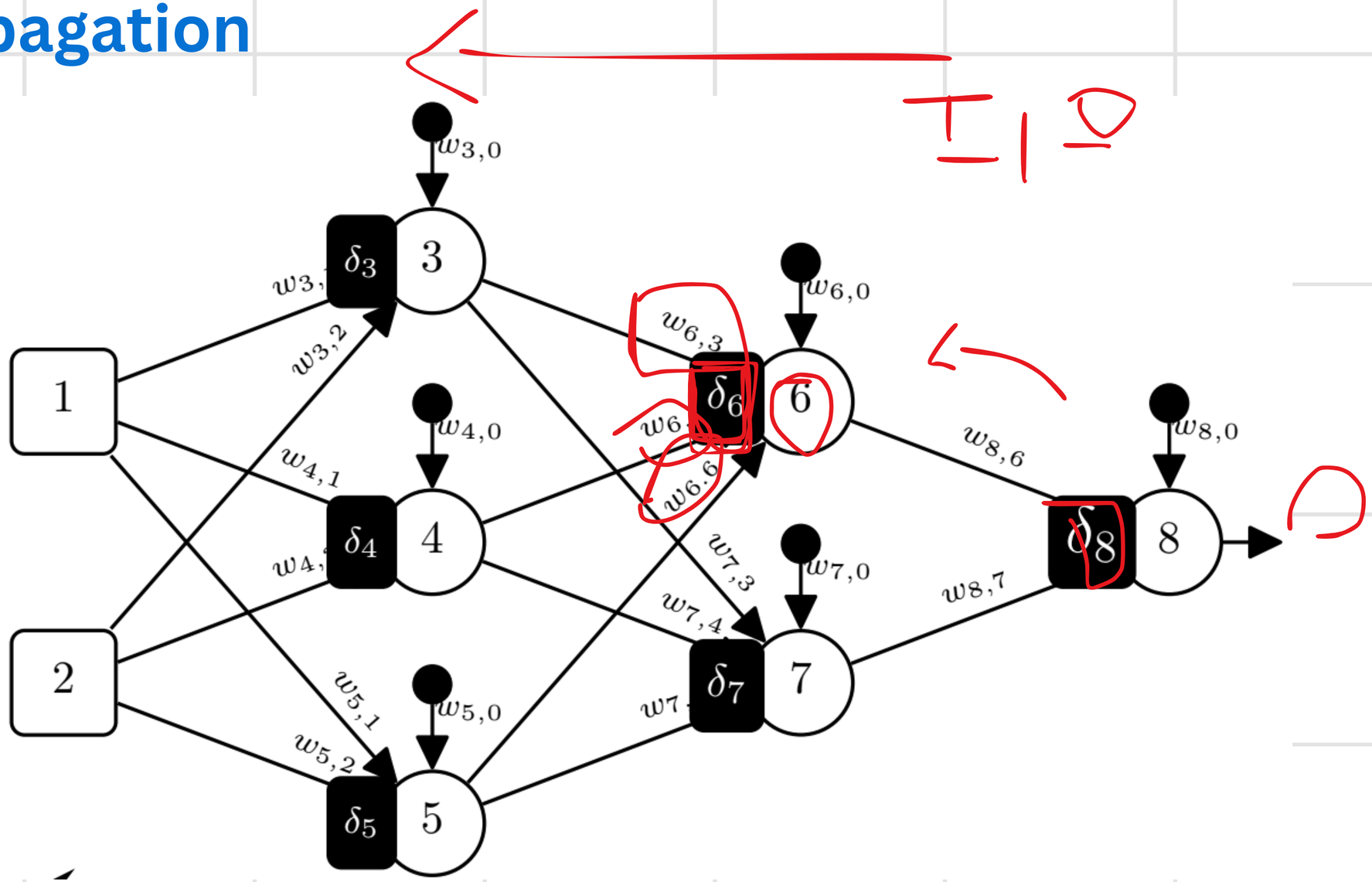




Forward Propagation



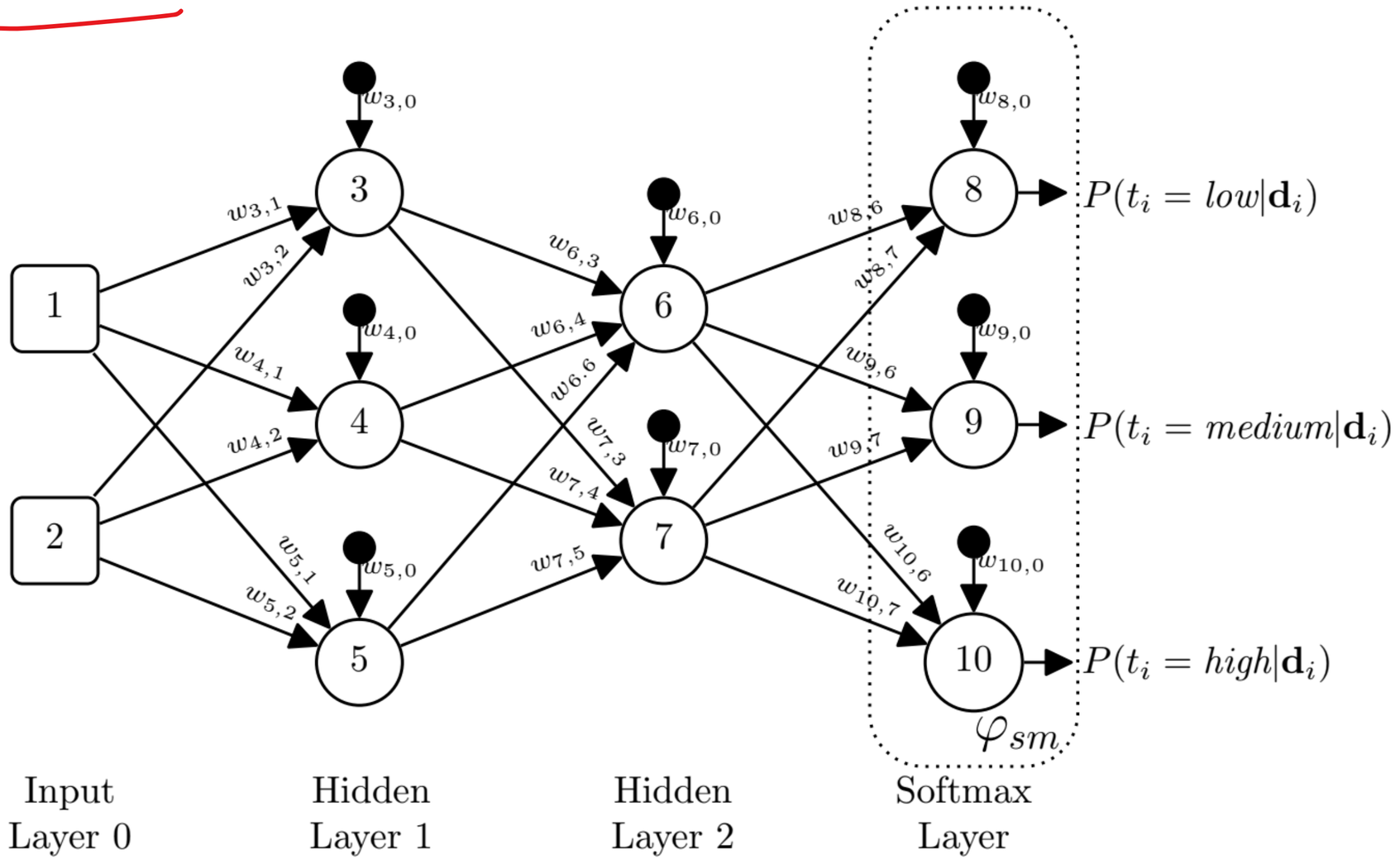
Backpropagation



Backpropagation

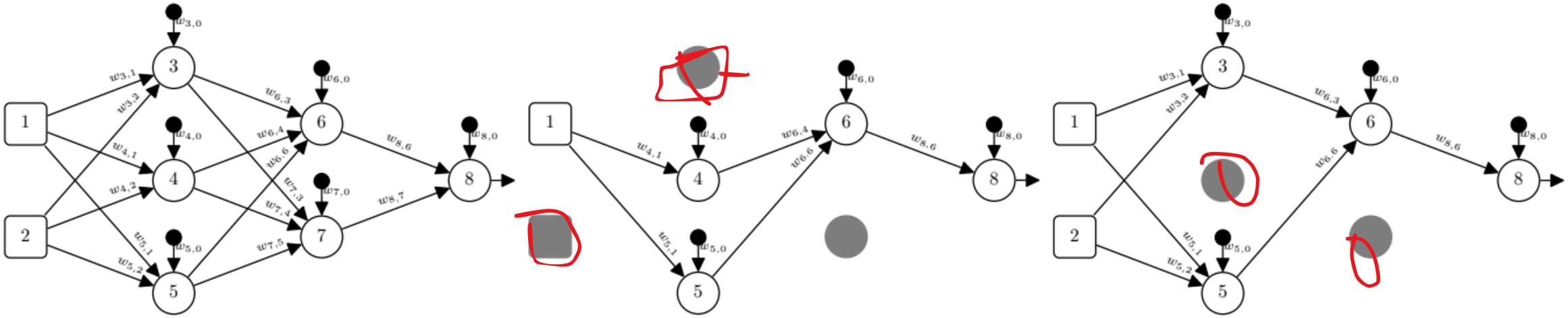
Shuffle $\mathcal{D}$ and create the mini-batches: $[(\mathbf{X}^{(1)}, \mathbf{Y}^{(1)}), \ldots, (\mathbf{X}^{k}, \mathbf{Y}^{k})]$
Initialize the weight matrices for each layer: $\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L)}$
repeat ▷ Each repeat loop is one epoch
  for t=1 to number of mini-batches **do** ▷ Each for loop is one iteration
    $\mathbf{A}^{(0)} \leftarrow \mathbf{X}^{(t)}$
    for l=1 to L **do**
      $\mathbf{v} \leftarrow [1_0, \ldots 1_m]$ ▷ Create $\mathbf{v}$ the vector of bias terms
