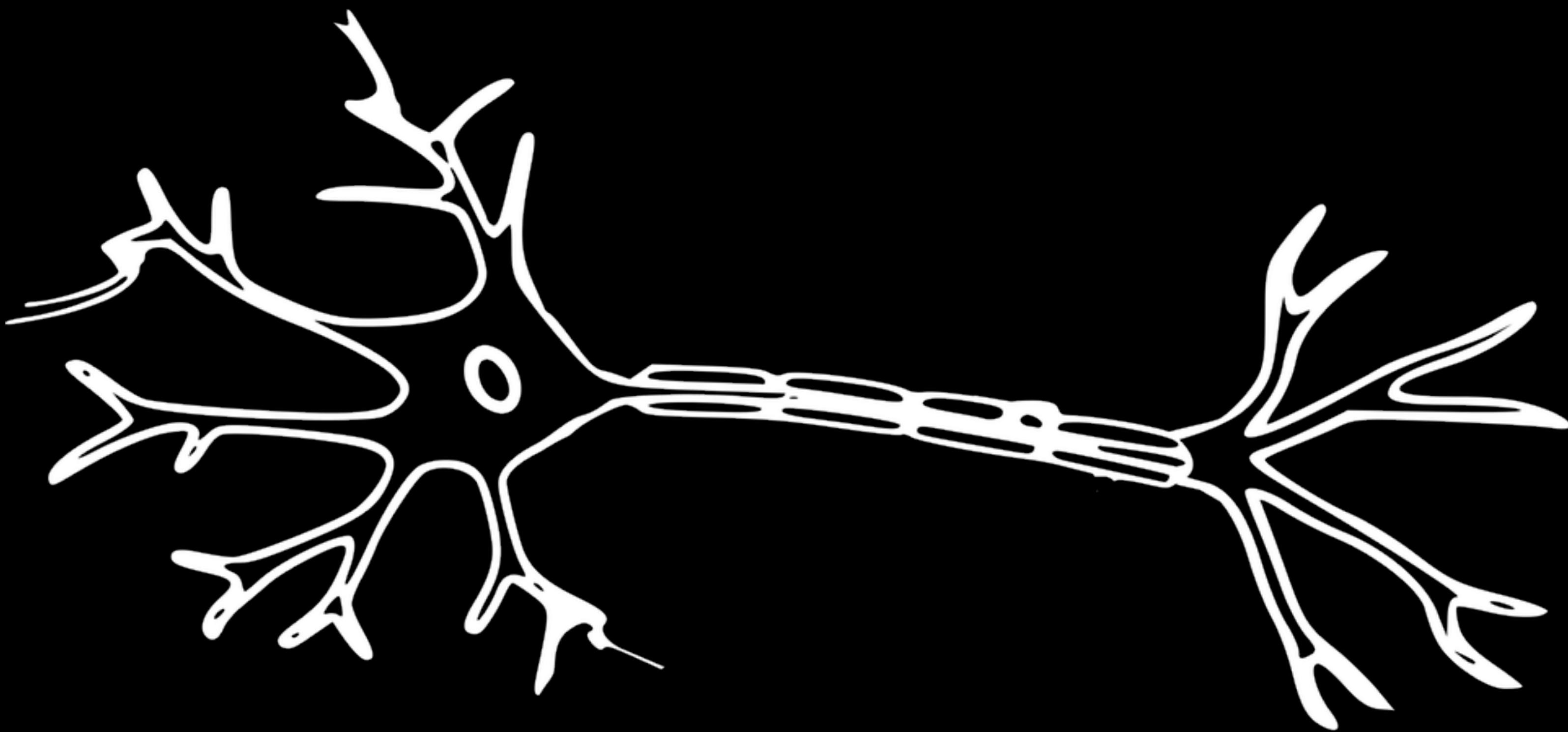


Introduction to
Artificial Intelligence
with Python

Neural Networks





Neural Networks

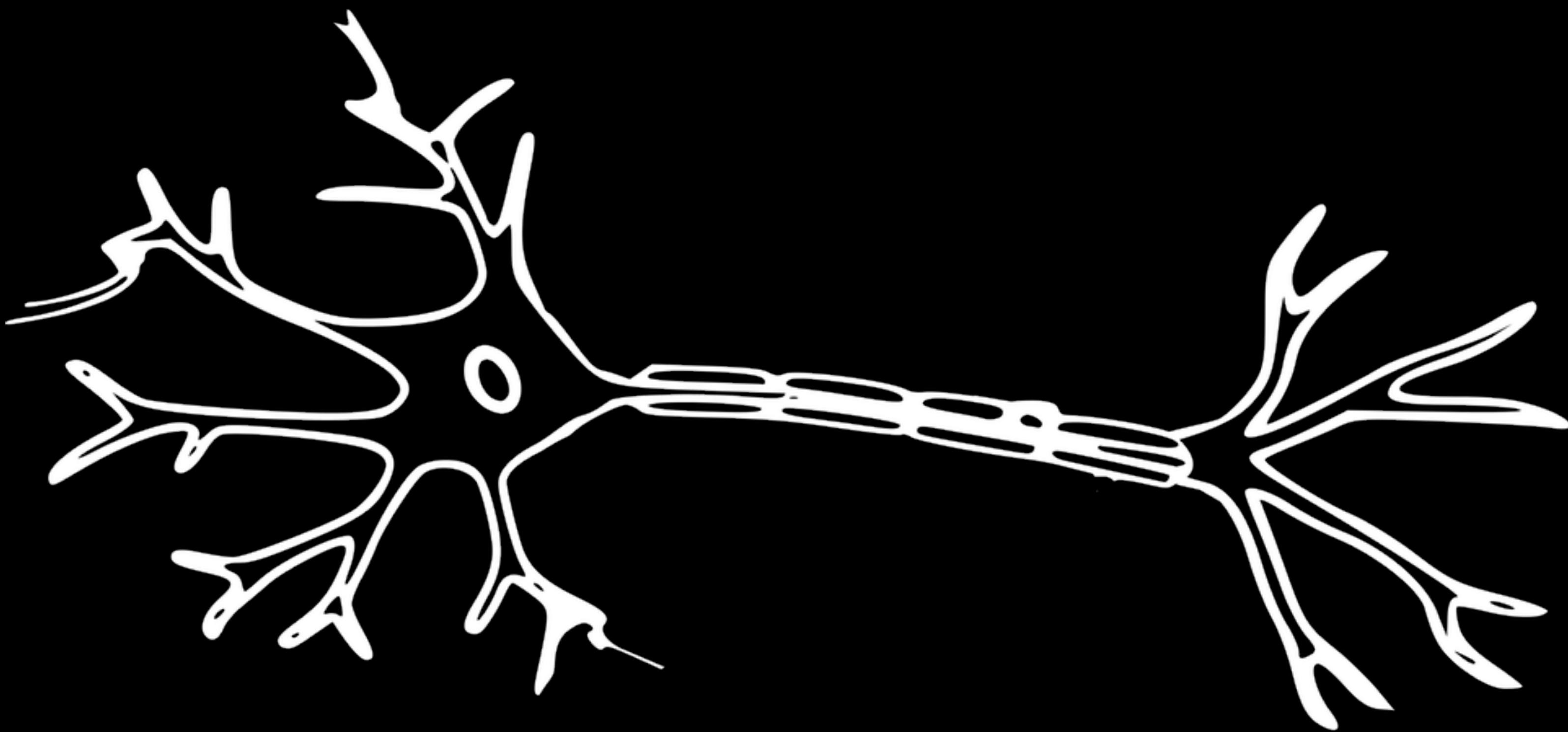
- Neurons are connected to and receive electrical signals from other neurons.
- Neurons process input signals and can be activated.

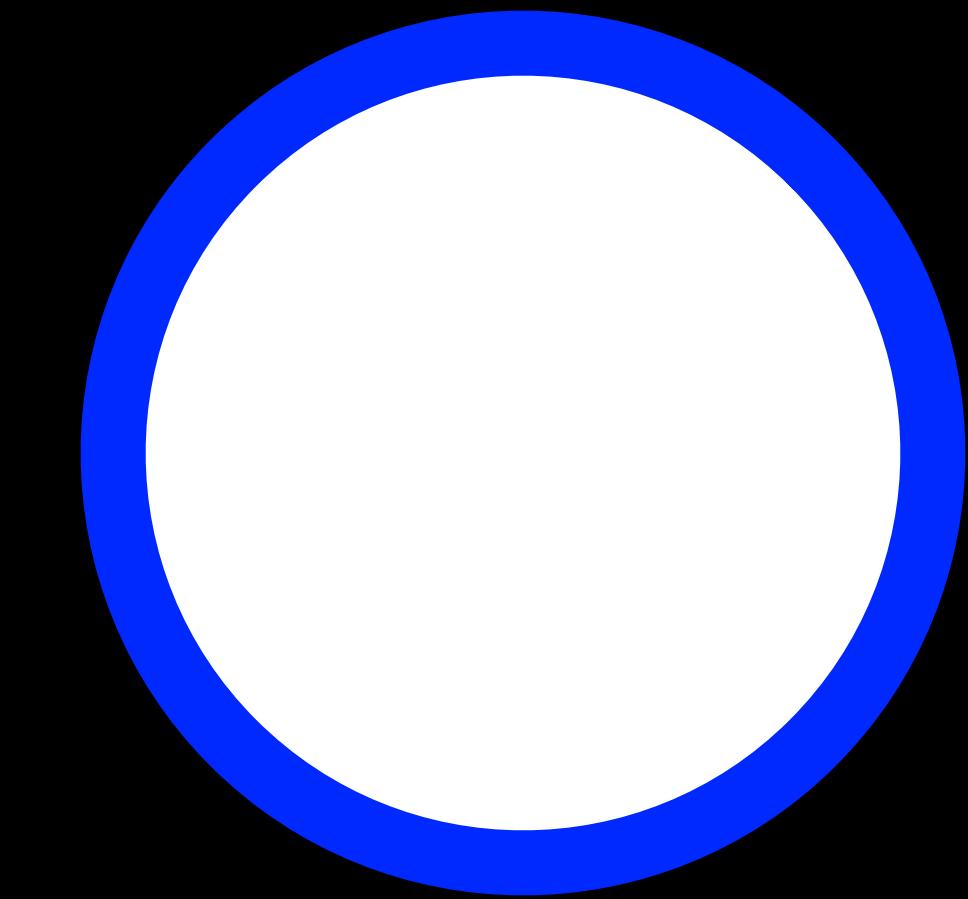
artificial neural network

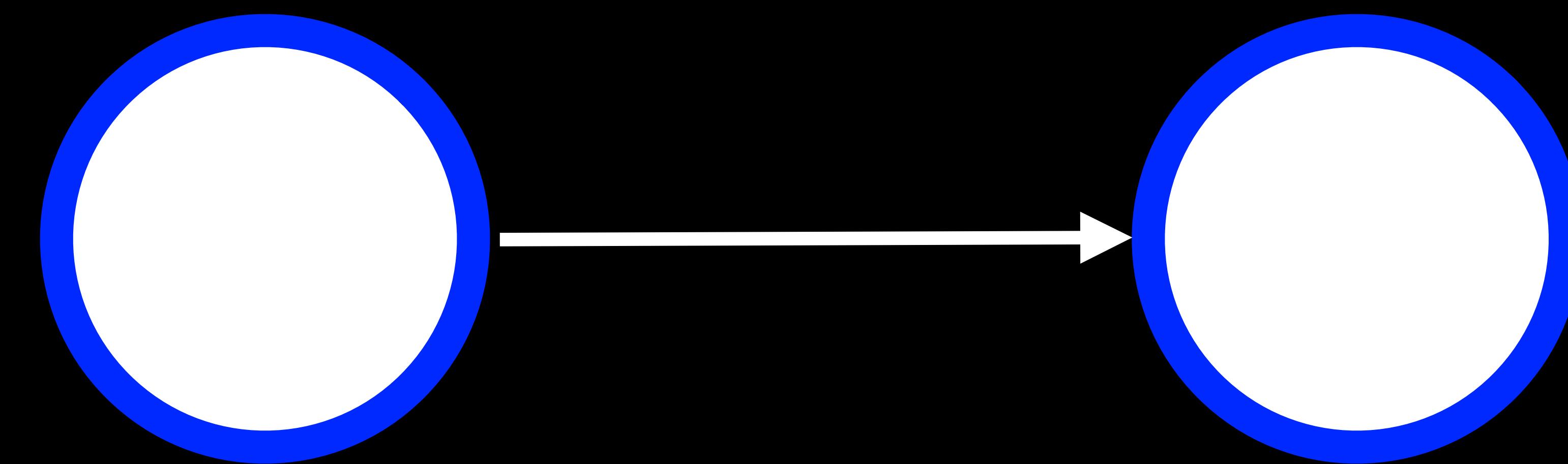
mathematical model for learning inspired by
biological neural networks

Artificial Neural Networks

- Model mathematical function from inputs to outputs based on the structure and parameters of the network.
- Allows for learning the network's parameters based on data.







(x_1, x_2) 

$$h(x_1, x_2) = w_0 + w_1x_1 + w_2x_2$$

$$h(x_1, x_2) = w_0 + w_1x_1 + w_2x_2$$

$$h(x_1, x_2) = w_0 + w_1x_1 + w_2x_2$$

step function

$$g(x) = 1 \text{ if } x \geq 0, \text{ else } 0$$

output

0

$\mathbf{w} \cdot \mathbf{x}$

step function

$$g(x) = 1 \text{ if } x \geq 0, \text{ else } 0$$

output

0

$\mathbf{w} \cdot \mathbf{x}$

logistic sigmoid

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

output

1

0

$\mathbf{w} \cdot \mathbf{x}$

rectified linear unit (ReLU)

$$g(x) = \max(0, x)$$

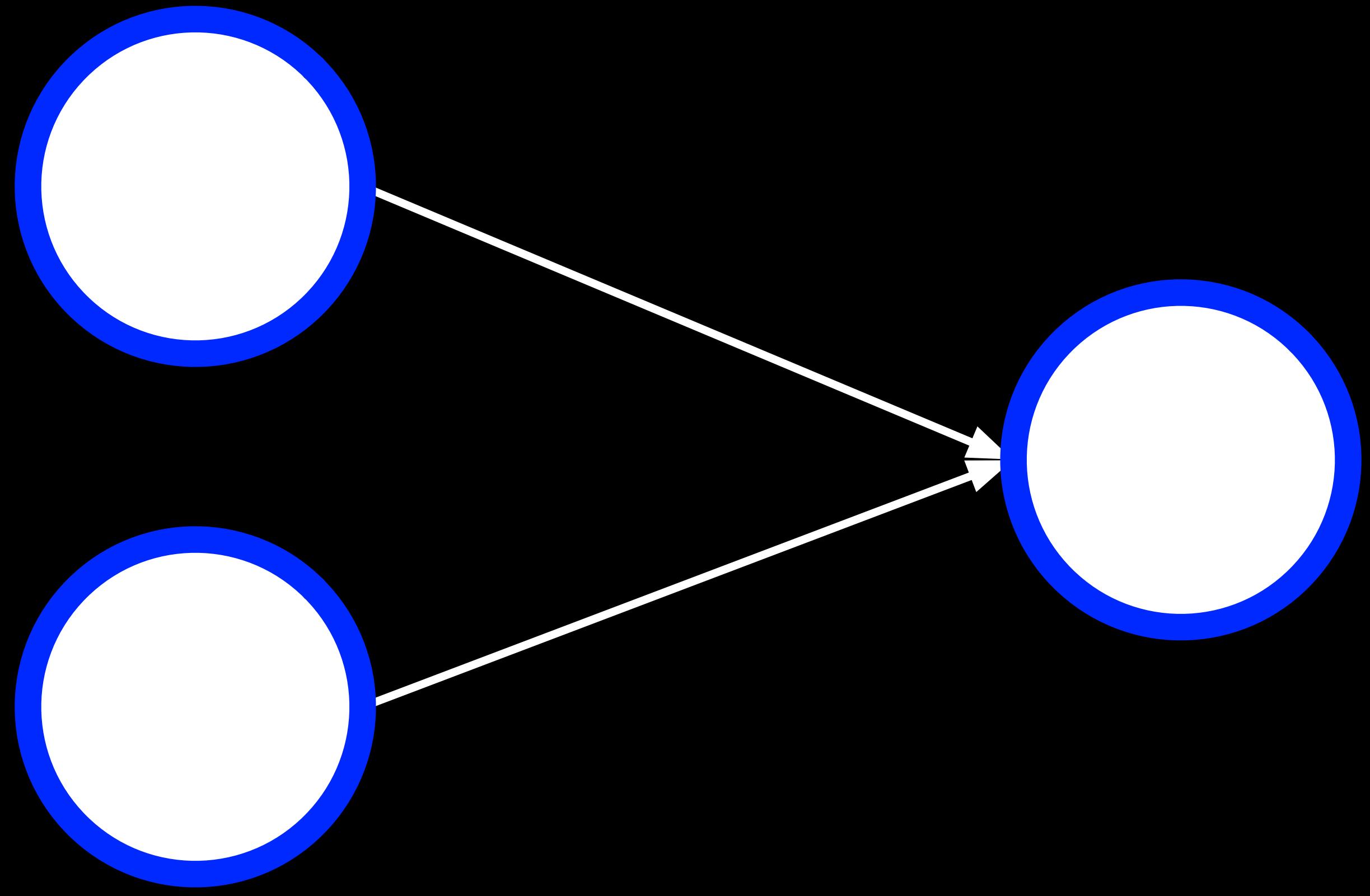
output

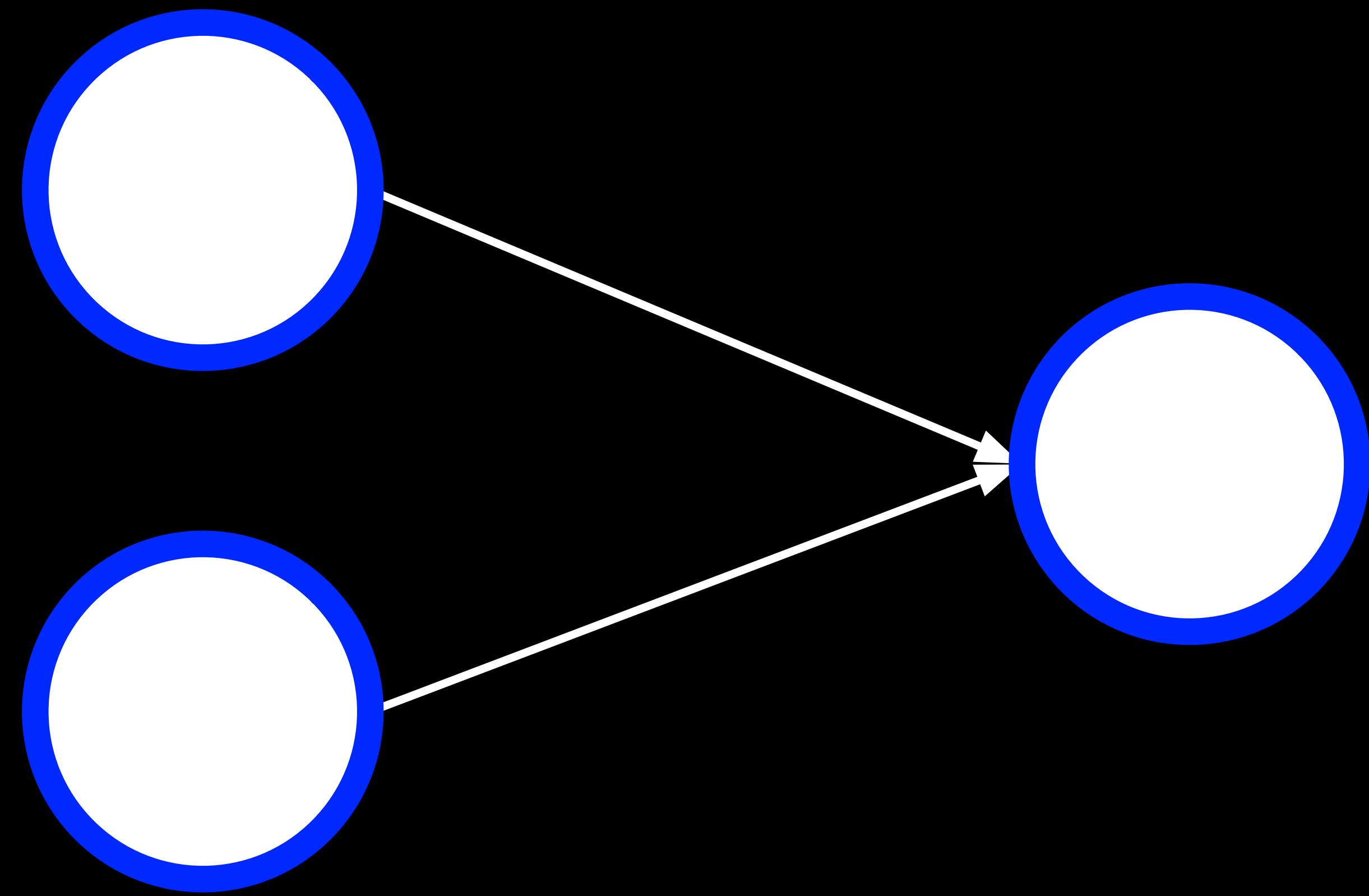
0

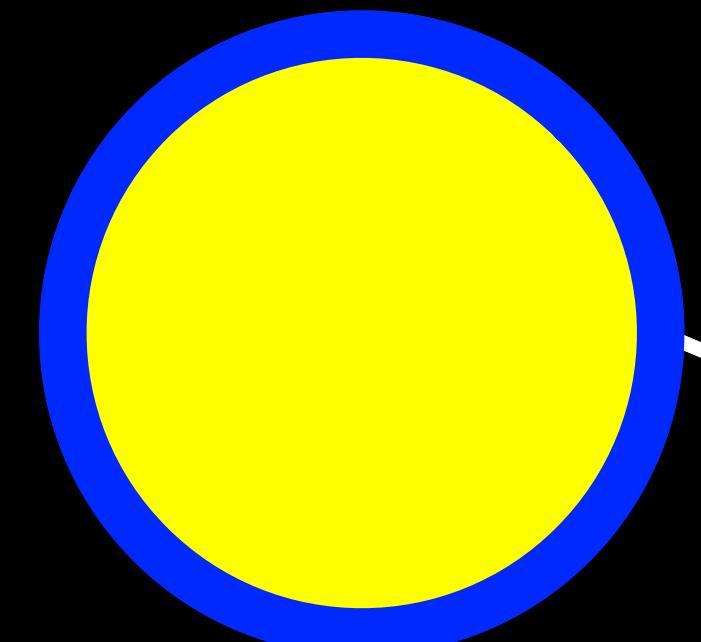
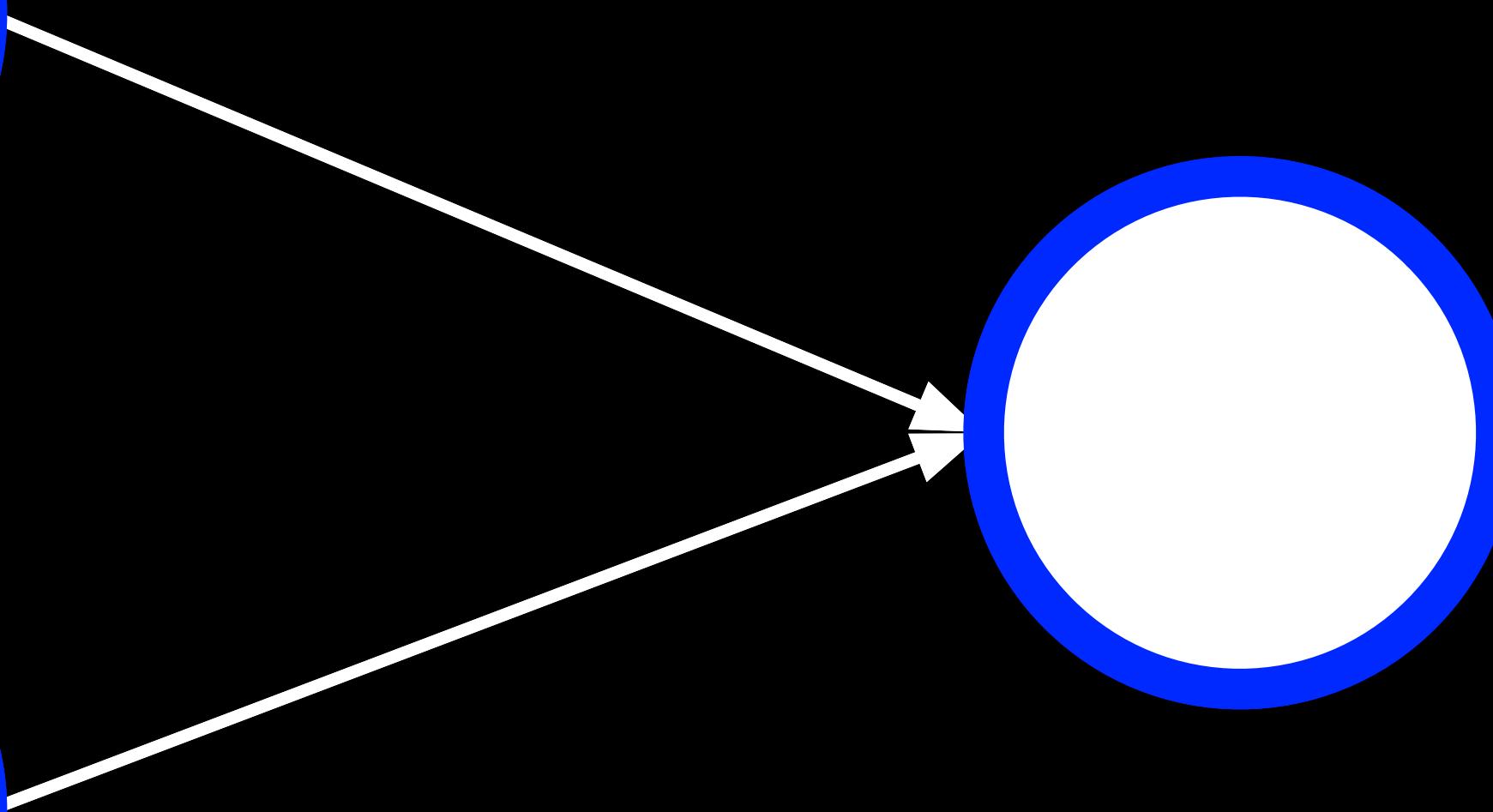
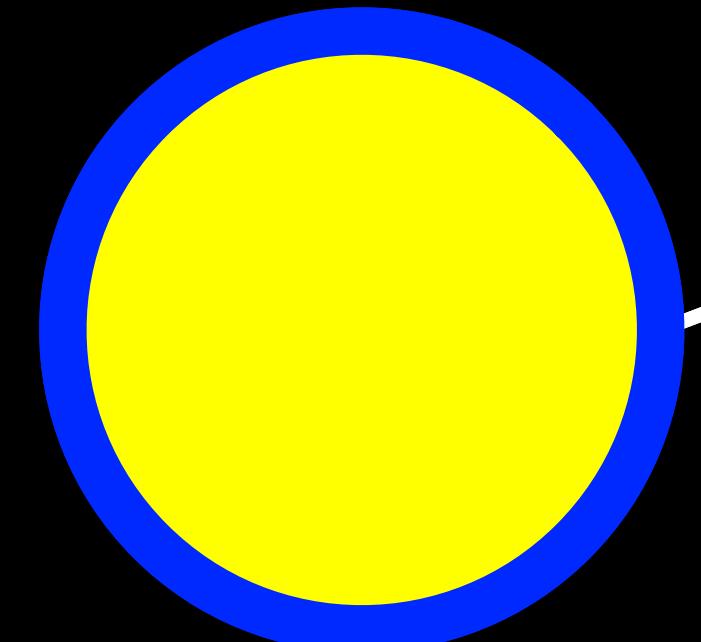
$\mathbf{w} \cdot \mathbf{x}$

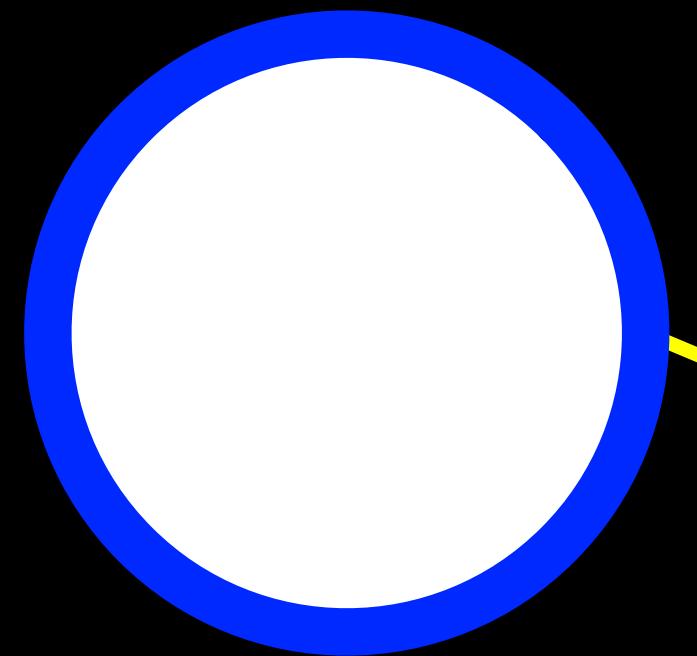
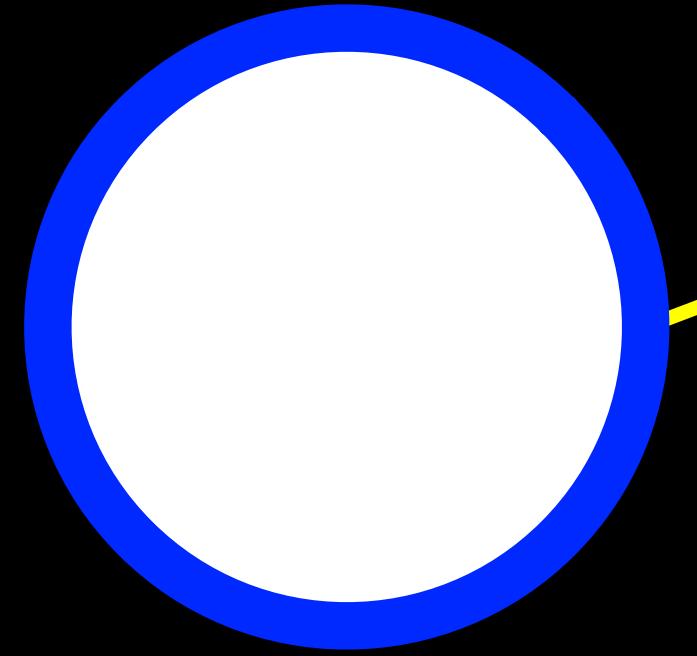
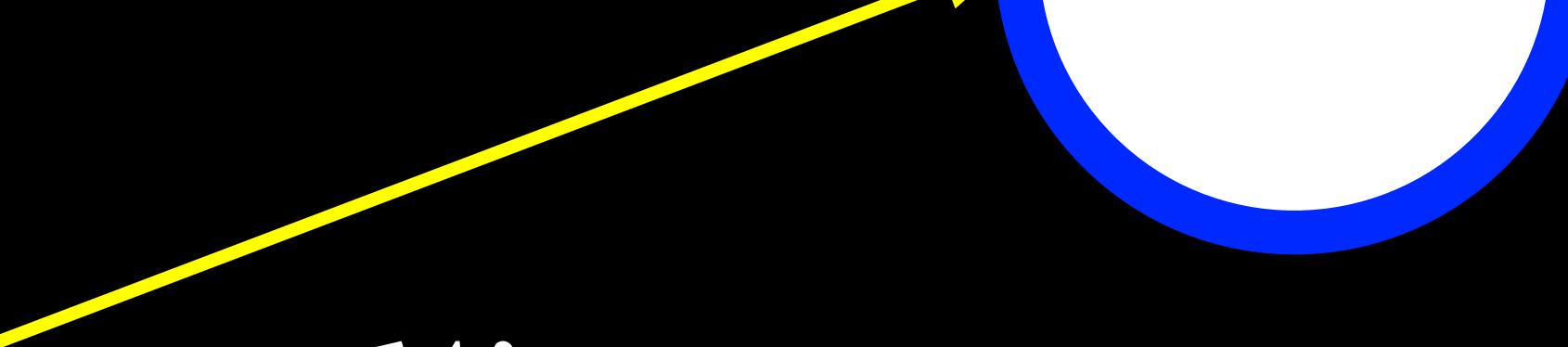
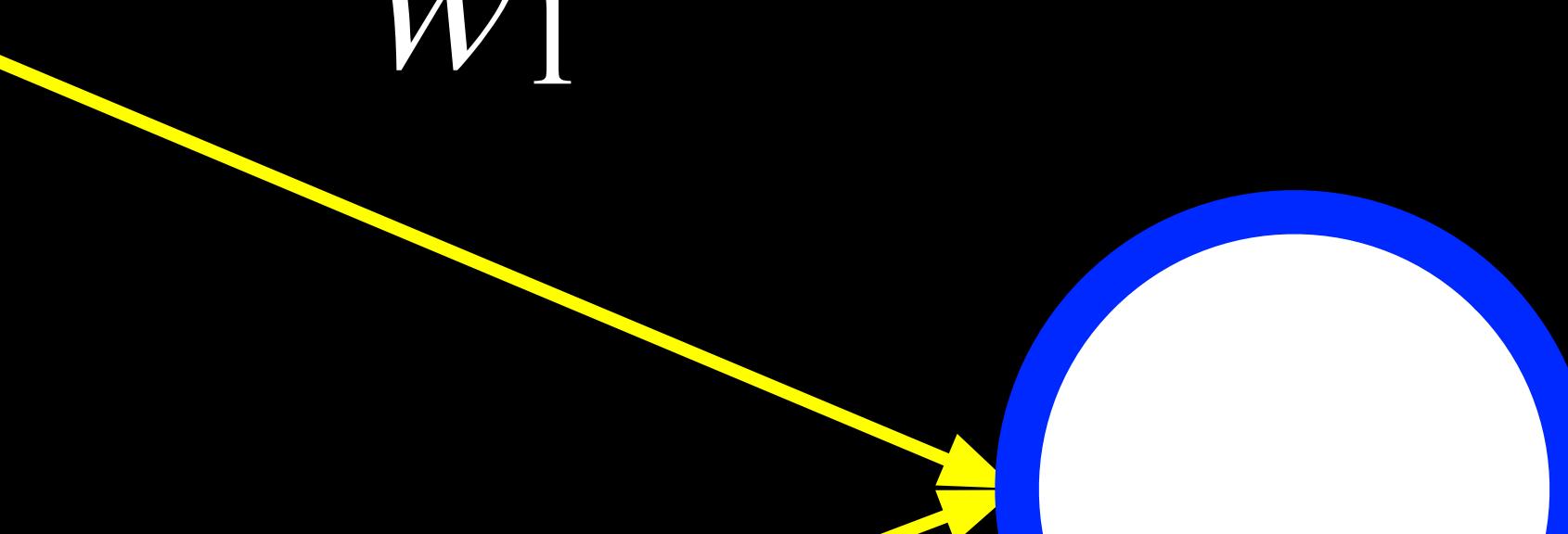
$$h(x_1, x_2) = w_0 + w_1x_1 + w_2x_2$$

