If you have it, please take out:

- **pen/pencil**
- **paper**
- **laptop**

Sit close to at least one other person!

Understanding how artificial neural networks understand

Lauren Hayward Sierens

PERIMETER INSTITUTE

CBC news

Kitchener-Waterloo

Snow, ice pellets, freezing rain and wind: More winter weather on the way

Photo from kitchener.ctvnews.ca

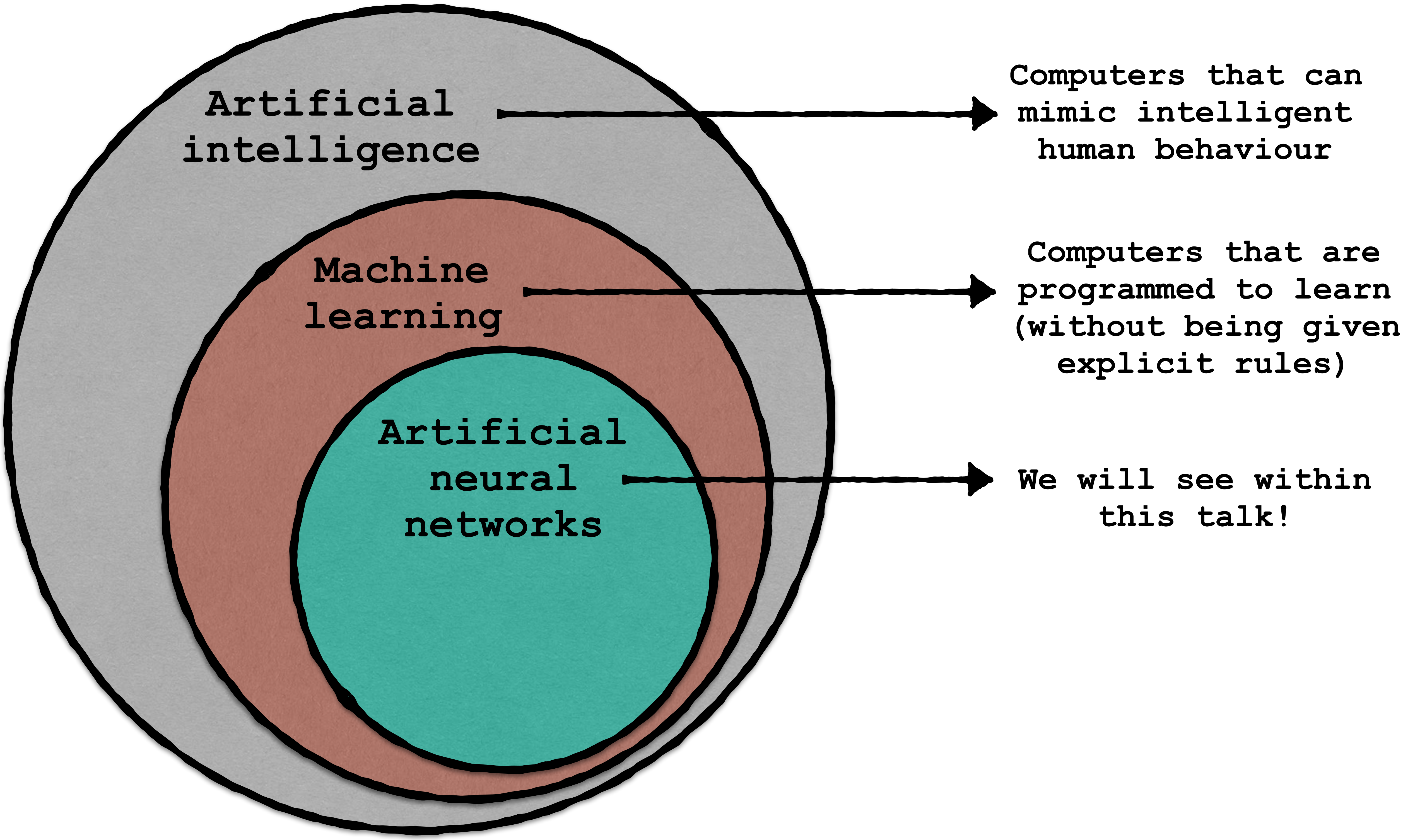
PERIMETER INSTITUTE

What comes to mind when you hear about “machine learning”?

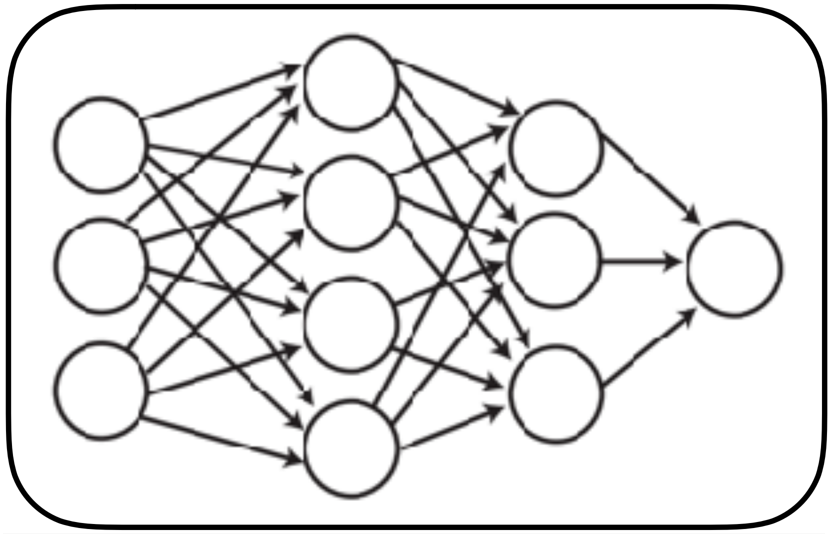
- Artificial intelligence
- Neural networks
- Deep learning
- Computer science

⋮

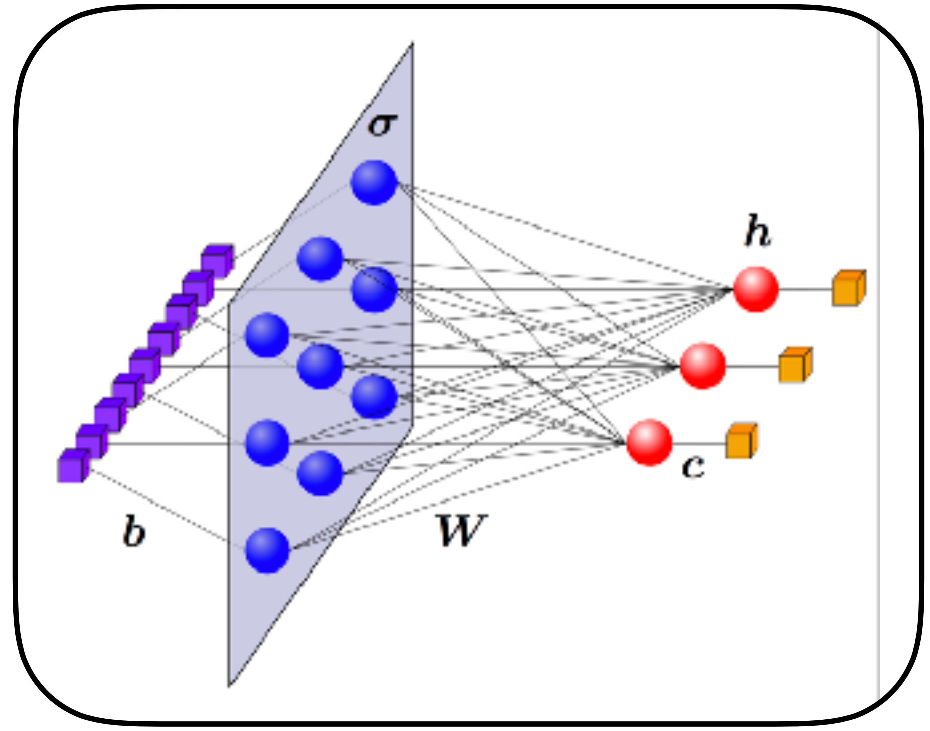
- Linear algebra
- Calculus


Artificial intelligence
Machine learning
Artificial neural networks
Computers that can mimic intelligent human behaviour
Computers that are programmed to learn (without being given explicit rules)
We will see within this talk!

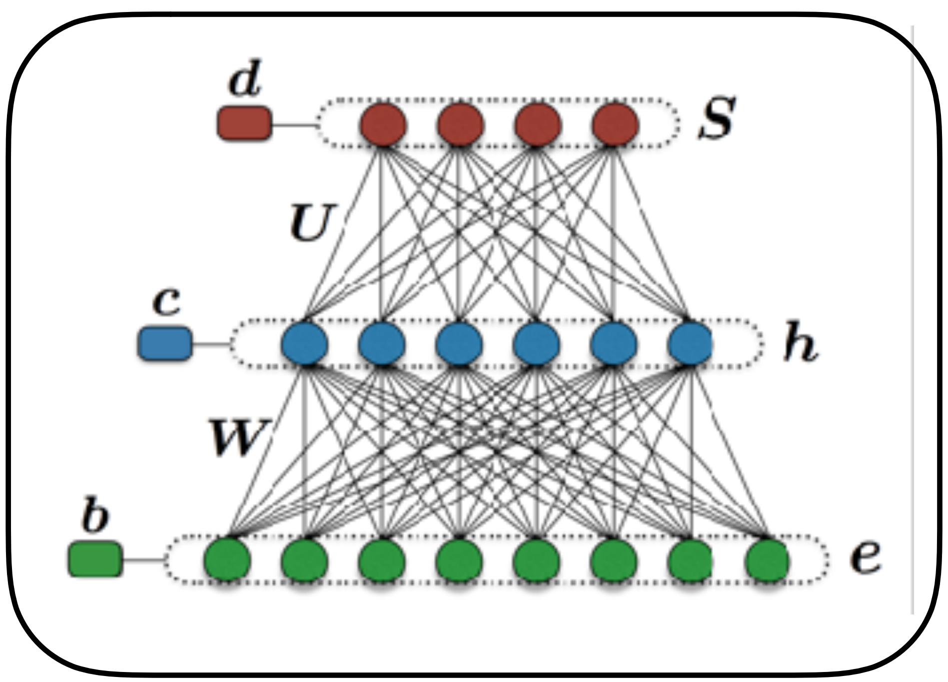
Artificial neural networks



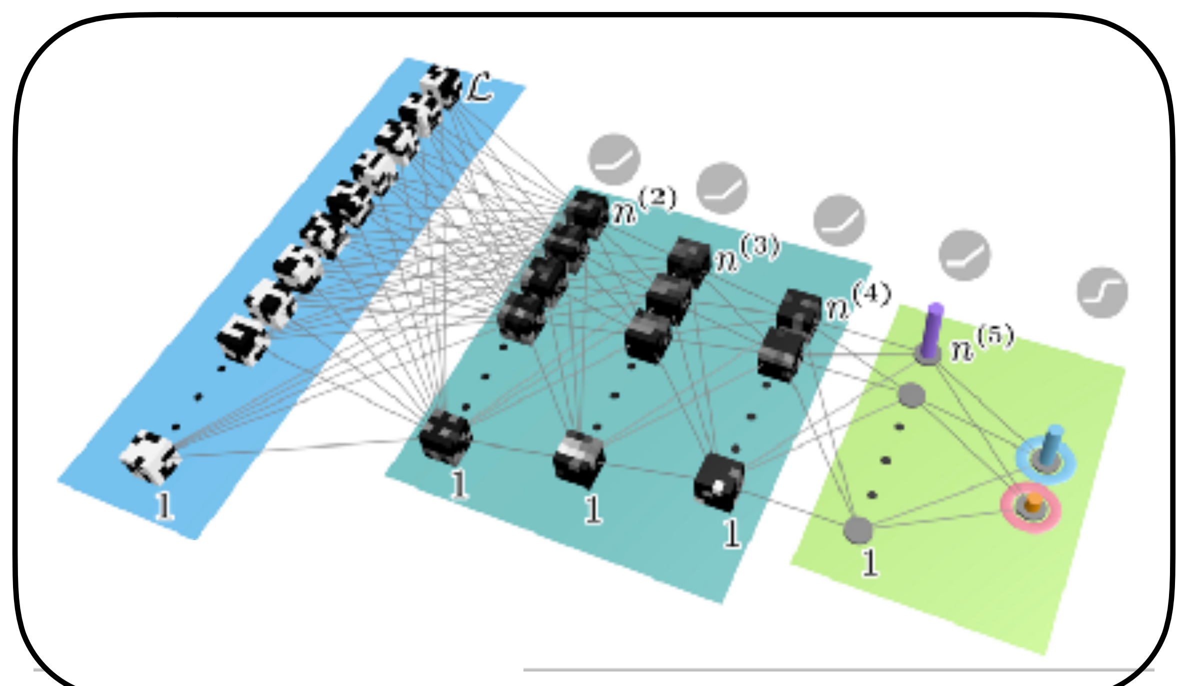
arXiv:1803.08823



arXiv:1606.02718

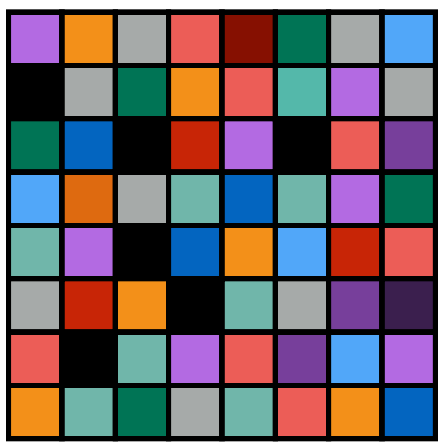


arXiv:1610.04238



arXiv:1609.02552

Application of neural networks: classifying images



Input: an image

Computer

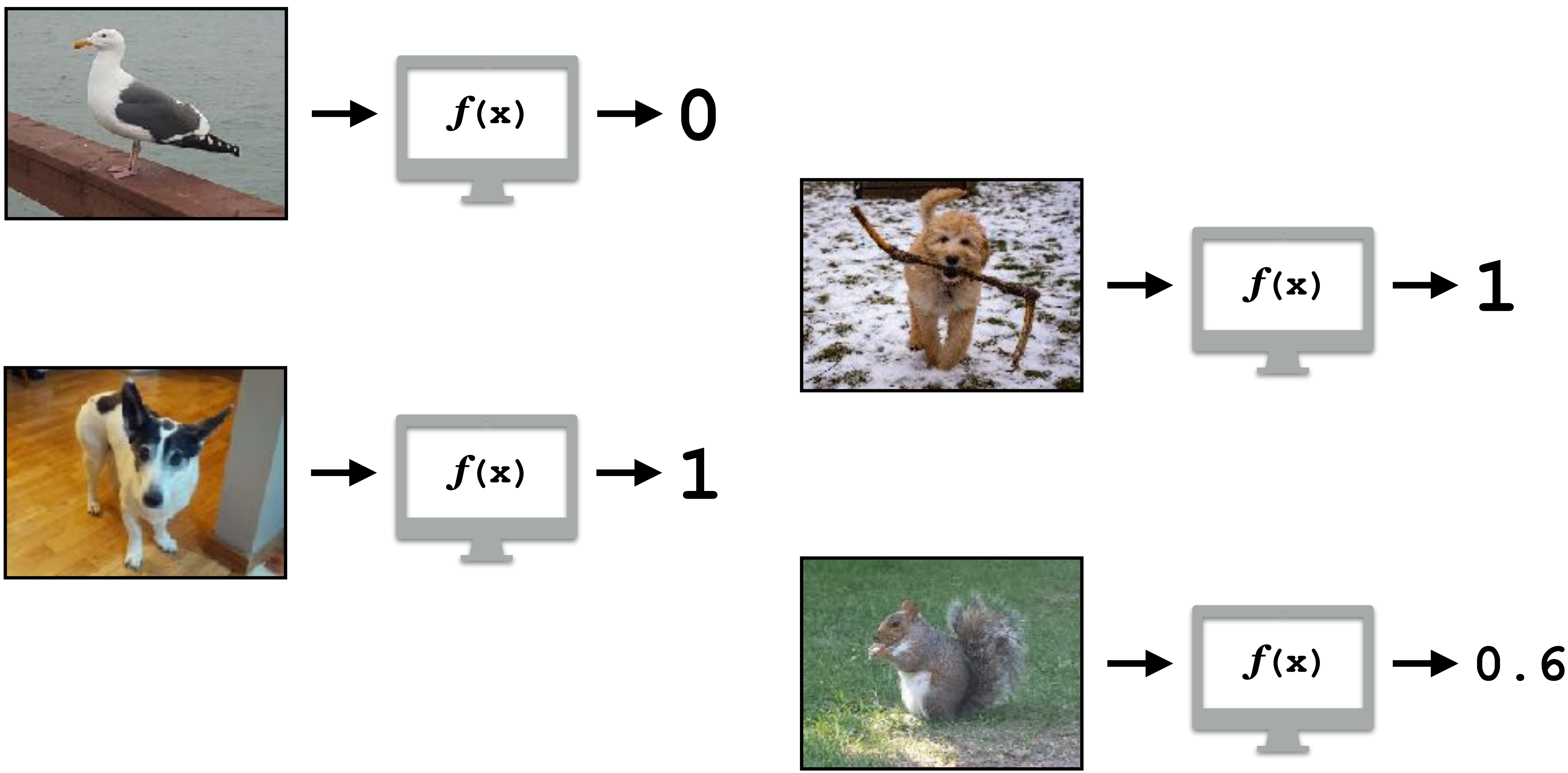
Output: some description of the image

Or, more mathematically:

Input: $\mathbf{x}$ → Function f → Output: $\mathbf{y} = f(\mathbf{x})$

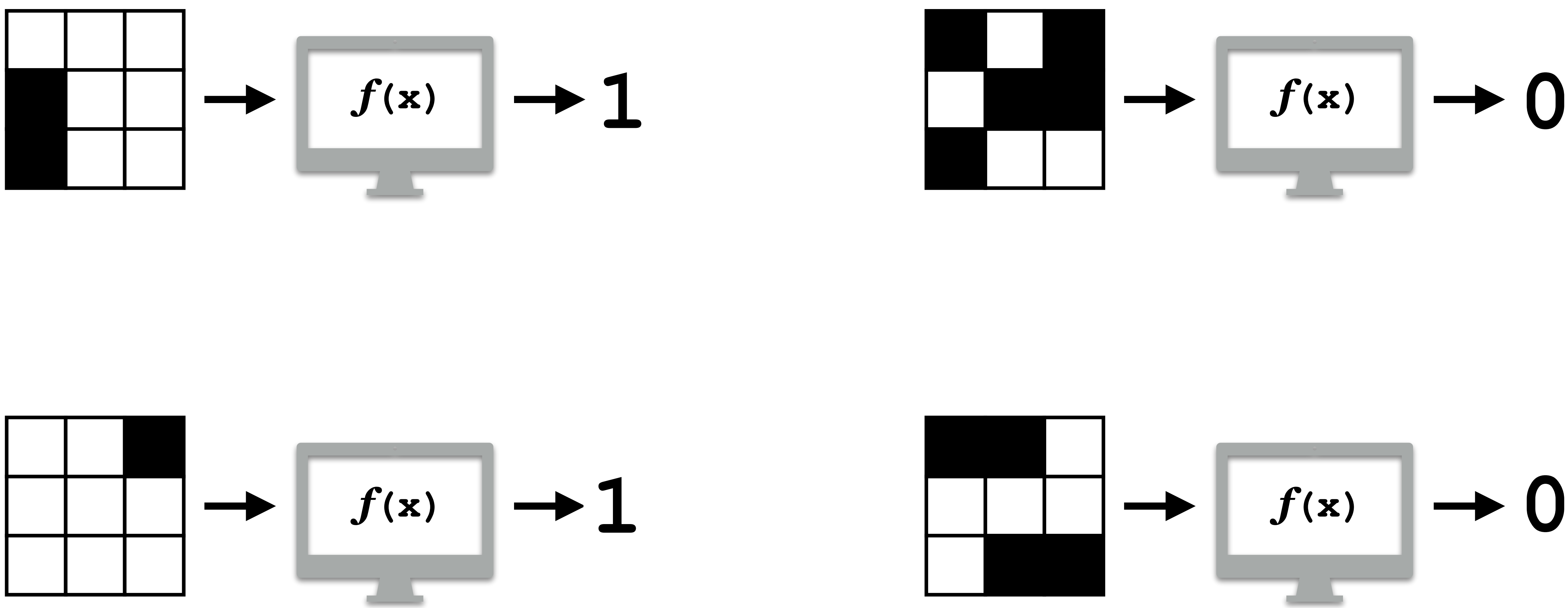
Application of neural networks: classifying images

Example: classifying images of birds and dogs

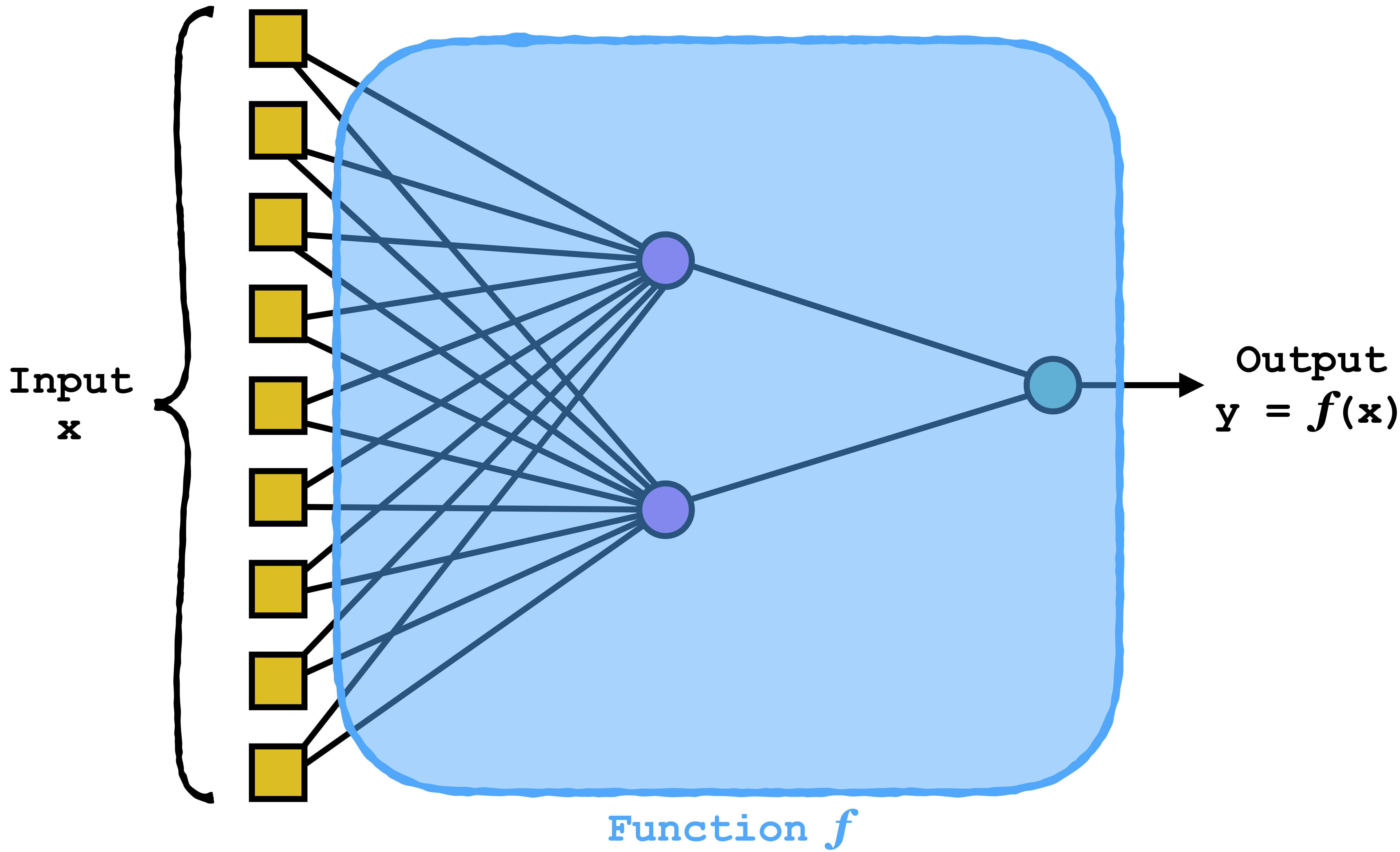


Application of neural networks: classifying images

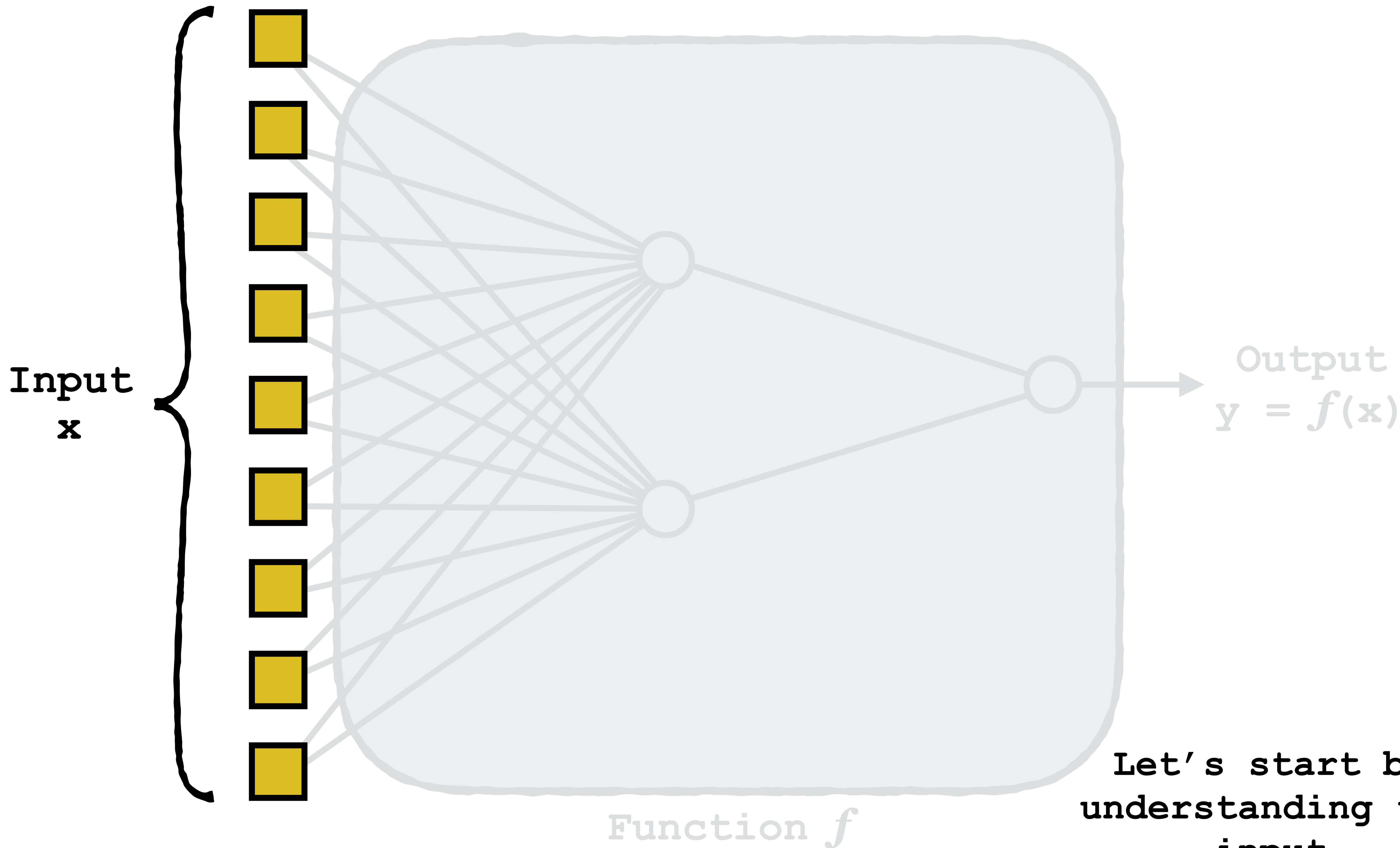
Example: identifying whether an image contains one rectangle



Neural networks



Neural networks



Let's start by understanding the input

Neural network input

Input
$\mathbf{x}$

$\mathbf{x}_1 = 0$

$\mathbf{x}_2 = 1$

$\mathbf{x}_3 = 1$

$\mathbf{x}_4 = 0$

$\mathbf{x}_5 = 1$

$\mathbf{x}_6 = 0$

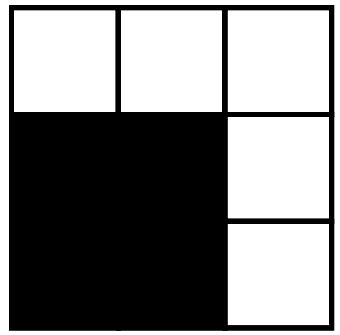
$\mathbf{x}_7 = 0$

$\mathbf{x}_8 = 0$

$\mathbf{x}_9 = 1$

$$\mathbf{x} = [0,1,1,0,1,0,0,0,1]$$

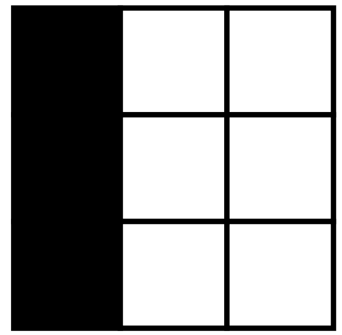
Neural network input

How do we translate  into an input for the neural network?

$x_1 = 0$	$x_2 = 0$	$x_3 = 0$
$x_4 = 1$	$x_5 = 1$	$x_6 = 0$
$x_7 = 1$	$x_8 = 1$	$x_9 = 0$

$$\mathbf{x} = [0,0,0,1,1,0,1,1,0]$$

Neural network input

How do we translate  into an input for the neural network?

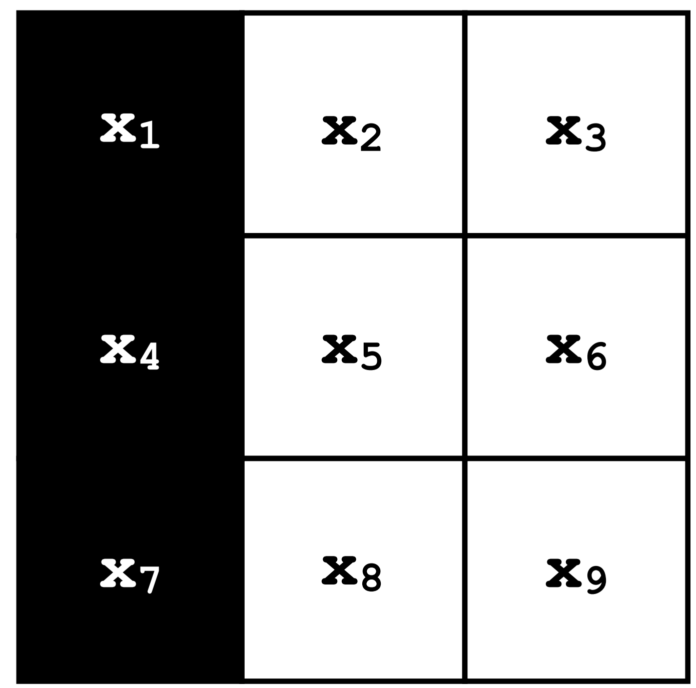
a) [1,1,1,0,0,0,0,0,0]

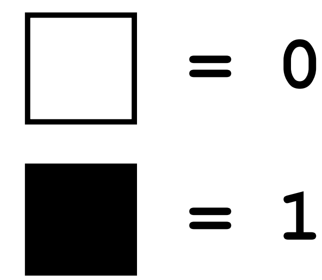
b) [0,1,1,0,1,1,0,1,1]

c) [0,0,0,1,1,1,1,1,1]

d) [1,0,0,1,0,0,1,0,0]

Hint:

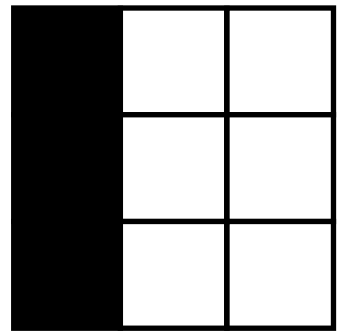




$\mathbf{x} = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9]$

$= ?$

Neural network input

How do we translate  into an input for the neural network?