      $\mathbf{A}^{(l-1)} \leftarrow [\mathbf{v}; \mathbf{A}^{(l-1)}]$ ▷ Insert $\mathbf{v}$ into the activation matrix
      $\mathbf{Z}^{(l)} \leftarrow \mathbf{W}^{t}\mathbf{A}^{(l-1)}$
      $\mathbf{A}^{(l)} \leftarrow \varphi(\mathbf{Z}^{(l)})$ ▷ Elementwise application of φ to $\mathbf{Z}^{(l)}$
    end for
    for each weight $w_{i,k}$ in the network **do**
      $\Delta w_{i,k} = 0$
    end for
    for each example in the mini-batch **do** ▷ Backpropagate the δs
      for each neuron i in the output layer **do**
        $\delta_i = \frac{\partial \mathcal{E}}{\partial a_i} \times \frac{\partial a_i}{\partial z_i}$ ▷ See Equation (9)[10]
      end for
      for l = L-1 to 1 **do**
        for each neuron i in the layer l **do**
          $\delta_i = \frac{\partial \mathcal{E}}{\partial a_i} \times \frac{\partial a_i}{\partial z_i}$ ▷ See Equation (11)[11]
        end for
      end for
      for each weight $w_{i,k}$ in the network **do**
        $\Delta w_{i,k} = \Delta w_{i,k} + (\delta_i \times a_k)$ ▷ Equation (17)[15]
      end for
    end for
    for each weight $w_{i,k}$ in the network **do**
      $w_{i,k} \leftarrow w_{i,k} - \alpha \times \Delta w_{i,k}$ ▷ Equation (18)[15]
    end for
  end for
  shuffle($[(\mathbf{X}^{(1)}, \mathbf{Y}^{(1)}), \ldots, (\mathbf{X}^{k}, \mathbf{Y}^{k})]$)
until convergence occurs

Softmax Output Layer



How to prevent Overfitting?

Dropout



How to prevent Overfitting?