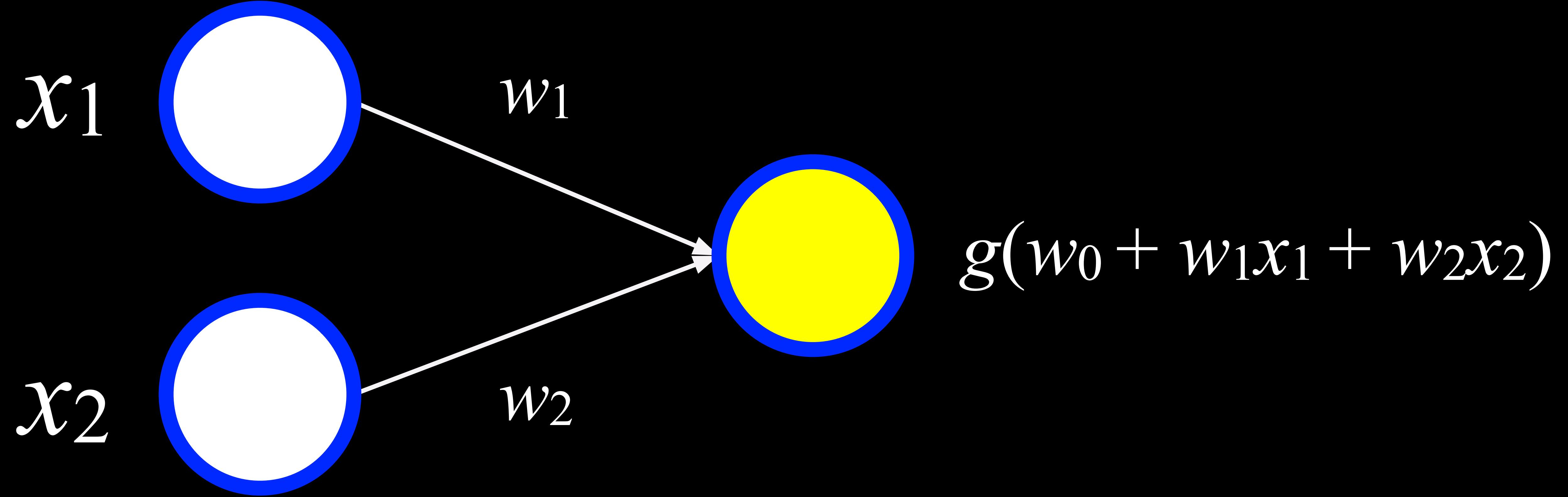
$$h(x_1,x_2)=g(w_0+w_1x_1+w_2x_2)$$

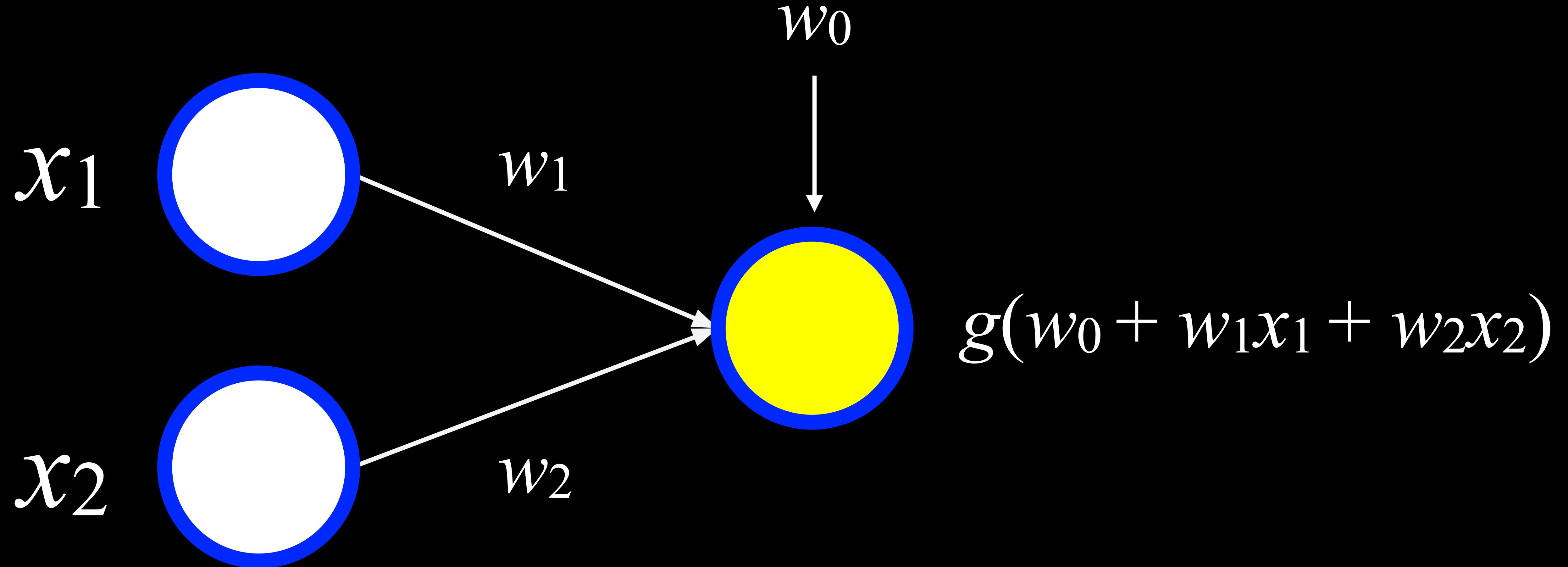


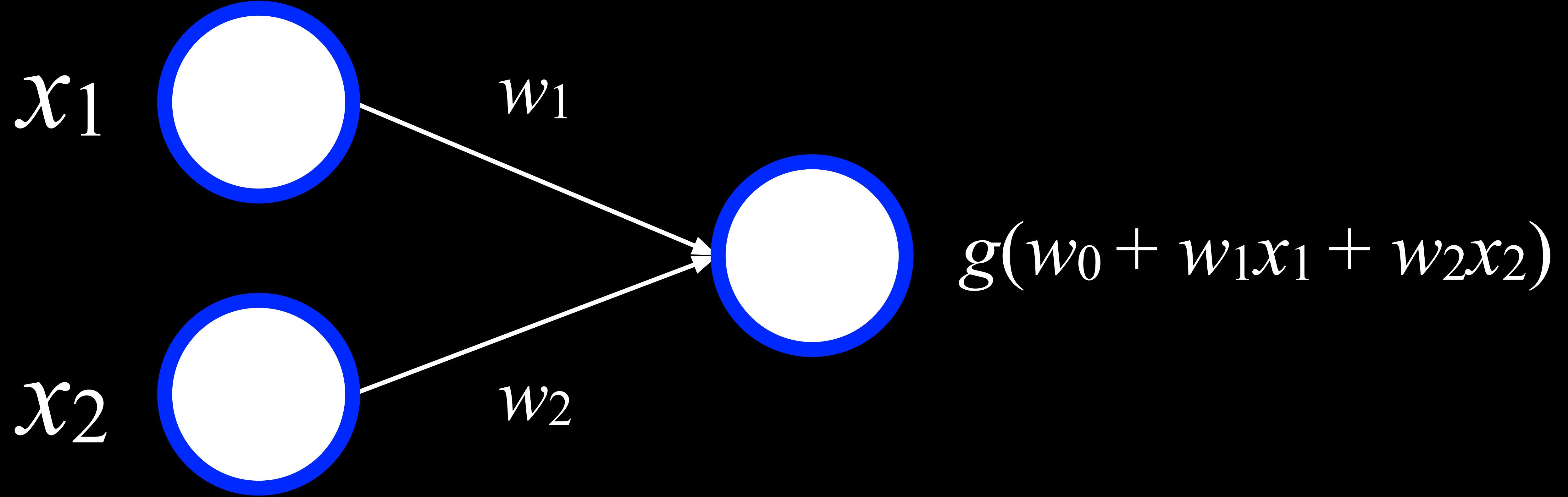


x_1  x_2 

x_1  w_1 x_2  w_2 

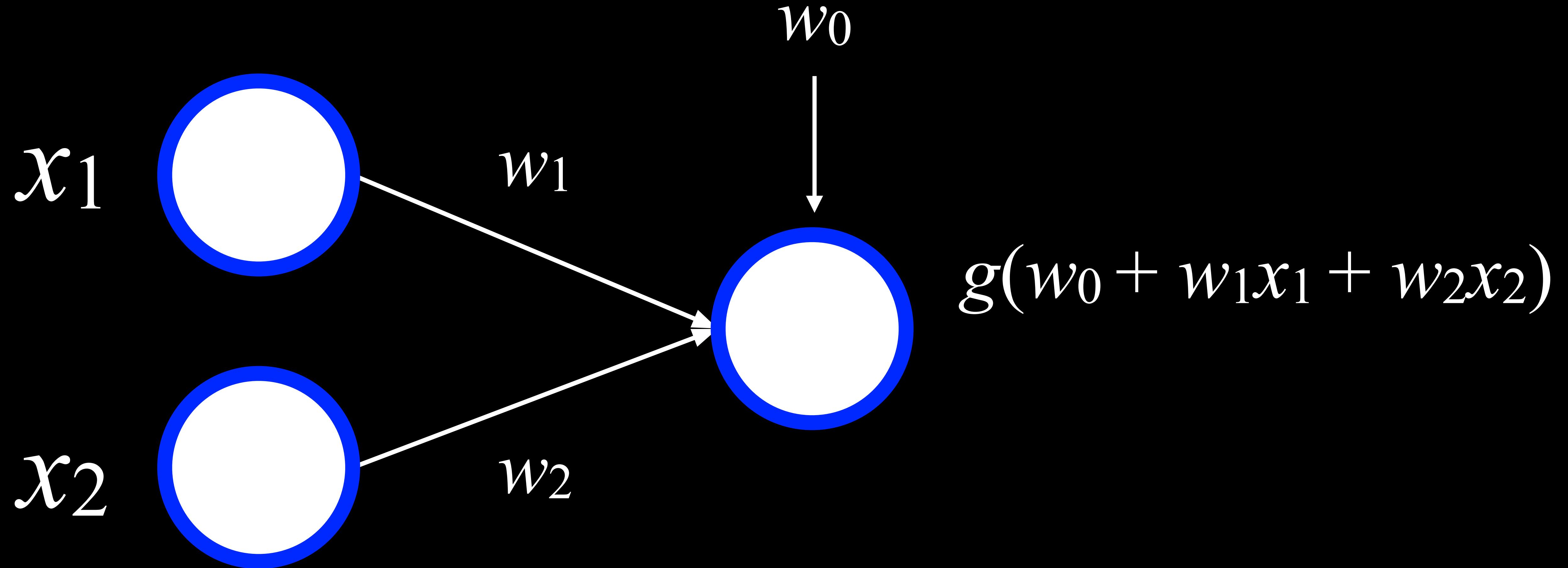


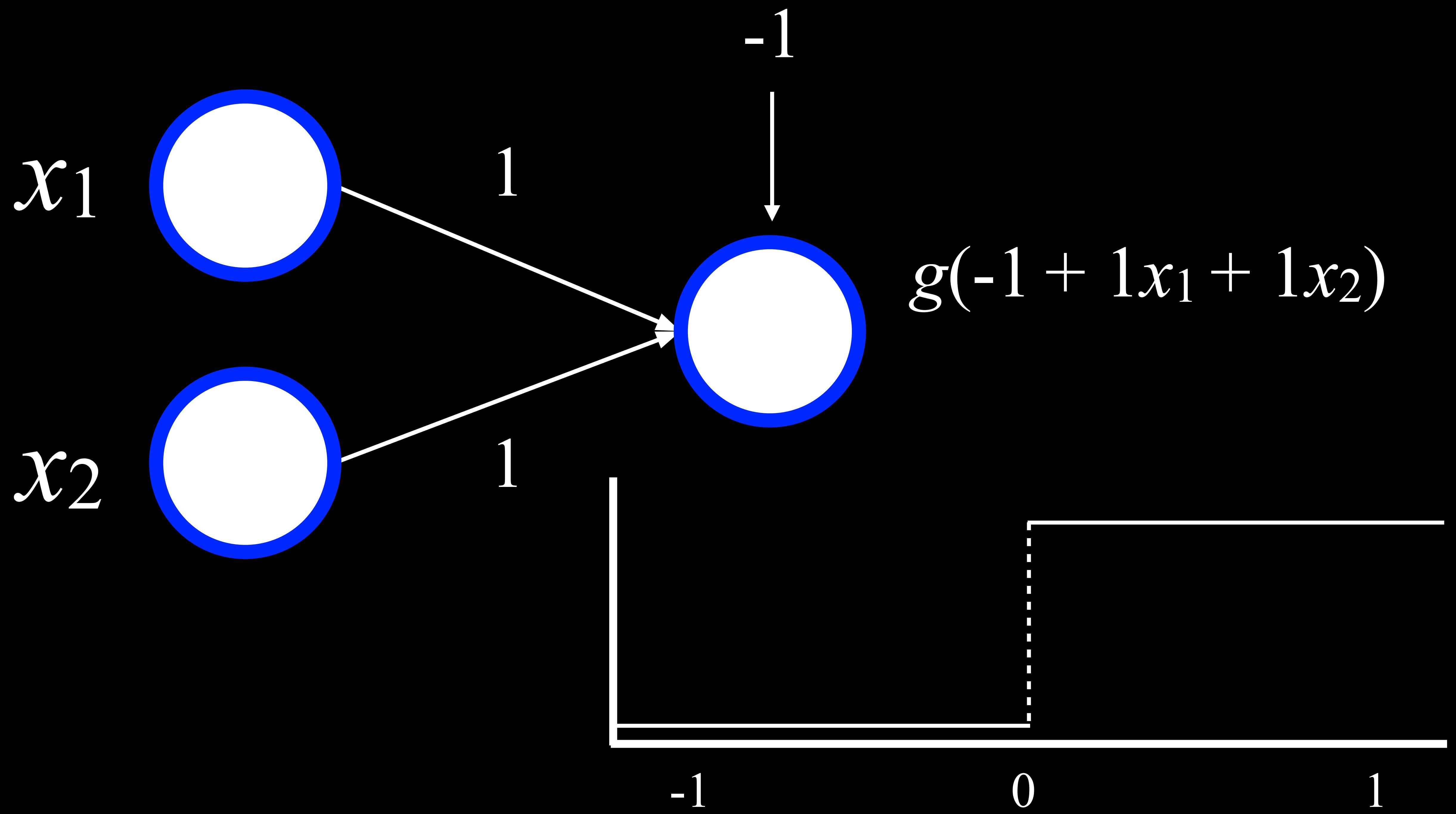


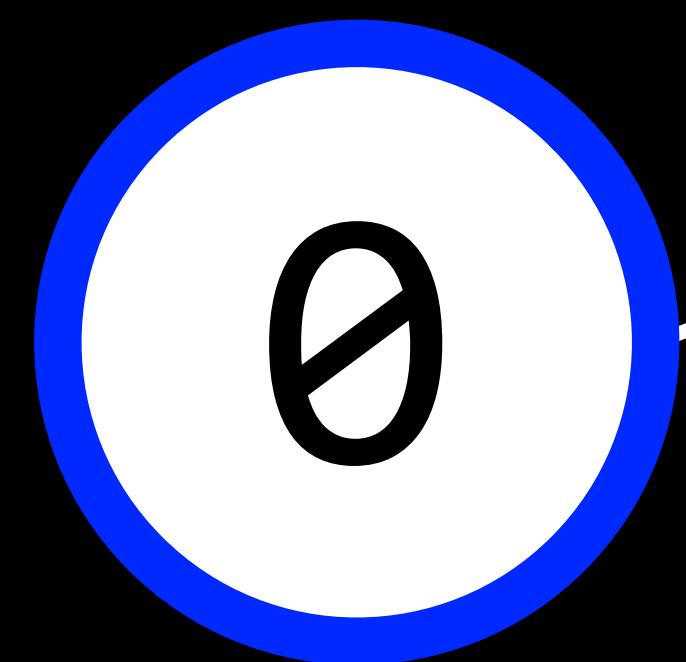
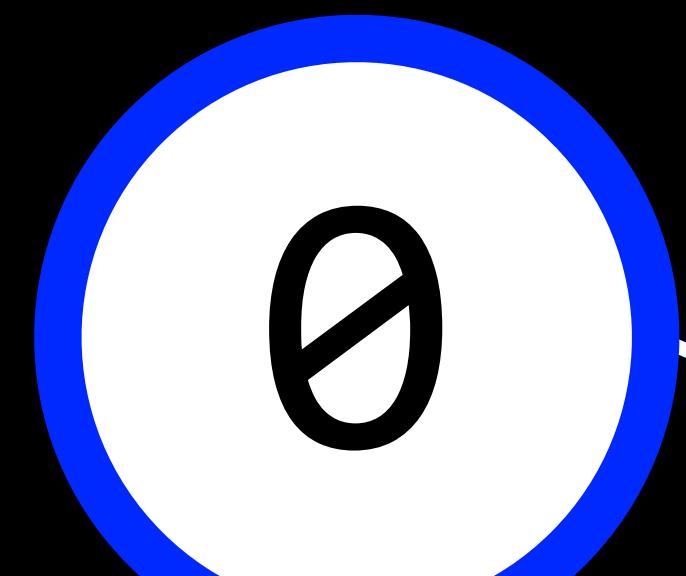