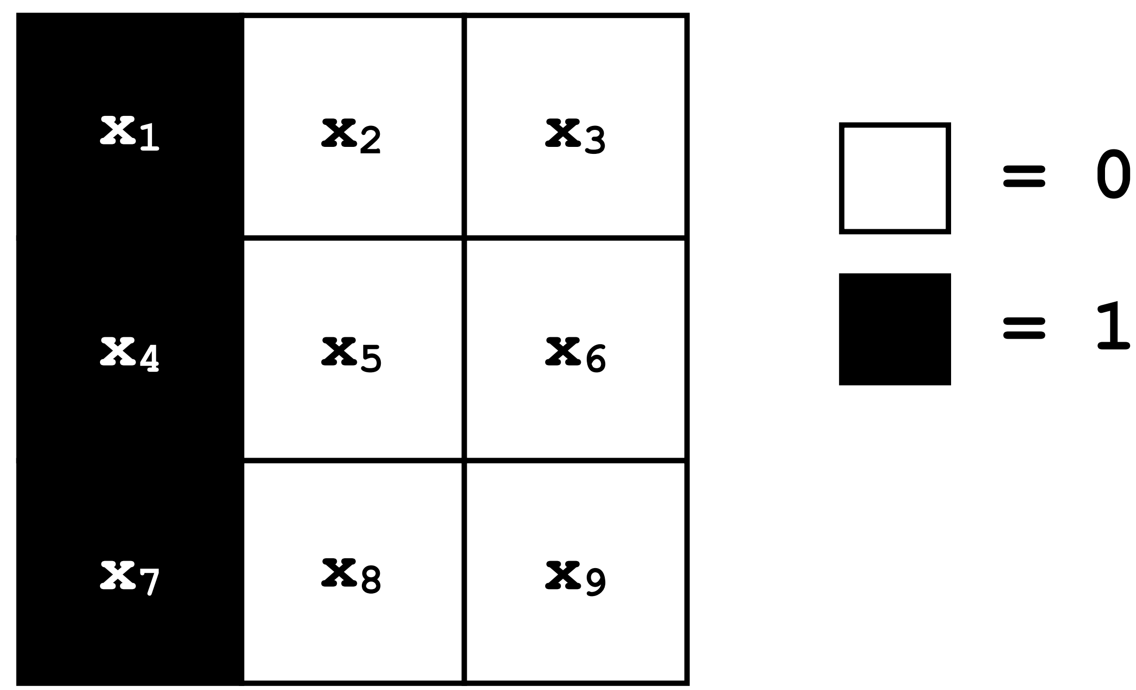
a) [1,1,1,0,0,0,0,0,0]

b) [0,1,1,0,1,1,0,1,1]

c) [0,0,0,1,1,1,1,1,1]

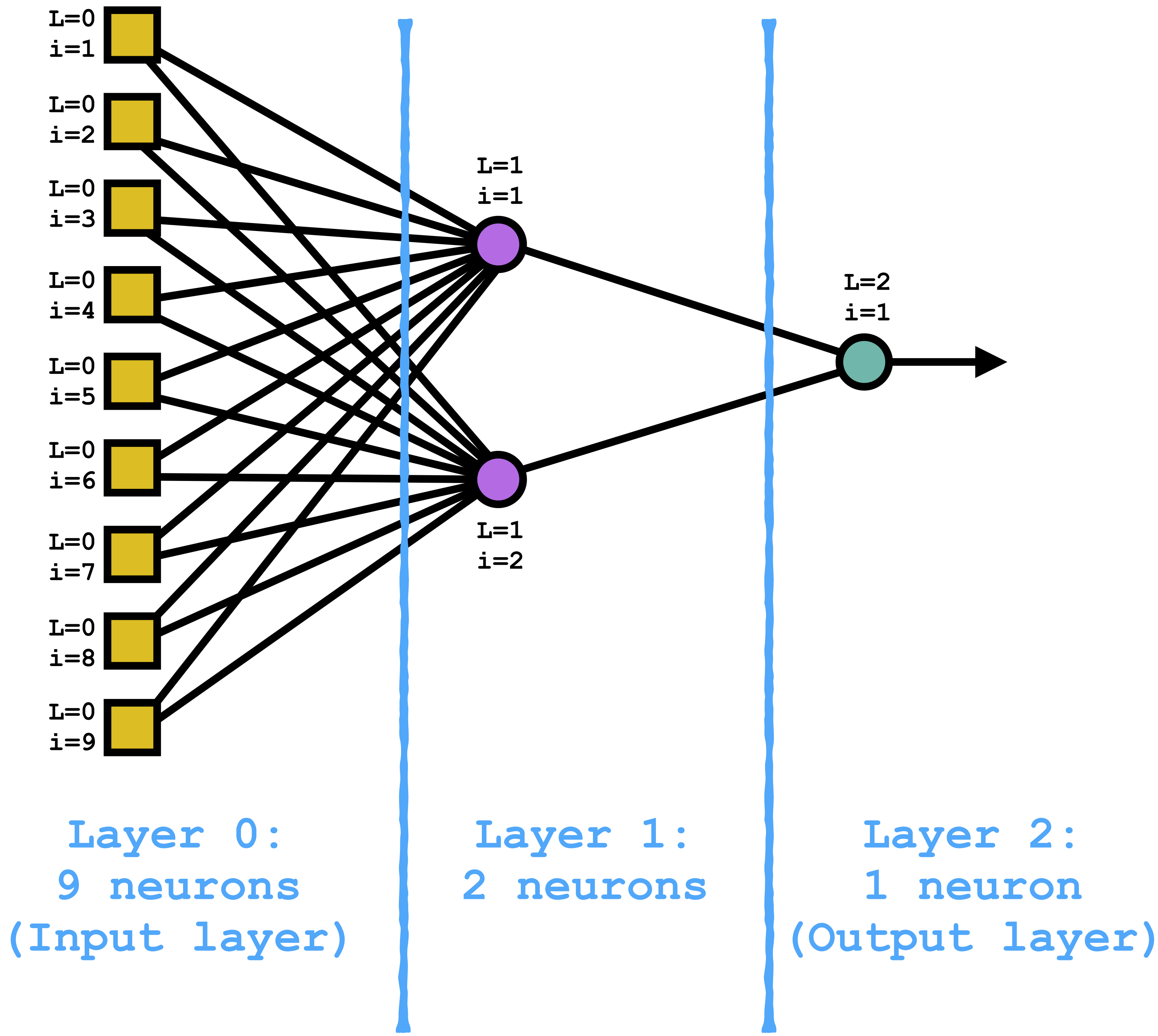
d) [1,0,0,1,0,0,1,0,0]

Hint:



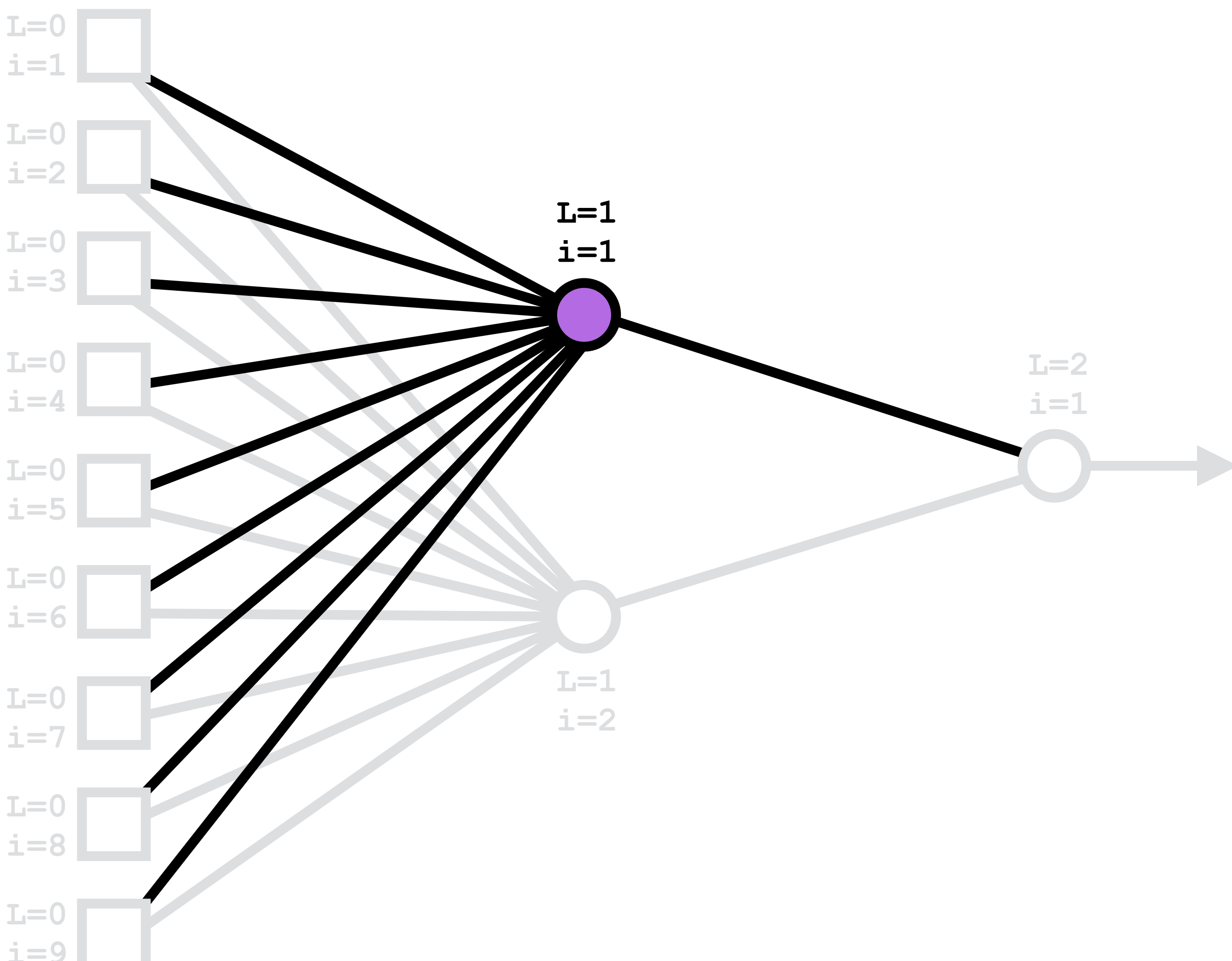
$$\mathbf{x} = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9]$$
$$= ?$$

Neural network layers

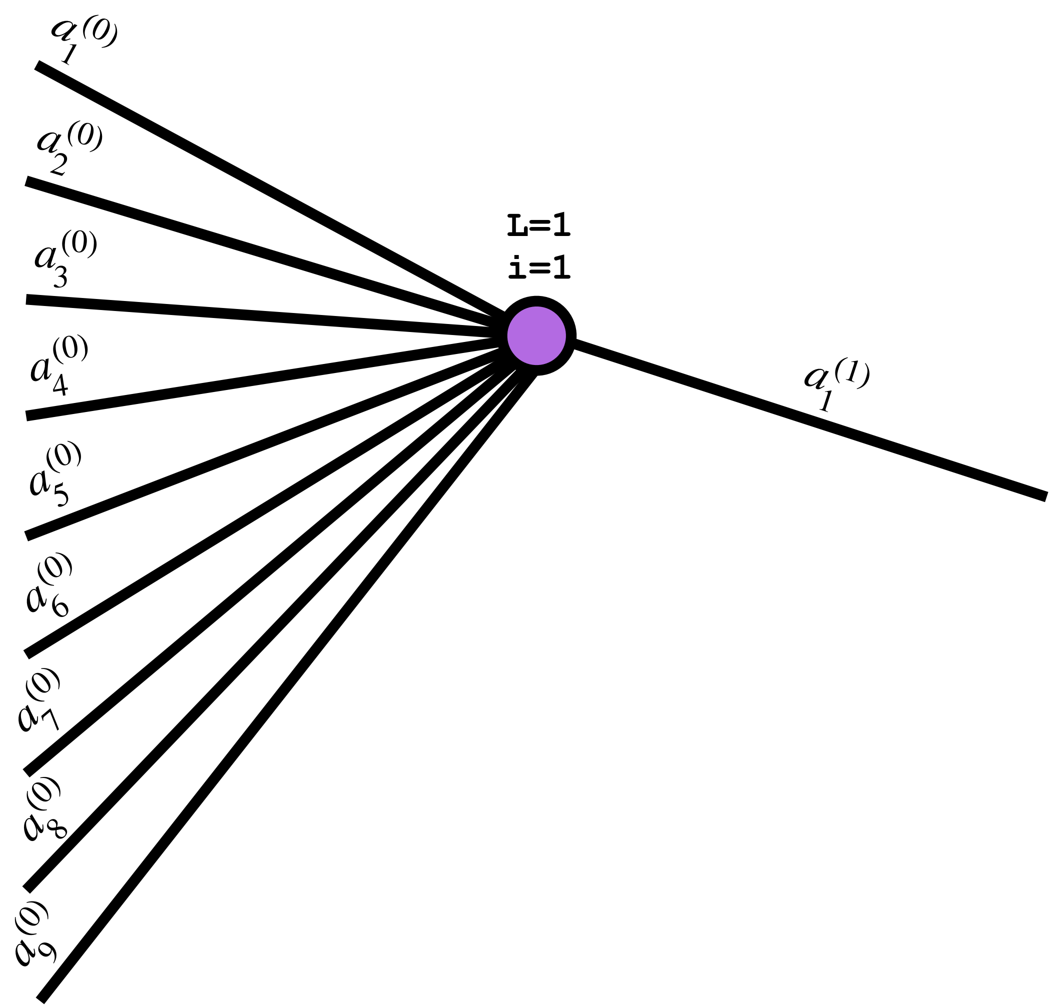


- **The neurons are labelled by their layer L and by another number i**
- **The output from neuron i in layer L is given by the variable** $a_i^{(L)}$

Neuron output



Neuron output



The output $a_i^{(L)}$ is given by

$$a_i^{(L)} = g\left(\boxed{\sum_j a_j^{(L-1)} W_{ji}^{(L)} + b_i^{(L)}}\right)$$

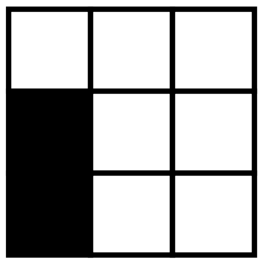
Linear algebra!