Dropout

Require: ρ probability that a neuron in a layer will not be dropped

▷ Forward Pass

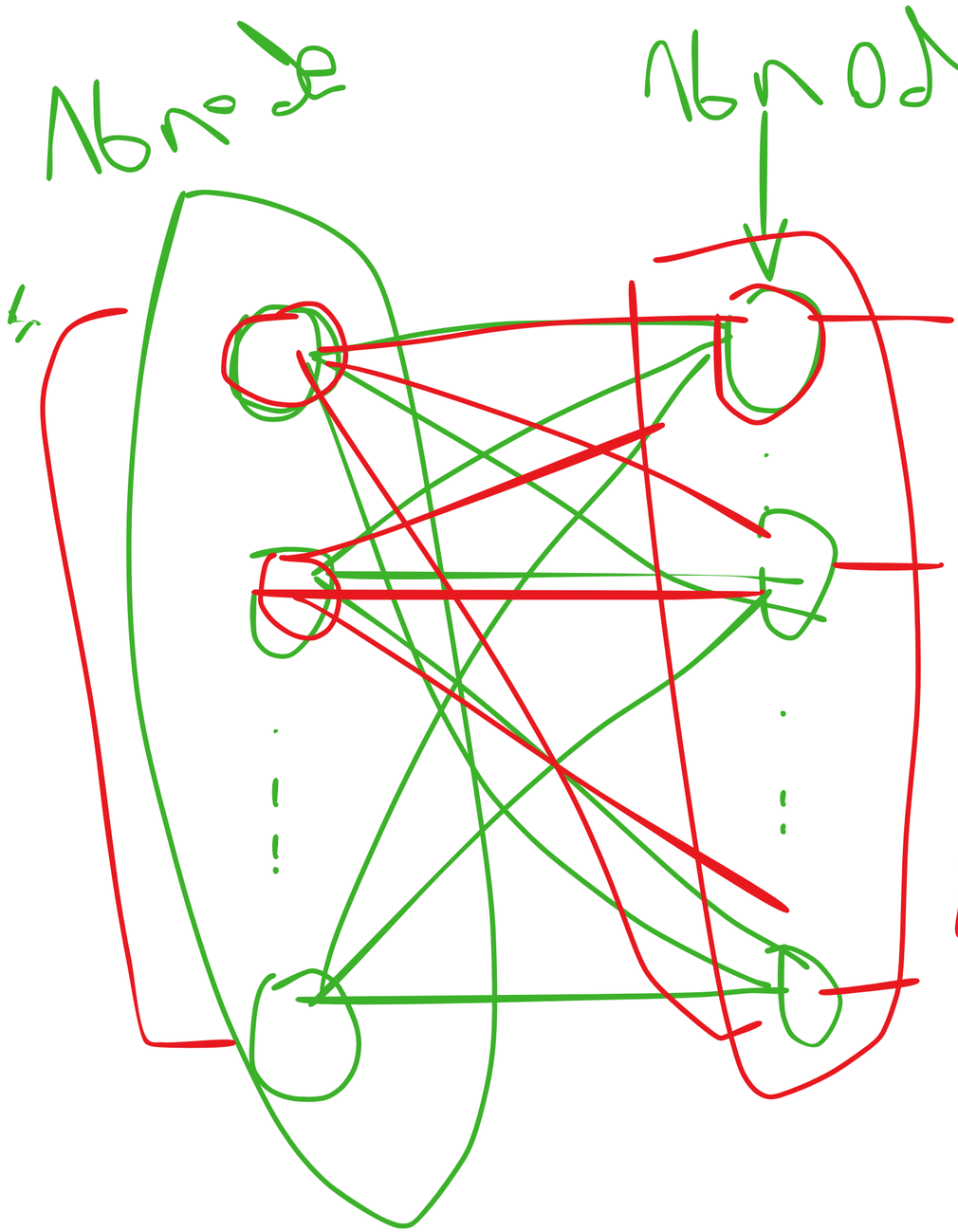
1: **for** each input or hidden layer l **do**
2: $\quad \mathbf{DropMask}^{(l)} = (m_1, \ldots, m_{size(l)}) \sim Bernoulli(\rho)$
3: $\quad \mathbf{a}^{(l)\prime} = \mathbf{a}^{(l)} \odot \mathbf{DropMax}^{(l)}$
4: $\quad \mathbf{a}^{(l)\prime\prime} = \frac{1}{\rho}\mathbf{a}^{(l)\prime}$
5: **end for**

▷ Backward Pass

6: **for** each layer l in backward pass **do**
7: $\quad \delta^{(l)} = \delta^{(l)} \odot \mathbf{DropMax}^{(l)}$
8: **end for**

Thank you!