Or

x	y	$f(x, y)$
0	0	0
0	1	1
1	0	1
1	1	1





x_1 x_2 

1

1

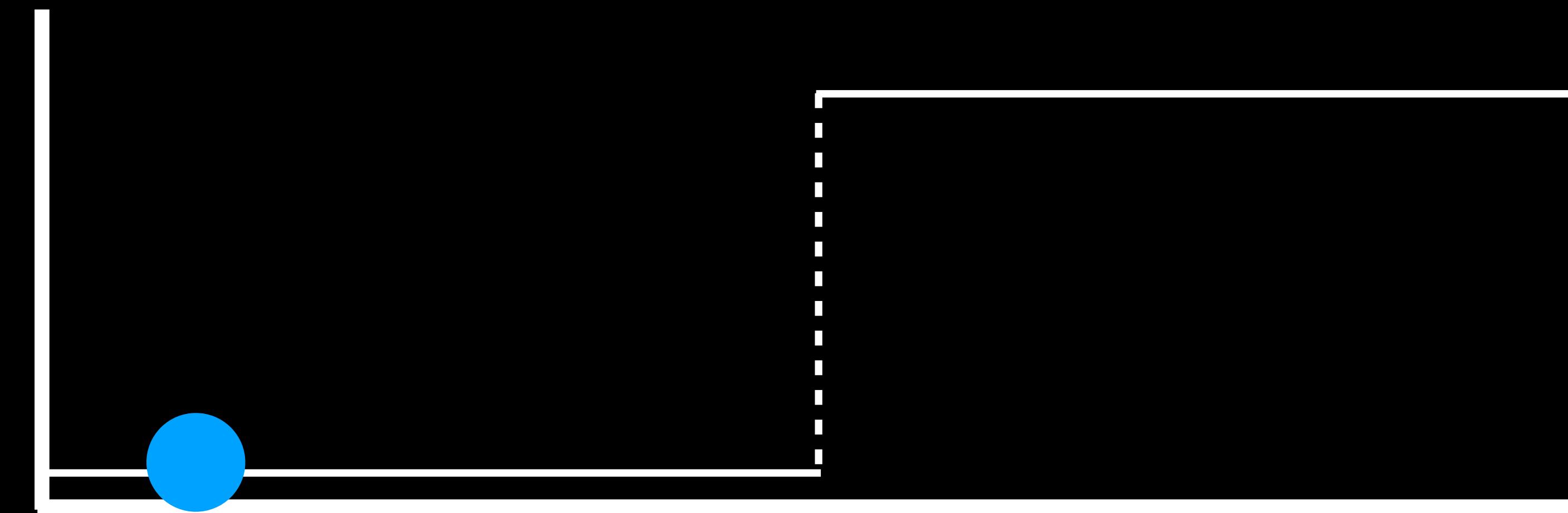
-1

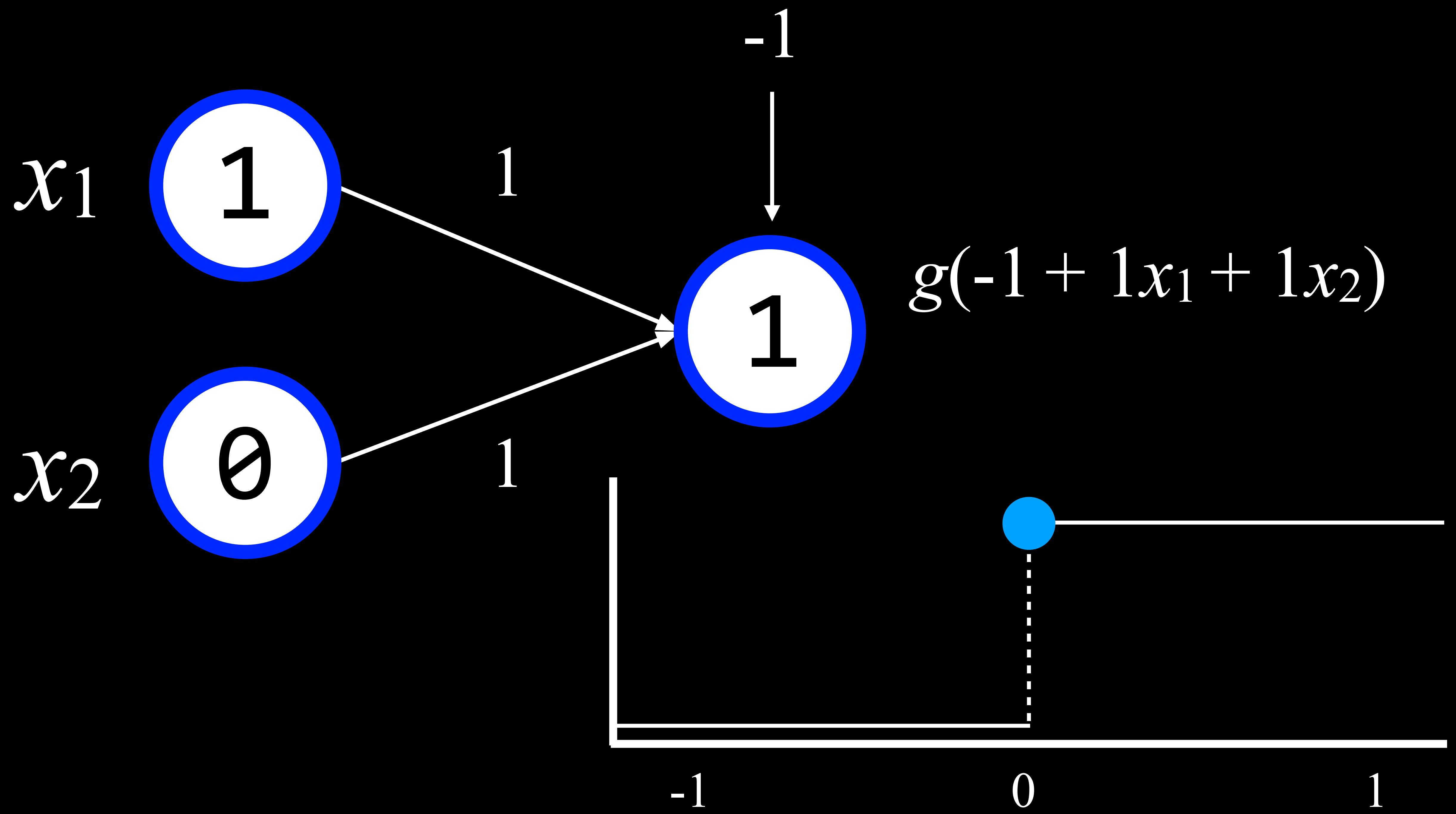
0

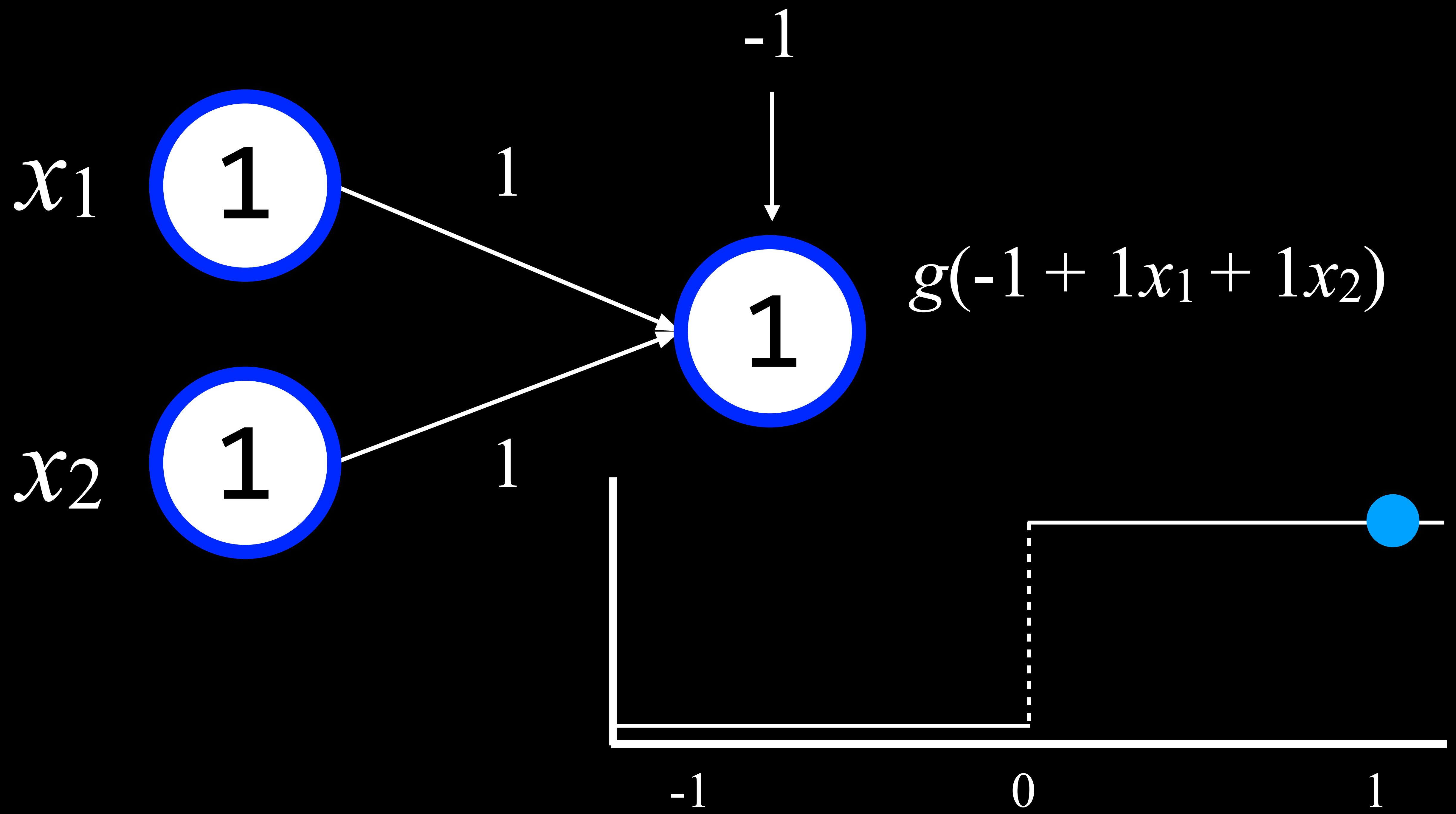
1

-1

$$g(-1 + 1x_1 + 1x_2)$$

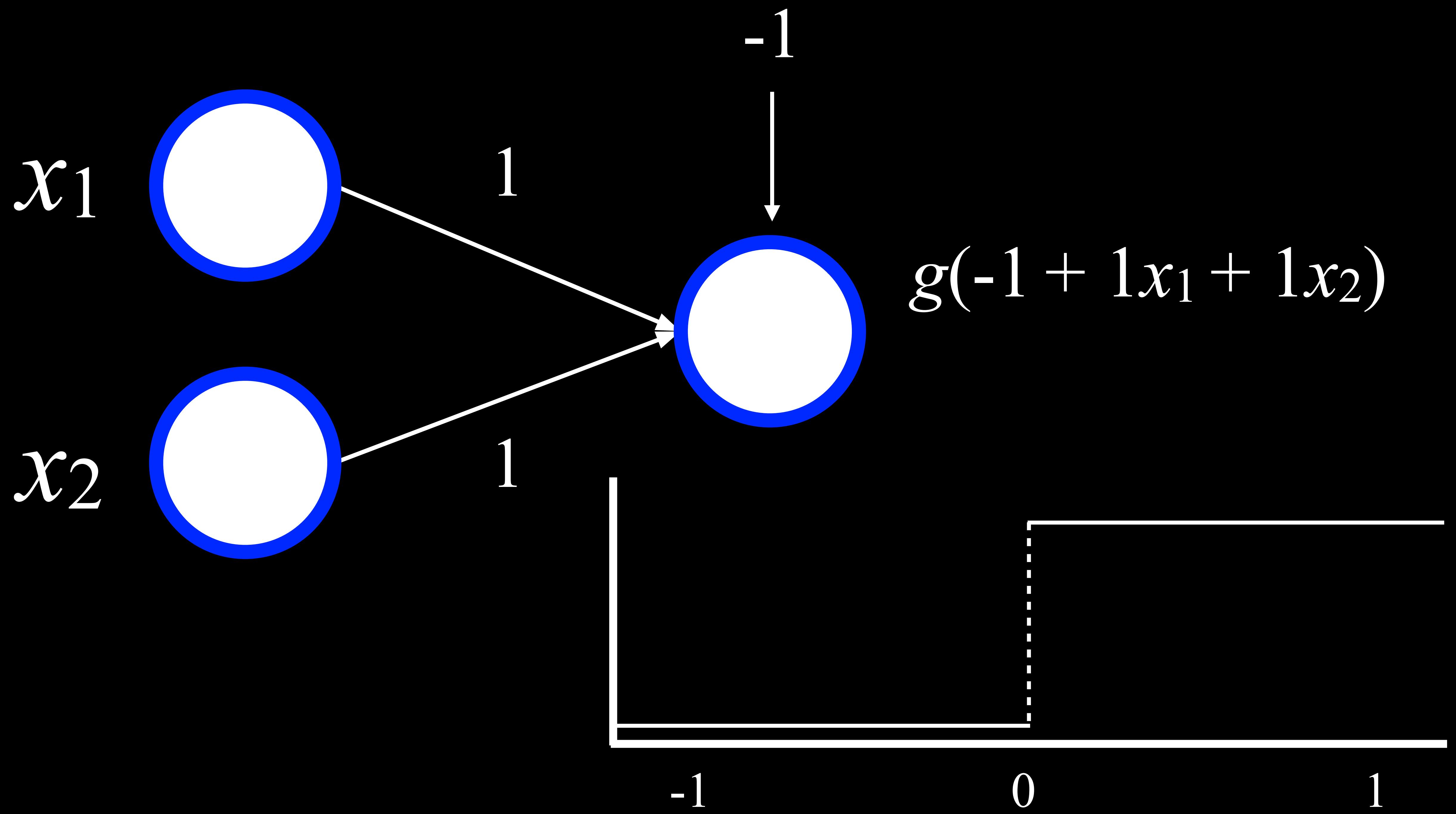


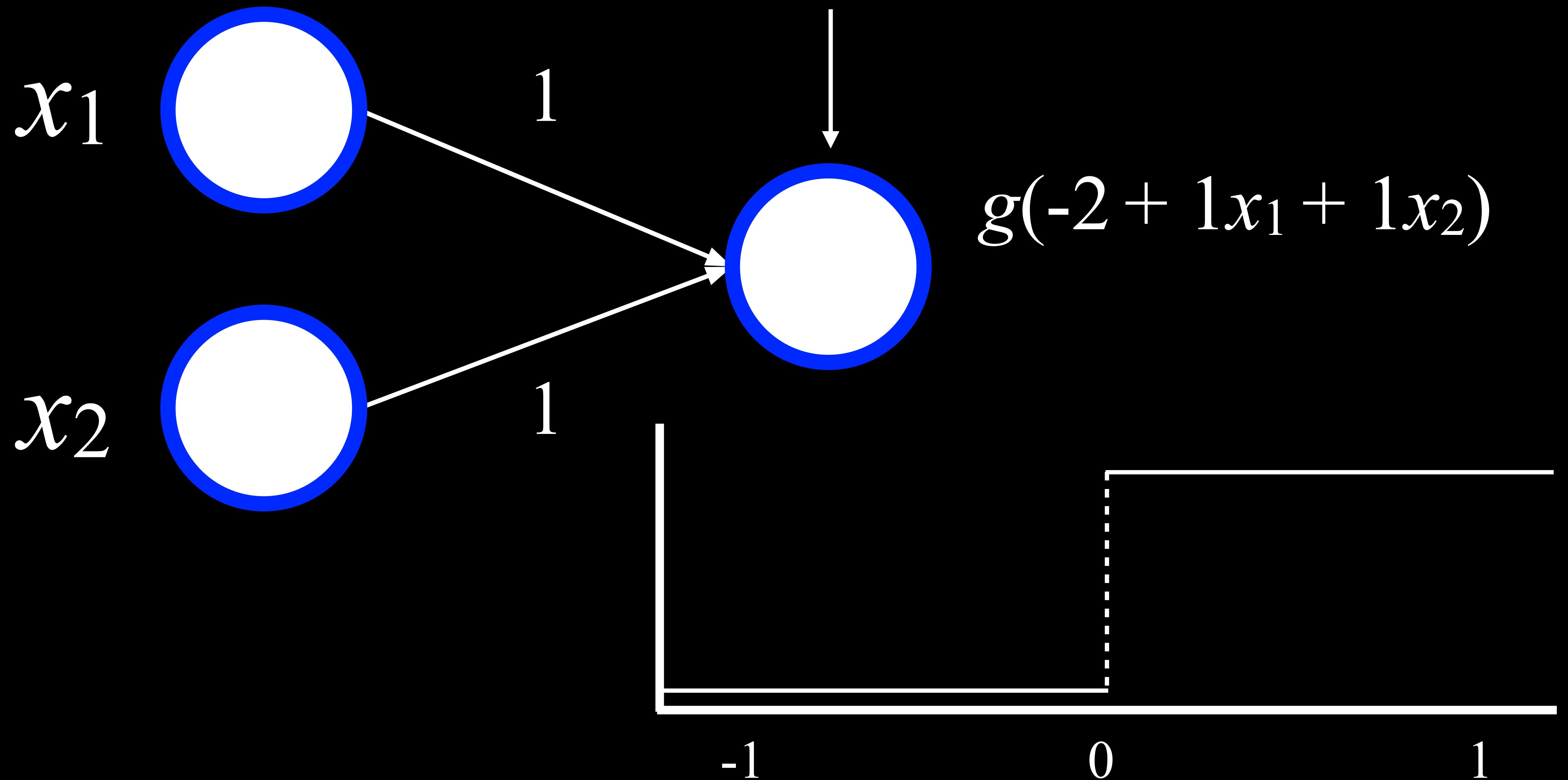


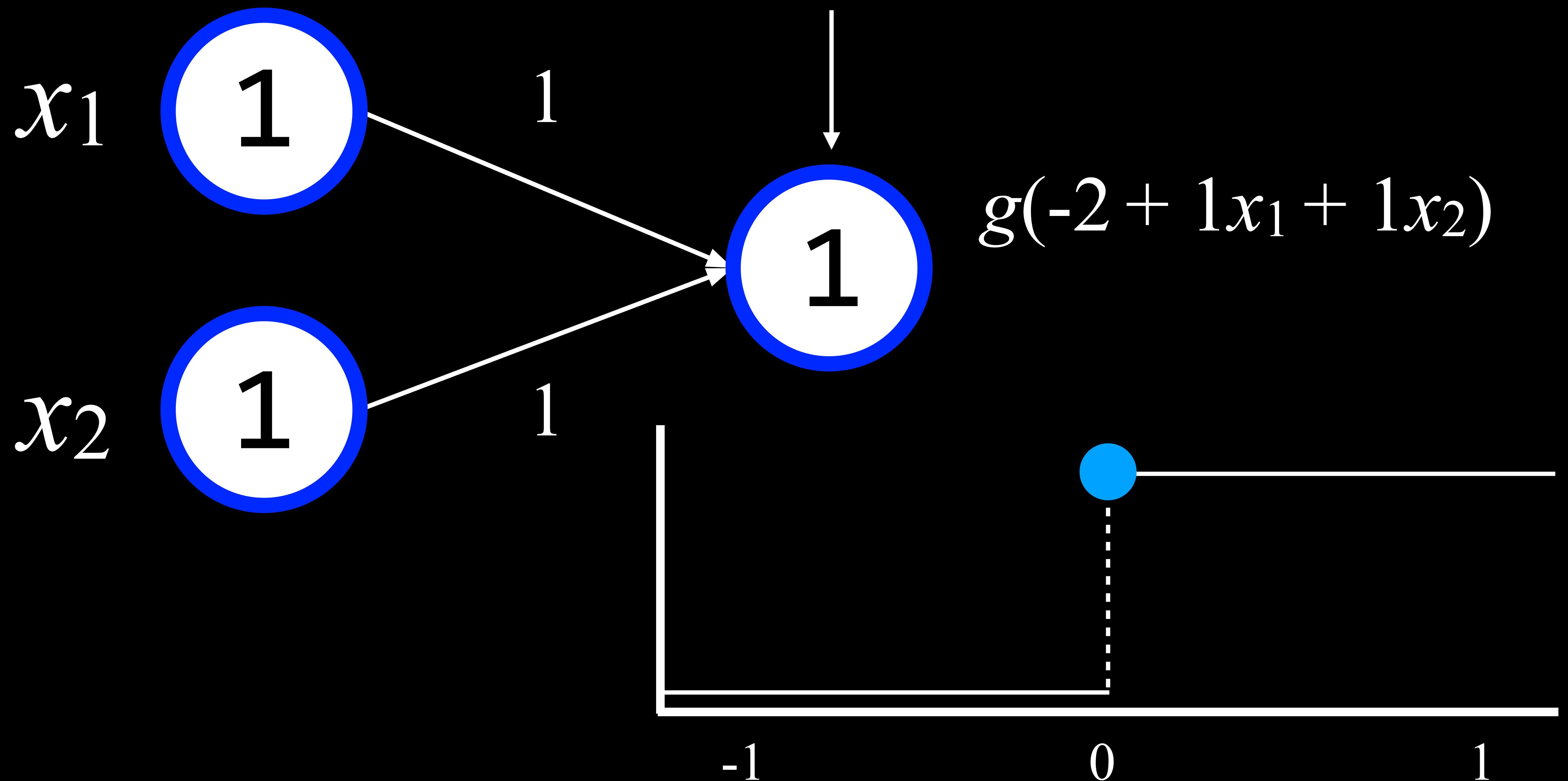


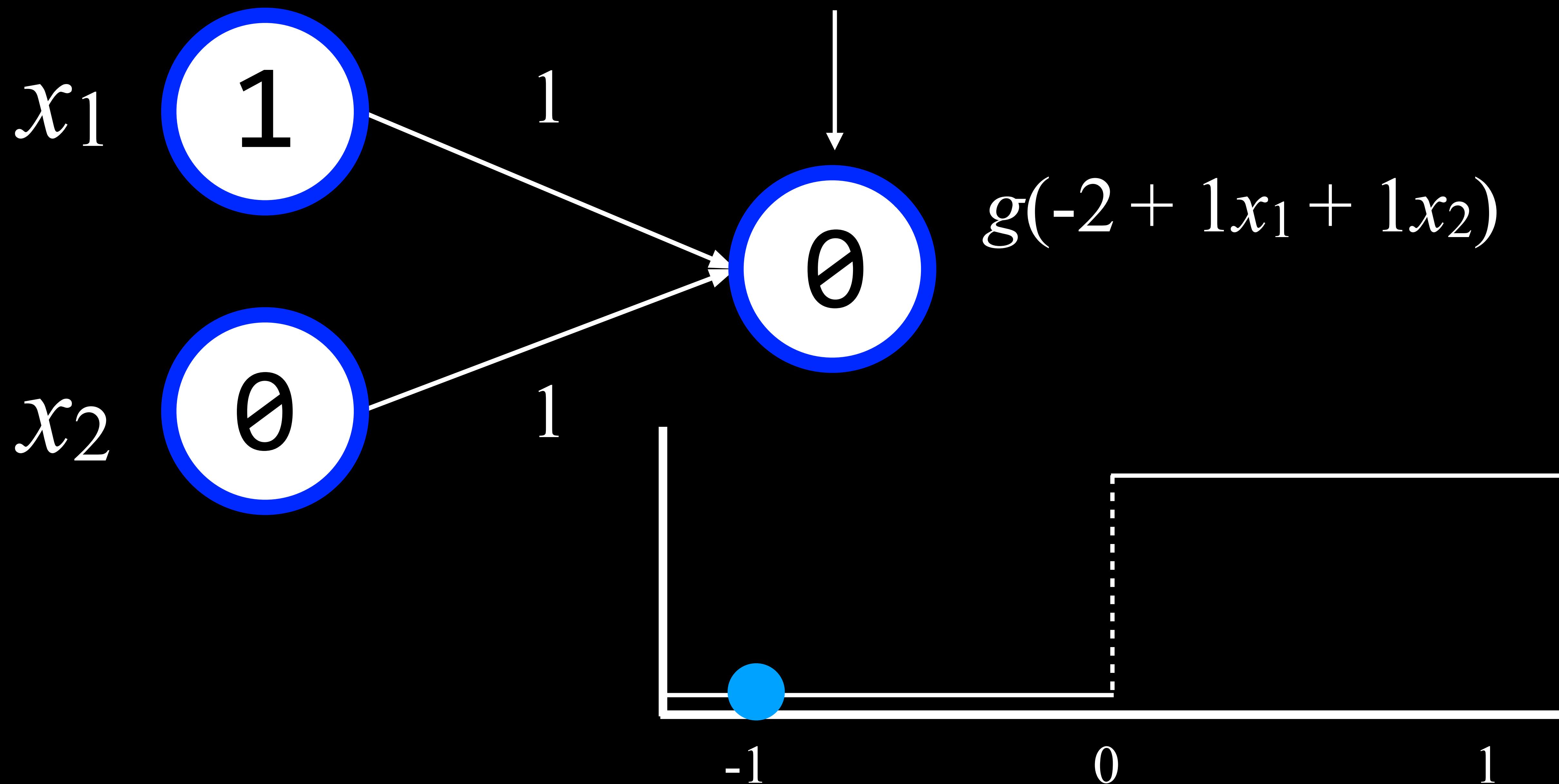
And

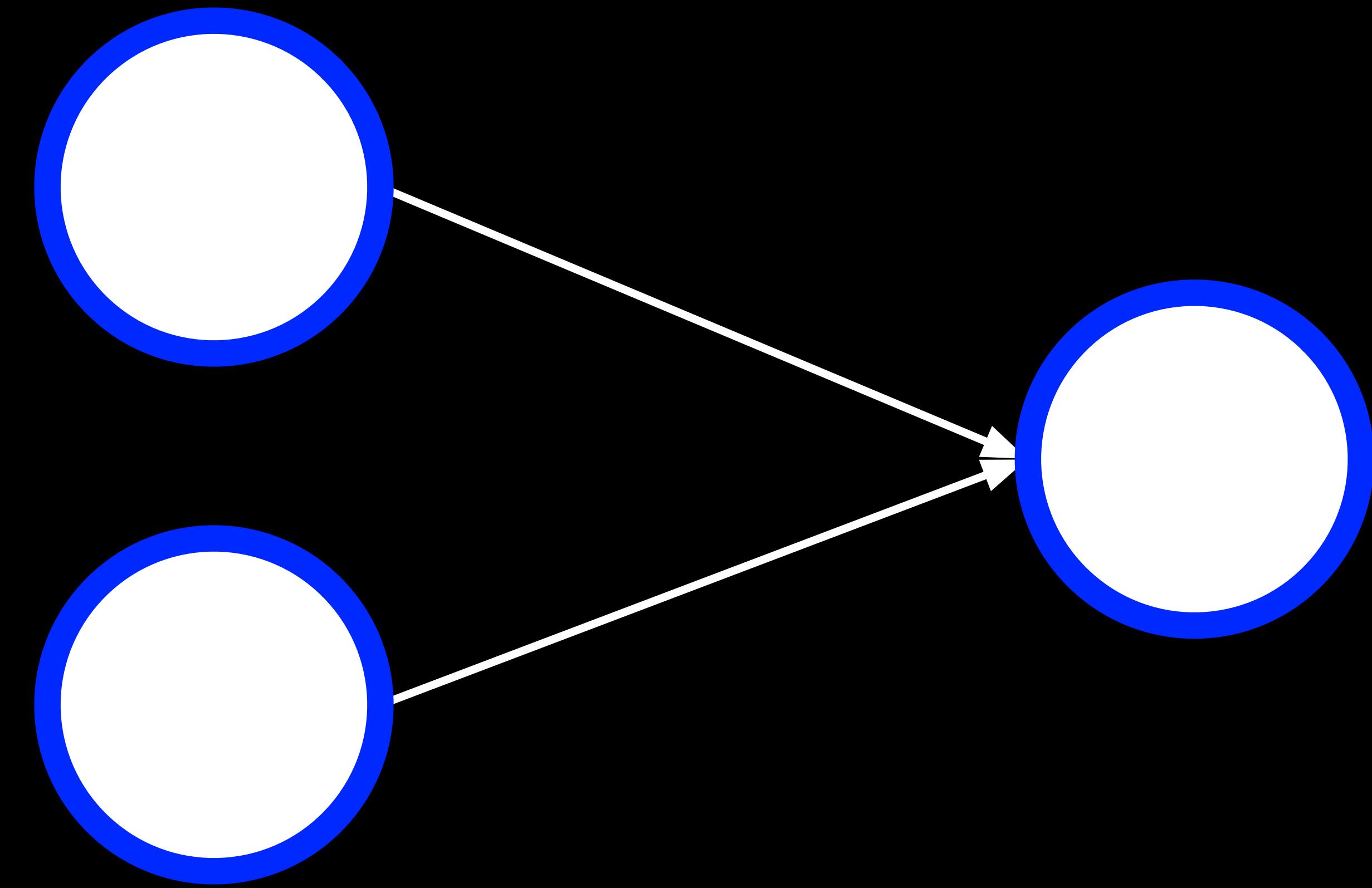
x	y	$f(x, y)$
0	0	0
0	1	0
1	0	0
1	1	1



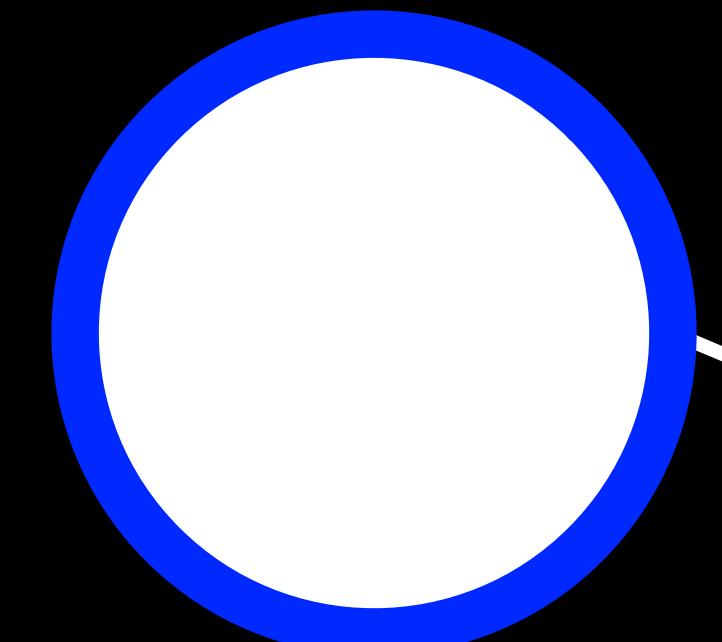




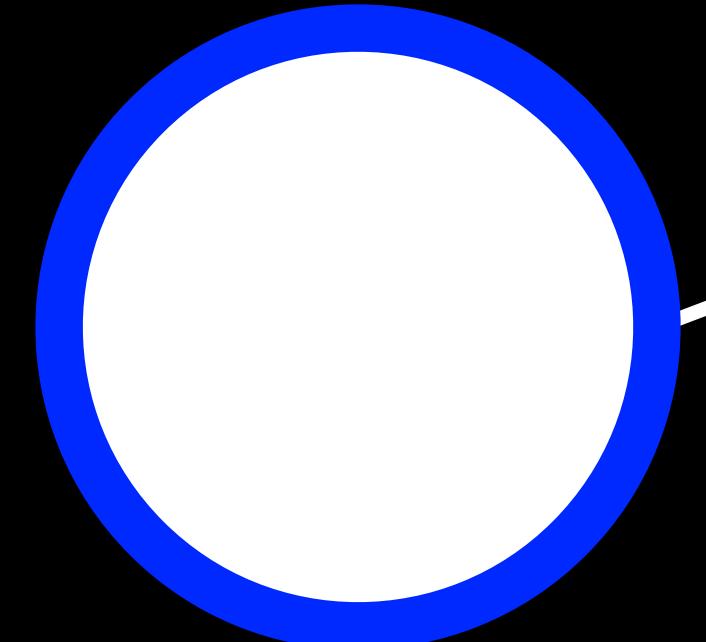




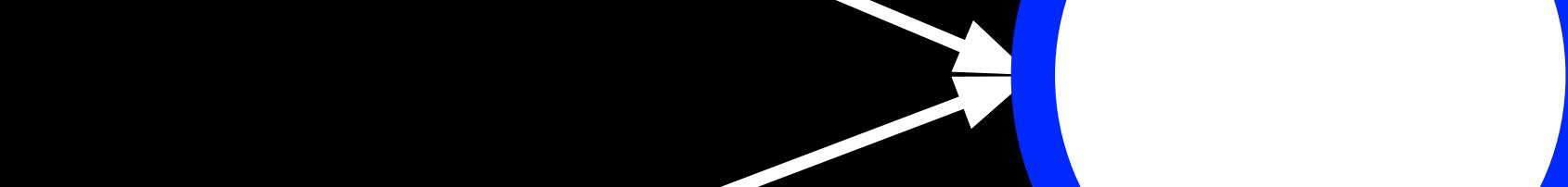
humidity



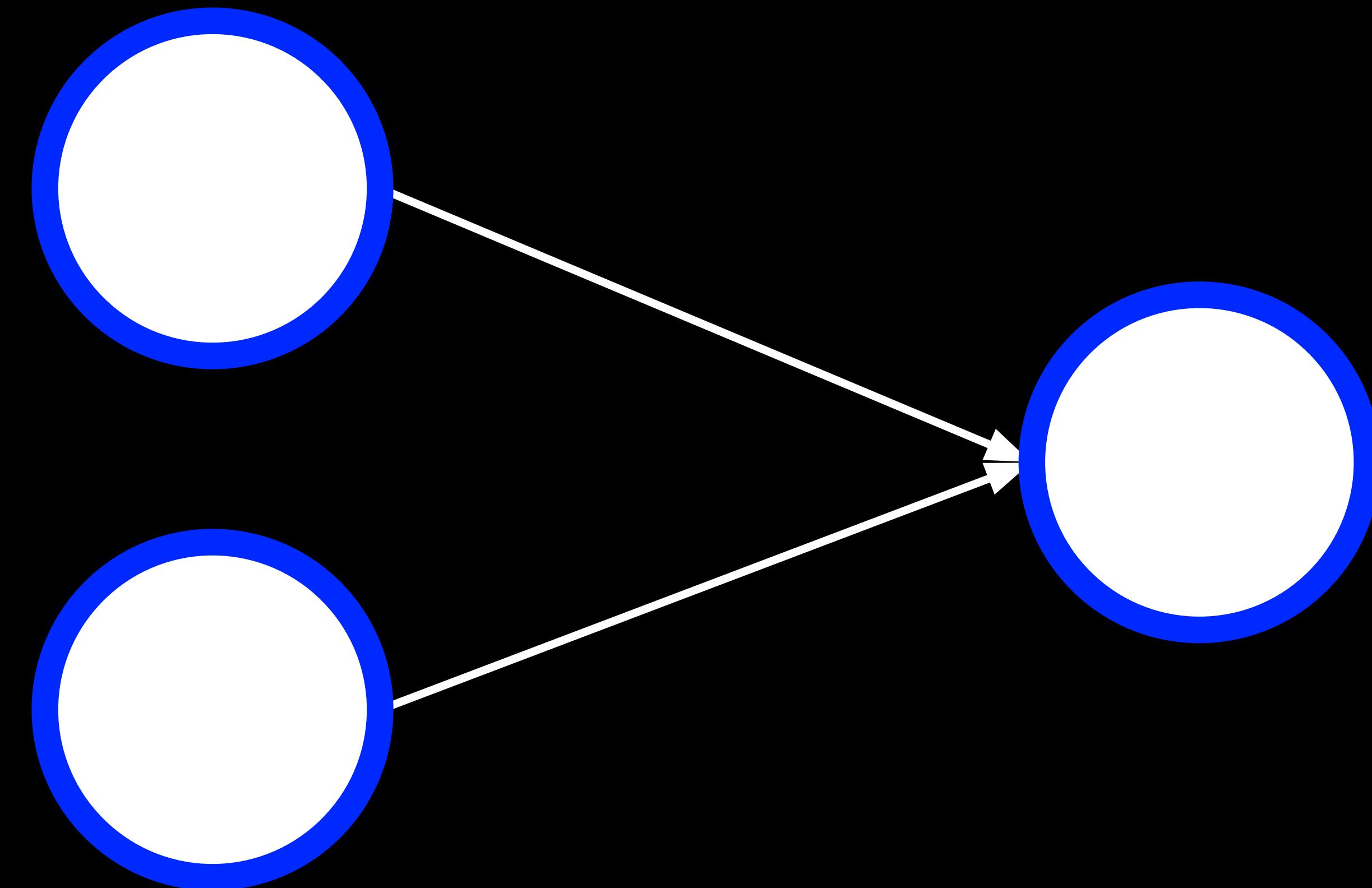
pressure



probability
of rain

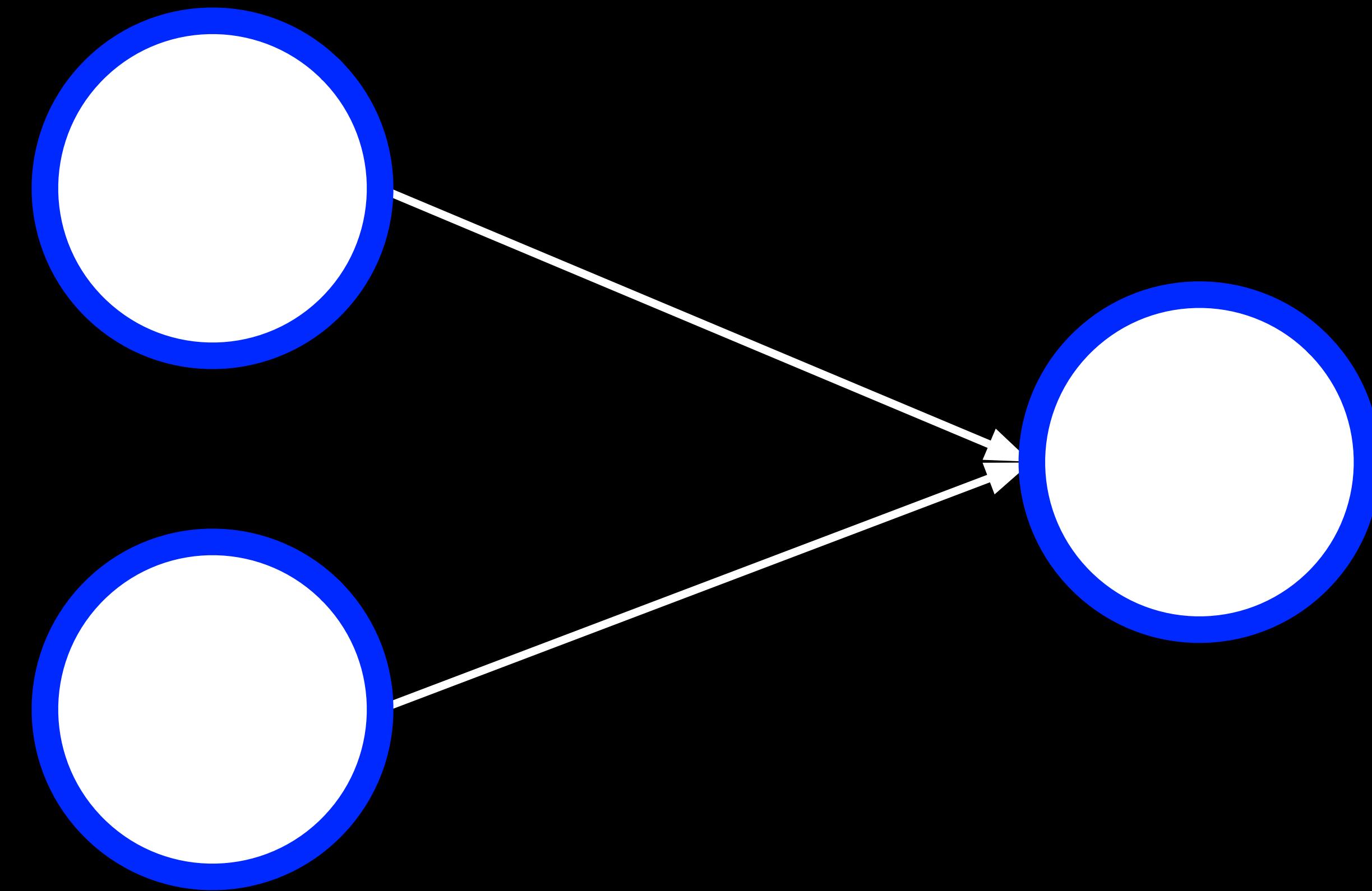


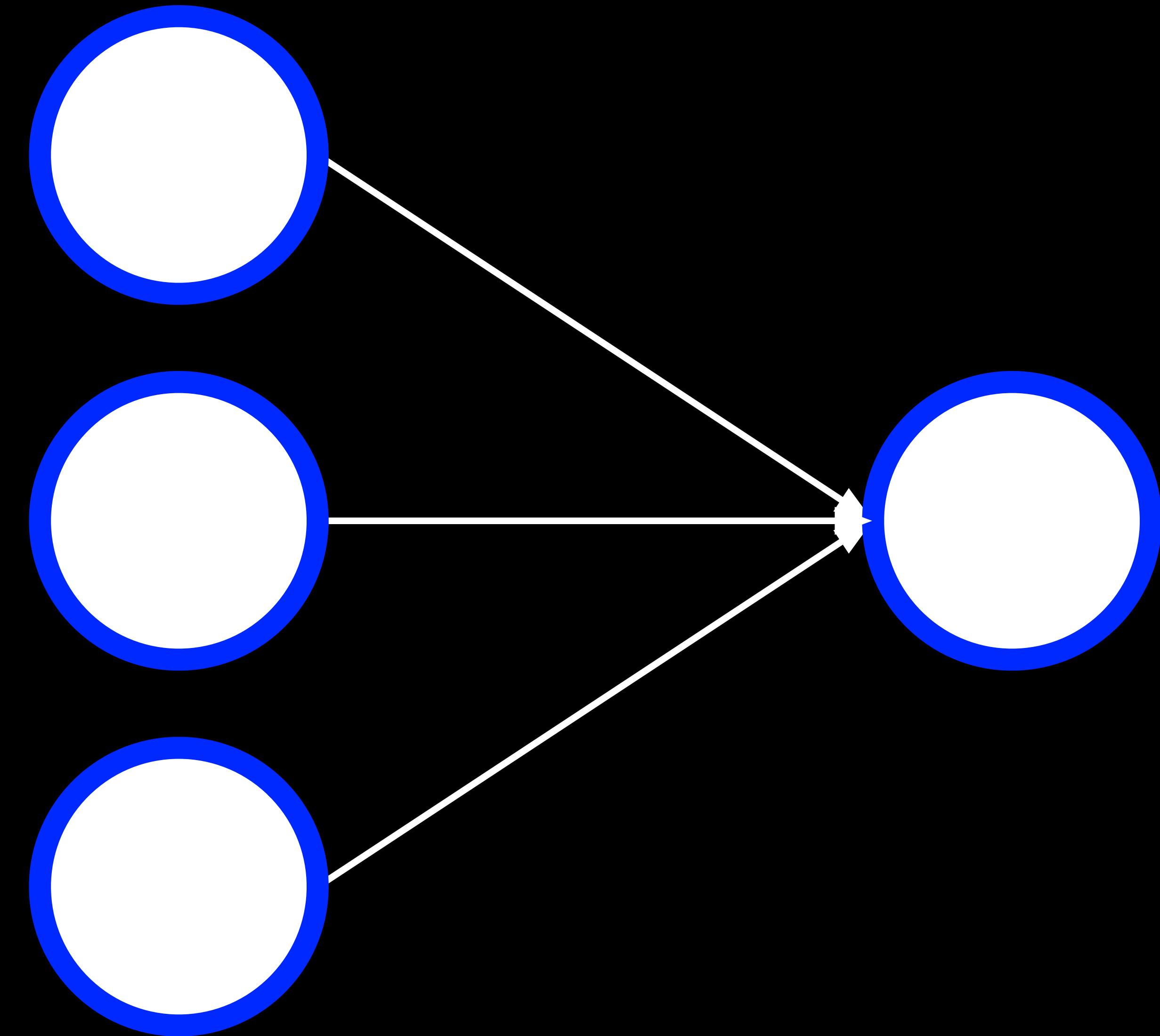
advertising

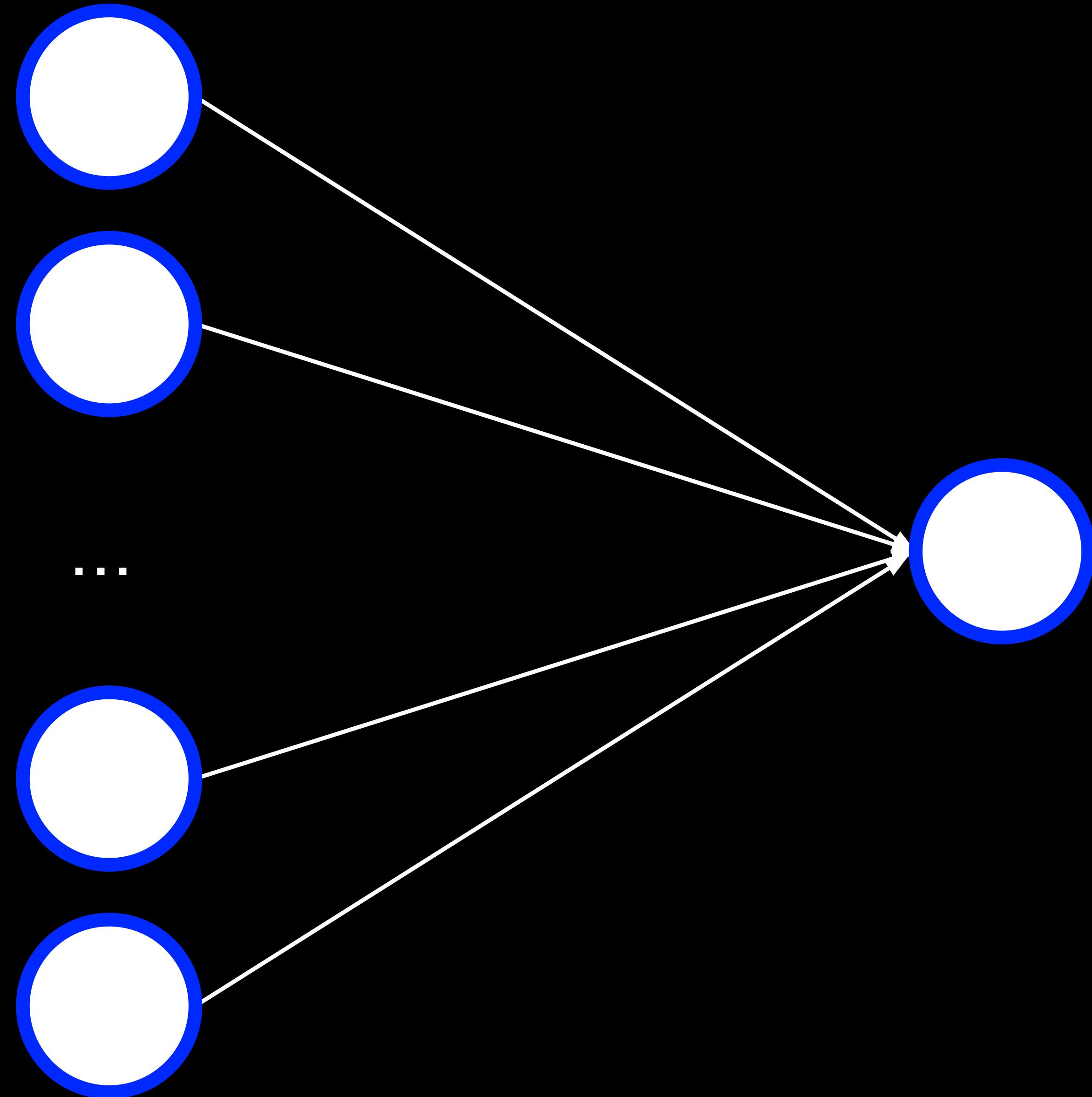


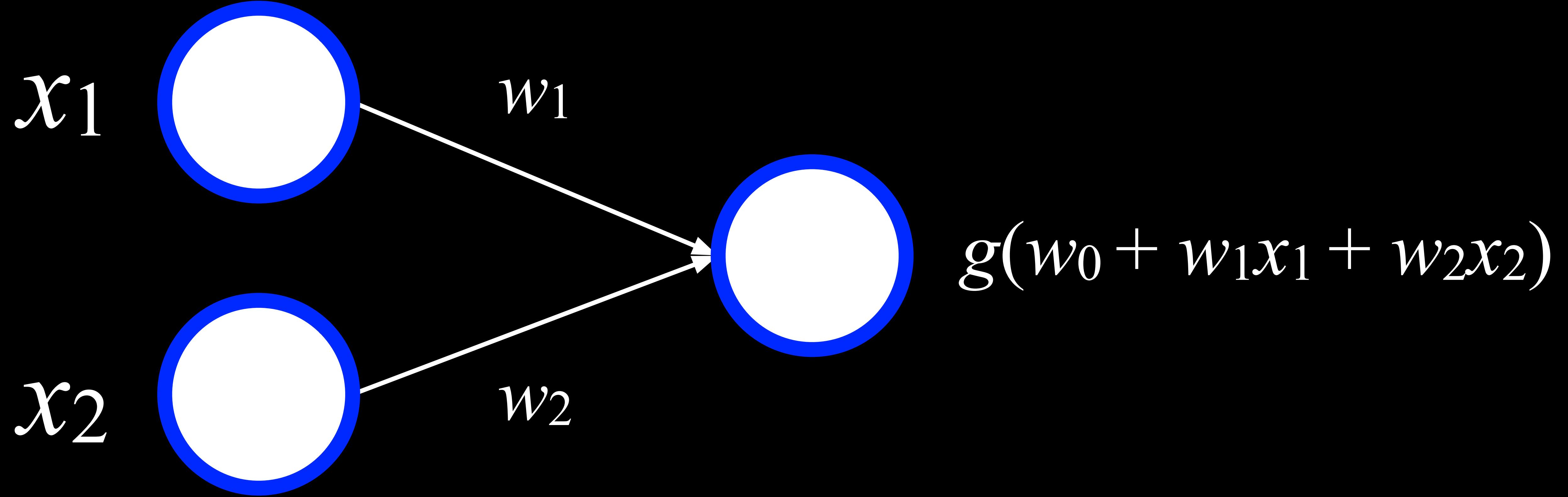
month

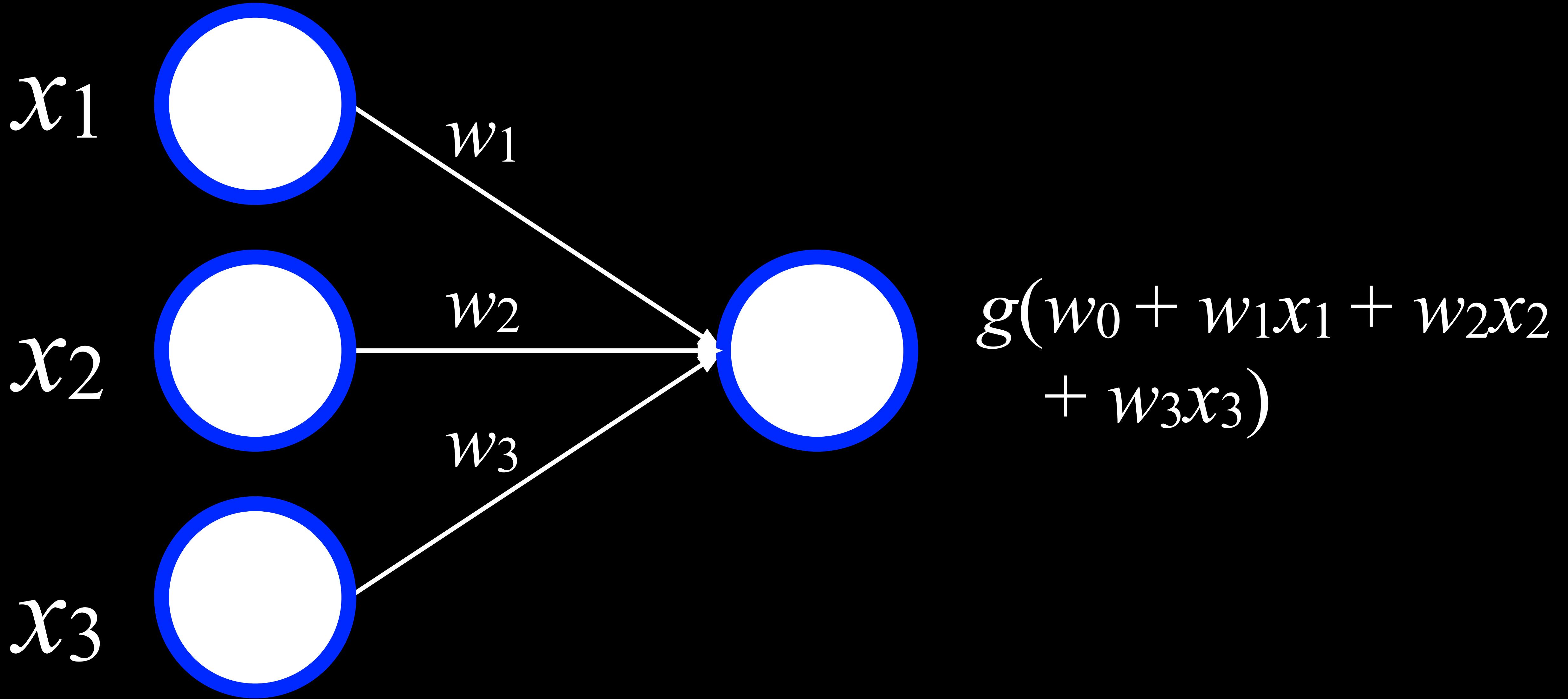
sales

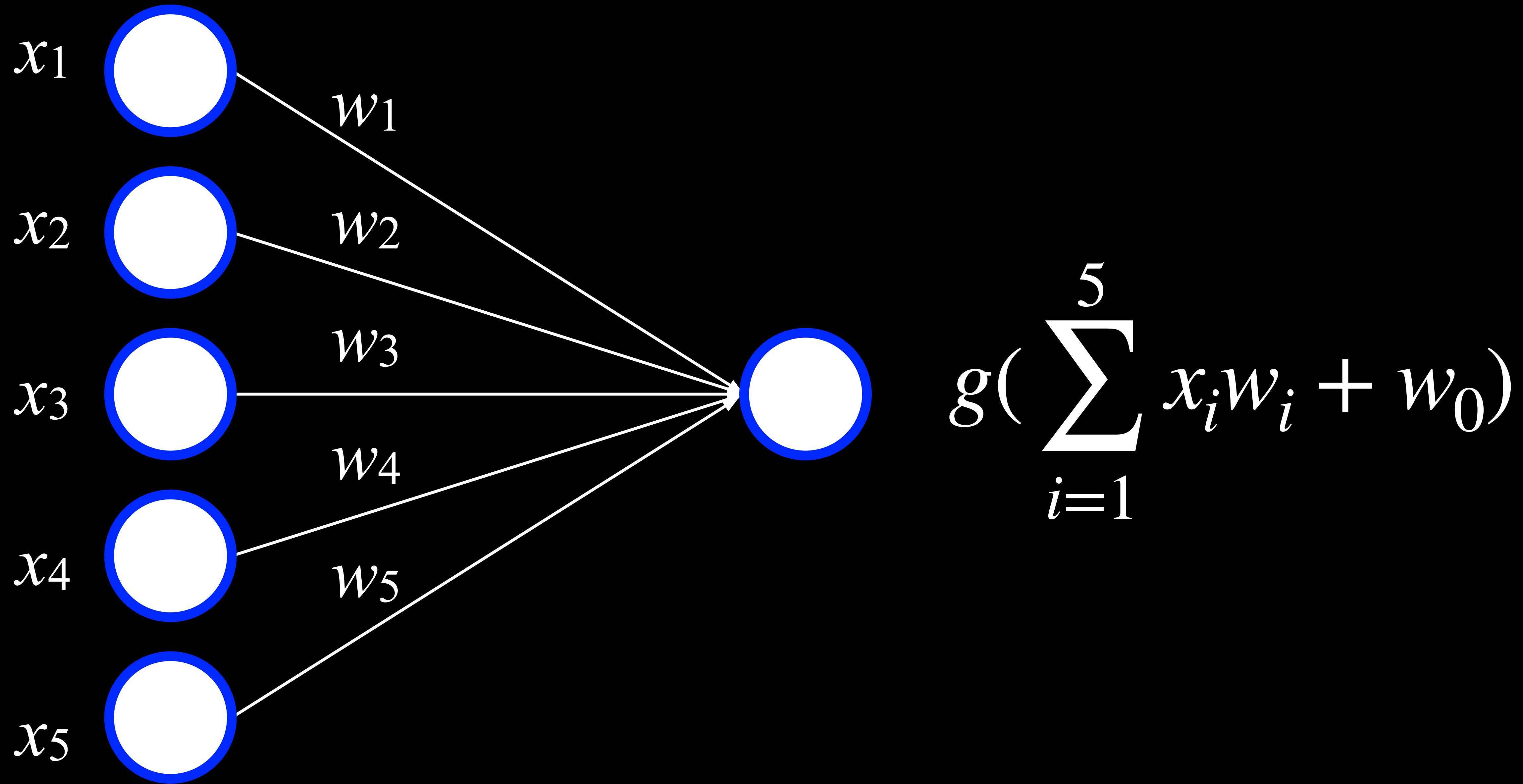


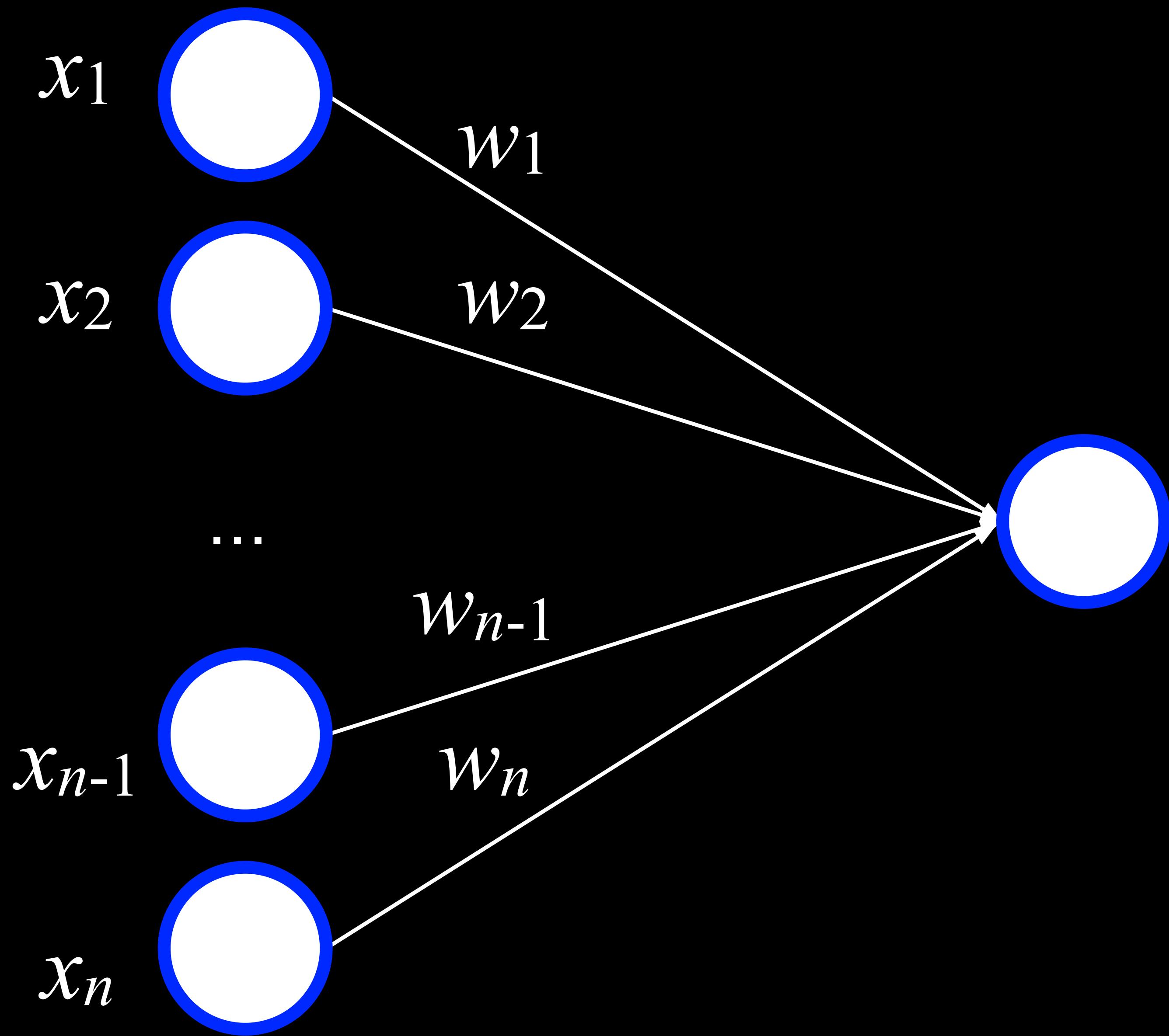












$$g\left(\sum_{i=1}^n x_i w_i + w_0\right)$$

gradient descent

algorithm for minimizing loss when training