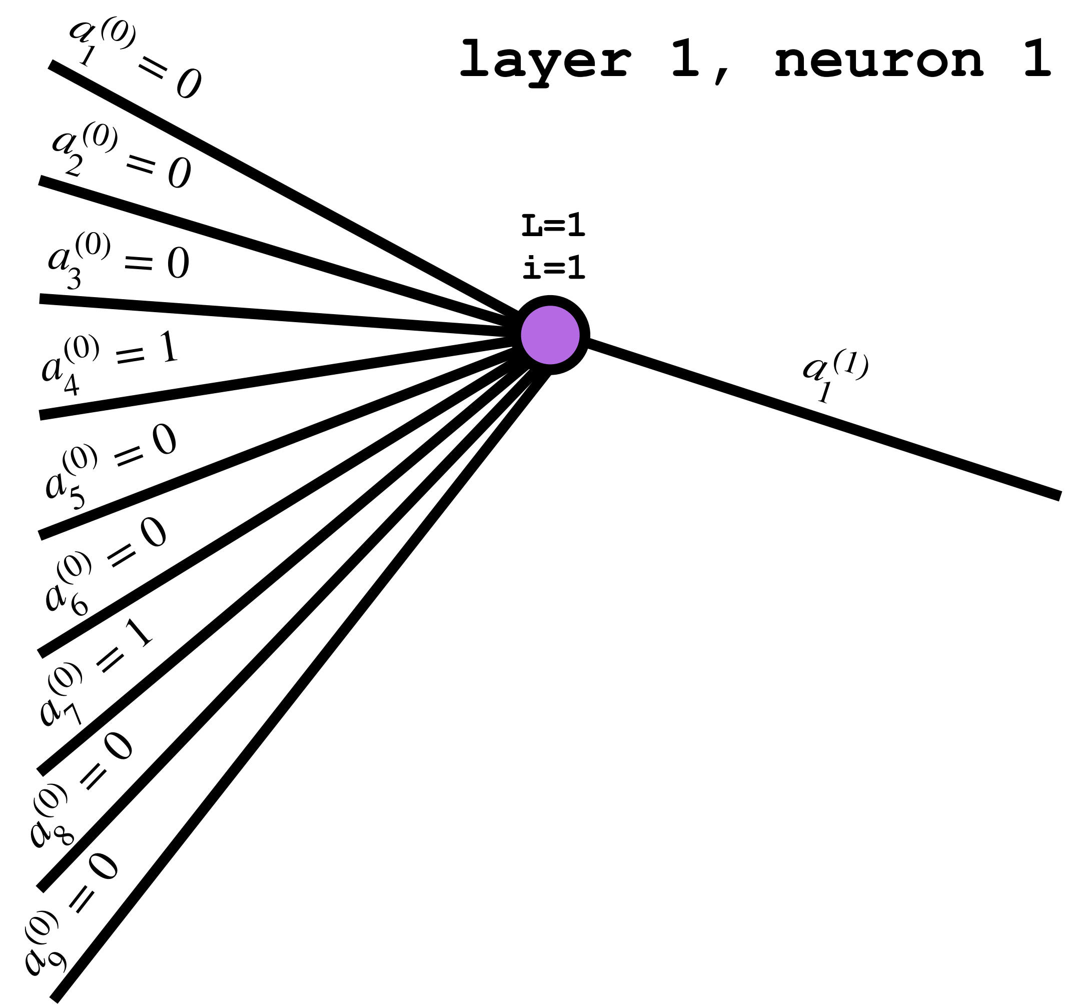
where:

- $g(z) = \dfrac{1}{1+e^{-z}}$
- **The variables $W_{ji}^{(L)}$ are called weights**
- **The variables $b_i^{(L)}$ are called biases**

Neuron output

Example:

$x =$  $= [0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0]$



$$W^{(1)} = \begin{bmatrix} 2.8 & 0.6 \\ 3.6 & 5.2 \\ 11.8 & -0.2 \\ -3.8 & 4.5 \\ -6.4 & 9.0 \\ 8.6 & 3.9 \\ -0.3 & 3.3 \\ 1.8 & 9.4 \\ 10.0 & 3.7 \end{bmatrix}$$

$$b^{(1)} = [-0.3 \quad -7.8]$$

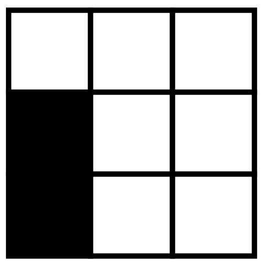
$$a_i^{(L)} = g\left(\sum_j a_j^{(L-1)} W_{ji}^{(L)} + b_i^{(L)}\right)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

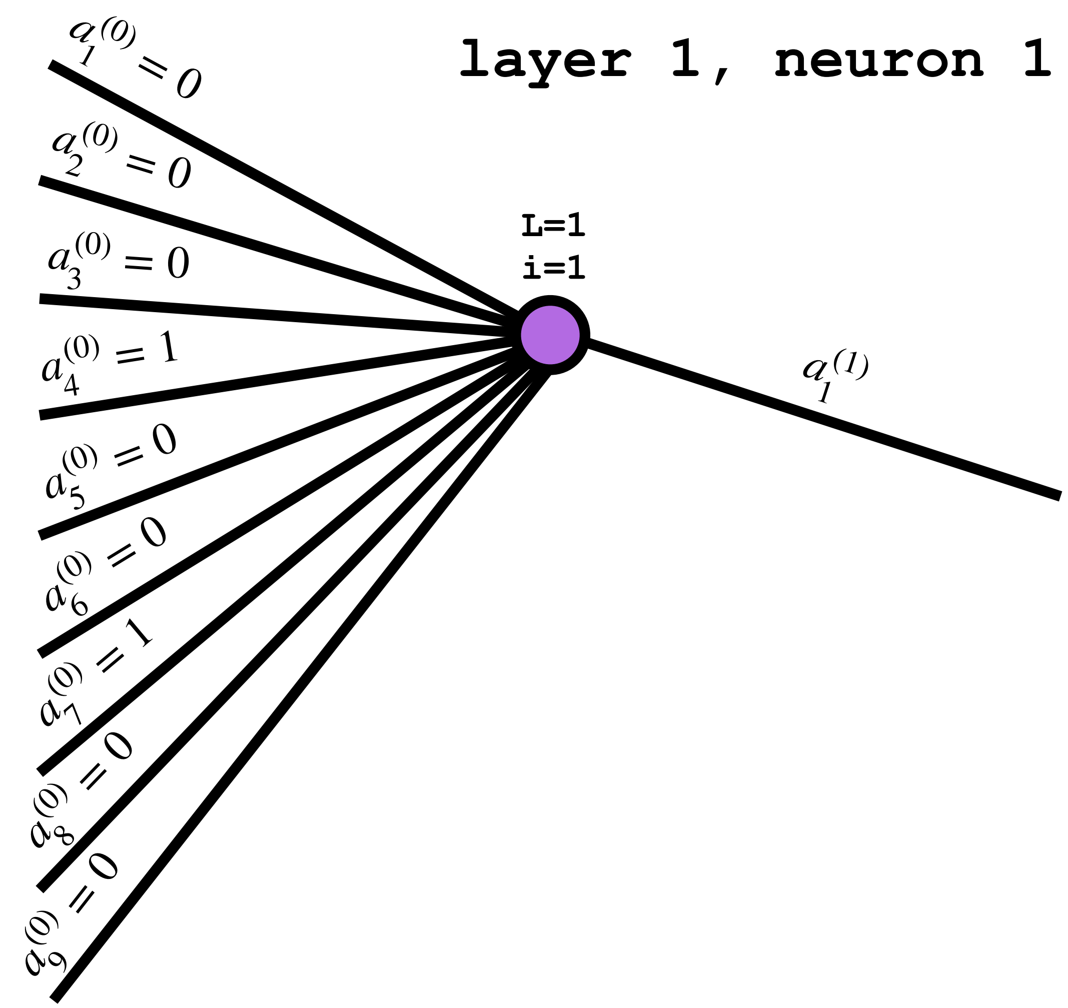
$$\sum_j a_j^{(0)} W_{j1}^{(1)} = 0 + 0 + 0 + (1 \times (-3.8)) + 0 + 0 + (1 \times (-0.3)) + 0 + 0 \ = -4.1$$

Neuron output

Example:

$x =$  $= [0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0]$

layer 1, neuron 1



$$W^{(1)} = \begin{bmatrix} 2.8 & 0.6 \\ 3.6 & 5.2 \\ 11.8 & -0.2 \\ -3.8 & 4.5 \\ -6.4 & 9.0 \\ 8.6 & 3.9 \\ -0.3 & 3.3 \\ 1.8 & 9.4 \\ 10.0 & 3.7 \end{bmatrix}$$

$$b^{(1)} = [-0.3 \quad -7.8]$$

$$a_i^{(L)} = g\left(\sum_j a_j^{(L-1)} W_{ji}^{(L)} + b_i^{(L)} \right)$$

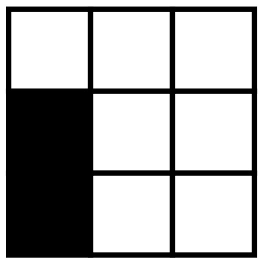
$$g(z) = \frac{1}{1 + e^{-z}}$$

$$\sum_j a_j^{(0)} W_{j1}^{(1)} = 0 + 0 + 0 + (1 \times (-3.8)) + 0 + 0 + (1 \times (-0.3)) + 0 + 0 = -4.1$$

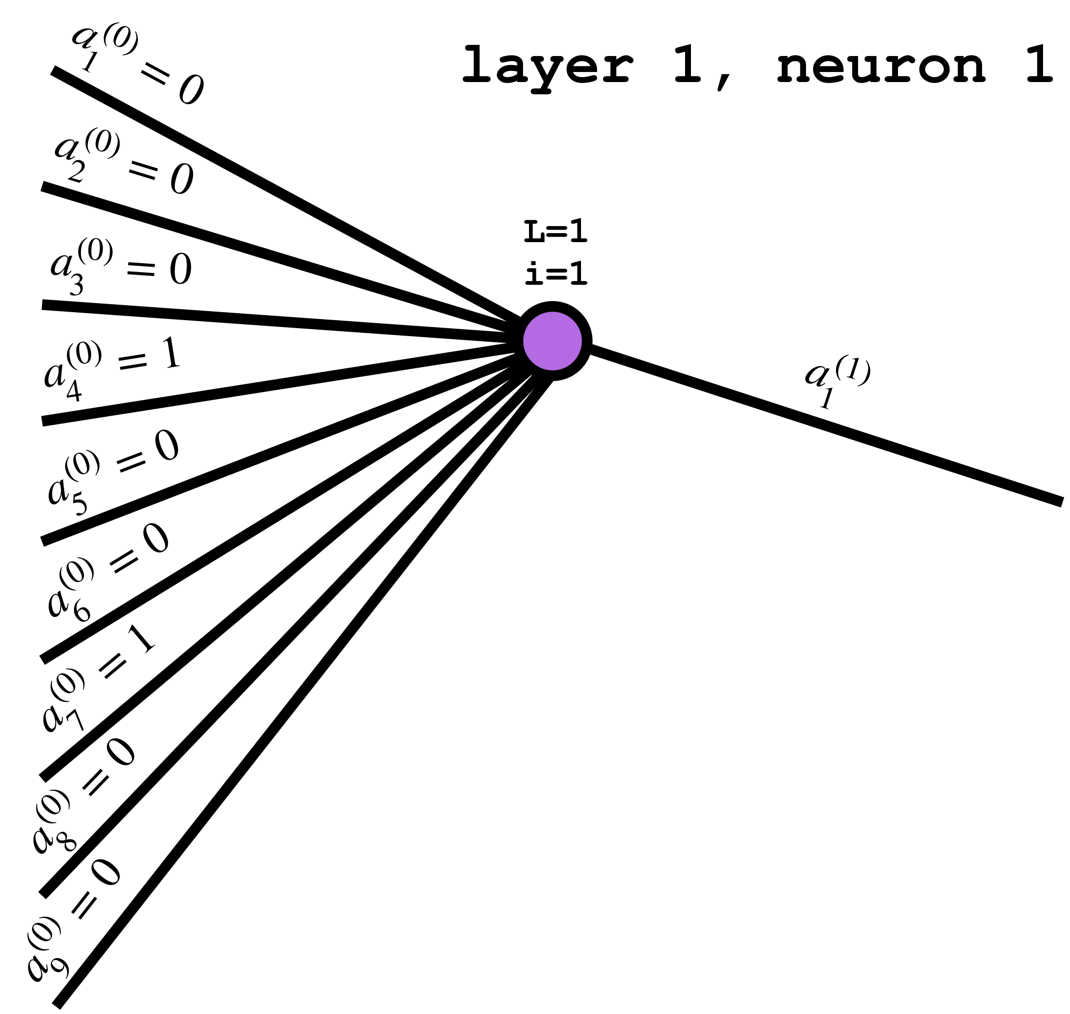
$$\sum_j a_j^{(0)} W_{j1}^{(1)} + b_1^{(1)} = -4.1 + (-0.3) = -4.4$$

Neuron output

Example:

$x =$  $= [0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0]$

$$W^{(1)} = \begin{bmatrix} 2.8 & 0.6 \\ 3.6 & 5.2 \\ 11.8 & -0.2 \\ -3.8 & 4.5 \\ -6.4 & 9.0 \\ 8.6 & 3.9 \\ -0.3 & 3.3 \\ 1.8 & 9.4 \\ 10.0 & 3.7 \end{bmatrix} \qquad b^{(1)} = [-0.3 \quad -7.8]$$



$$a_i^{(L)} = g\left(\sum_j a_j^{(L-1)} W_{ji}^{(L)} + b_i^{(L)}\right)$$

$$g(z) = \frac{1}{1+e^{-z}}$$

$$\sum_j a_j^{(0)} W_{j1}^{(1)} = 0+0+0+(1\times(-3.8))+0+0+(1\times(-0.3))+0+0 = -4.1$$

$$\sum_j a_j^{(0)} W_{j1}^{(1)} + b_1^{(1)} = -4.1 + (-0.3) = -4.4$$

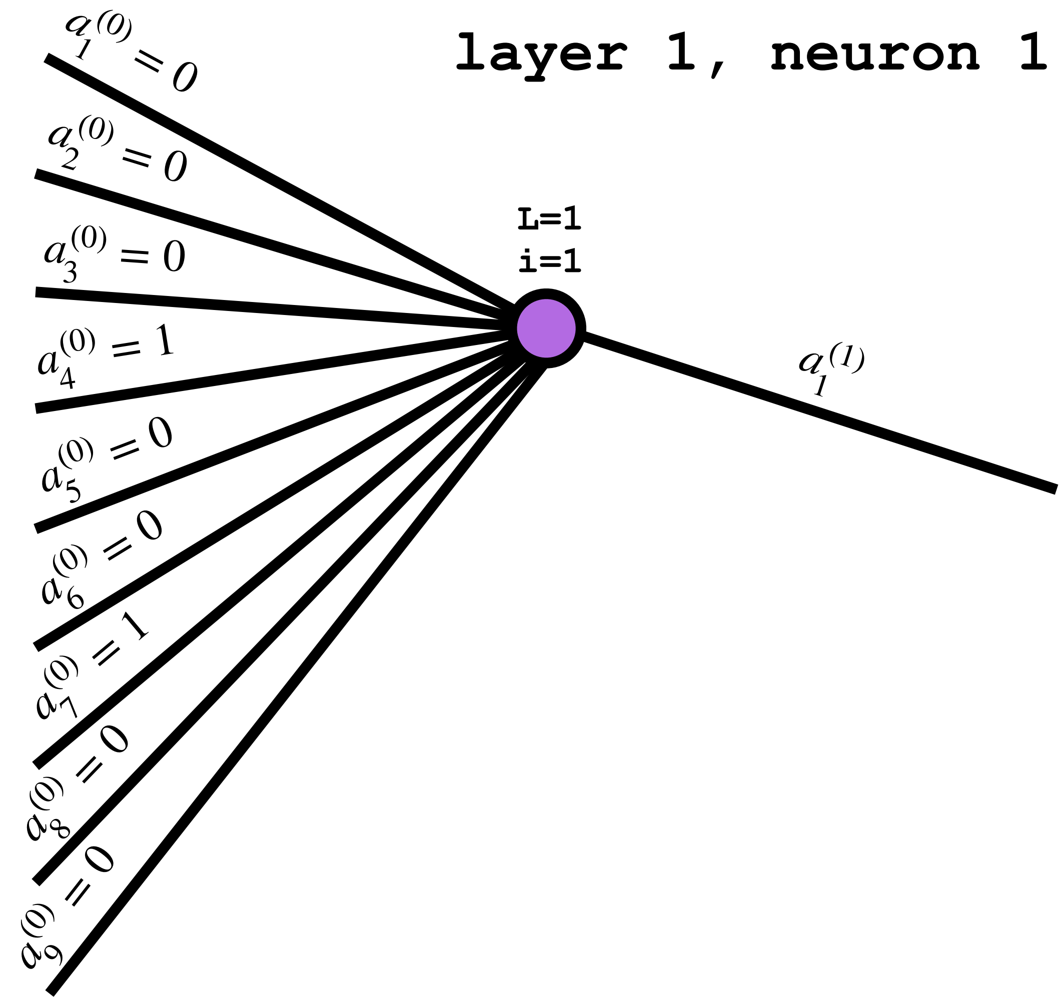
$$a_1^{(1)} = g\left(\sum_j a_j^{(0)} W_{j1}^{(1)} + b_1^{(1)}\right) = g(-4.4) = \frac{1}{1+e^{-(-4.4)}} = 0.012$$