I hope you enjoyed 🎀



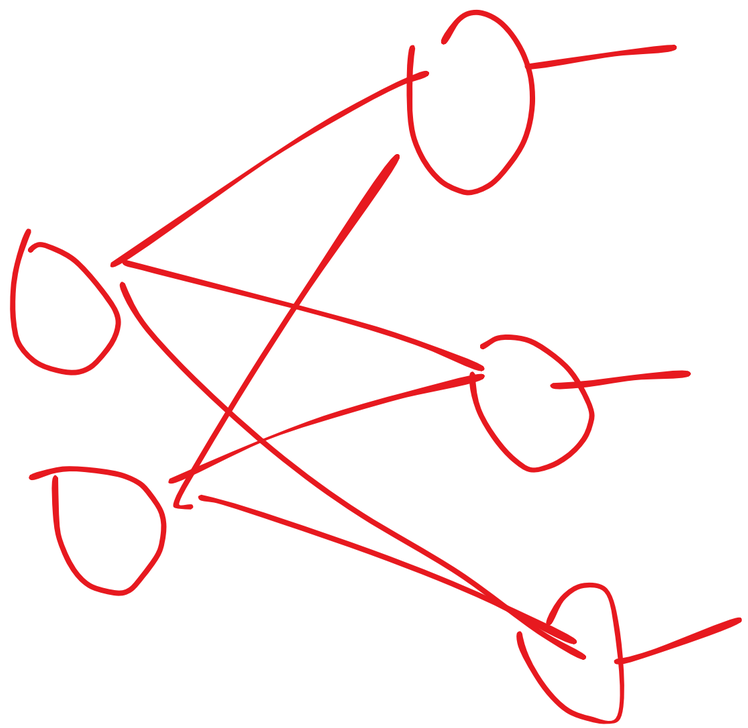
16 x 16

16 + 16 x 16

16 + 16 + ... + 16

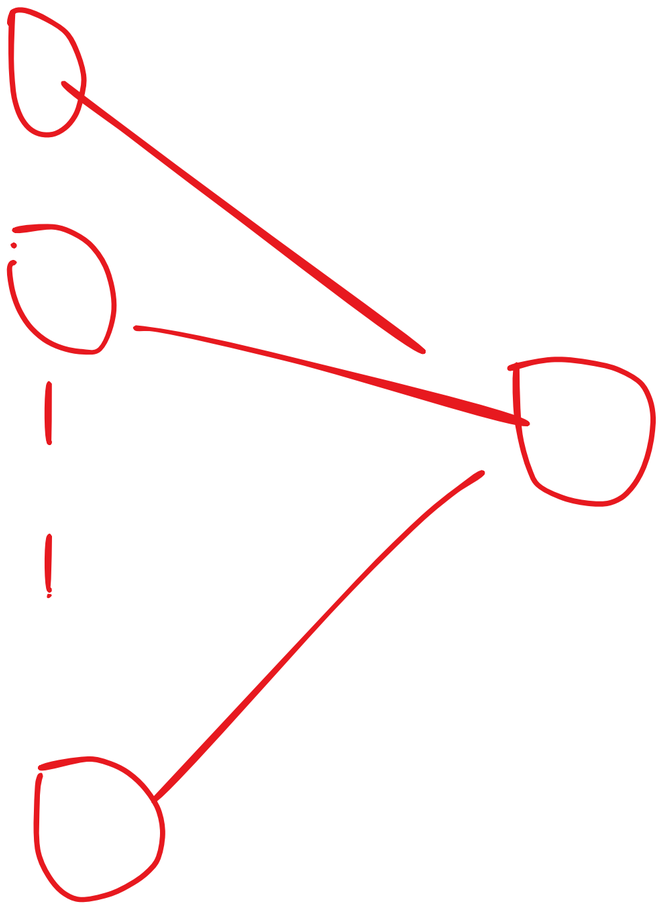
16

+ 16



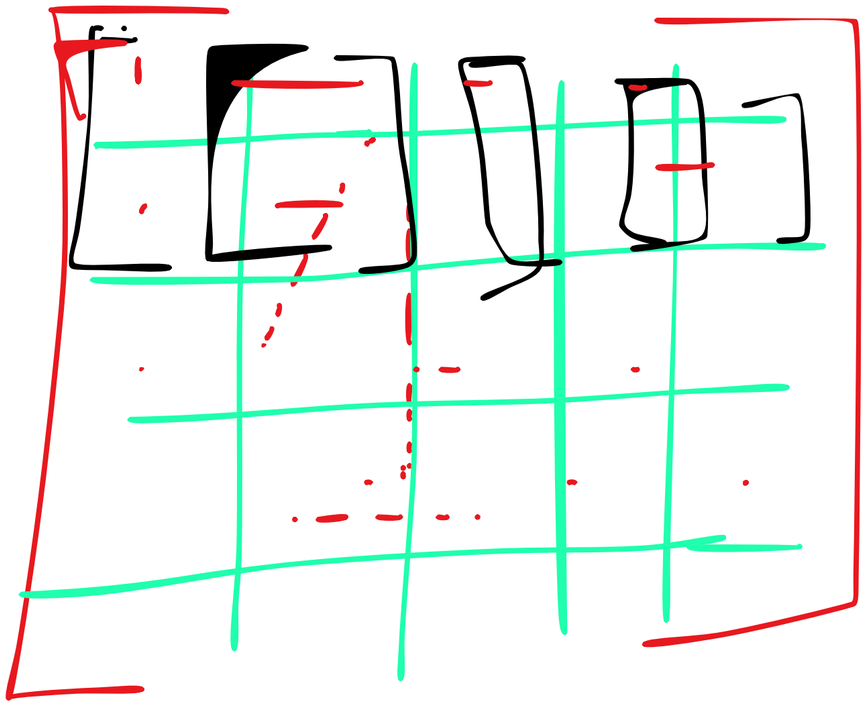
$6+3=$

$12\times3+3=$



$$16 \times 1 + 1$$

$$= 17$$




Age