neural network

Gradient Descent

- Start with a random choice of weights.
- Repeat:
 - Calculate the gradient based on all data points: direction that will lead to decreasing loss.
 - Update weights according to the gradient.

Gradient Descent

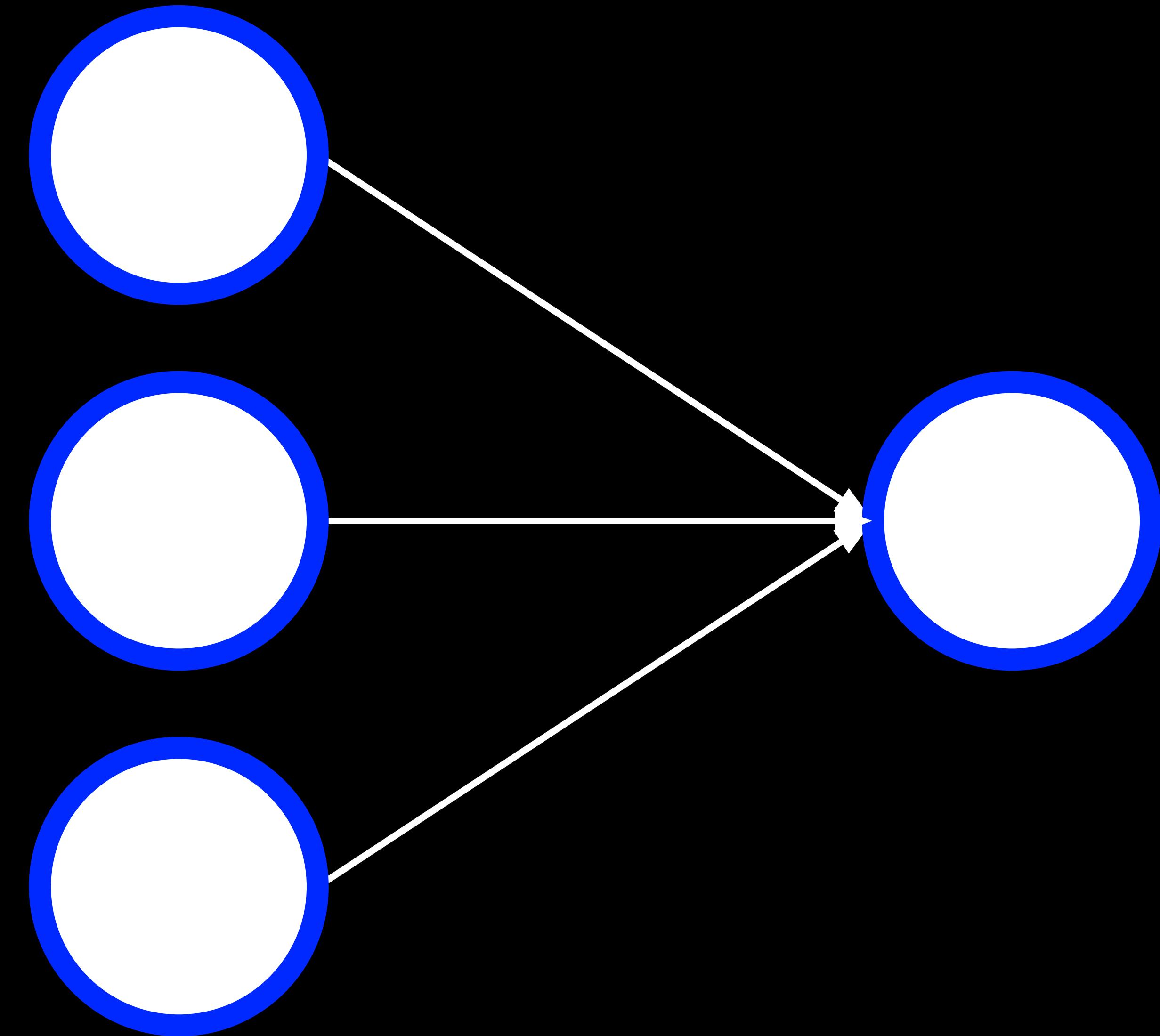
- Start with a random choice of weights.
- Repeat:
 - Calculate the gradient based on **all data points**: direction that will lead to decreasing loss.
 - Update weights according to the gradient.

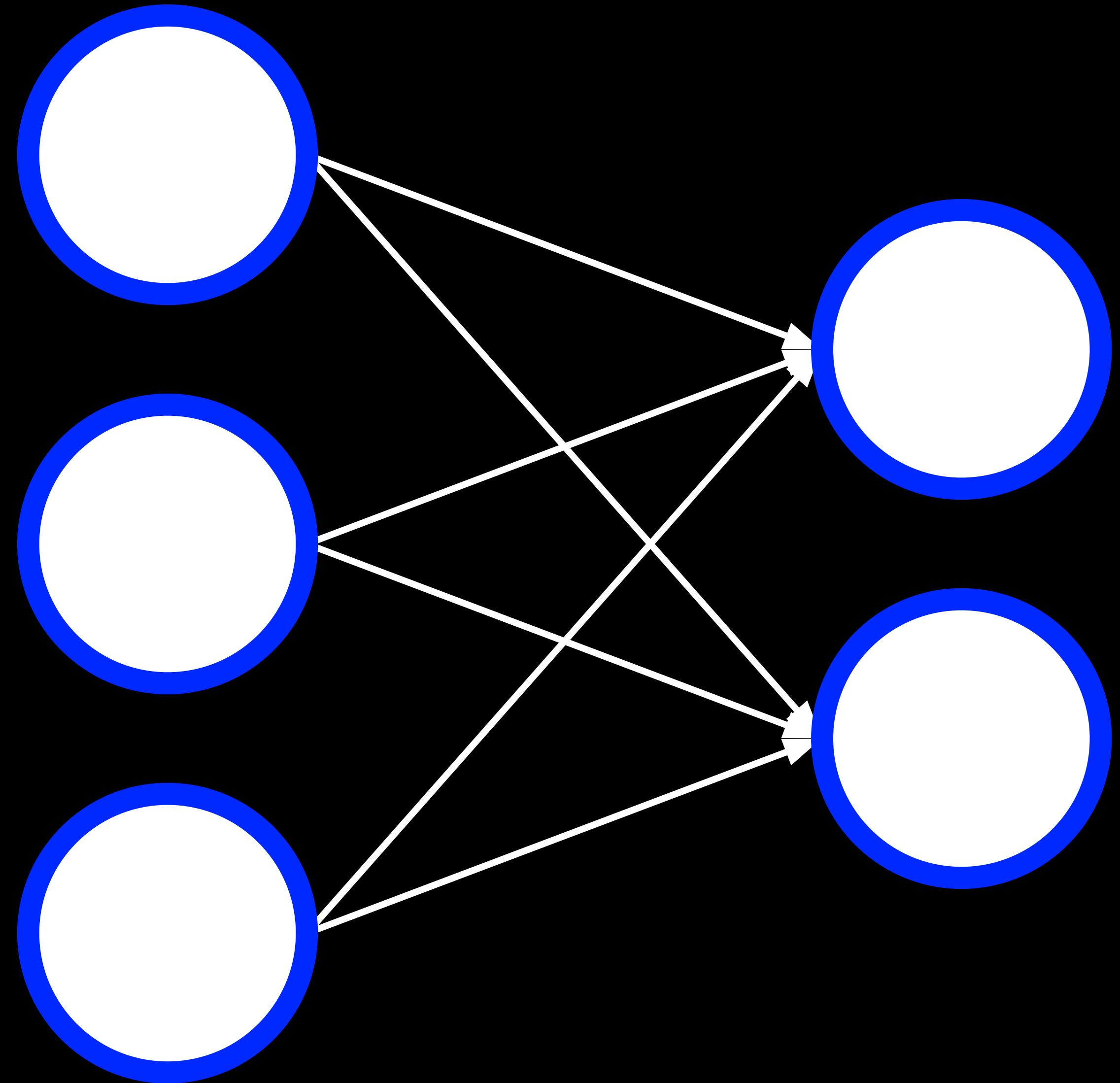
Stochastic Gradient Descent

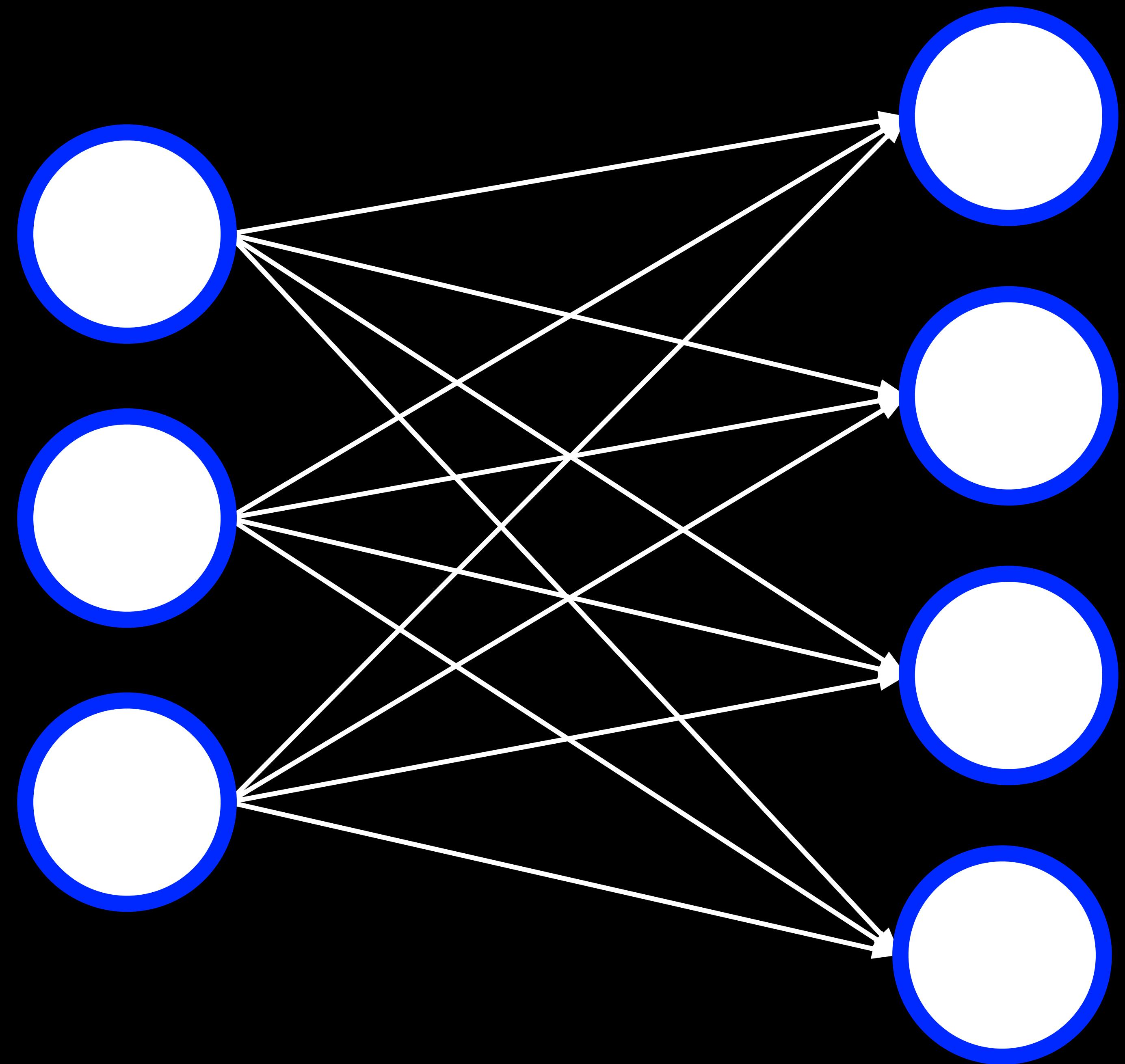
- Start with a random choice of weights.
- Repeat:
 - Calculate the gradient based on **one data point**: direction that will lead to decreasing loss.
 - Update weights according to the gradient.

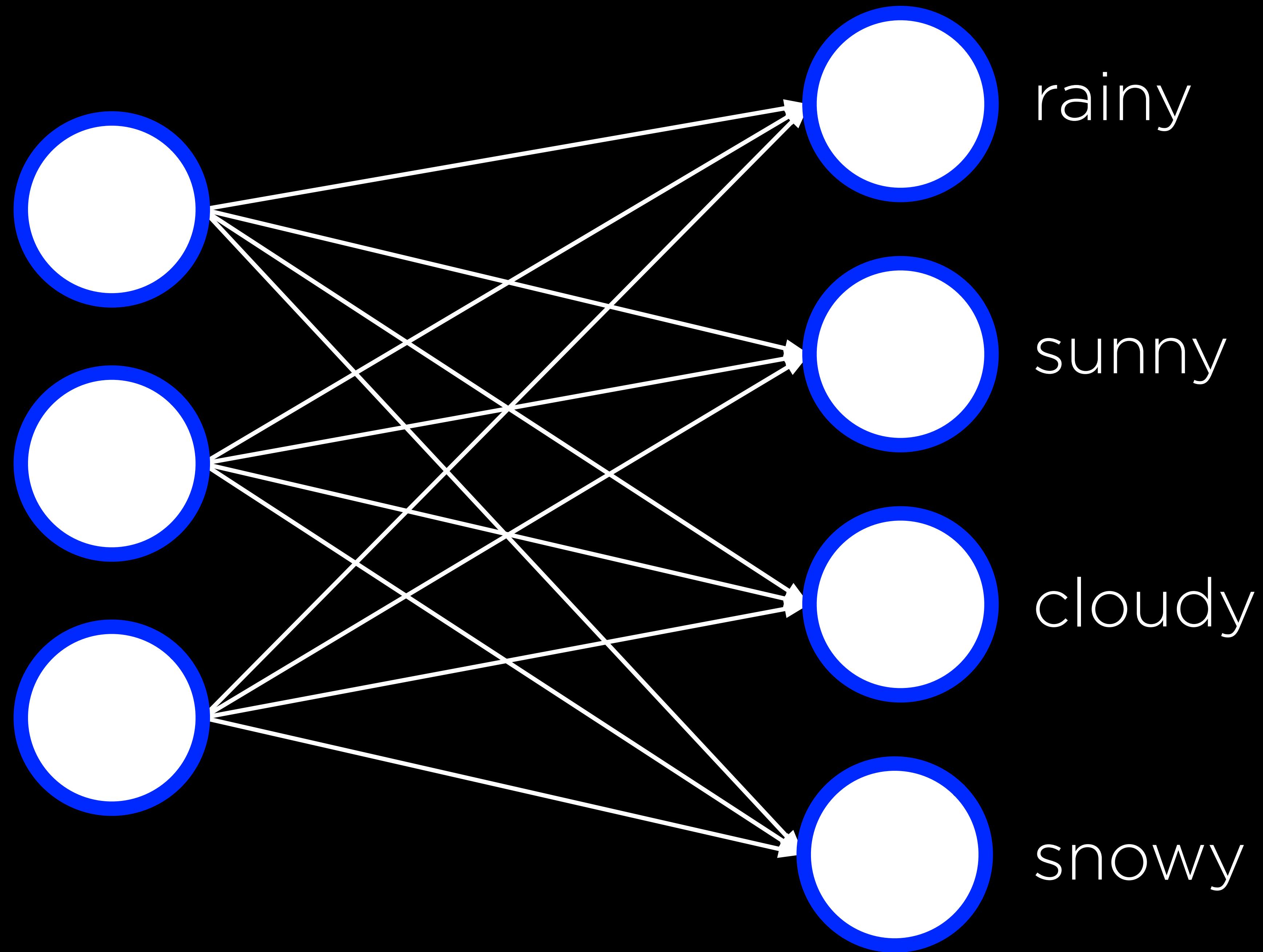
Mini-Batch Gradient Descent

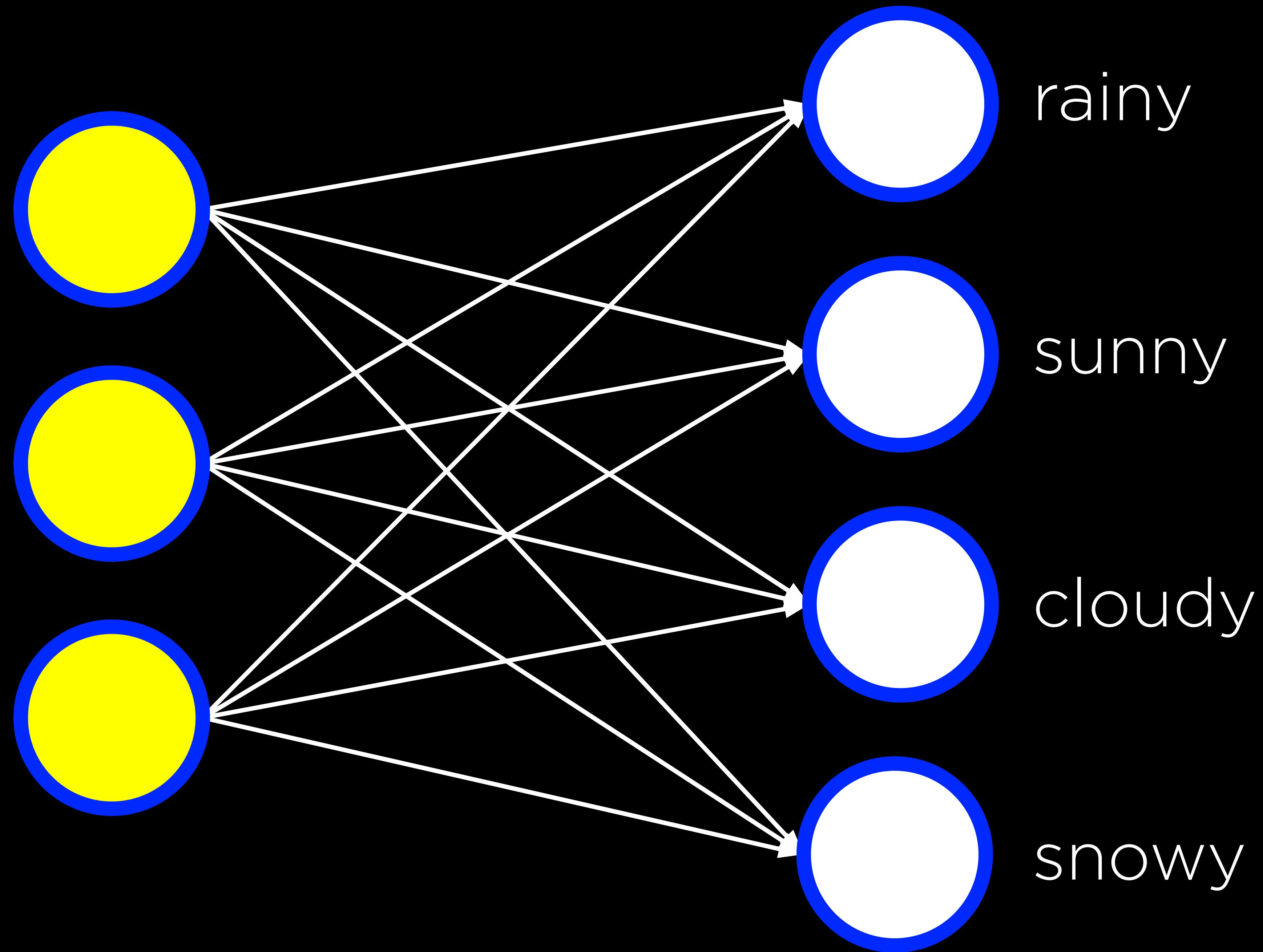
- Start with a random choice of weights.
- Repeat:
 - Calculate the gradient based on **one small batch**: direction that will lead to decreasing loss.
 - Update weights according to the gradient.