Neuron output

Example:

$x =$ $= [0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0]$

$$W^{(1)} = \begin{bmatrix} 2.8 & 0.6 \\ 3.6 & 5.2 \\ 11.8 & -0.2 \\ -3.8 & 4.5 \\ -6.4 & 9.0 \\ 8.6 & 3.9 \\ -0.3 & 3.3 \\ 1.8 & 9.4 \\ 10.0 & 3.7 \end{bmatrix} \qquad b^{(1)} = [-0.3 \quad -7.8]$$



$$a_i^{(L)} = g\left(\sum_j a_j^{(L-1)} W_{ji}^{(L)} + b_i^{(L)}\right)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

$$\sum_j a_j^{(0)} W_{j1}^{(1)} = 0 + 0 + 0 + (1 \times (-3.8)) + 0 + 0 + (1 \times (-0.3)) + 0 + 0 = -4.1$$

$$\sum_j a_j^{(0)} W_{j1}^{(1)} + b_1^{(1)} = -4.1 + (-0.3) = -4.4$$

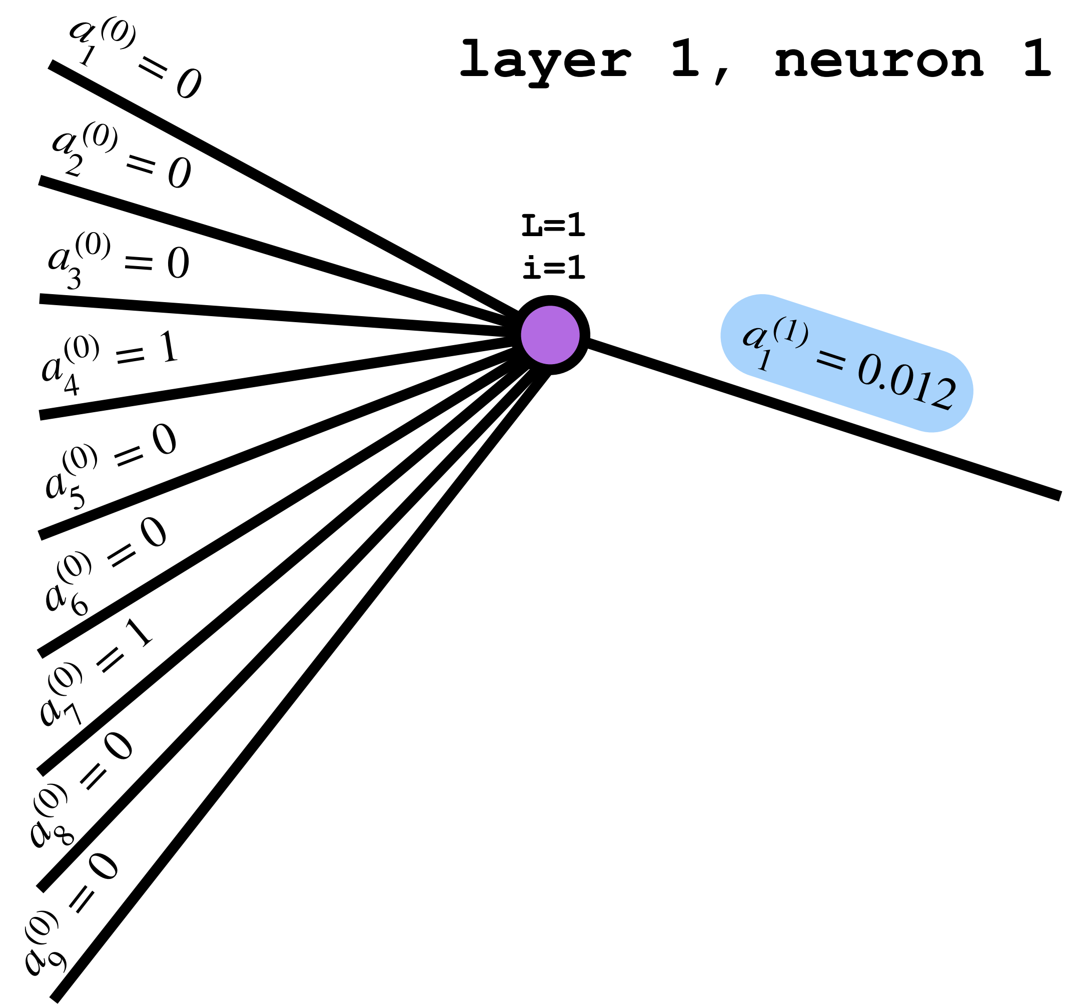
$$a_1^{(1)} = g\left(\sum_j a_j^{(0)} W_{j1}^{(1)} + b_1^{(1)}\right) = g(-4.4) = \frac{1}{1 + e^{-(-4.4)}} = \boxed{0.012}$$

Output from layer 1, neuron 1

Neuron output

Example:

$x =$ $= [0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0]$



$$W^{(1)} = \begin{bmatrix} 2.8 & 0.6 \\ 3.6 & 5.2 \\ 11.8 & -0.2 \\ -3.8 & 4.5 \\ -6.4 & 9.0 \\ 8.6 & 3.9 \\ -0.3 & 3.3 \\ 1.8 & 9.4 \\ 10.0 & 3.7 \end{bmatrix} \qquad b^{(1)} = [-0.3 \quad -7.8]$$

$$a_i^{(L)} = g\left(\sum_j a_j^{(L-1)} W_{ji}^{(L)} + b_i^{(L)}\right)$$

$$g(z) = \frac{1}{1+e^{-z}}$$

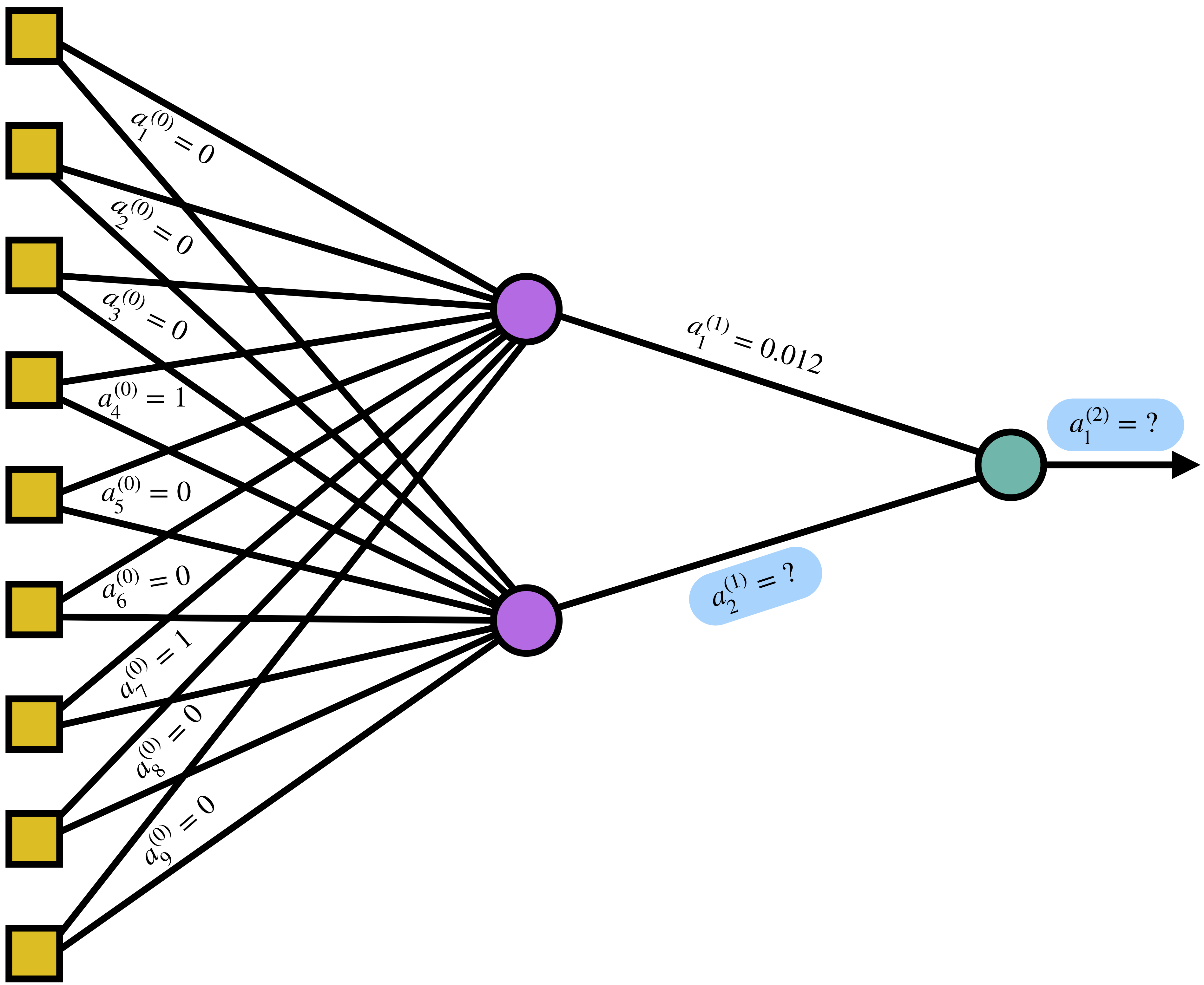
$$\sum_j a_j^{(0)} W_{j1}^{(1)} = 0 + 0 + 0 + (1 \times (-3.8)) + 0 + 0 + (1 \times (-0.3)) + 0 + 0 = -4.1$$

$$\sum_j a_j^{(0)} W_{j1}^{(1)} + b_1^{(1)} = -4.1 + (-0.3) = -4.4$$

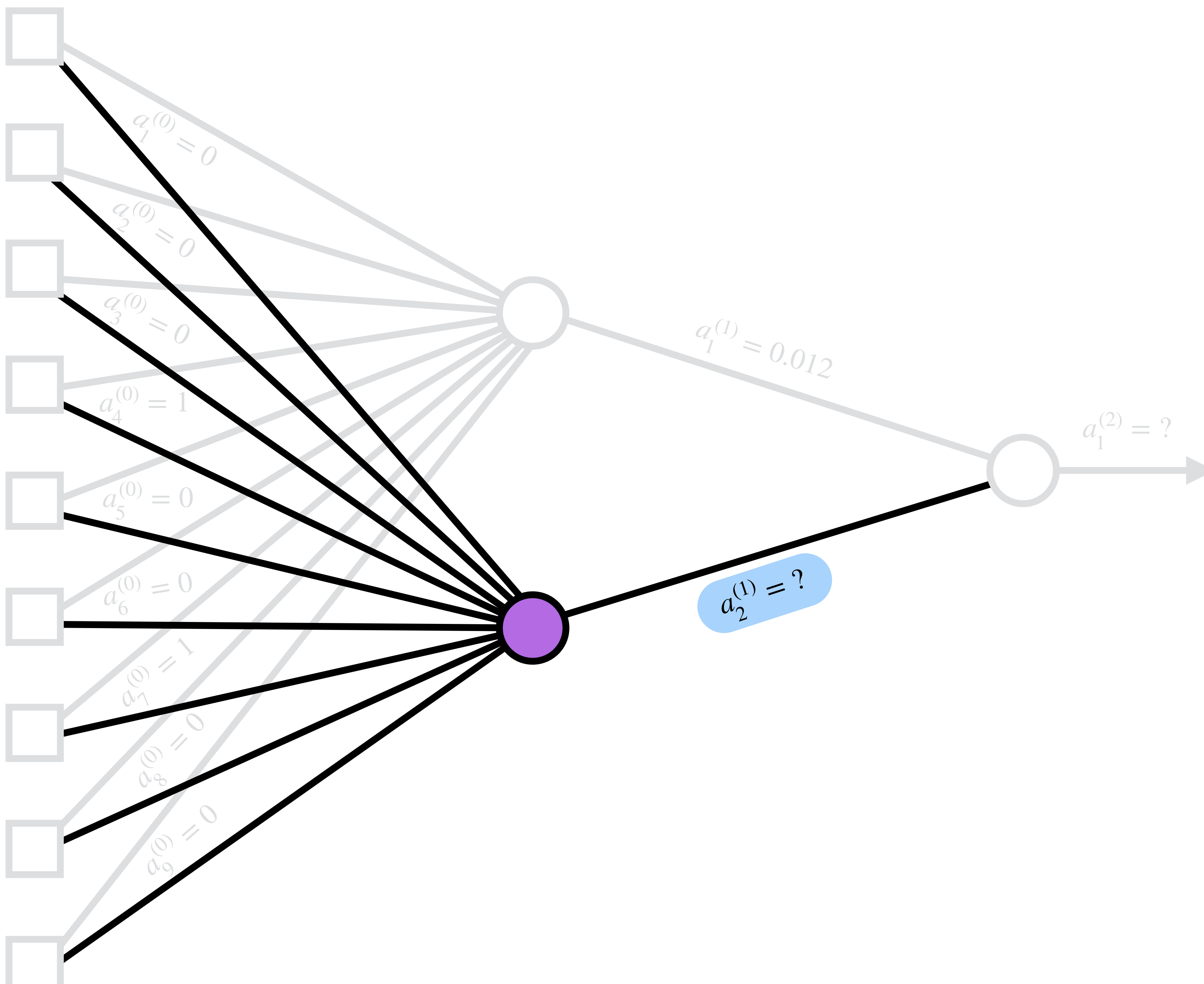
$$a_1^{(1)} = g\left(\sum_j a_j^{(0)} W_{j1}^{(1)} + b_1^{(1)}\right) = g(-4.4) = \frac{1}{1+e^{-(-4.4)}} = 0.012$$