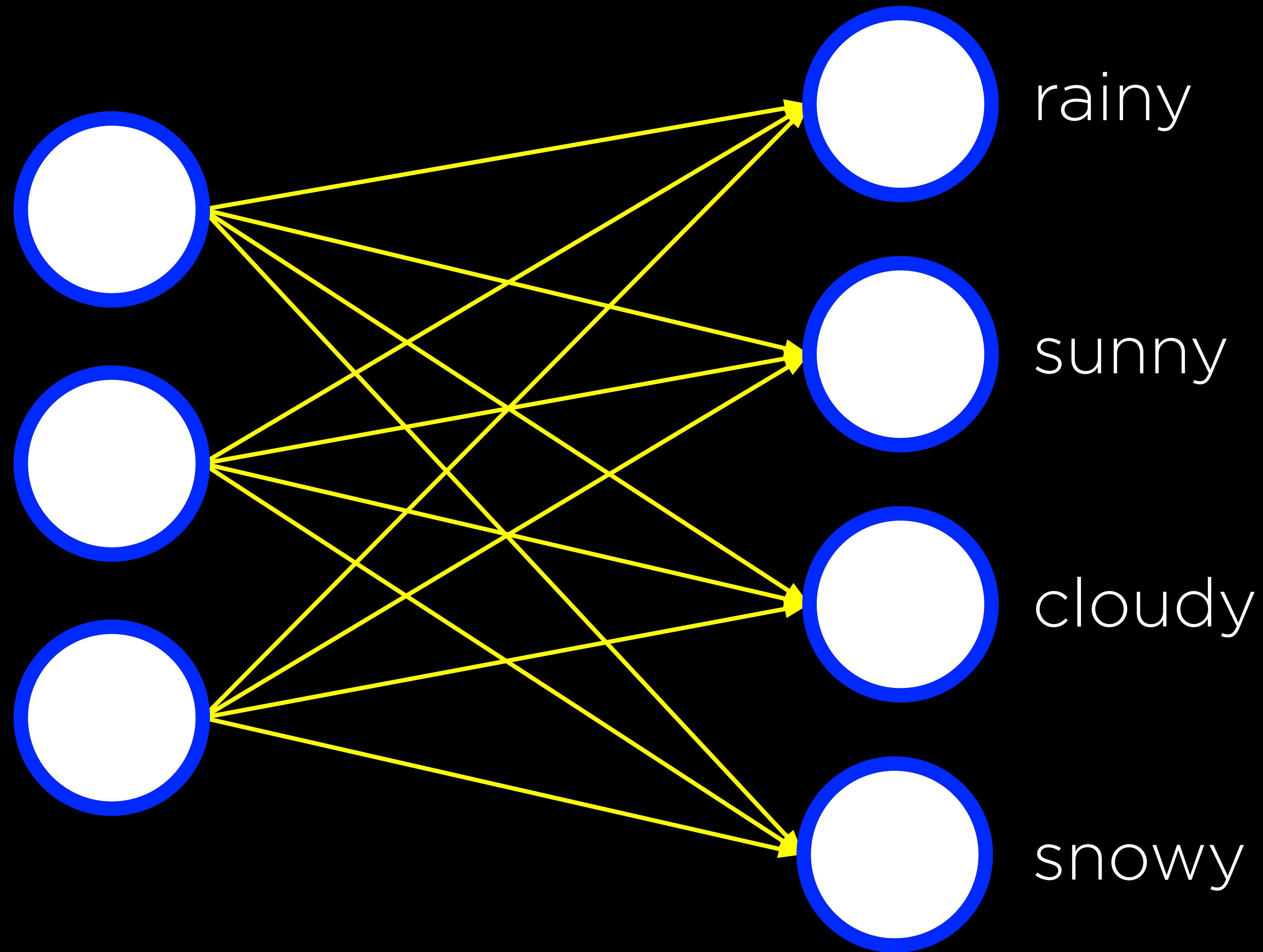


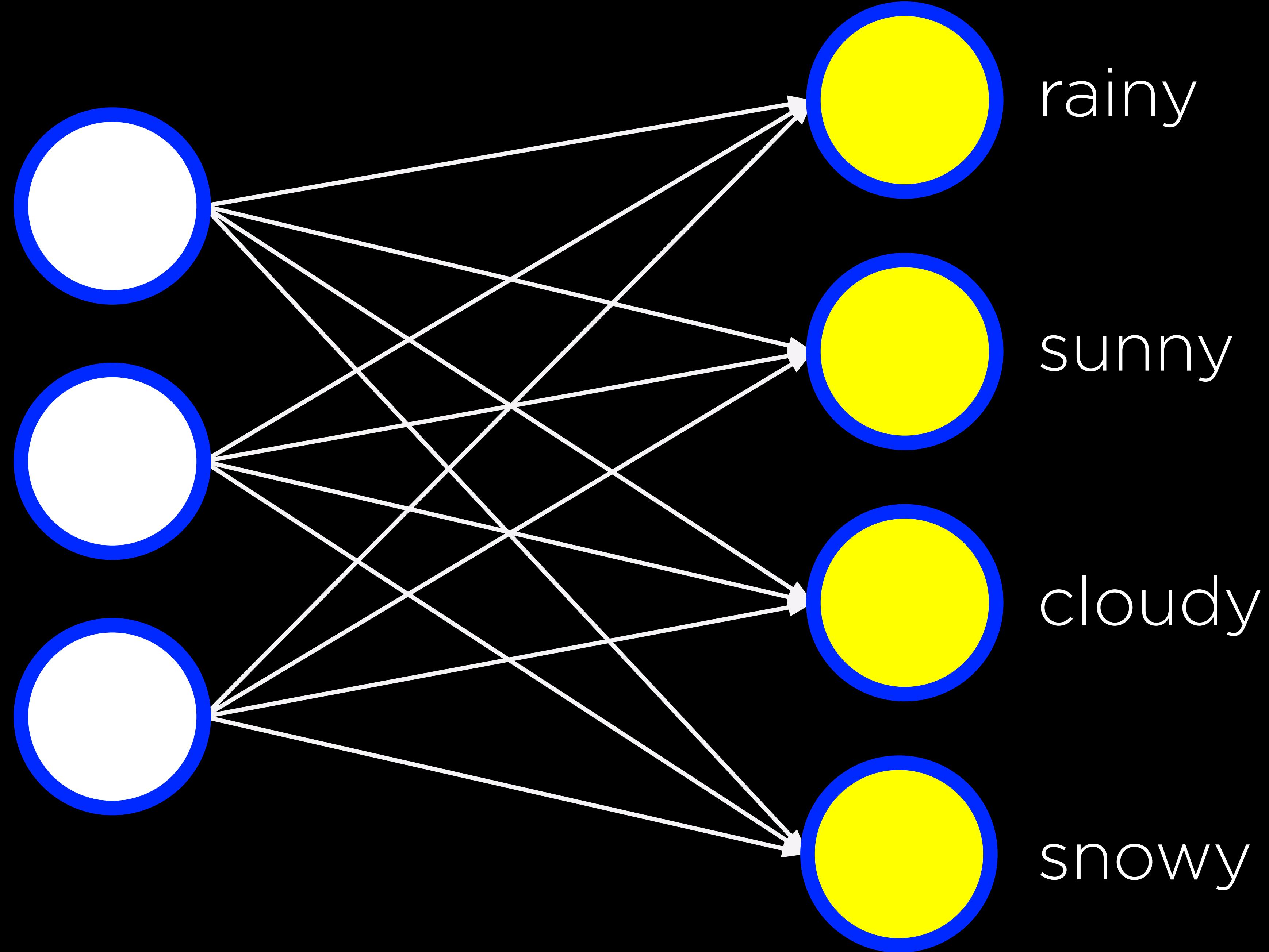


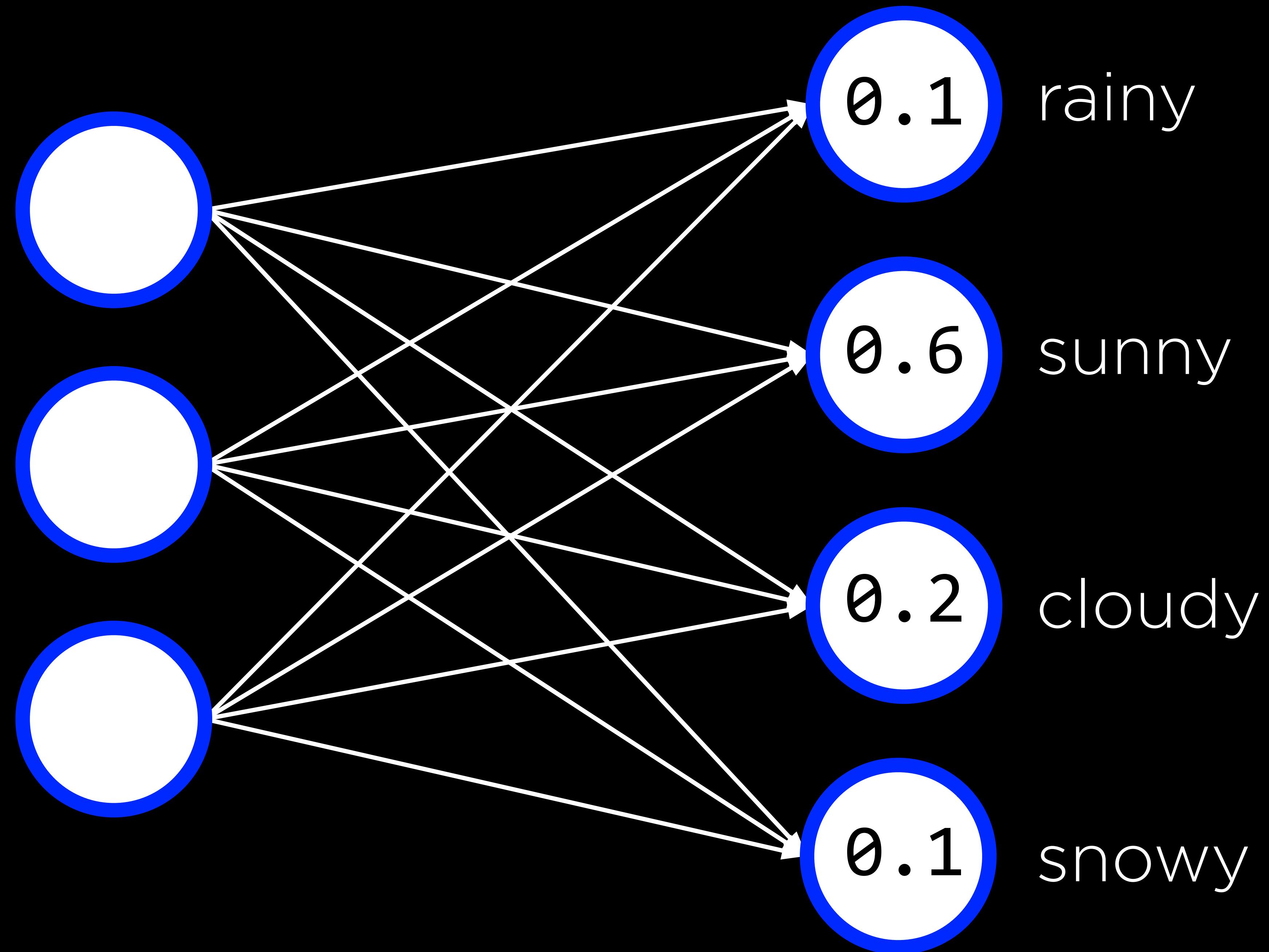


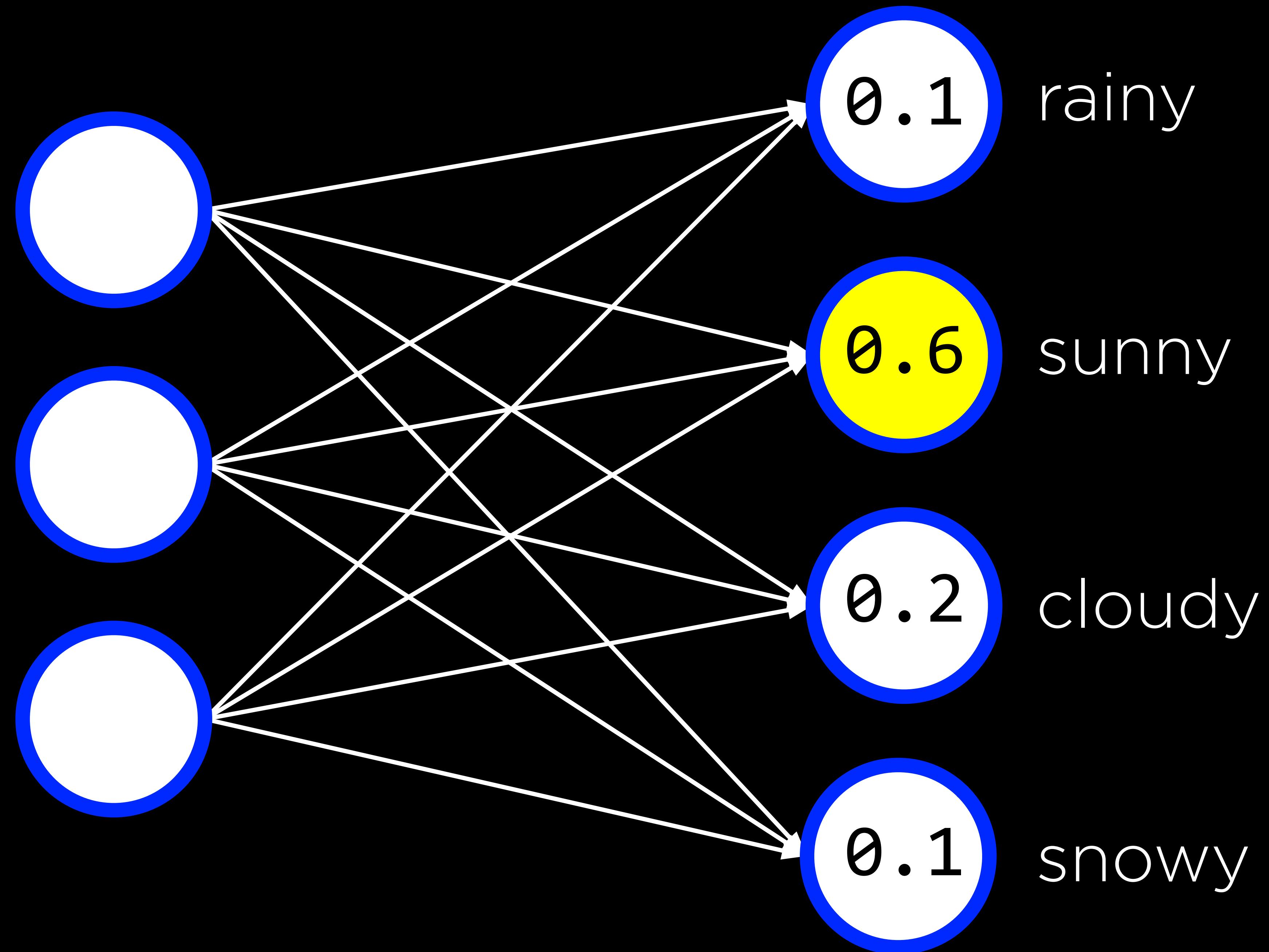


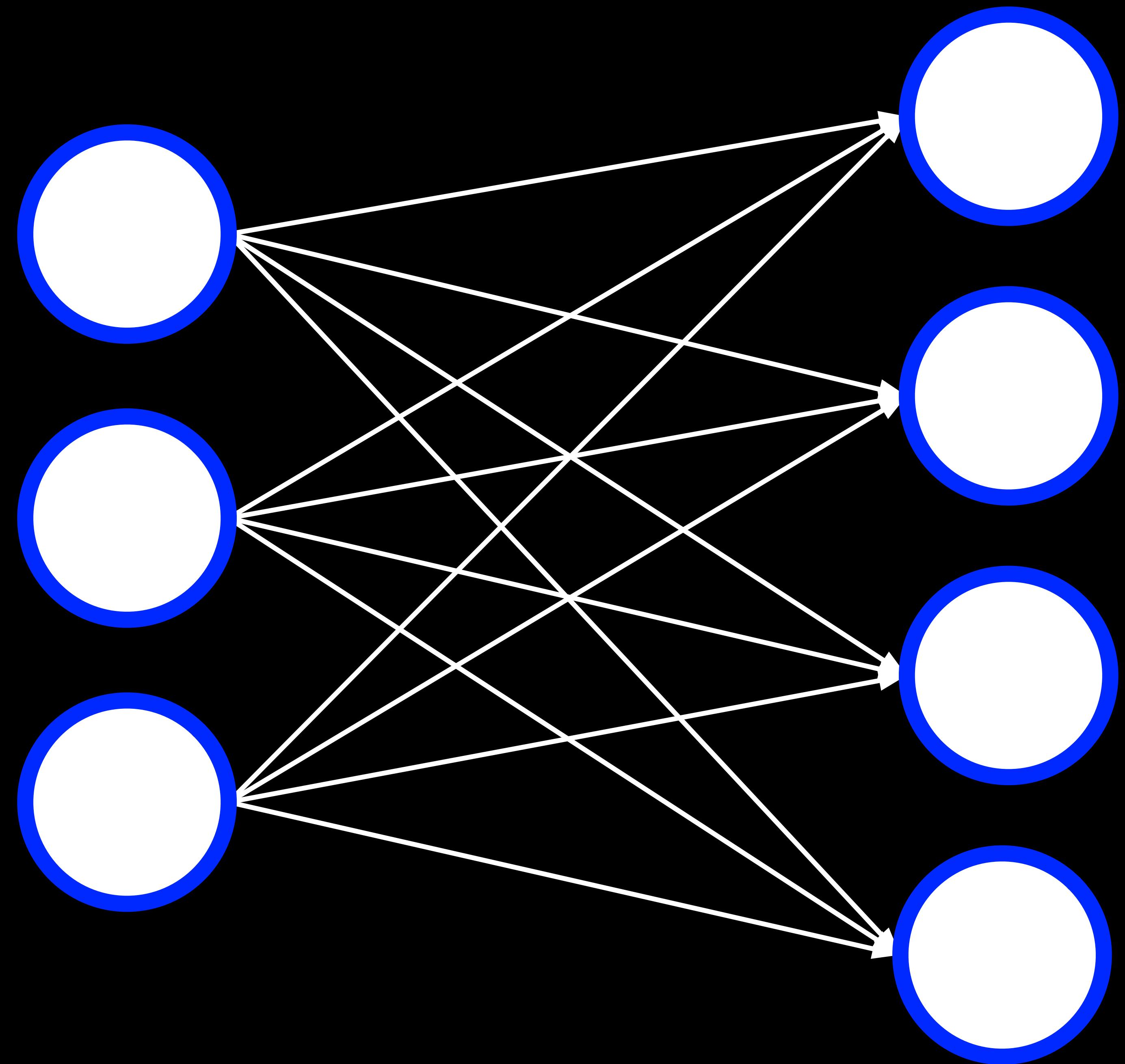


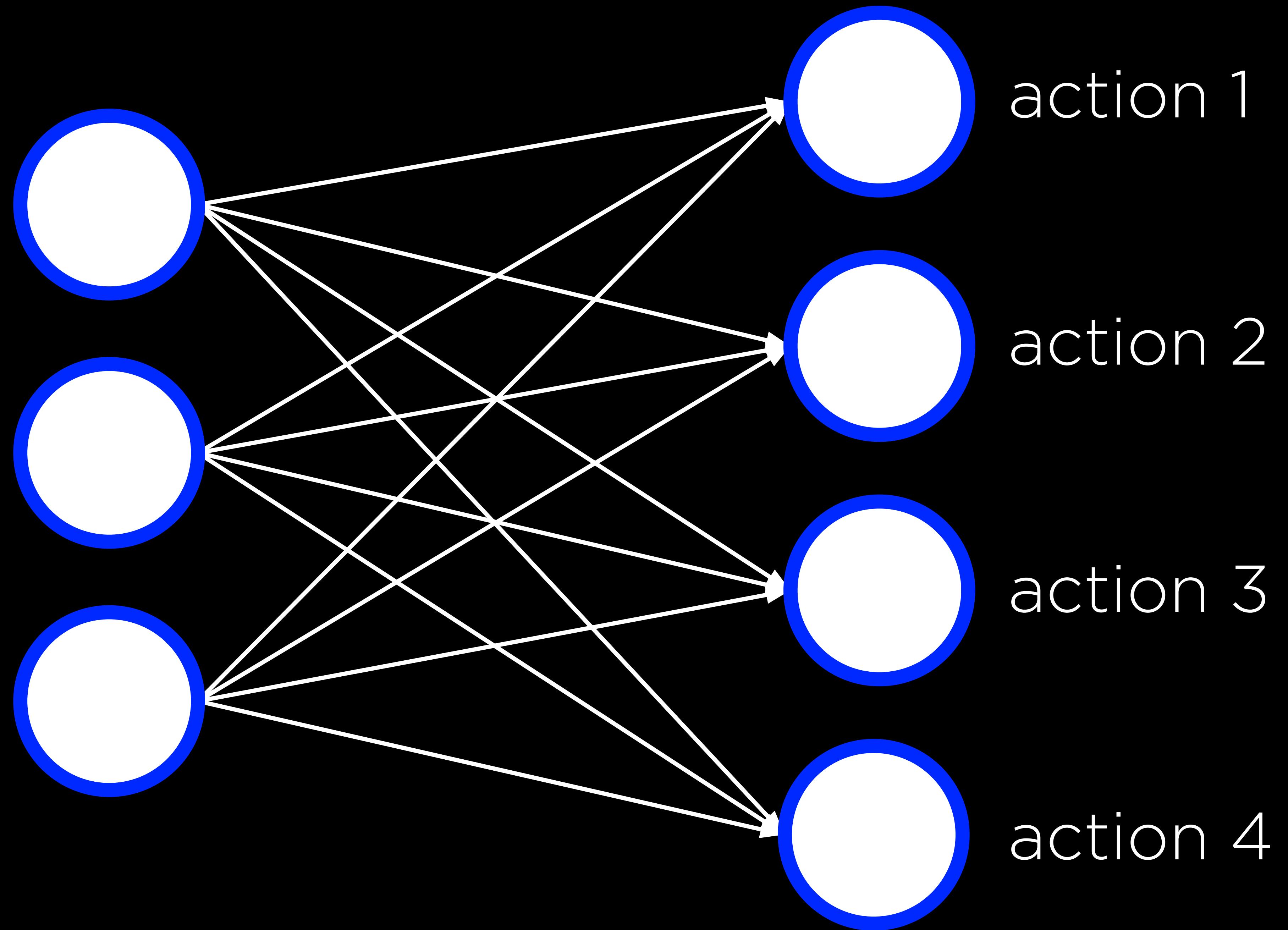


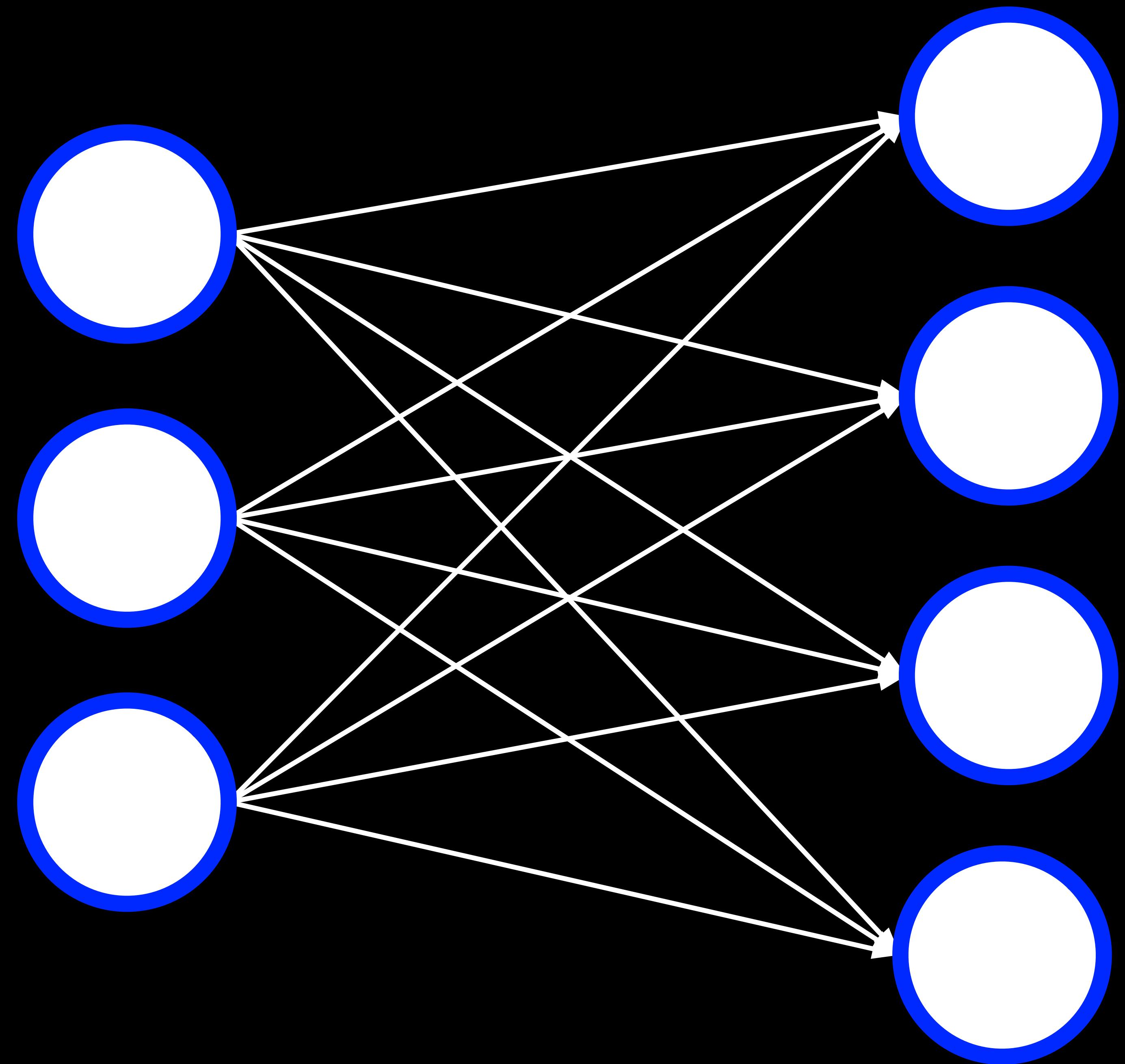


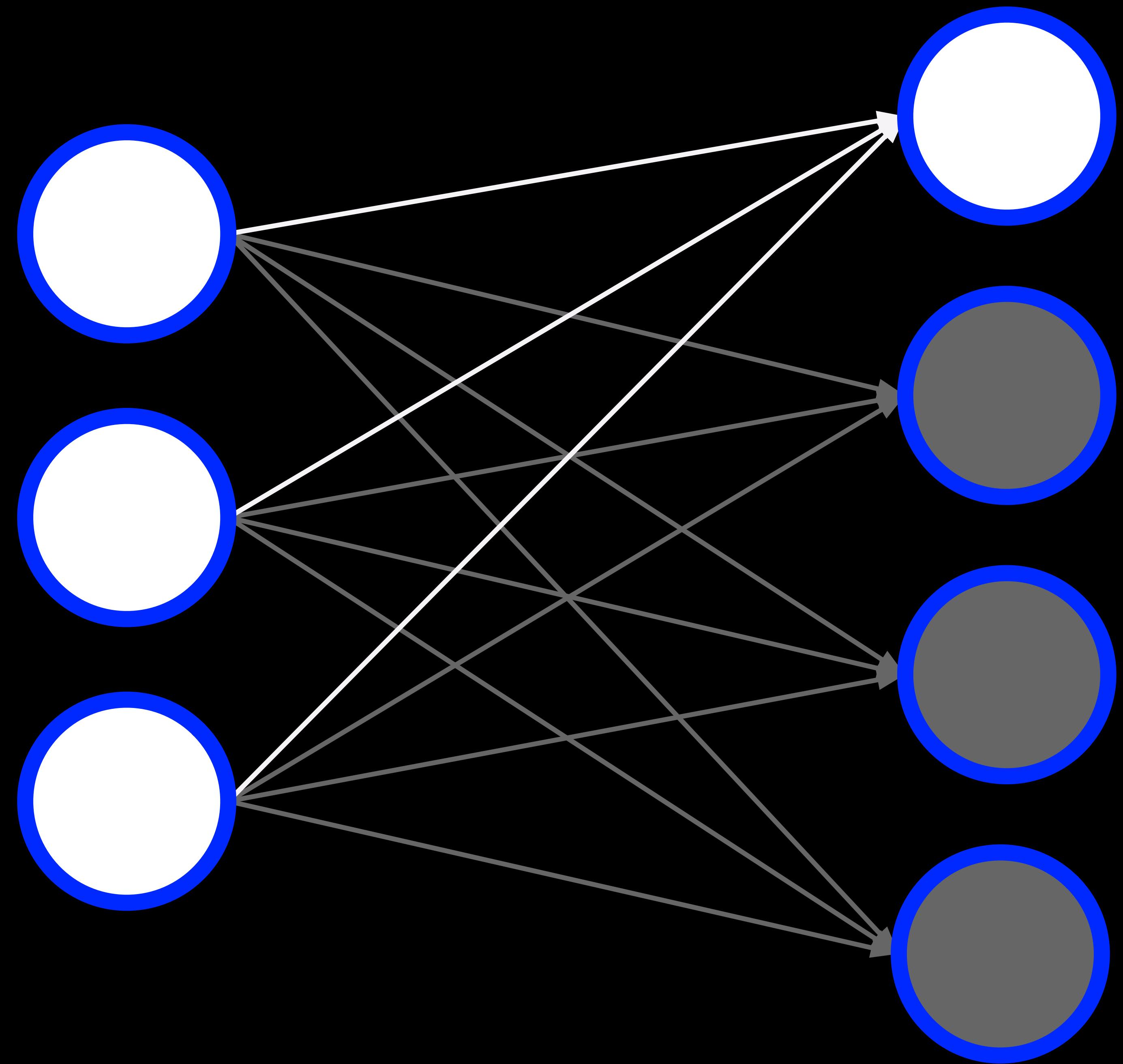


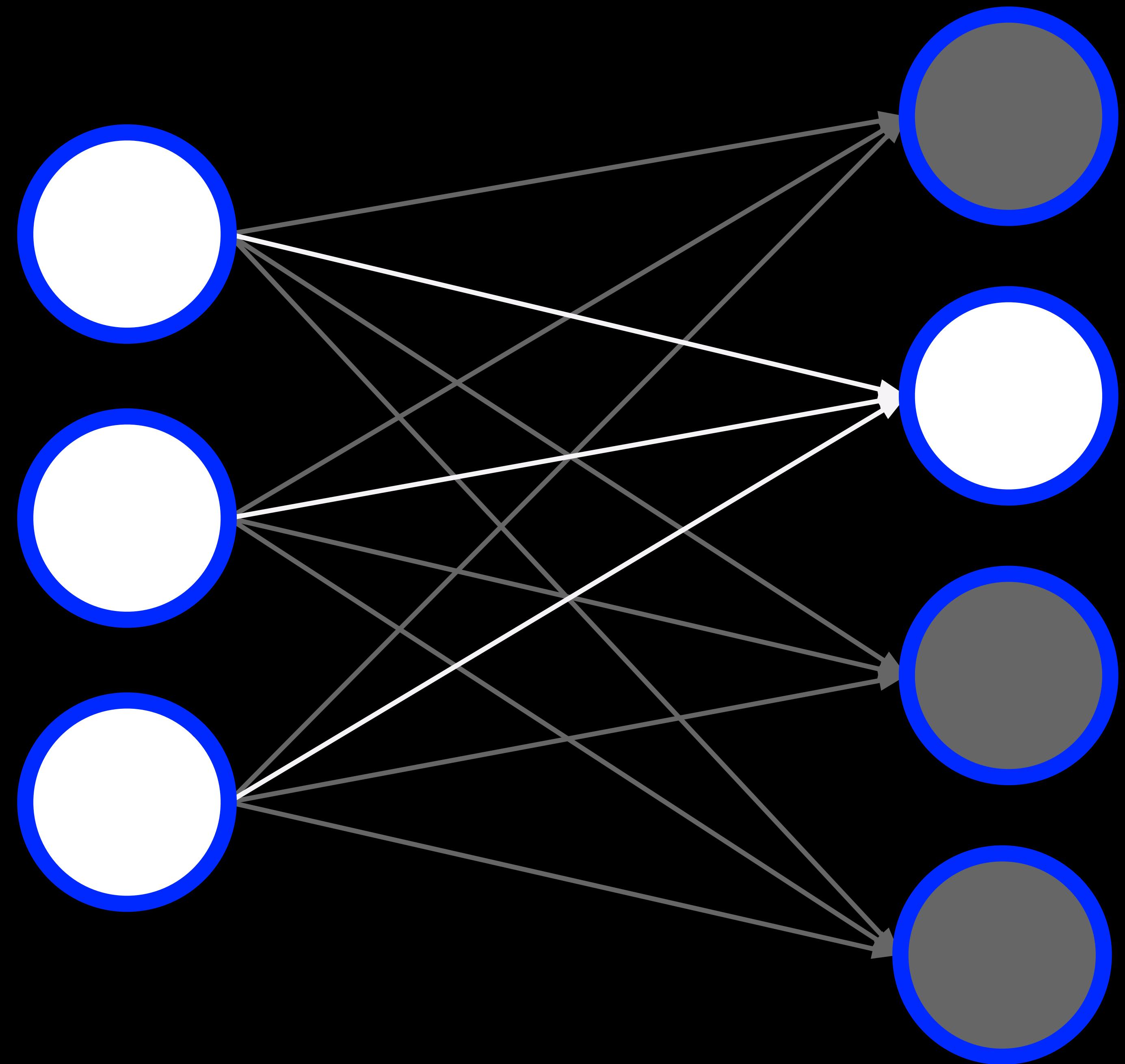


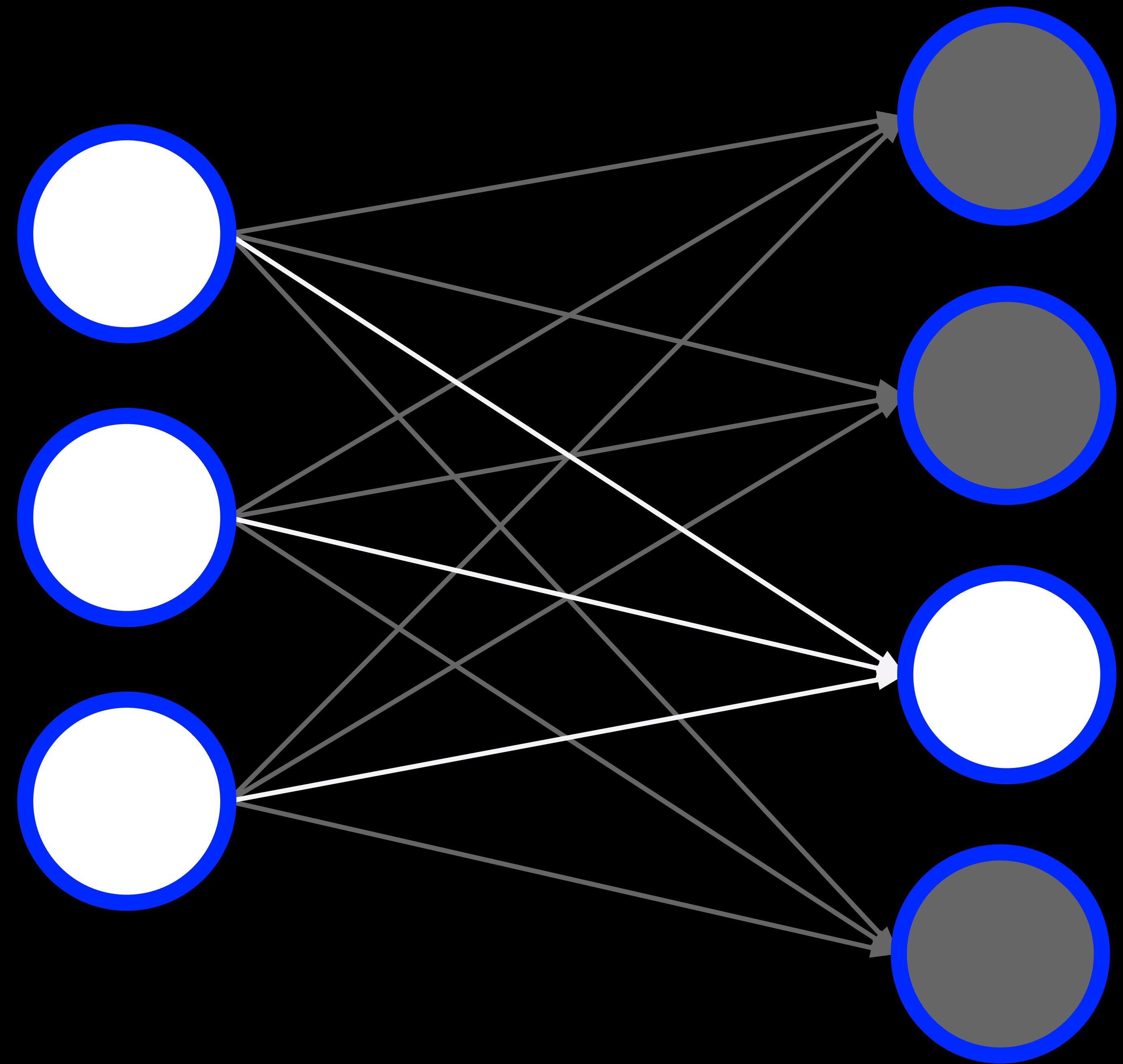


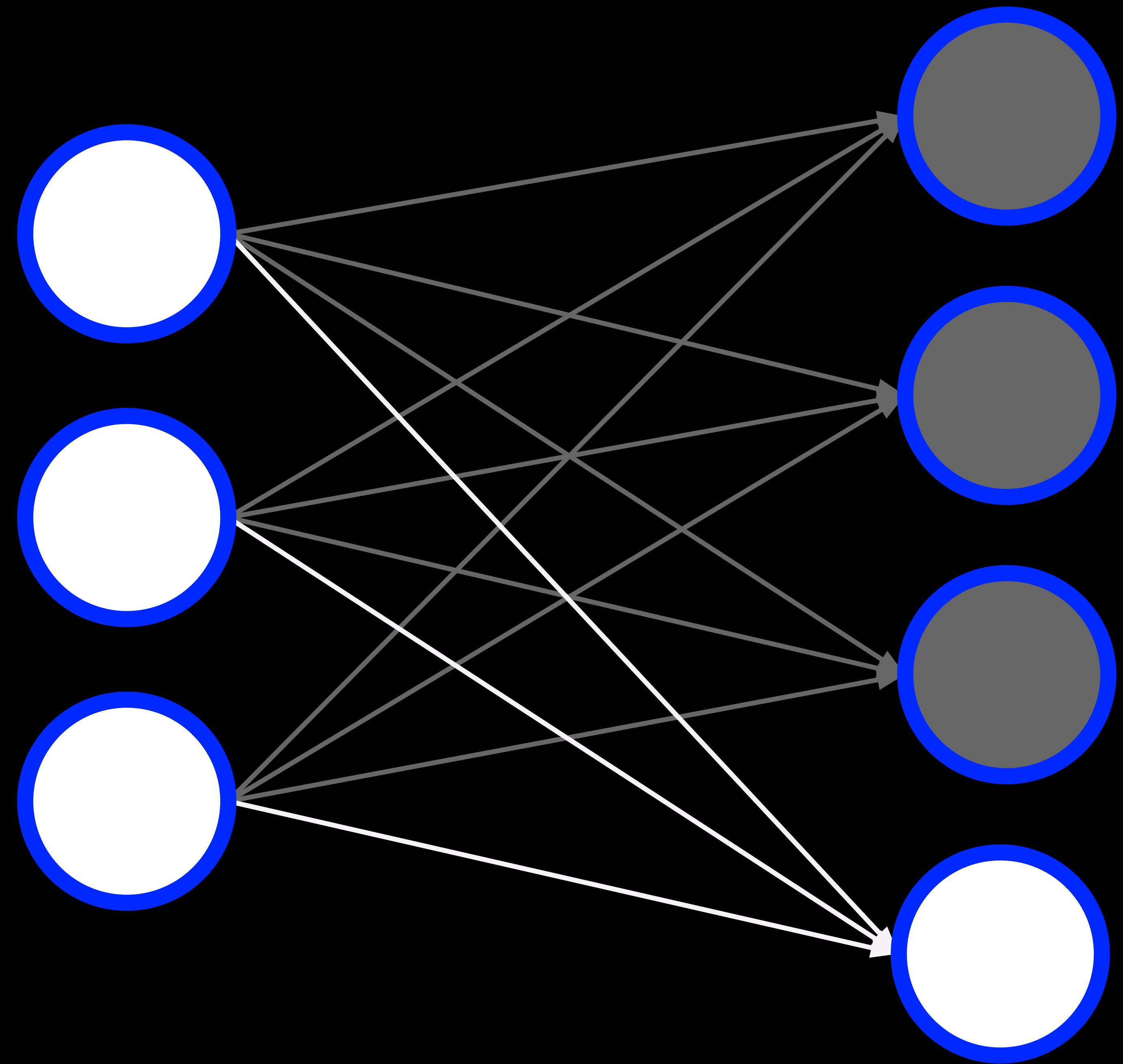


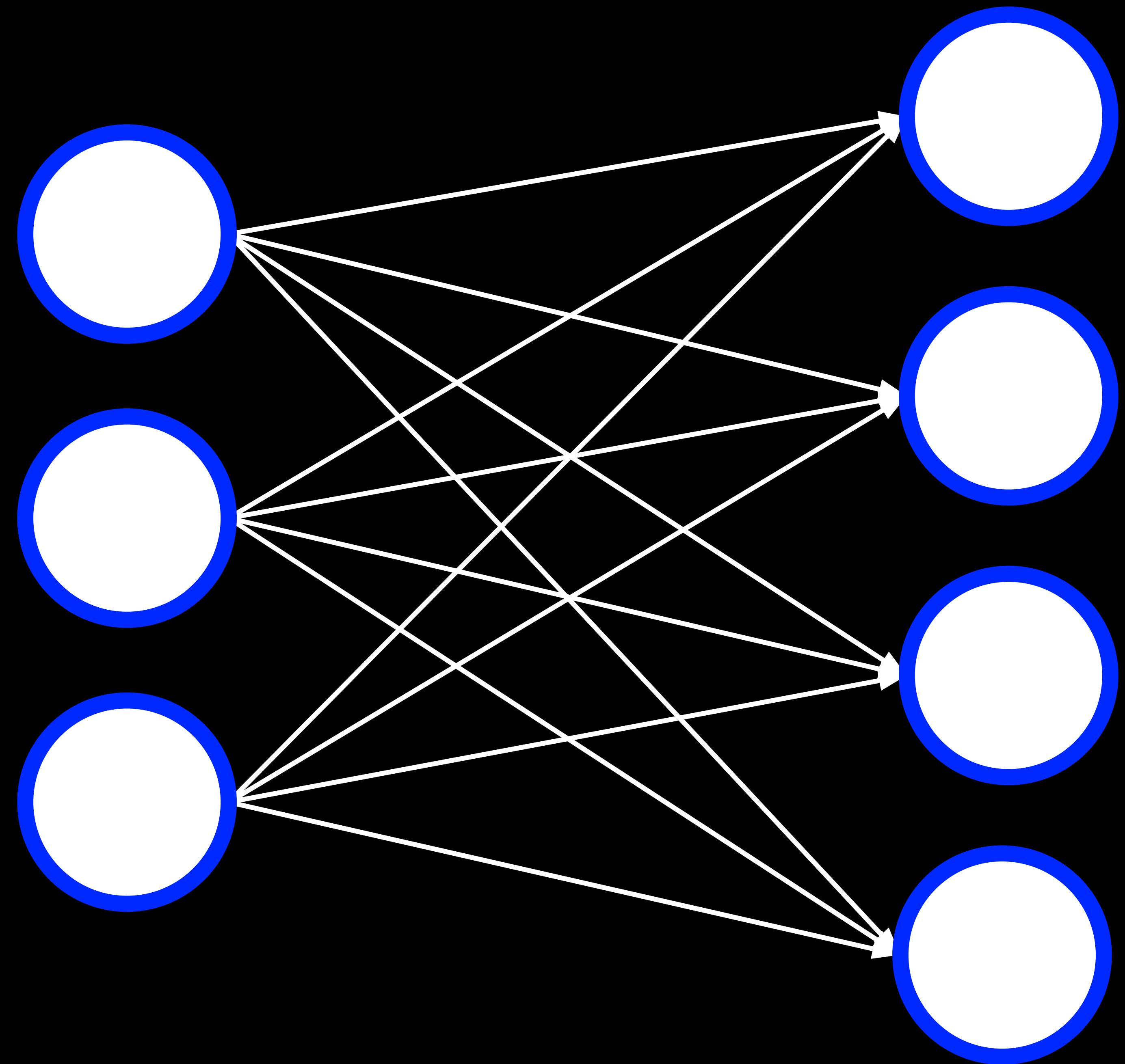


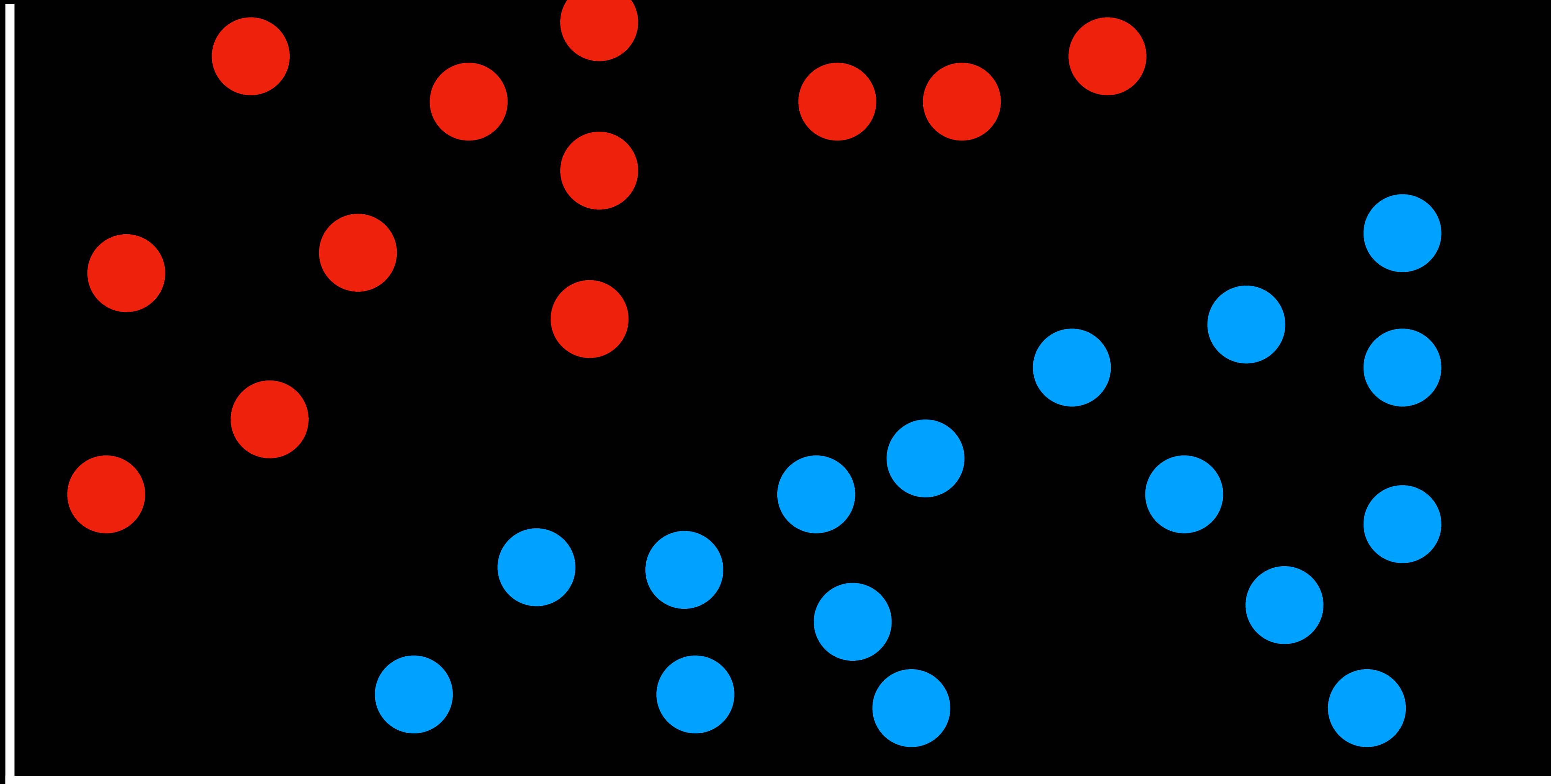


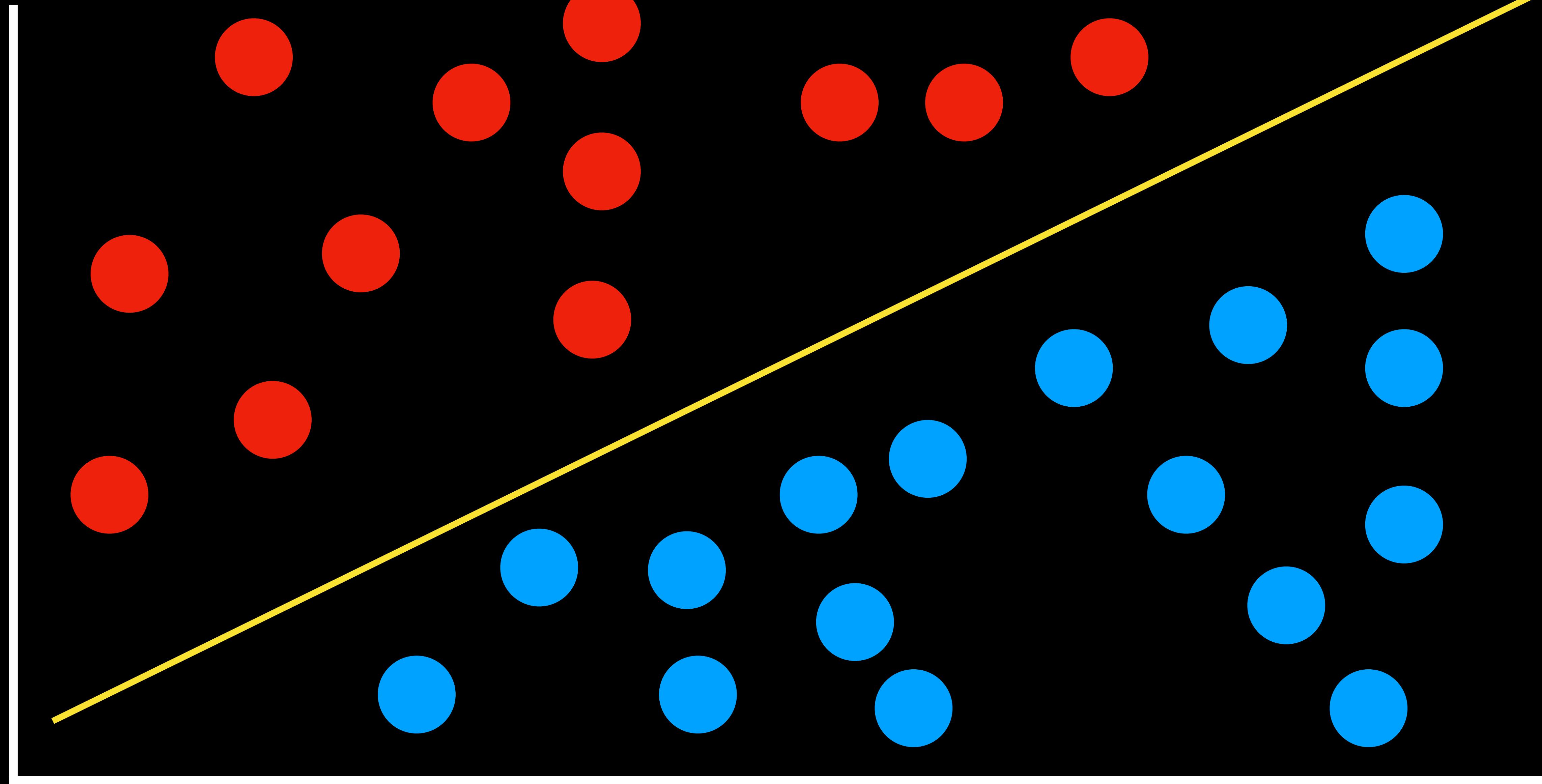


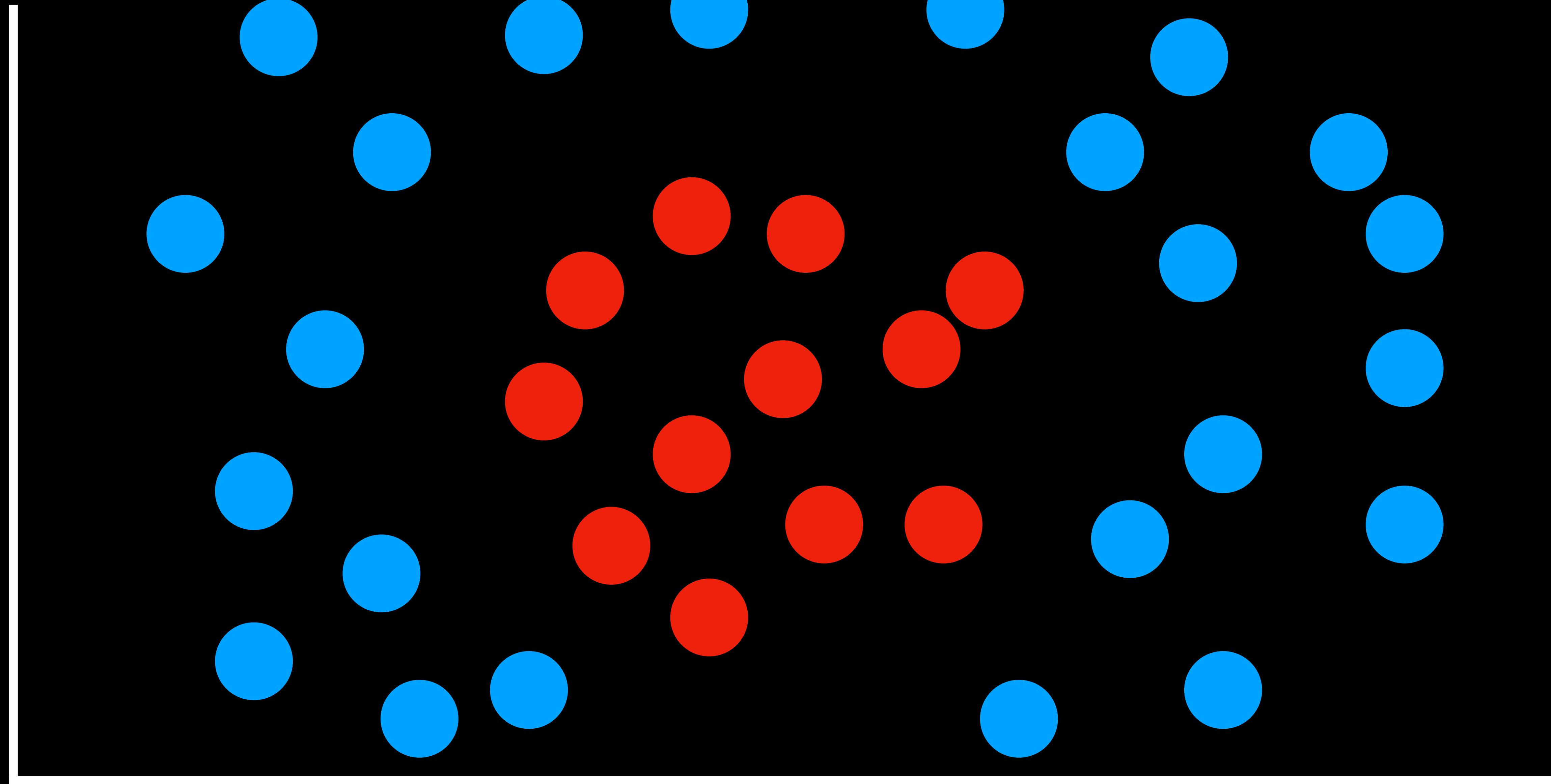










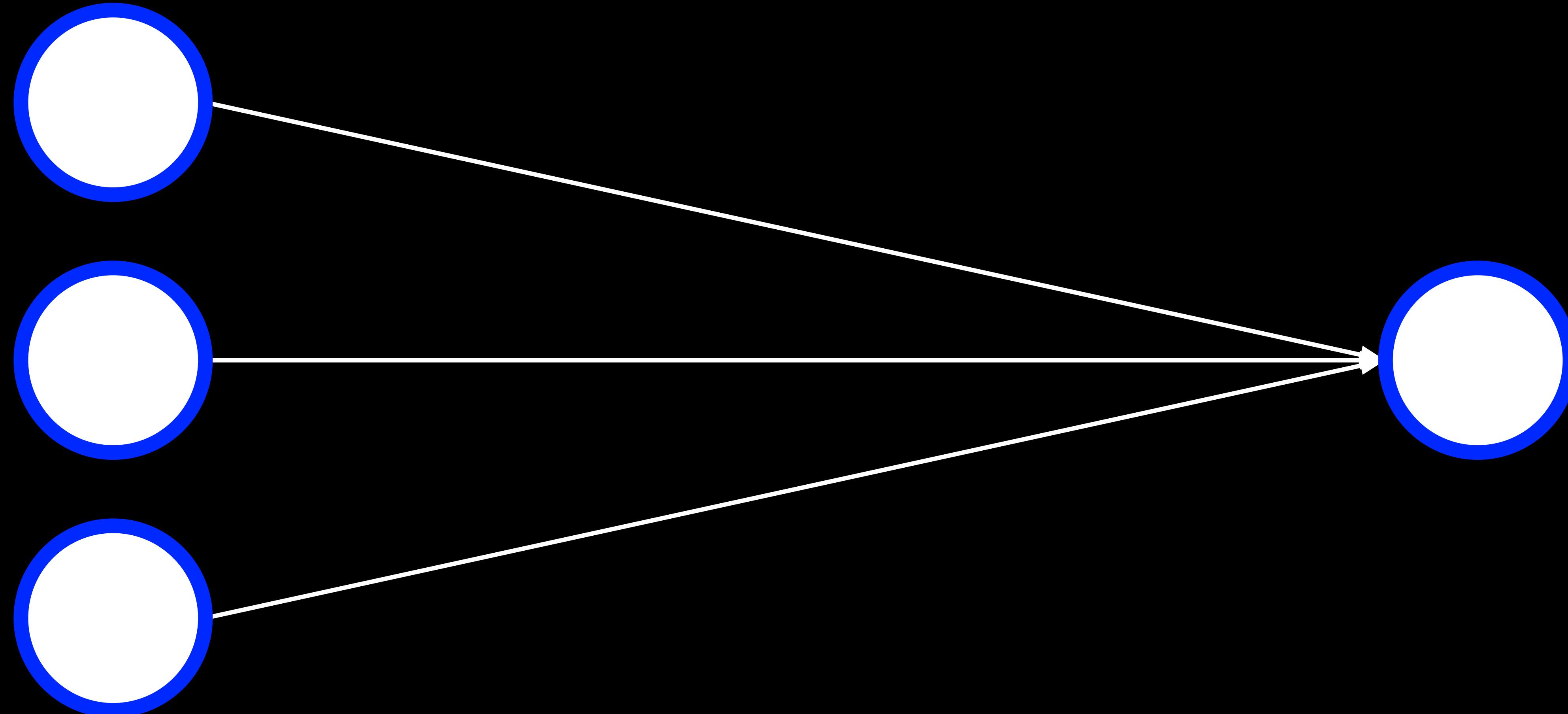


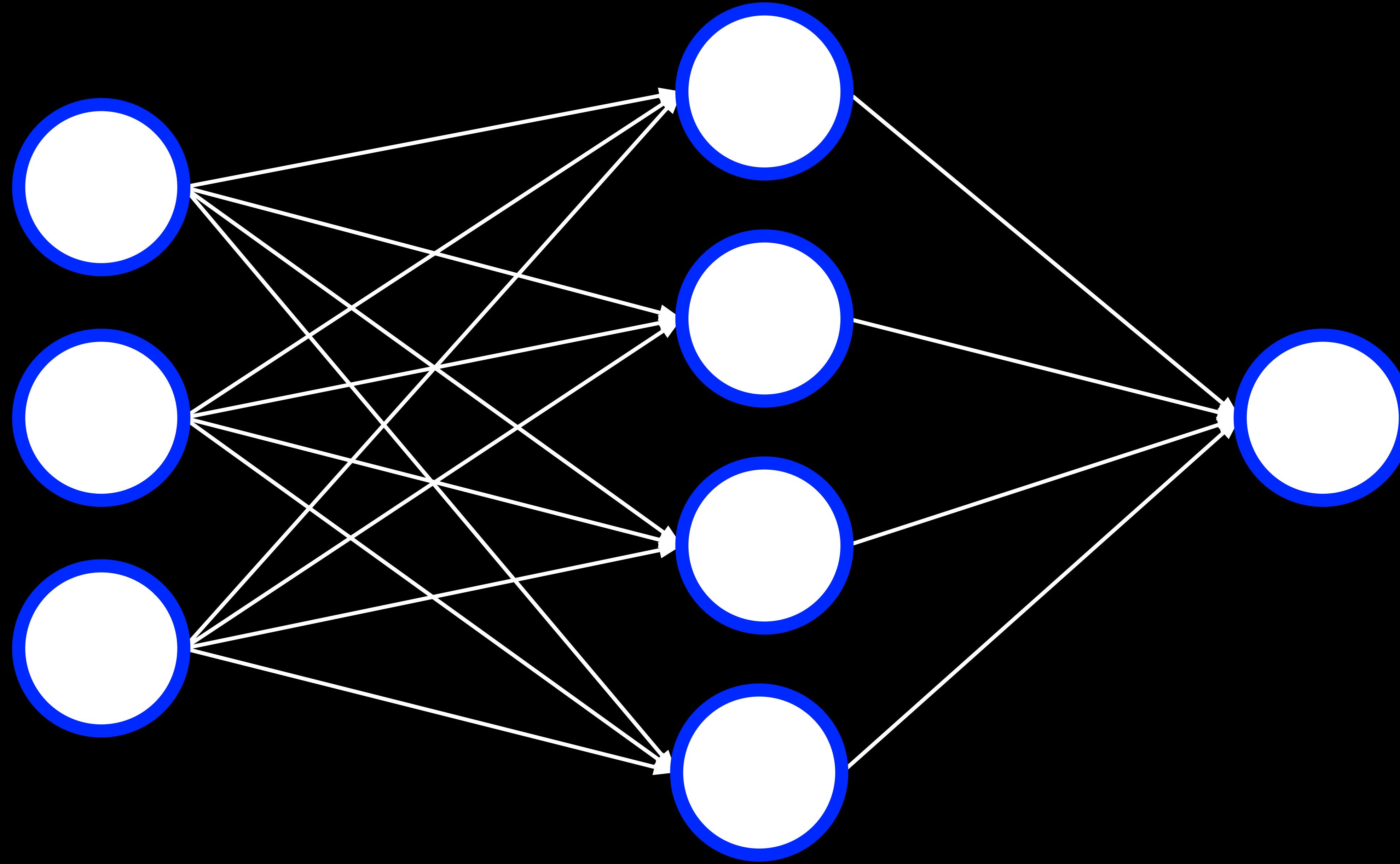
Perceptron

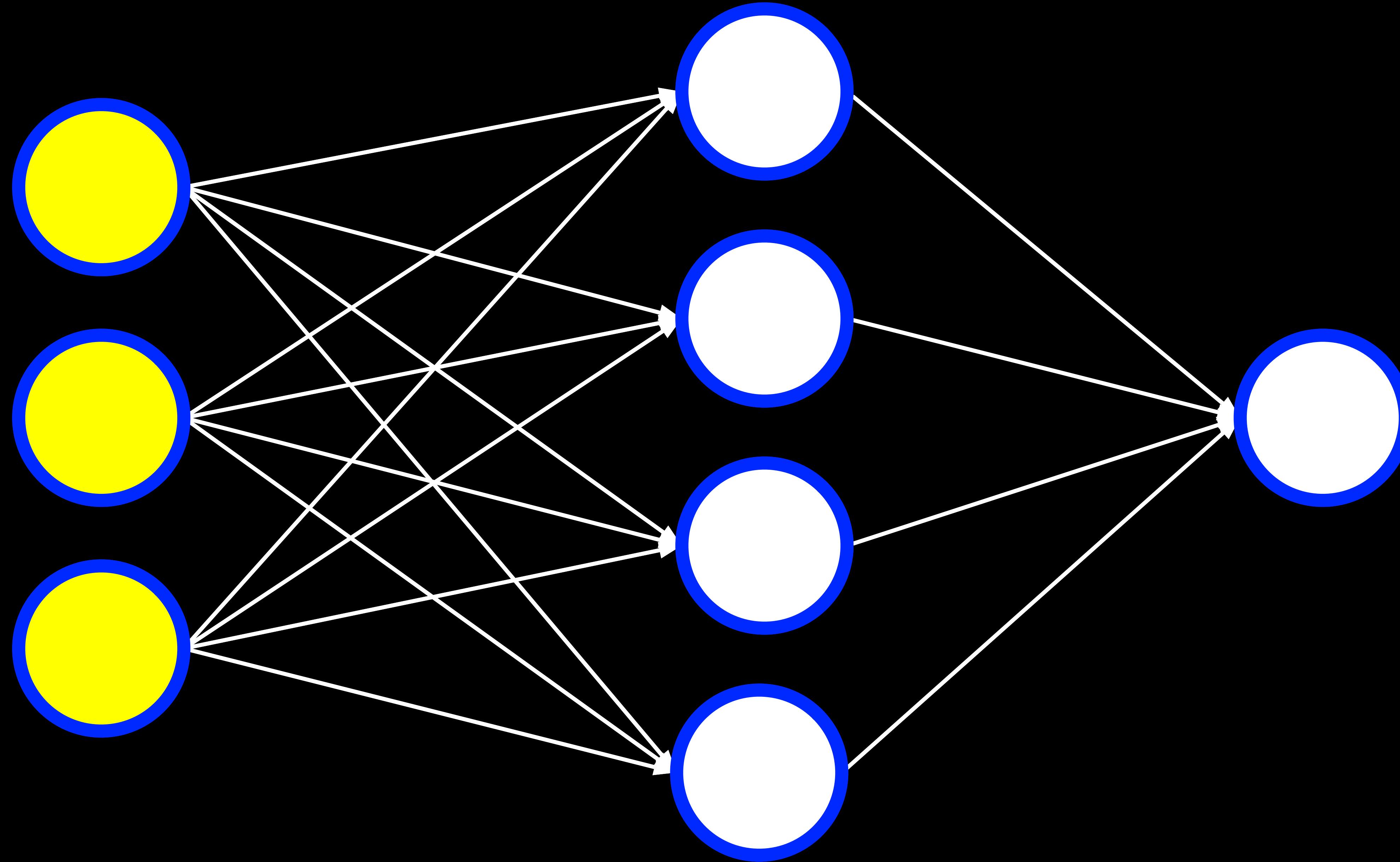
- Only capable of learning linearly separable decision boundary.