Output from layer 1, neuron 1

Neuron outputs

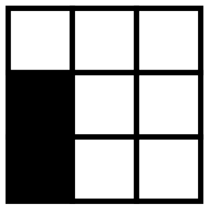


Neuron outputs

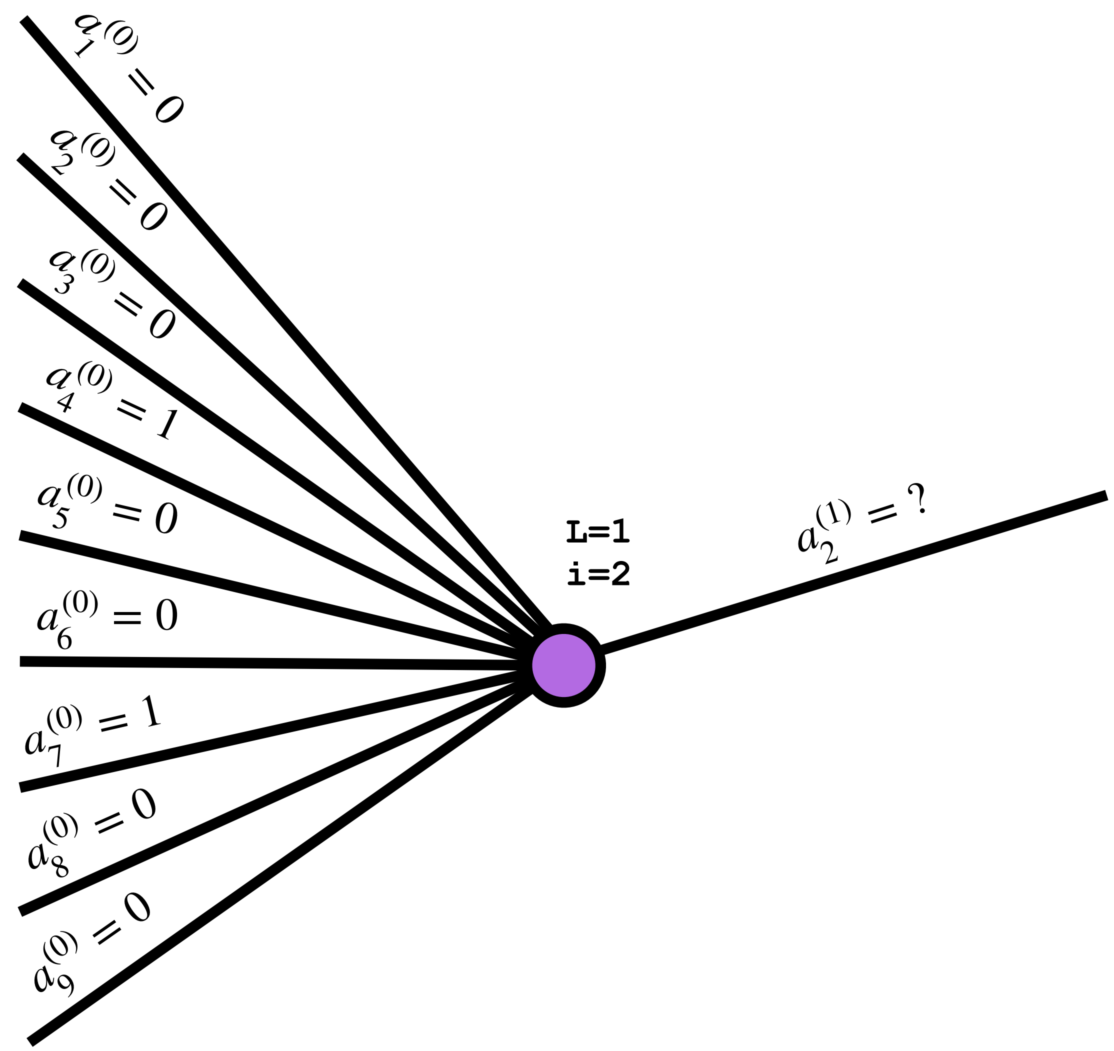


Neuron output

Now let's find the output from layer 1, neuron 2:

$x =$  $= [0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0]$

$$W^{(1)} = \begin{bmatrix} 2.8 & 0.6 \\ 3.6 & 5.2 \\ 11.8 & -0.2 \\ -3.8 & 4.5 \\ -6.4 & 9.0 \\ 8.6 & 3.9 \\ -0.3 & 3.3 \\ 1.8 & 9.4 \\ 10.0 & 3.7 \end{bmatrix} \qquad b^{(1)} = [-0.3 \quad -7.8]$$



$$a_i^{(L)} = g\left(\sum_j a_j^{(L-1)} W_{ji}^{(L)} + b_i^{(L)}\right)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

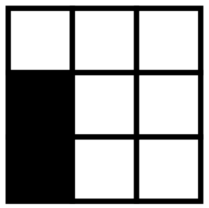
$$\sum_j a_j^{(0)} W_{j2}^{(1)} = ?$$

$$\sum_j a_j^{(0)} W_{j2}^{(1)} + b_2^{(1)} = ?$$

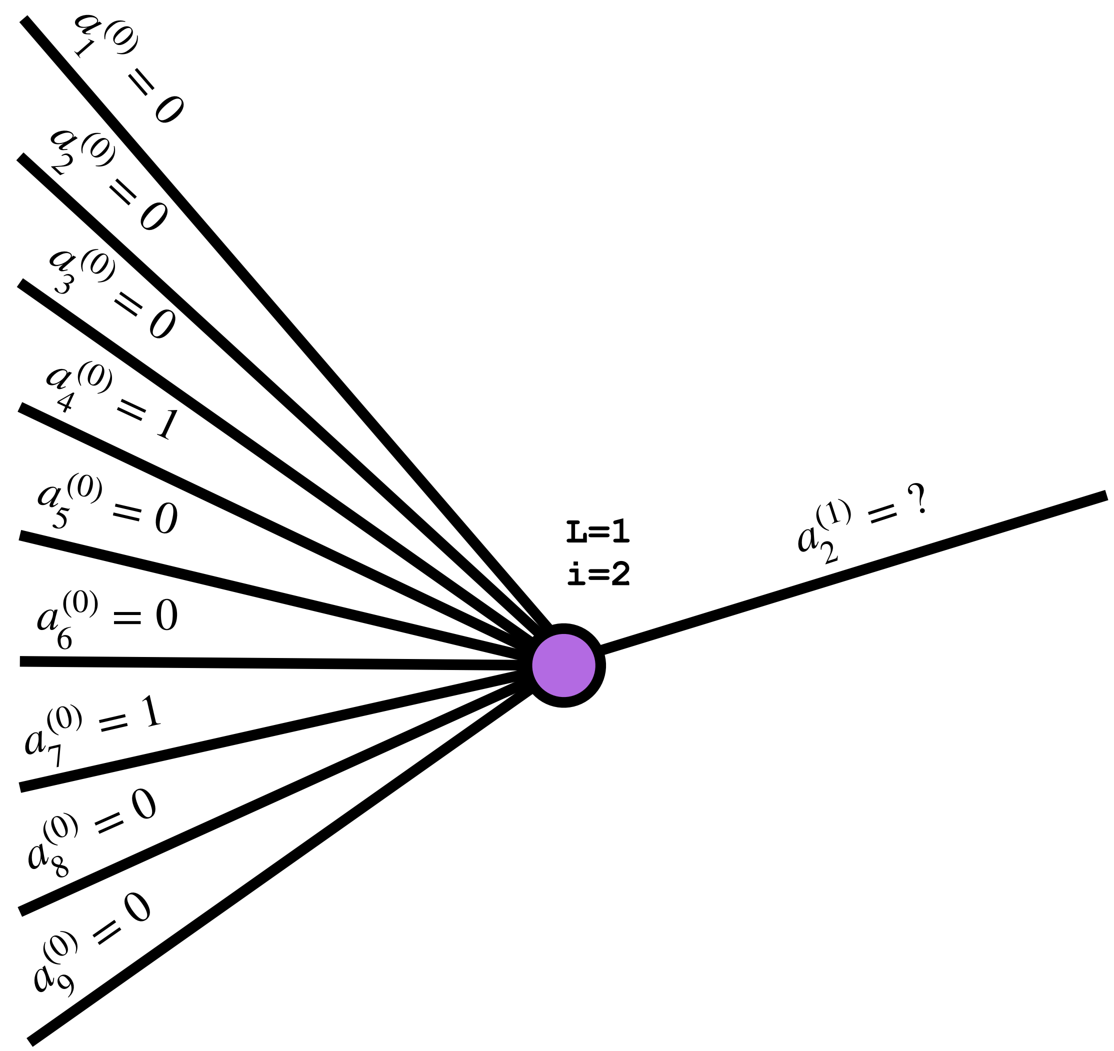
$$a_2^{(1)} = g\left(\sum_j a_j^{(0)} W_{j2}^{(1)} + b_2^{(1)}\right) = ?$$

Neuron output

Now let's find the output from layer 1, neuron 2:

$x =$  $= [0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0]$

$$W^{(1)} = \begin{bmatrix} 2.8 & 0.6 \\ 3.6 & 5.2 \\ 11.8 & -0.2 \\ -3.8 & 4.5 \\ -6.4 & 9.0 \\ 8.6 & 3.9 \\ -0.3 & 3.3 \\ 1.8 & 9.4 \\ 10.0 & 3.7 \end{bmatrix} \qquad b^{(1)} = [-0.3 \quad -7.8]$$



$$a_i^{(L)} = g\left(\sum_j a_j^{(L-1)} W_{ji}^{(L)} + b_i^{(L)}\right)$$

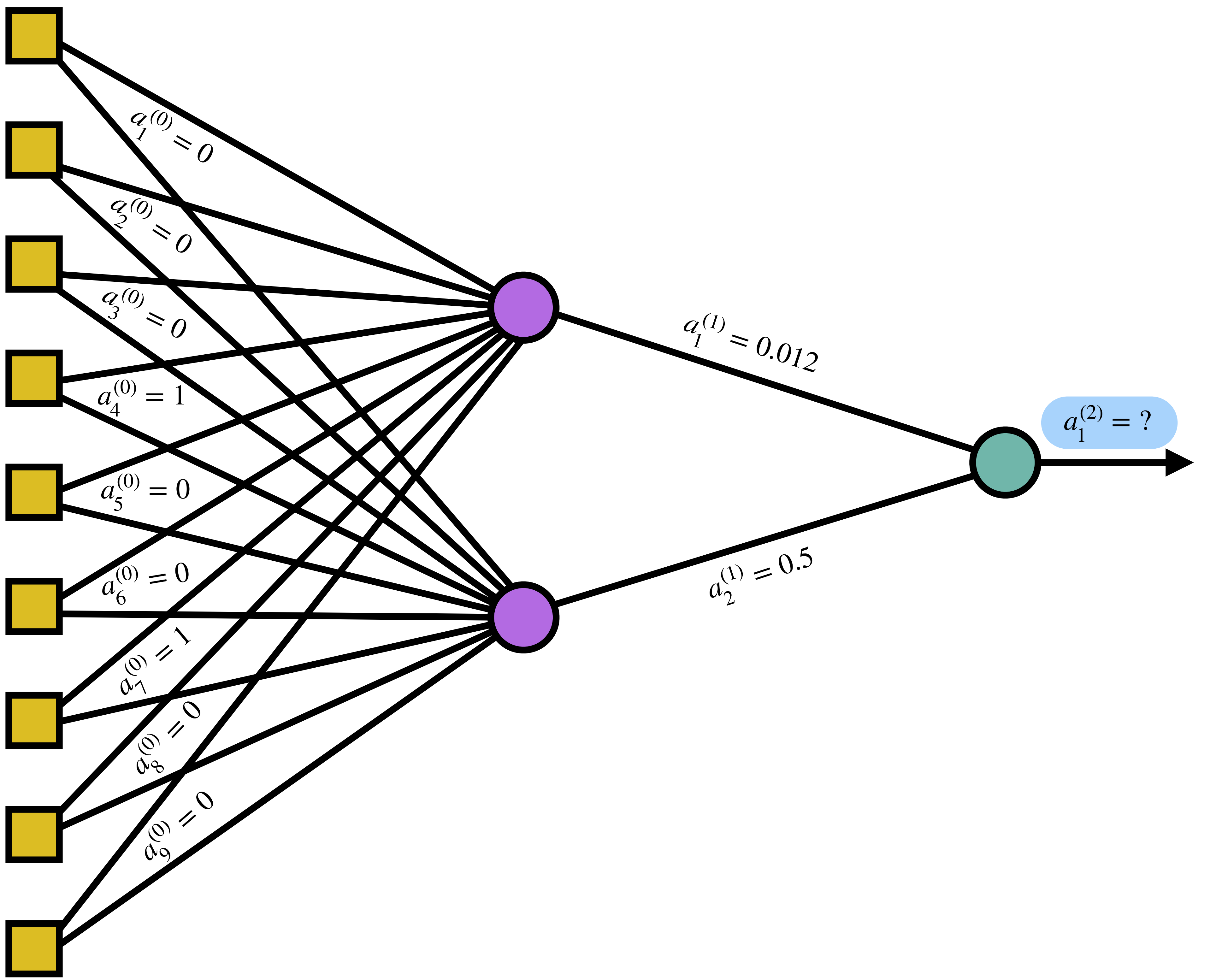
$$g(z) = \frac{1}{1 + e^{-z}}$$

$$\sum_j a_j^{(0)} W_{j2}^{(1)} = 7.8$$

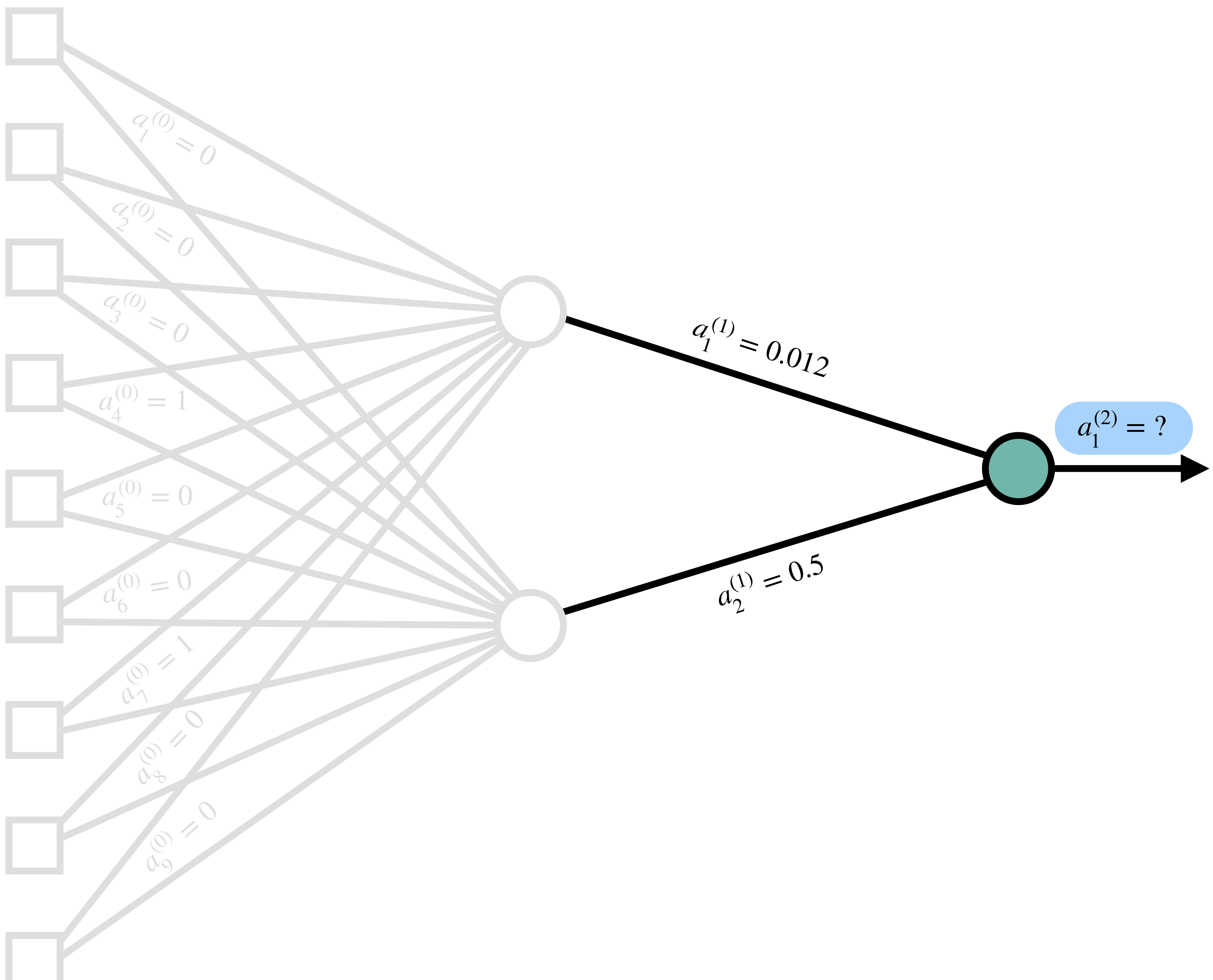
$$\sum_j a_j^{(0)} W_{j2}^{(1)} + b_2^{(1)} = 0$$