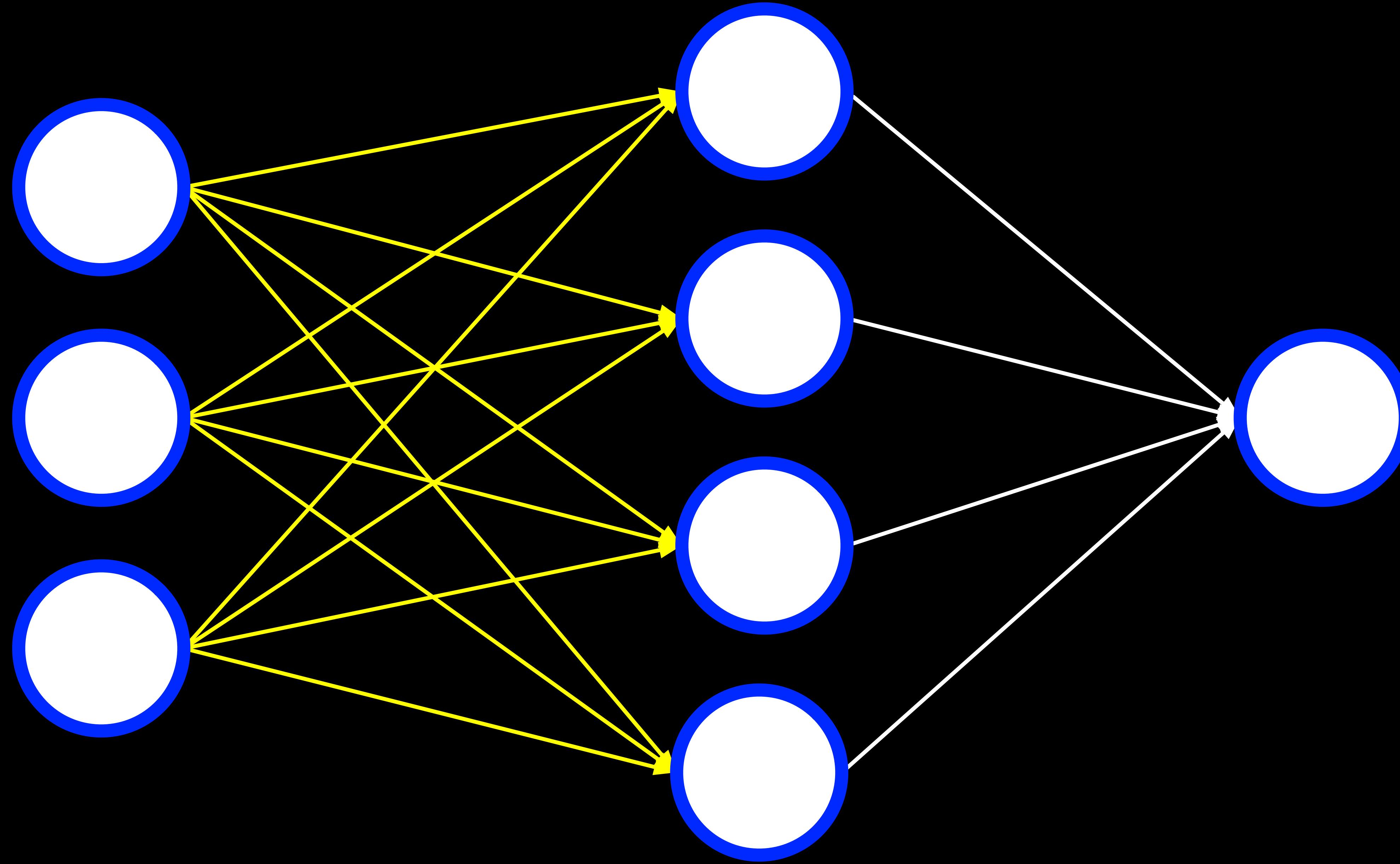
multilayer neural network

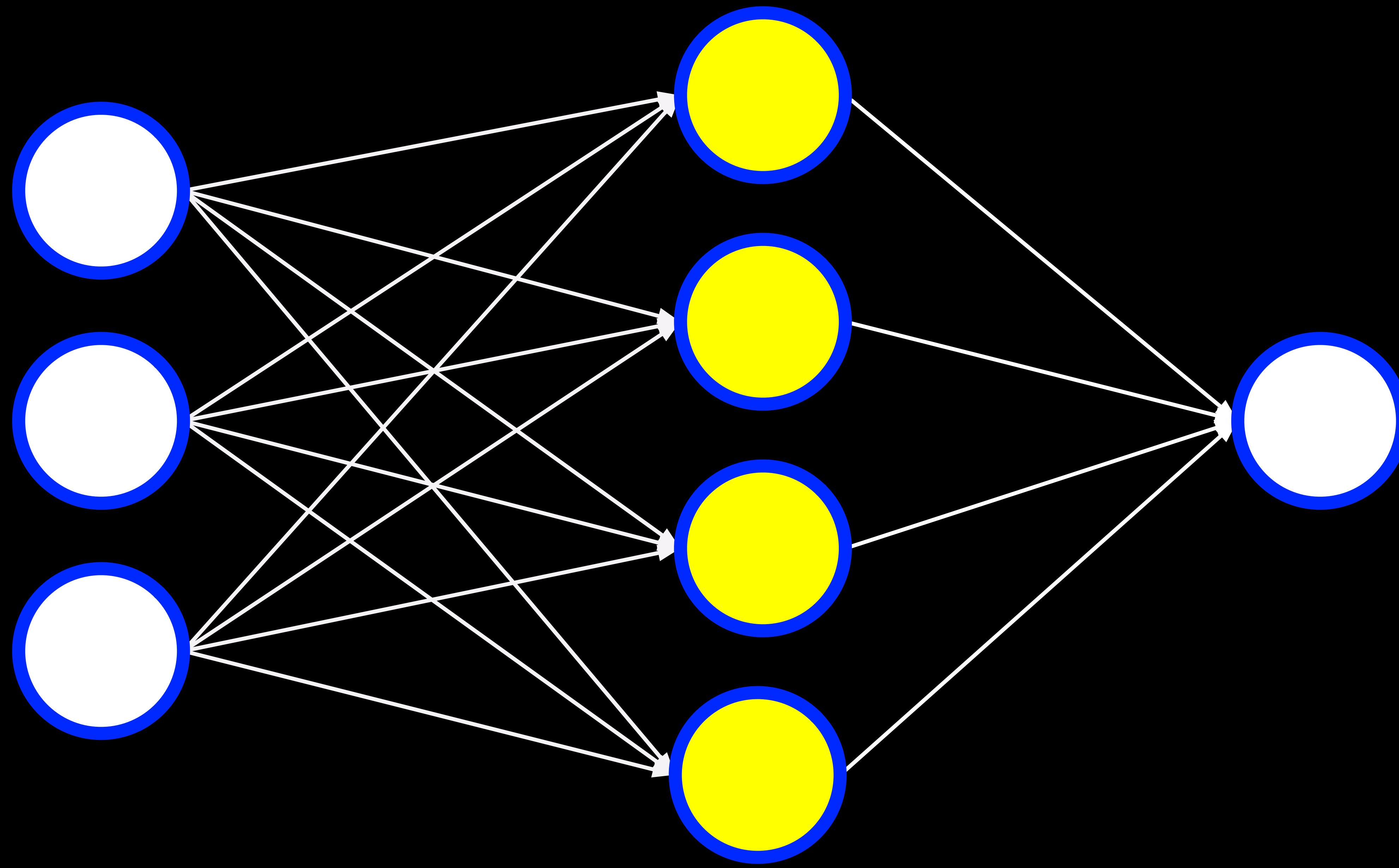
artificial neural network with an input layer,
an output layer, and at least one hidden layer

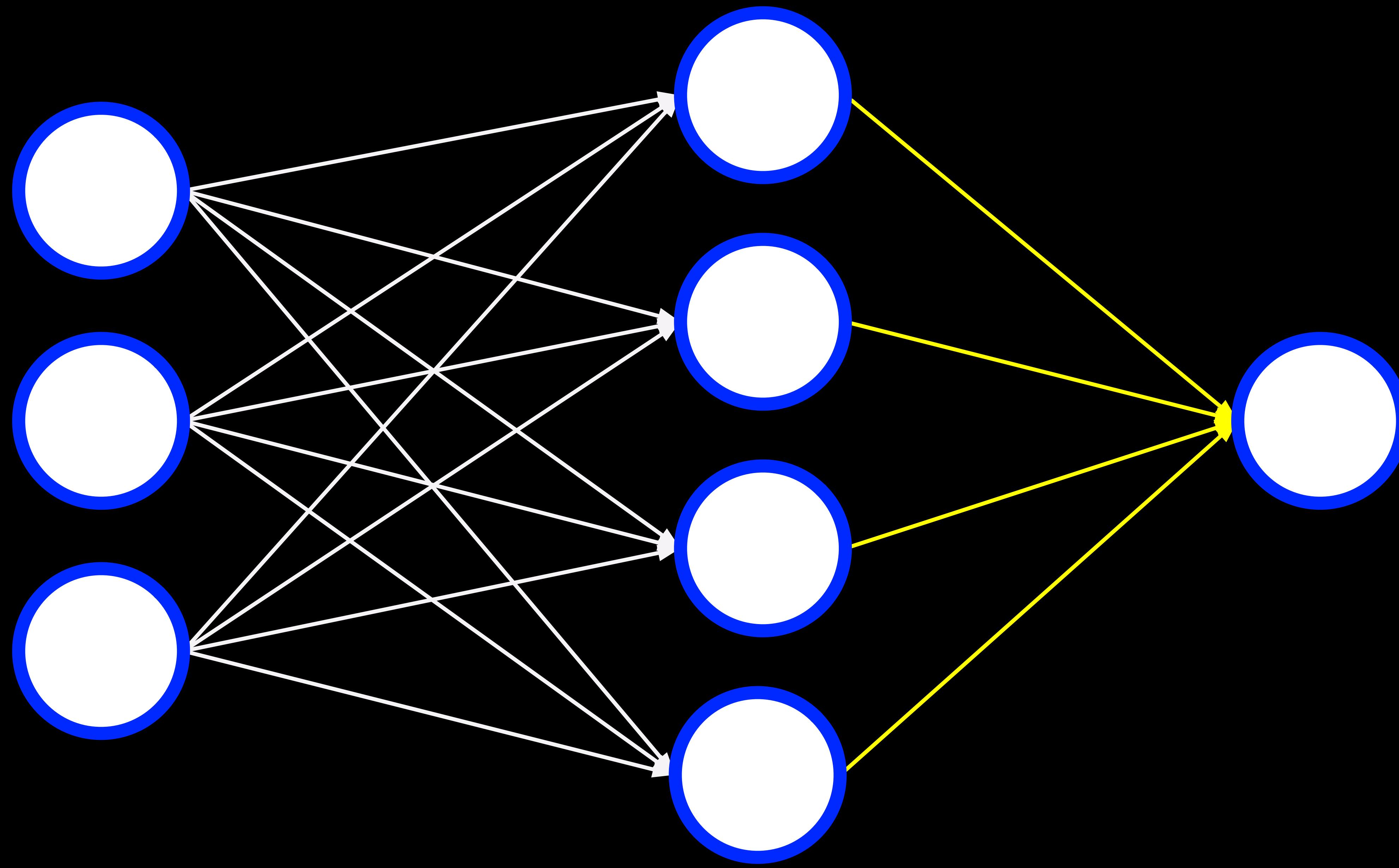


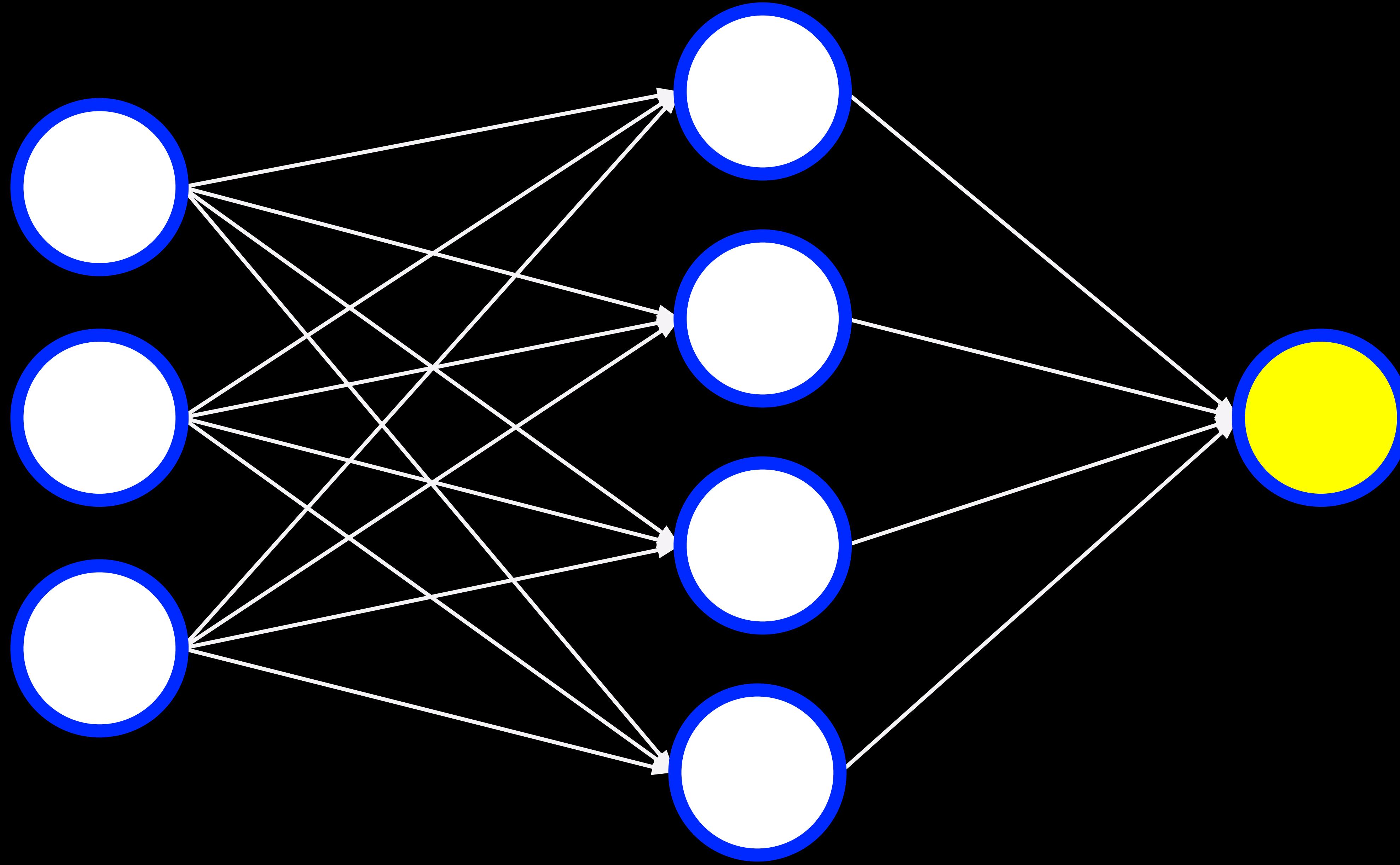


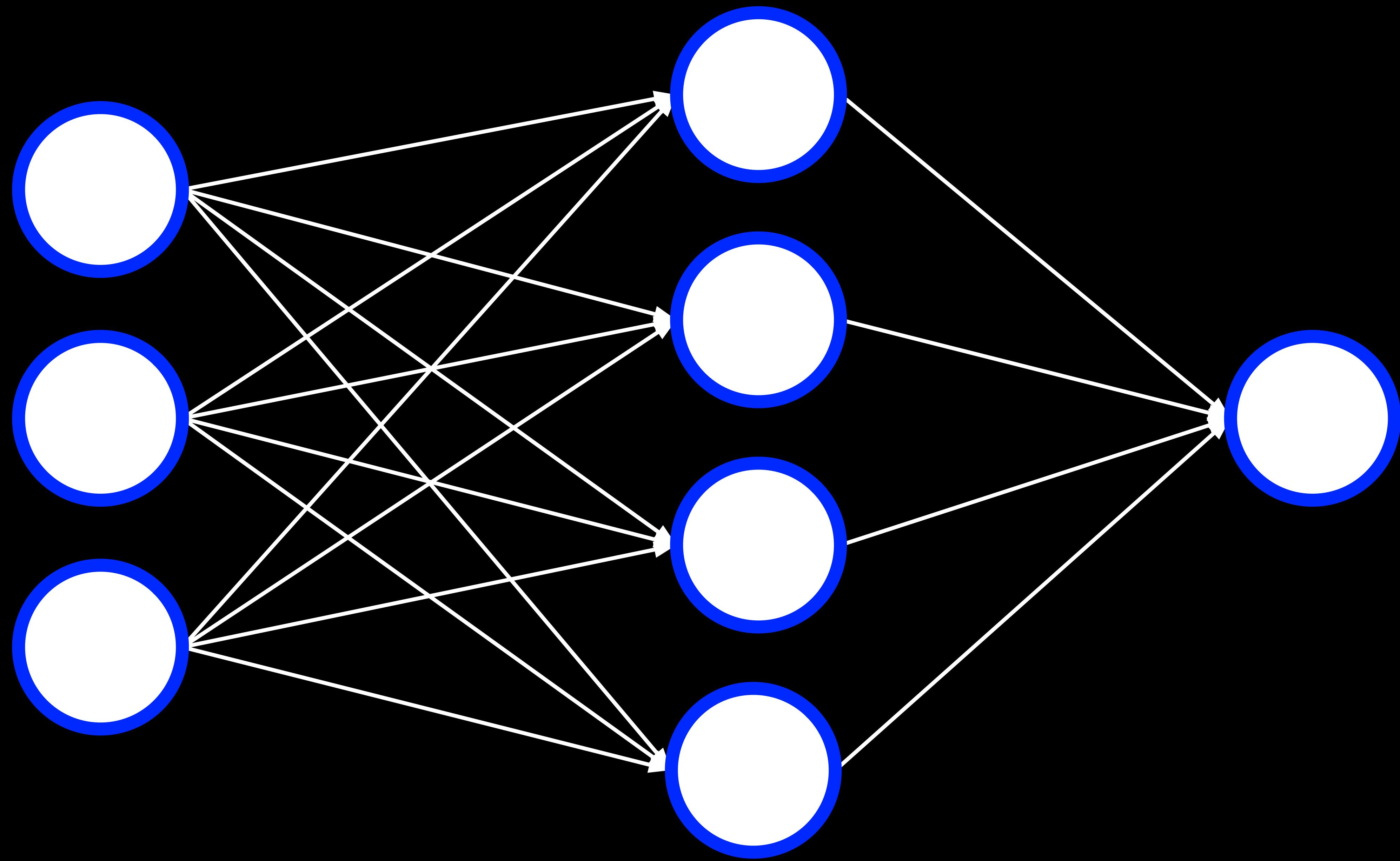










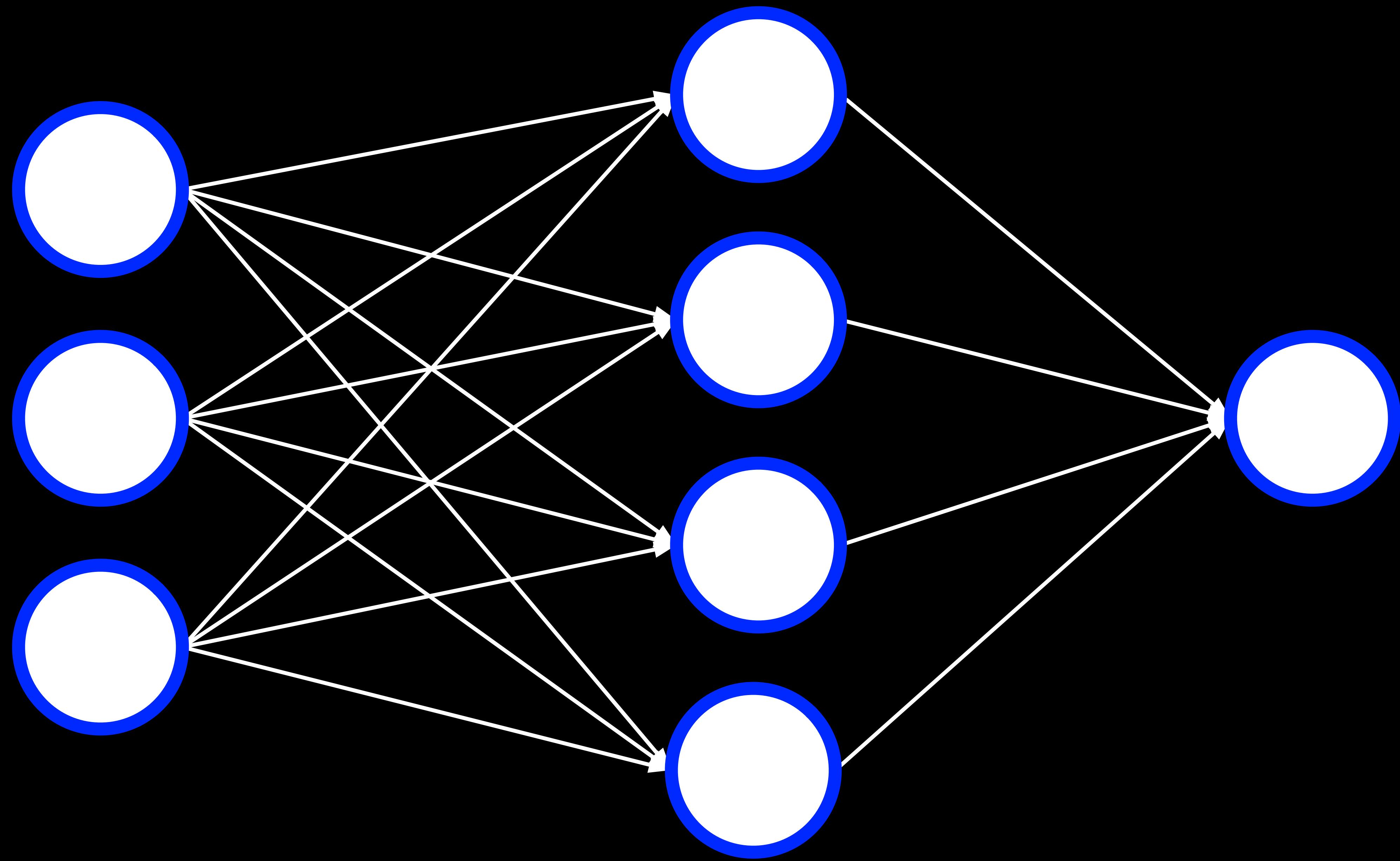


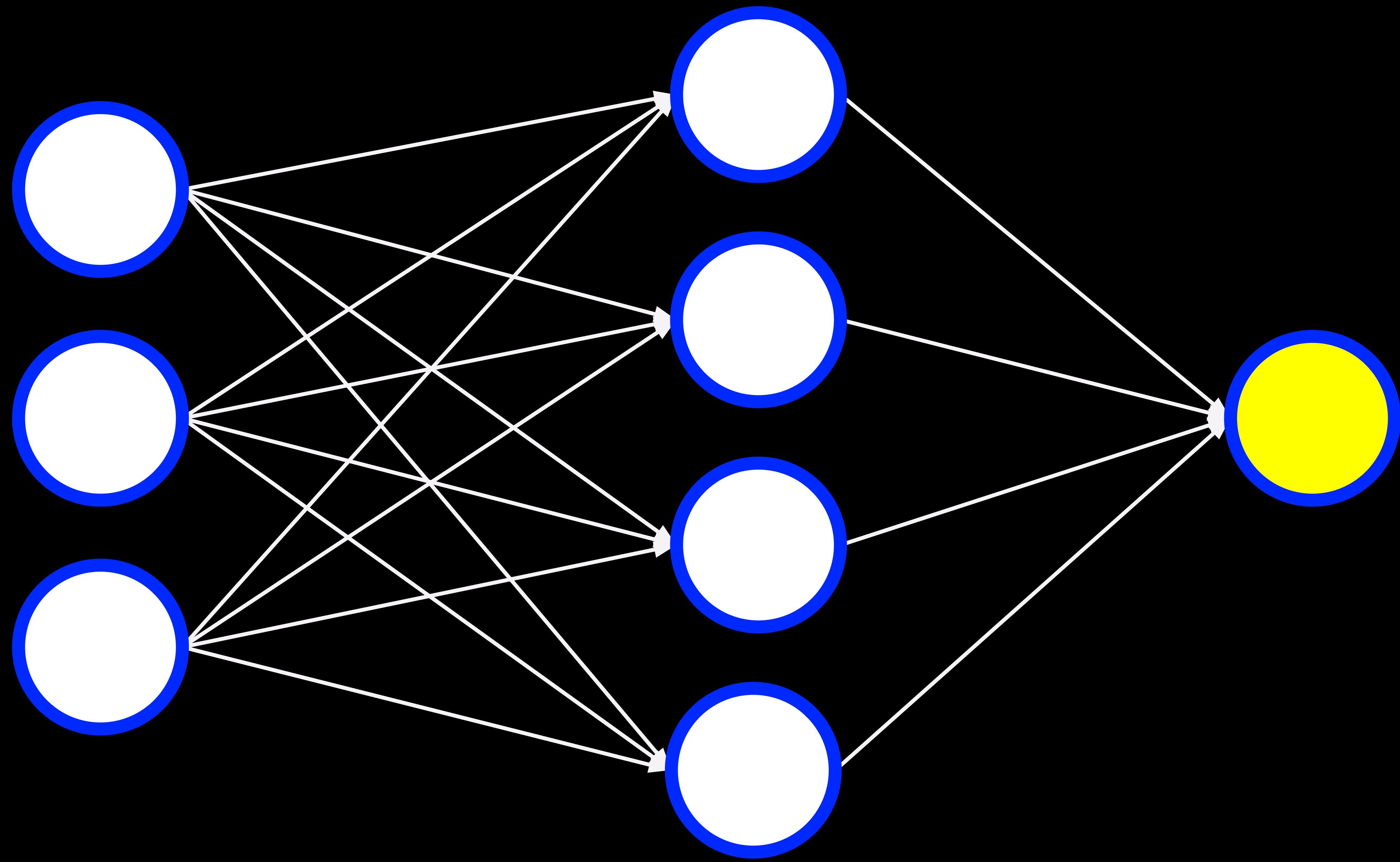
backpropagation

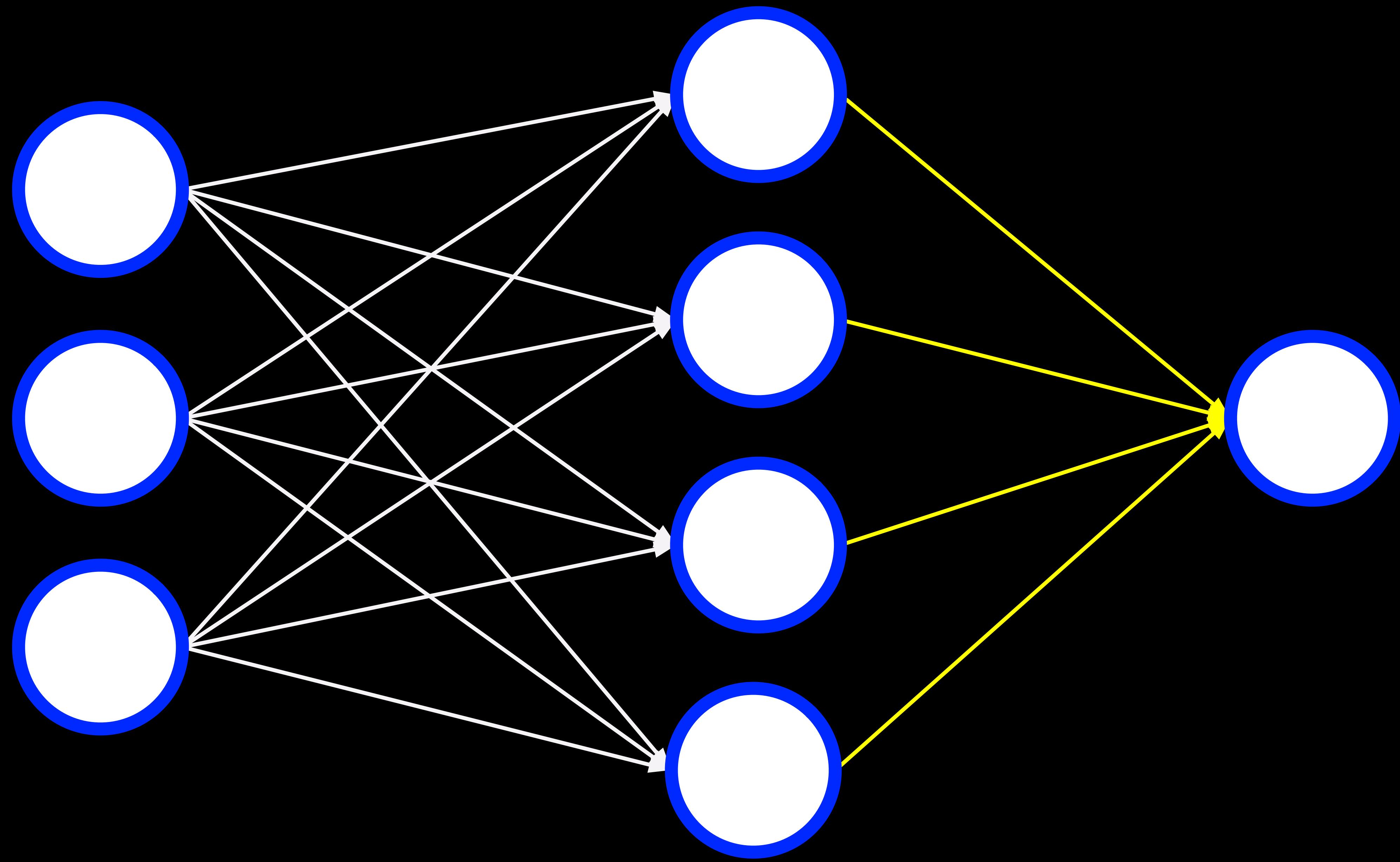
algorithm for training neural networks with
hidden layers

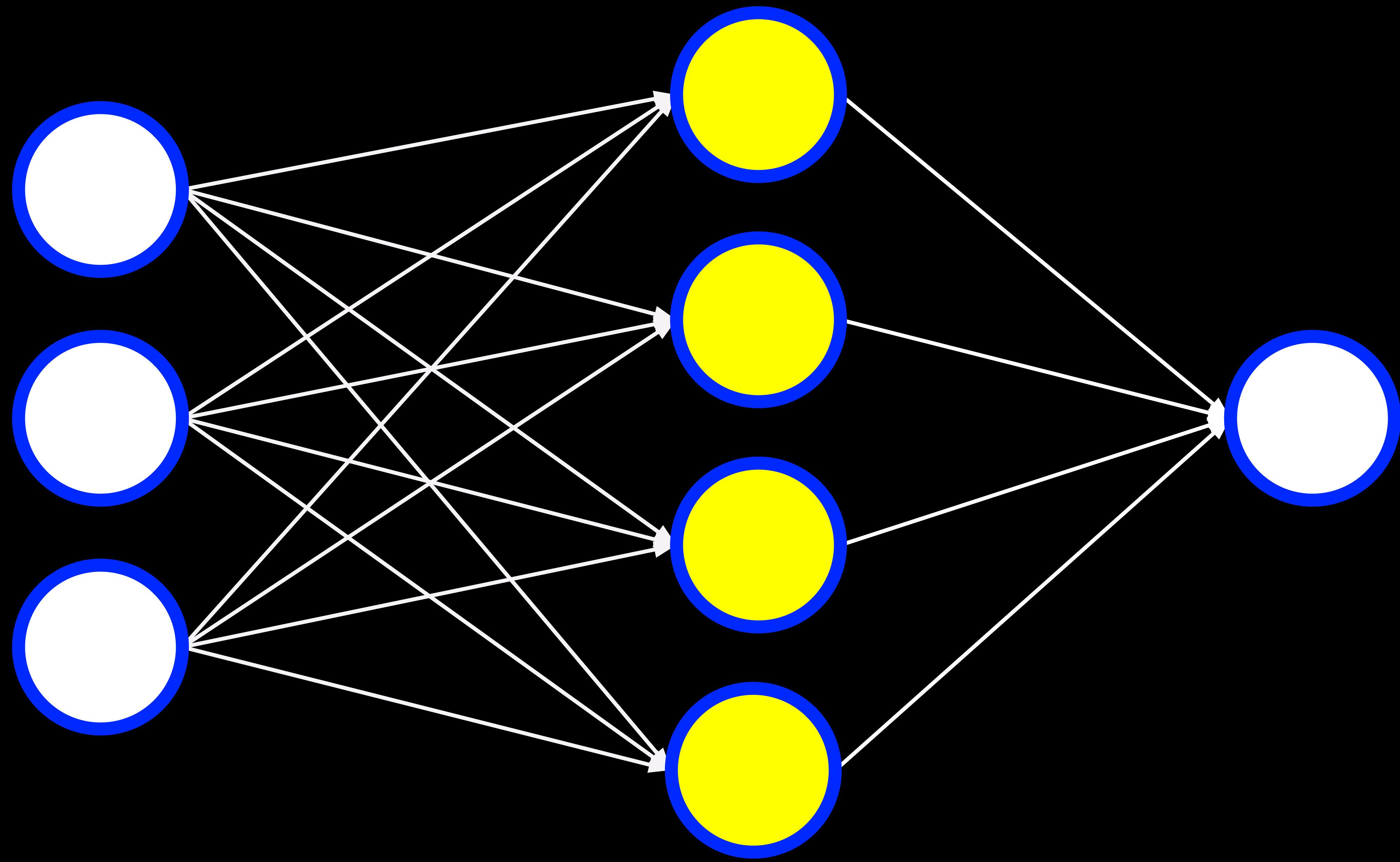
Backpropagation

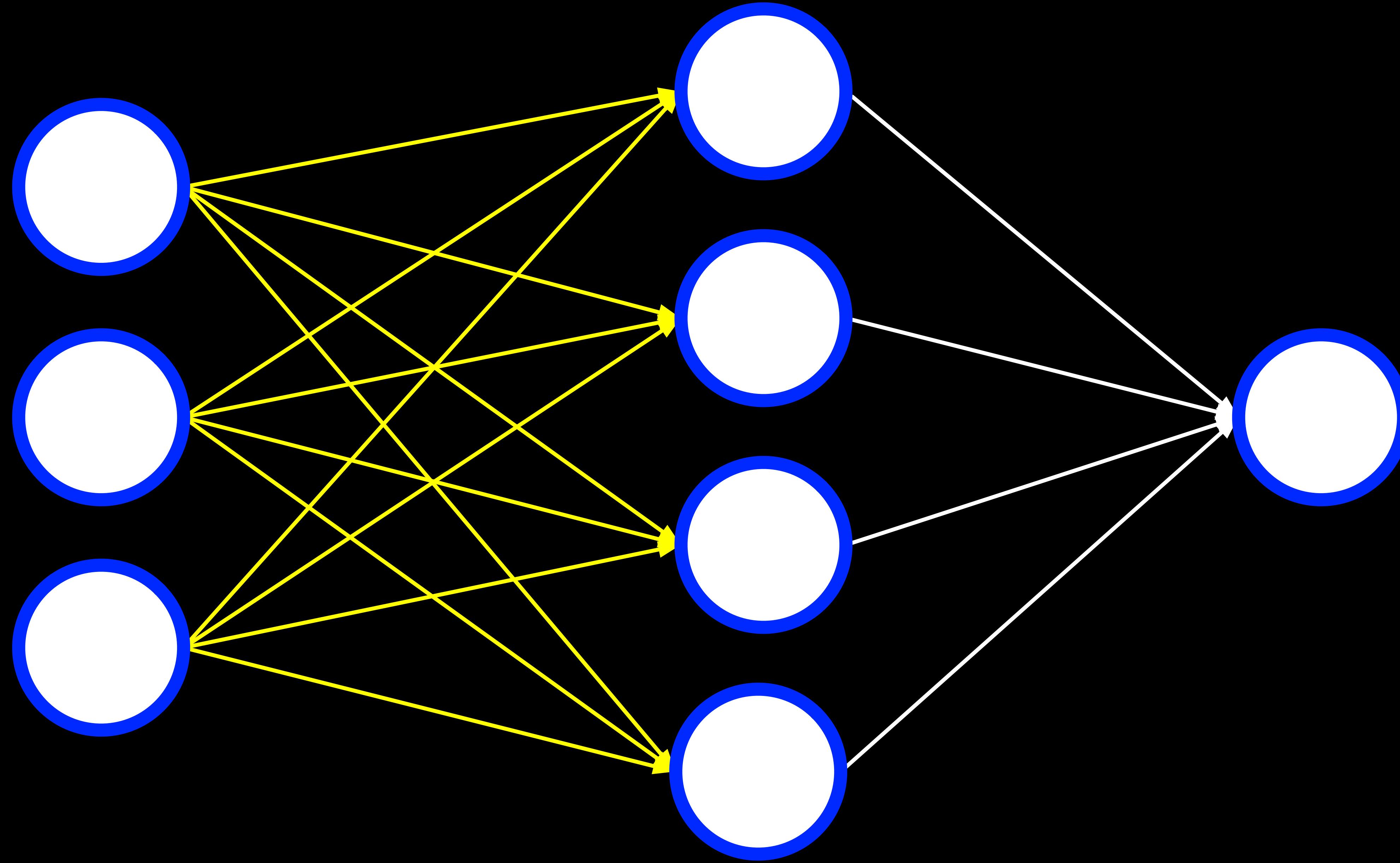
- Start with a random choice of weights.
- Repeat:
 - Calculate error for output layer.
 - For each layer, starting with output layer, and moving inwards towards earliest hidden layer:
 - Propagate error back one layer.
 - Update weights.