$$a_2^{(1)} = g\left(\sum_j a_j^{(0)} W_{j2}^{(1)} + b_2^{(1)}\right) = 0.5$$

Neuron outputs

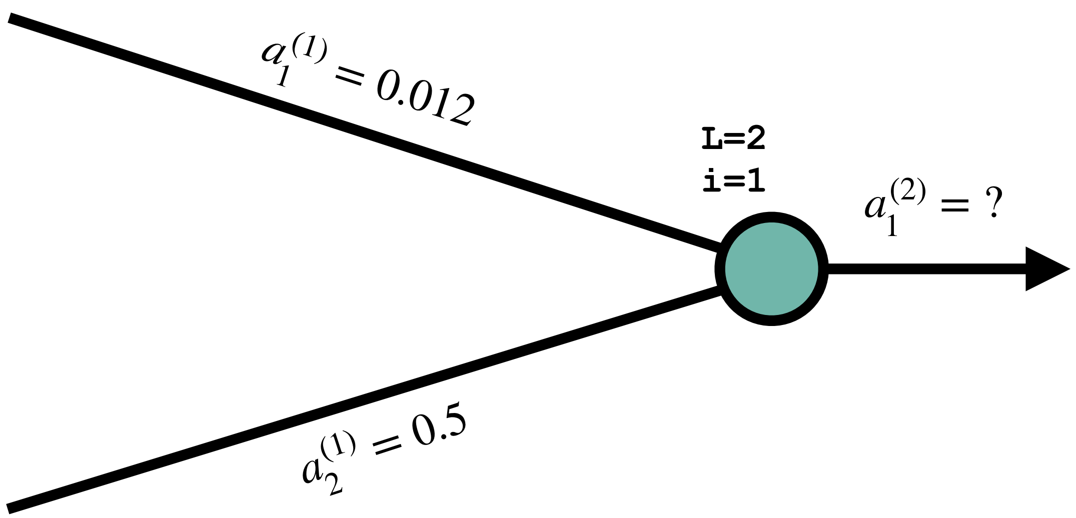


Neuron outputs



Neuron output

Finally, let's find the output from layer 2, neuron 1:



Previous layer output: $a^{(1)} = [0.012 \quad 0.5]$

$$W^{(2)} = \begin{bmatrix} -3.3 \\ -3.6 \end{bmatrix} \qquad b^{(2)} = [3.9]$$

$$a_i^{(L)} = g\left(\sum_j a_j^{(L-1)} W_{ji}^{(L)} + b_i^{(L)}\right)$$

$$g(z) = \frac{1}{1+e^{-z}}$$

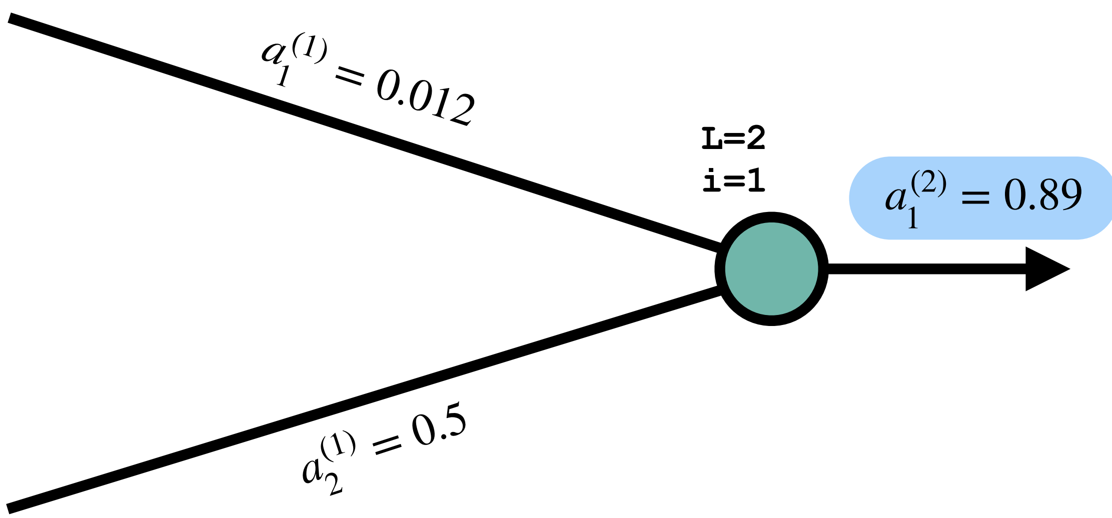
$$\sum_j a_j^{(1)} W_{j1}^{(2)} = (0.012 \times (-3.3)) + (0.5 \times (-3.6)) = -1.8$$

$$\sum_j a_j^{(1)} W_{j1}^{(2)} + b_1^{(2)} = -1.8 + 3.9 = 2.1$$

$$a_1^{(2)} = g\left(\sum_j a_j^{(1)} W_{j1}^{(2)} + b_1^{(2)}\right) = g(2.1) = \frac{1}{1+e^{-2.1}} = 0.89$$

Neuron output

Finally, let's find the output from layer 2, neuron 1:



Previous layer output: $a^{(1)} = [0.012 \quad 0.5]$

$$W^{(2)} = \begin{bmatrix} -3.3 \\ -3.6 \end{bmatrix} \qquad b^{(2)} = [3.9]$$

$$a_i^{(L)} = g\left(\sum_j a_j^{(L-1)} W_{ji}^{(L)} + b_i^{(L)}\right)$$

$$g(z) = \frac{1}{1+e^{-z}}$$

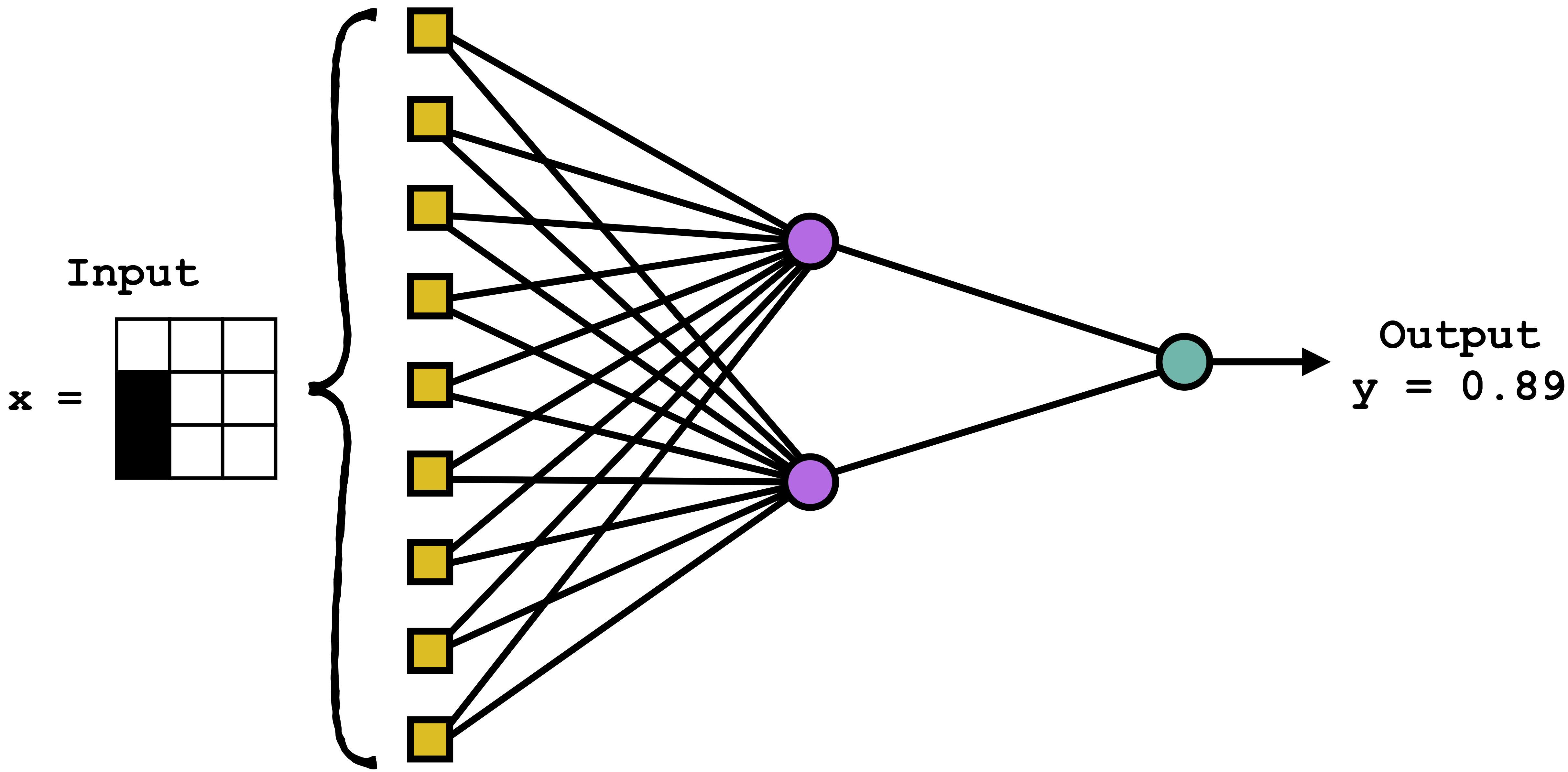
$$\sum_j a_j^{(1)} W_{j1}^{(2)} = (0.012 \times (-3.3)) + (0.5 \times (-3.6)) = -1.8$$

$$\sum_j a_j^{(1)} W_{j1}^{(2)} + b_1^{(2)} = -1.8 + 3.9 = 2.1$$

$$a_1^{(2)} = g\left(\sum_j a_j^{(1)} W_{j1}^{(2)} + b_1^{(2)}\right) = g(2.1) = \frac{1}{1+e^{-2.1}} = 0.89$$

Output from layer 2, neuron 1

Neural network output

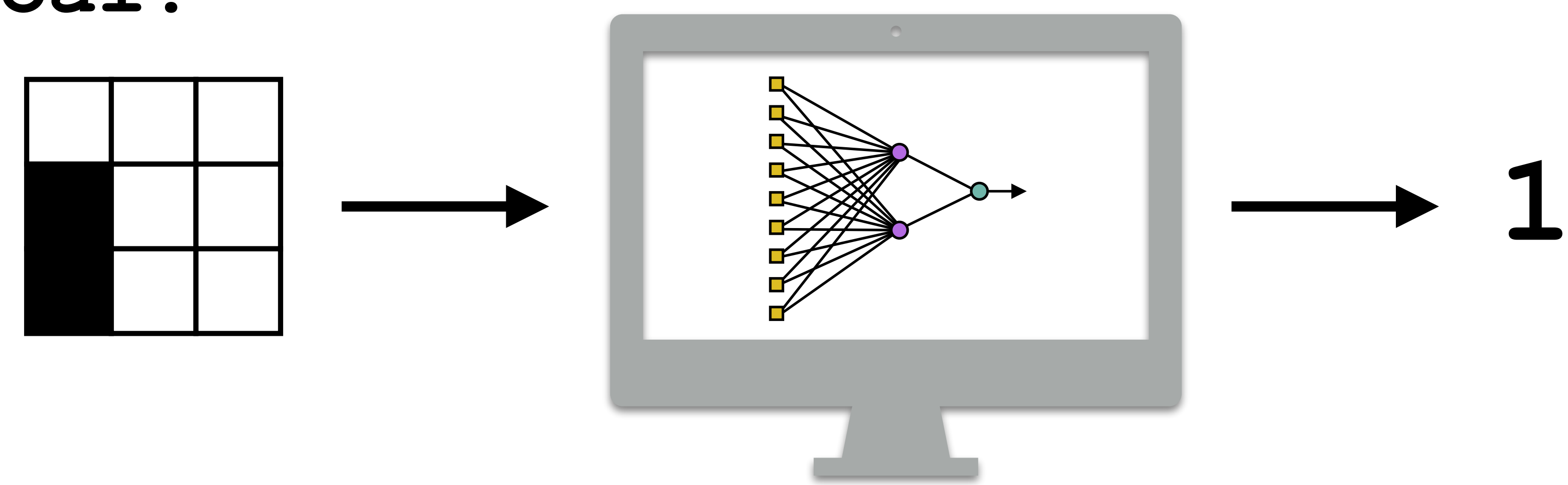


What does this output mean?

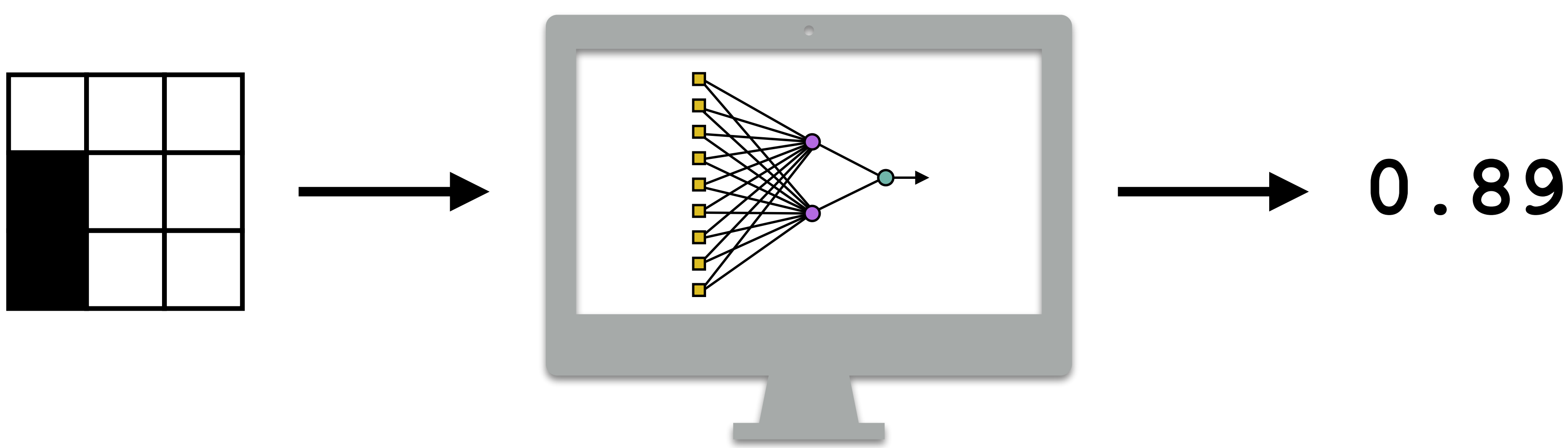
Neural network output

Recall that our goal is to identify whether an image contains one rectangle

Goal:



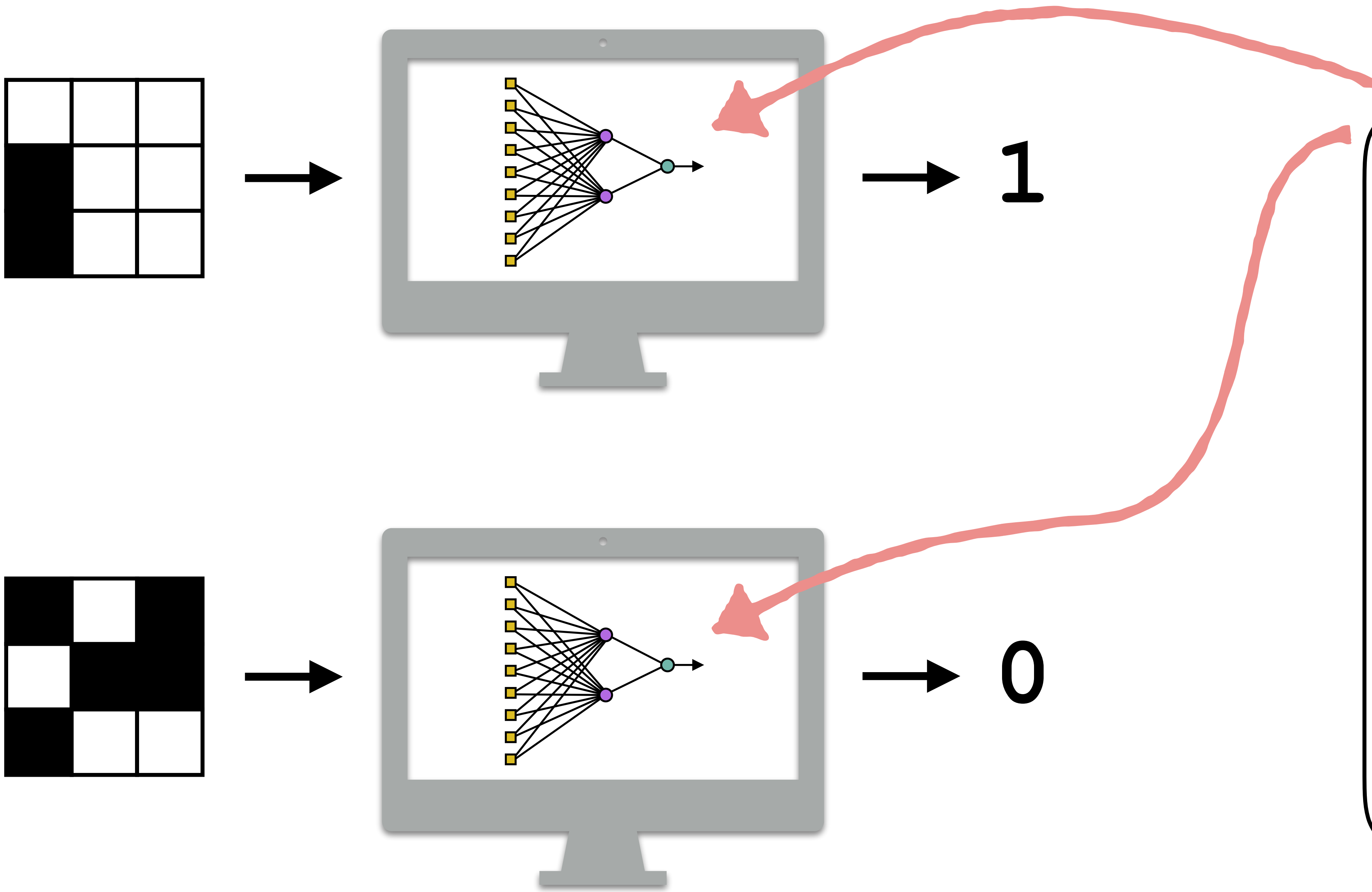
What we found:



Our network is 89% sure that the input contains one rectangle.

Neural network output

We would like our neural network to classify all possible inputs using the same weights and biases.



$$W^{(1)} = \begin{bmatrix} 2.8 & 0.6 \\ 3.6 & 5.2 \\ 11.8 & -0.2 \\ -3.8 & 4.5 \\ -6.4 & 9.0 \\ 8.6 & 3.9 \\ -0.3 & 3.3 \\ 1.8 & 9.4 \\ 10.0 & 3.7 \end{bmatrix}$$

$$b^{(1)} = [-0.3 \quad -7.8]$$

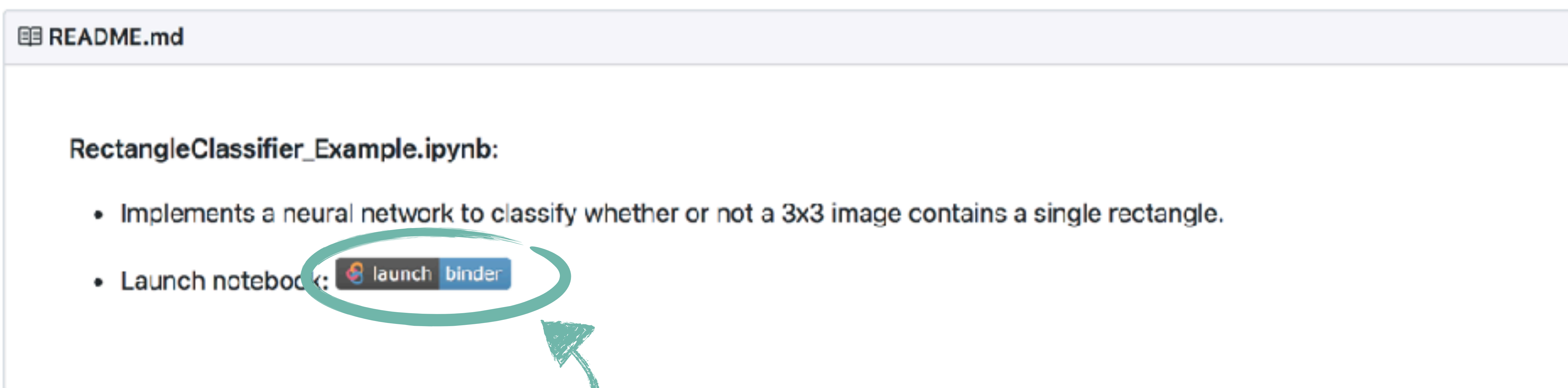
$$W^{(2)} = \begin{bmatrix} -3.3 \\ -3.6 \end{bmatrix}$$

$$b^{(2)} = [3.9]$$

Python code

Go to:

https://github.com/lhayward/RectangleClassifier


README.md

RectangleClassifier_Example.ipynb:

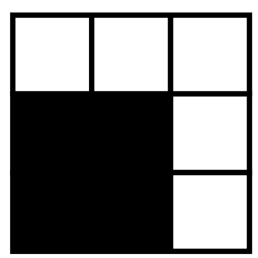
- Implements a neural network to classify whether or not a 3x3 image contains a single rectangle.
- Launch notebook: launch binder

Click here!

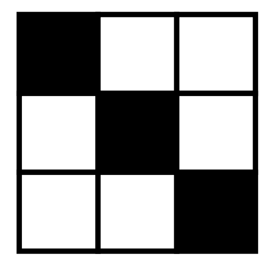
Python code

Exercises:

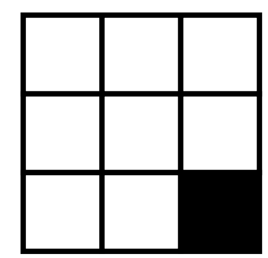
- Find the output from the neural network for the following inputs. In each case, does the output round to 0 or 1? Does the rounded output agree with what you would expect?



0.57



0.048



0.63

- Leave the weights and biases the same but change the input x. Can you find an input where the network gives the wrong prediction after rounding?

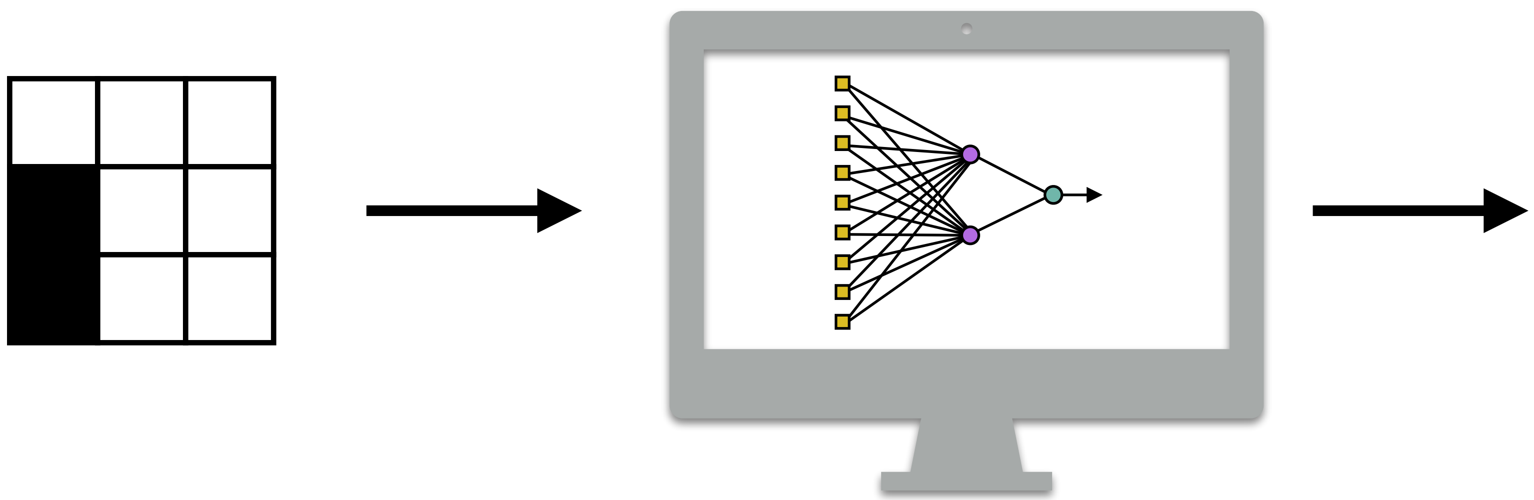
- Use the input x = np.array([0,0,0,1,0,0,1,0,0]). Adjust the weights and biases so that the network output is as close to 1 as possible.

https://github.com/lhayward/RectangleClassifier

Training neural networks

Our network might give the output:

$x =$



$y(x) = 0.89$

While the desired answer is: $y_{\text{true}}(x) = 1$

We would like to choose the weights W **and biases** b **such that the difference between** $y(x)$ **and** $y_{\text{true}}(x)$ **is as small as possible.**

Training neural networks

We measure how well our network is doing by calculating a cost function

$$C(W,b) = \sum_x \left[y(x) - y_{\text{true}}(x)\right]^2.$$

For a perfect classifier, $C(W,b) = 0.$

We use the cost function to modify the weights and biases:

$$W_{ij}^{(L)} \to W_{ij}^{(L)} - R \times \frac{\partial C}{\partial W_{ij}^{(L)}}$$

$$b_i^{(L)} \to b_i^{(L)} - R \times \frac{\partial C}{\partial b_i^{(L)}}$$

R: learning rate

Training neural networks

We measure how well our network is doing by calculating a cost function

$$C(W, b) = \sum_x \left[y(x) - y_{\text{true}}(x)\right]^2.$$

For a perfect classifier, $C(W, b) = 0.$

We use the cost function to modify the weights and biases:

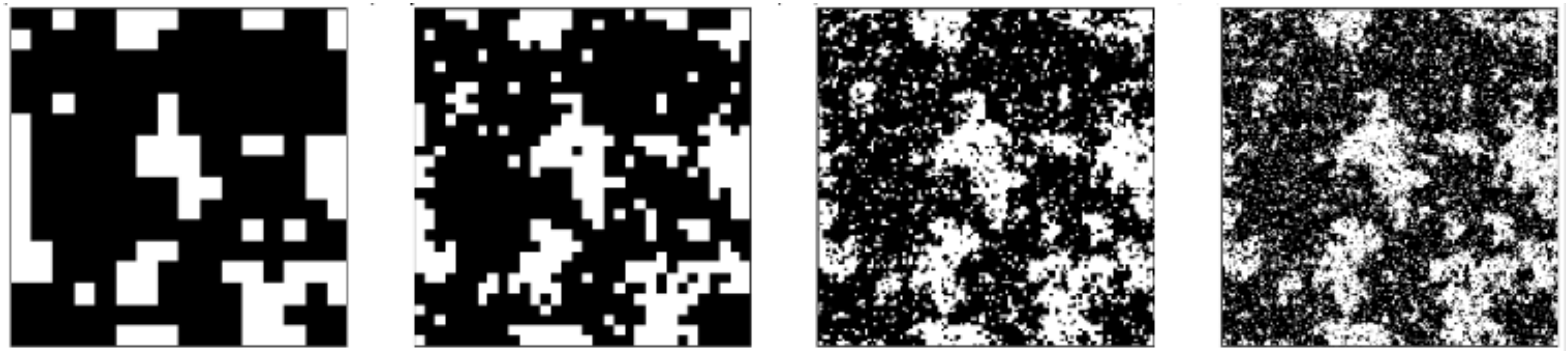
$$W_{ij}^{(L)} \to W_{ij}^{(L)} - R \times \frac{\partial C}{\partial W_{ij}^{(L)}}$$

Calculus!

$$b_i^{(L)} \to b_i^{(L)} - R \times \frac{\partial C}{\partial b_i^{(L)}}$$

Applications in physics

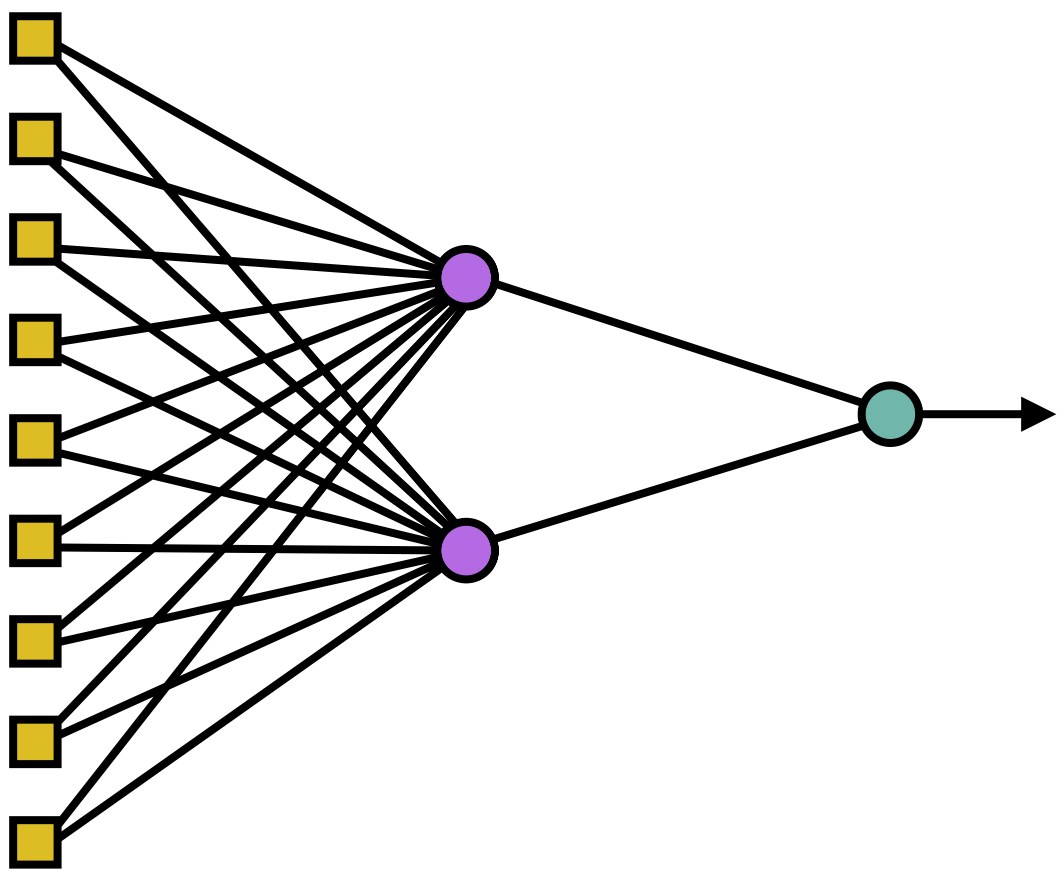
Many of the inputs are also images:



arXiv:1810.02372

Summary

- We have seen how artificial neural networks take in input and calculate output:



- For more examples with code, go to:
  https://www.tensorflow.org/tutorials