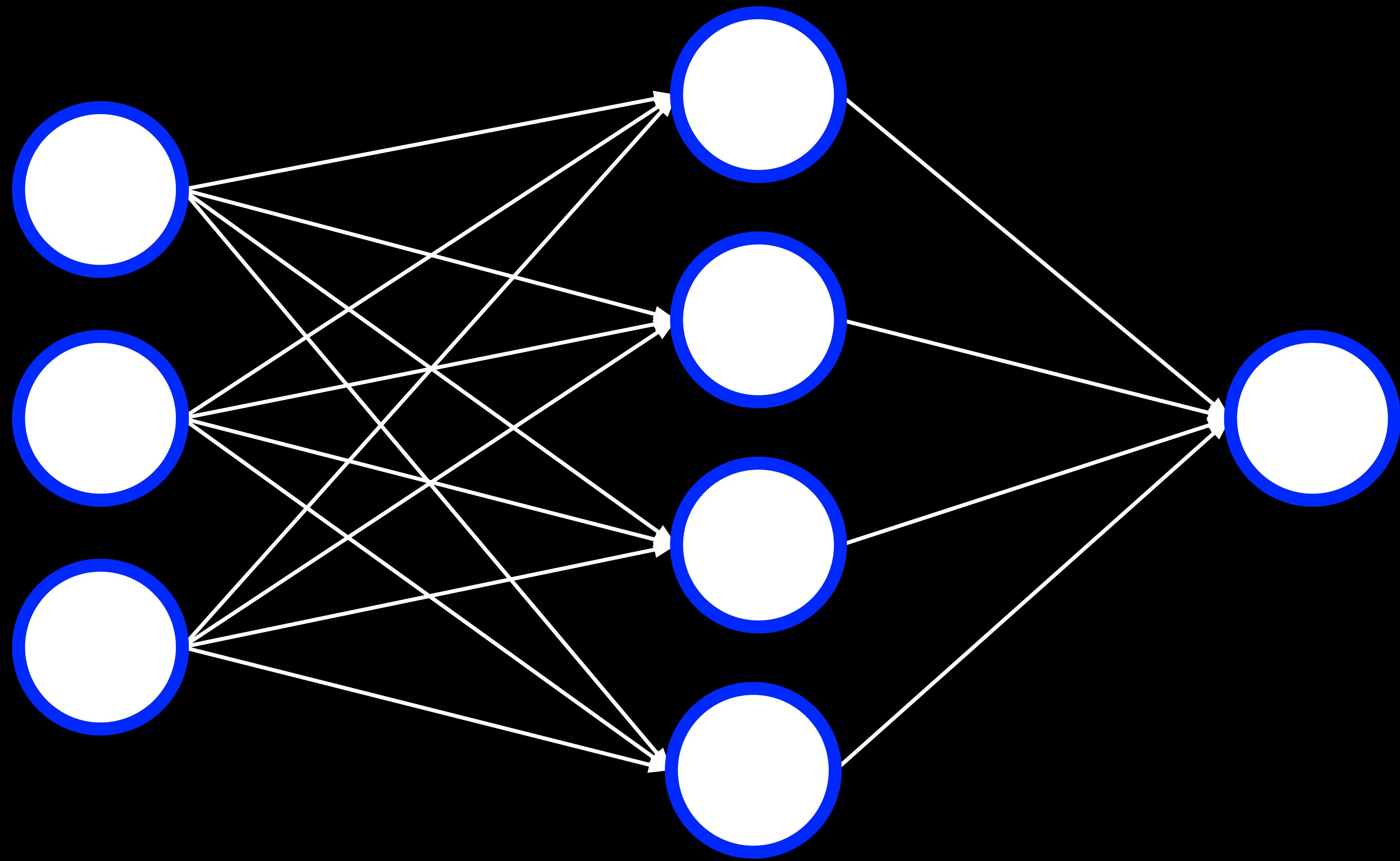


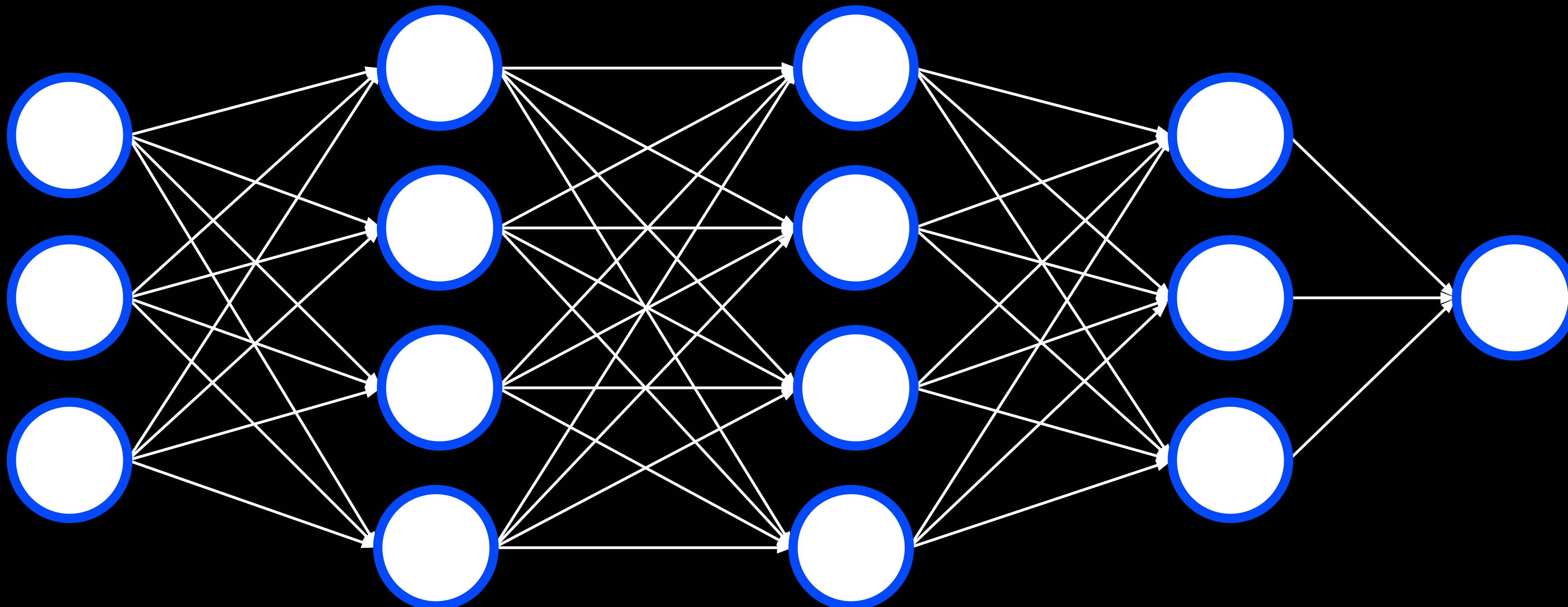






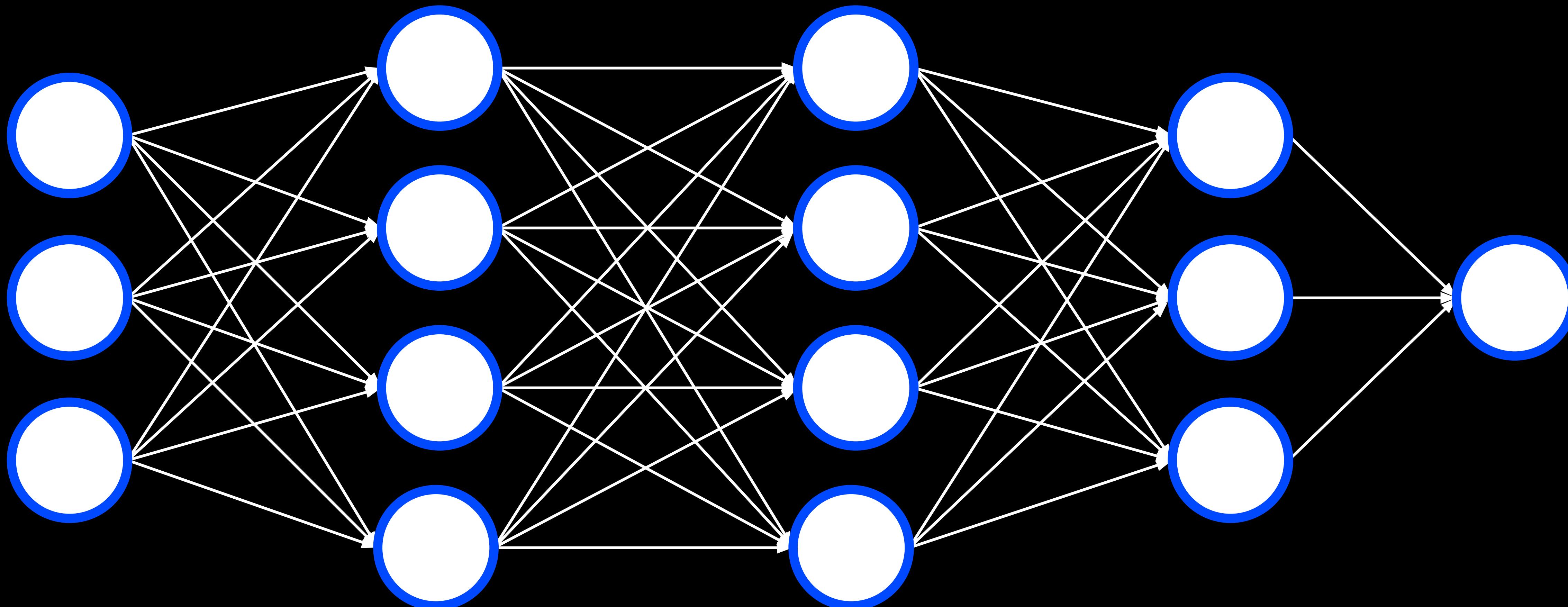






deep neural networks

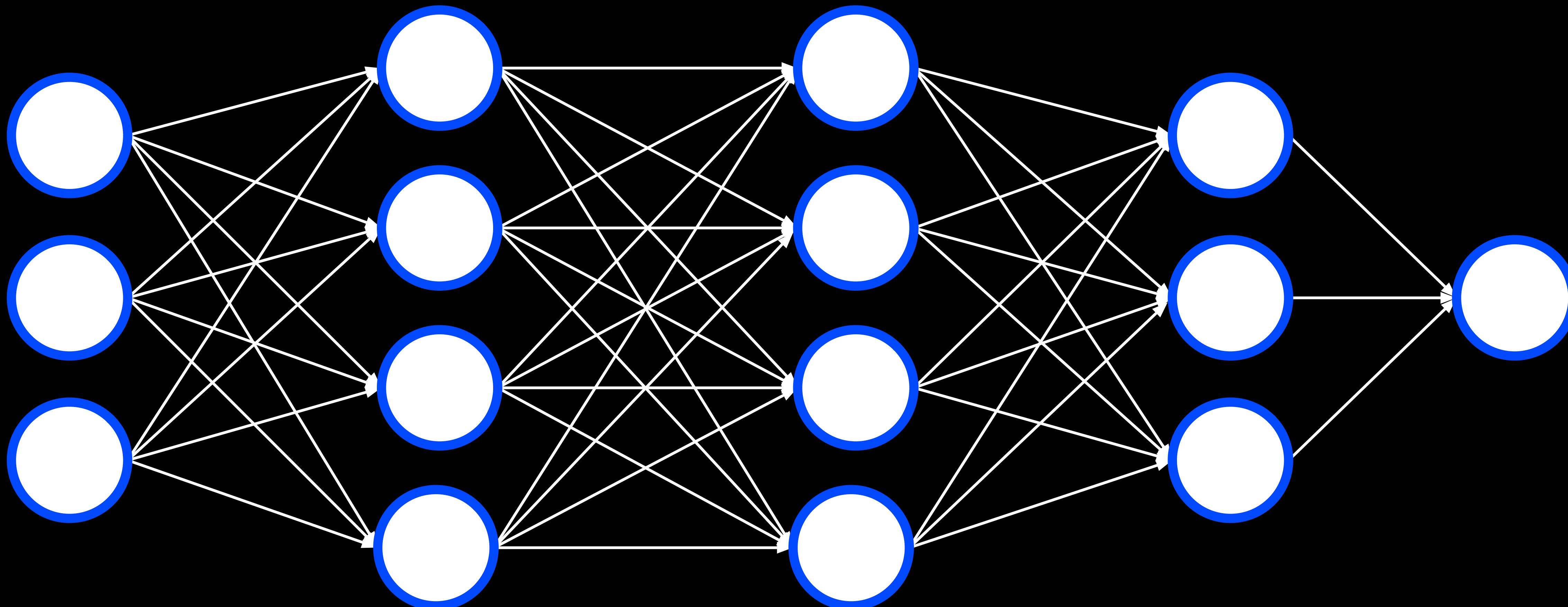
neural network with multiple hidden layers

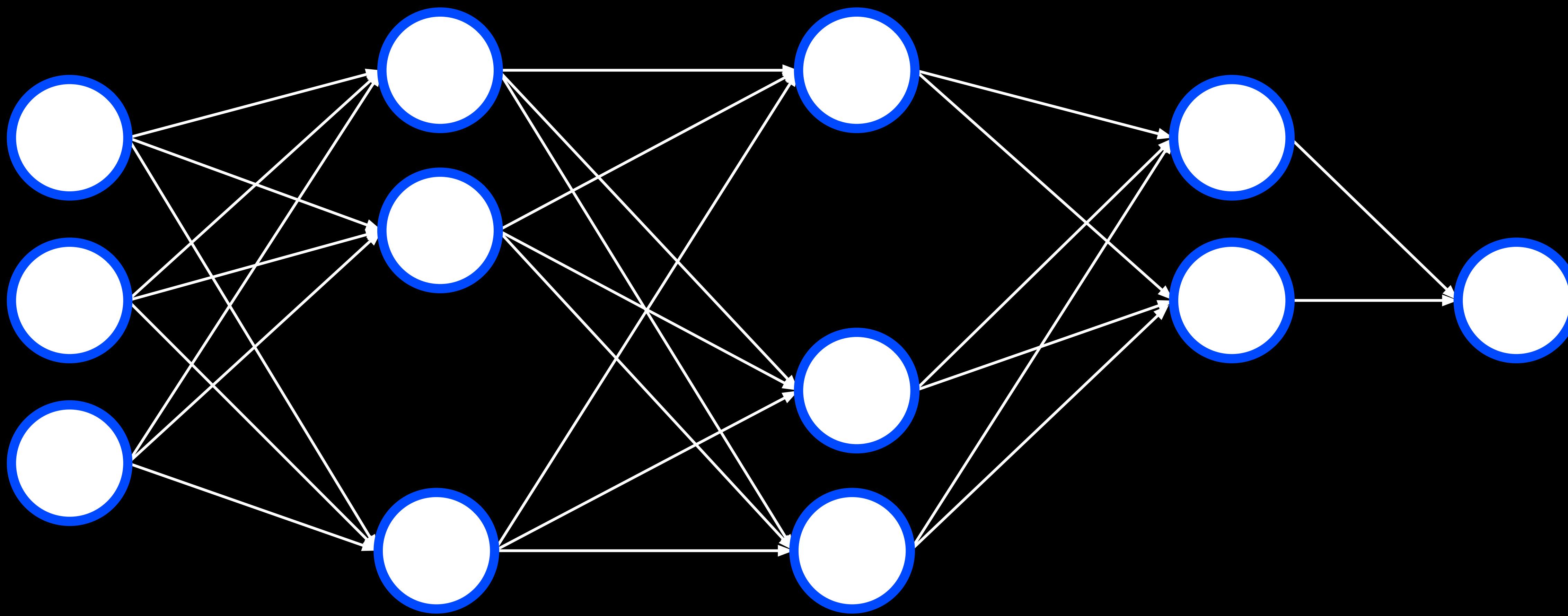


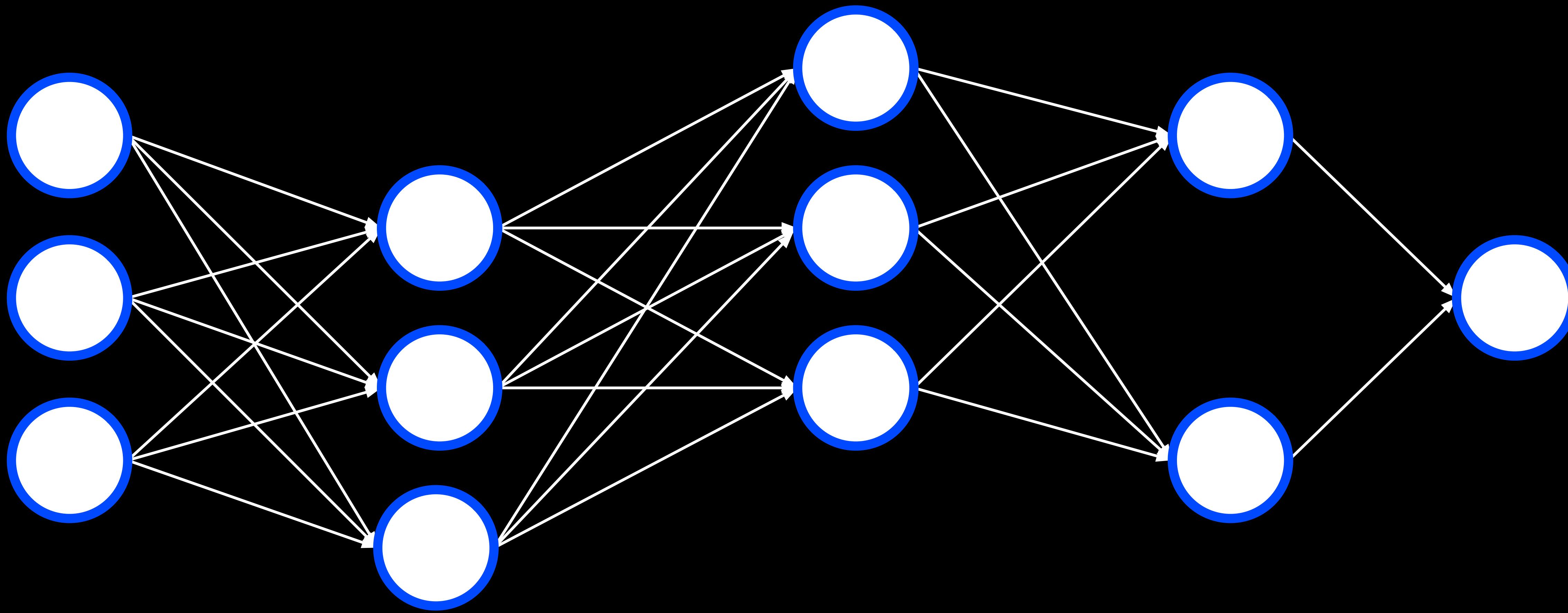
Overfitting

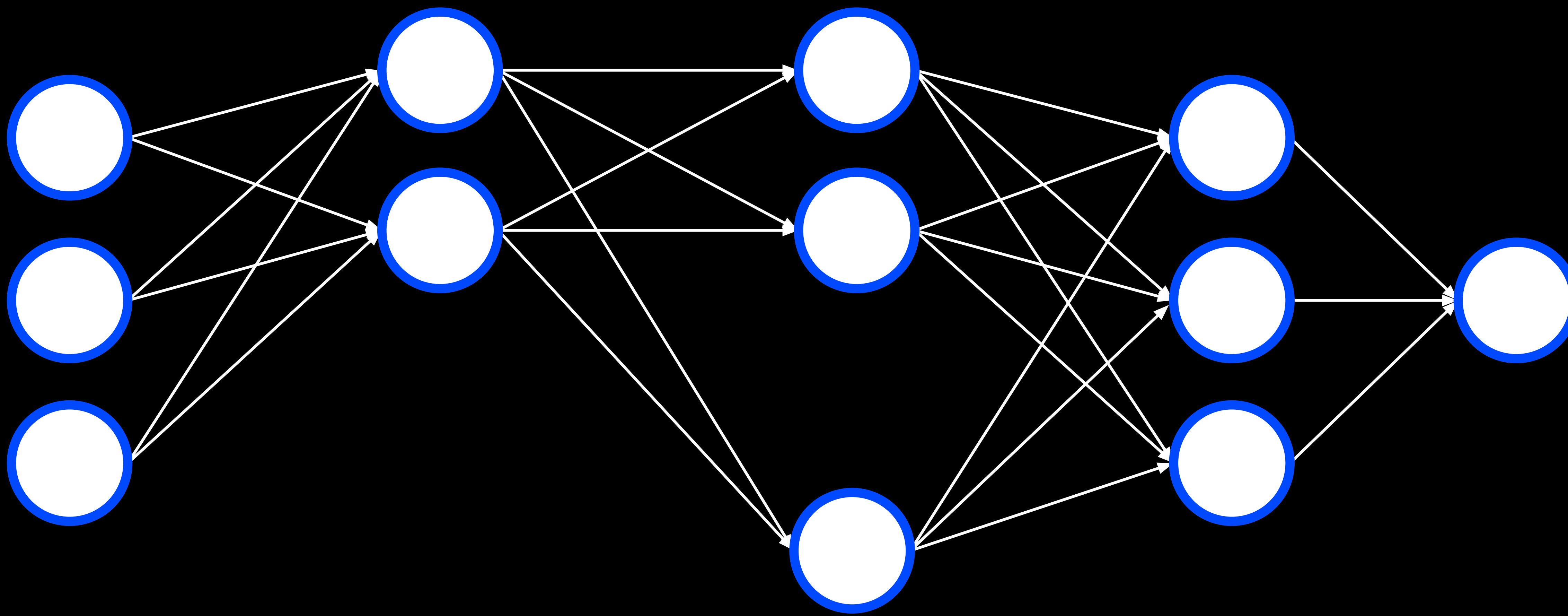
dropout

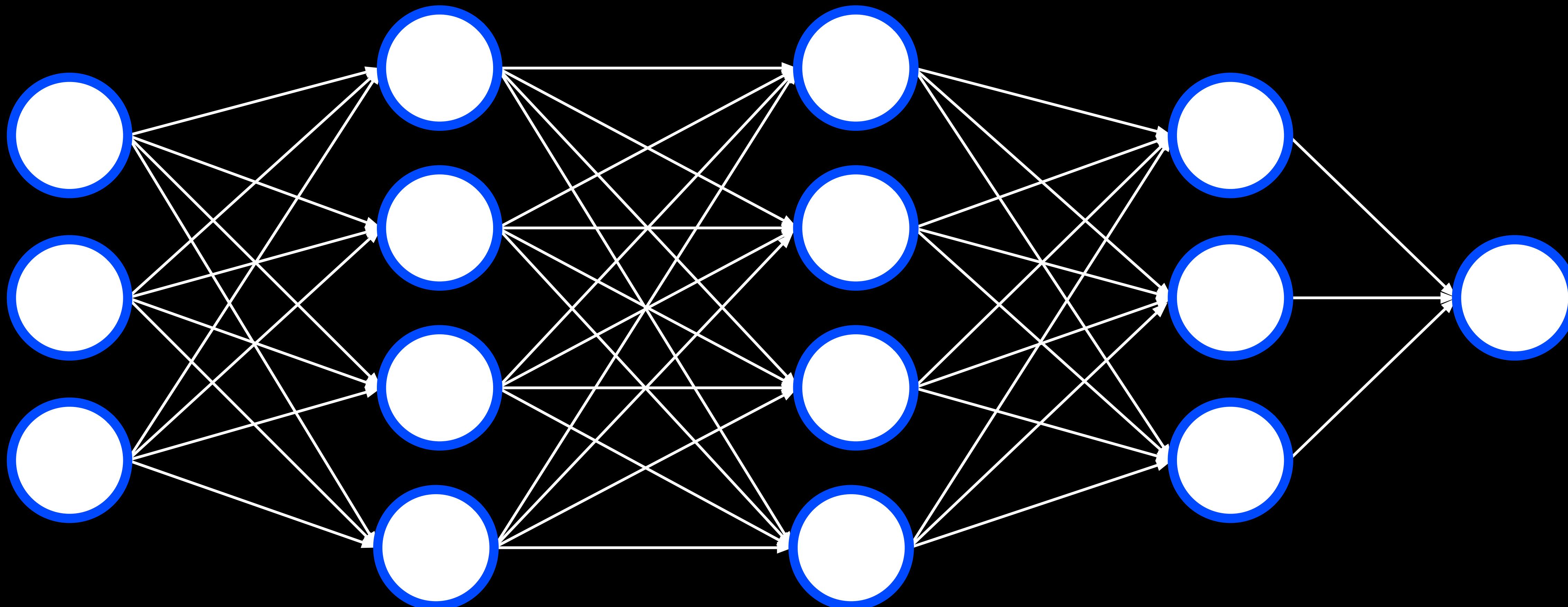
temporarily removing units – selected at random – from a neural network to prevent over-reliance on certain units











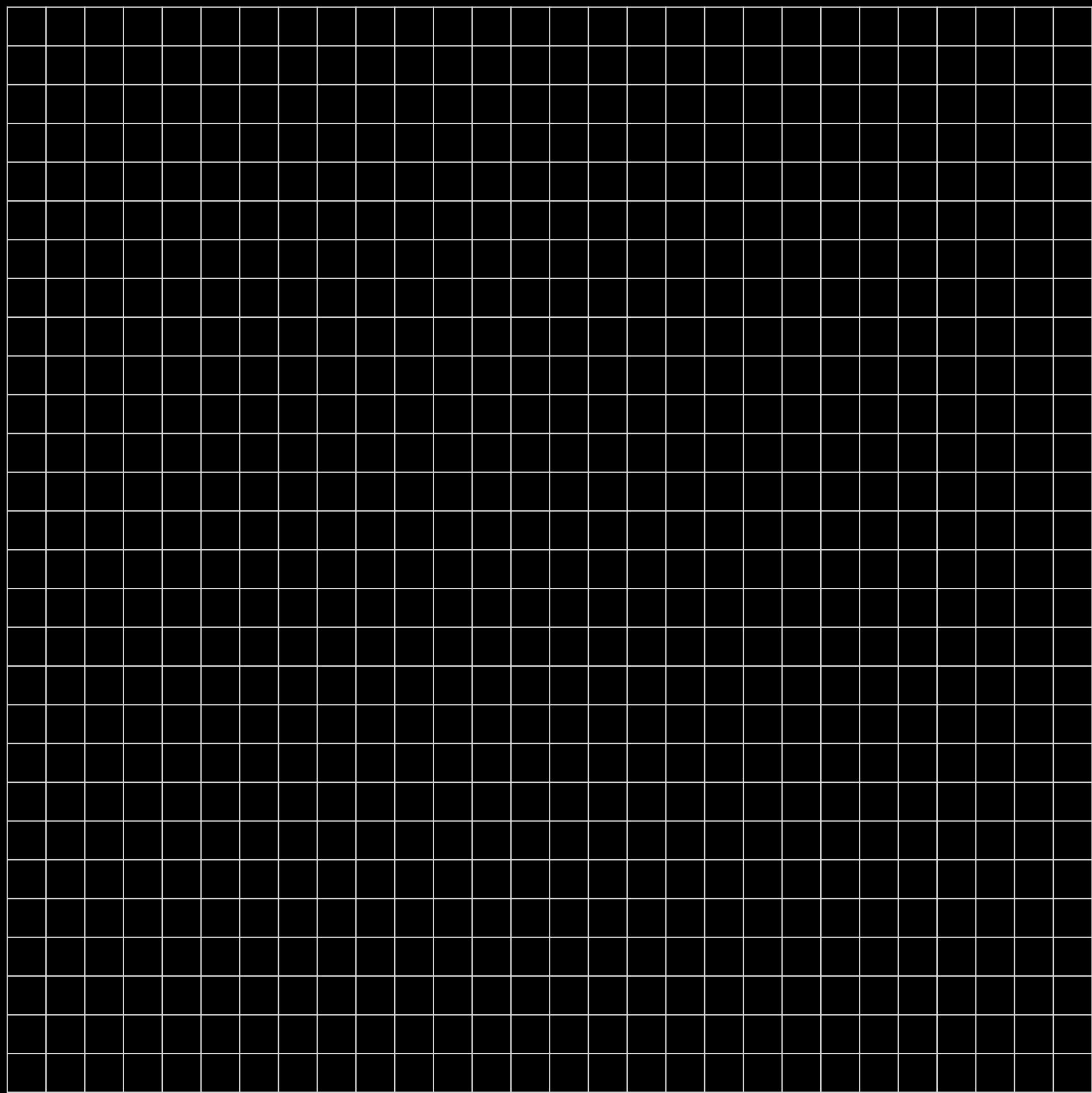
TensorFlow

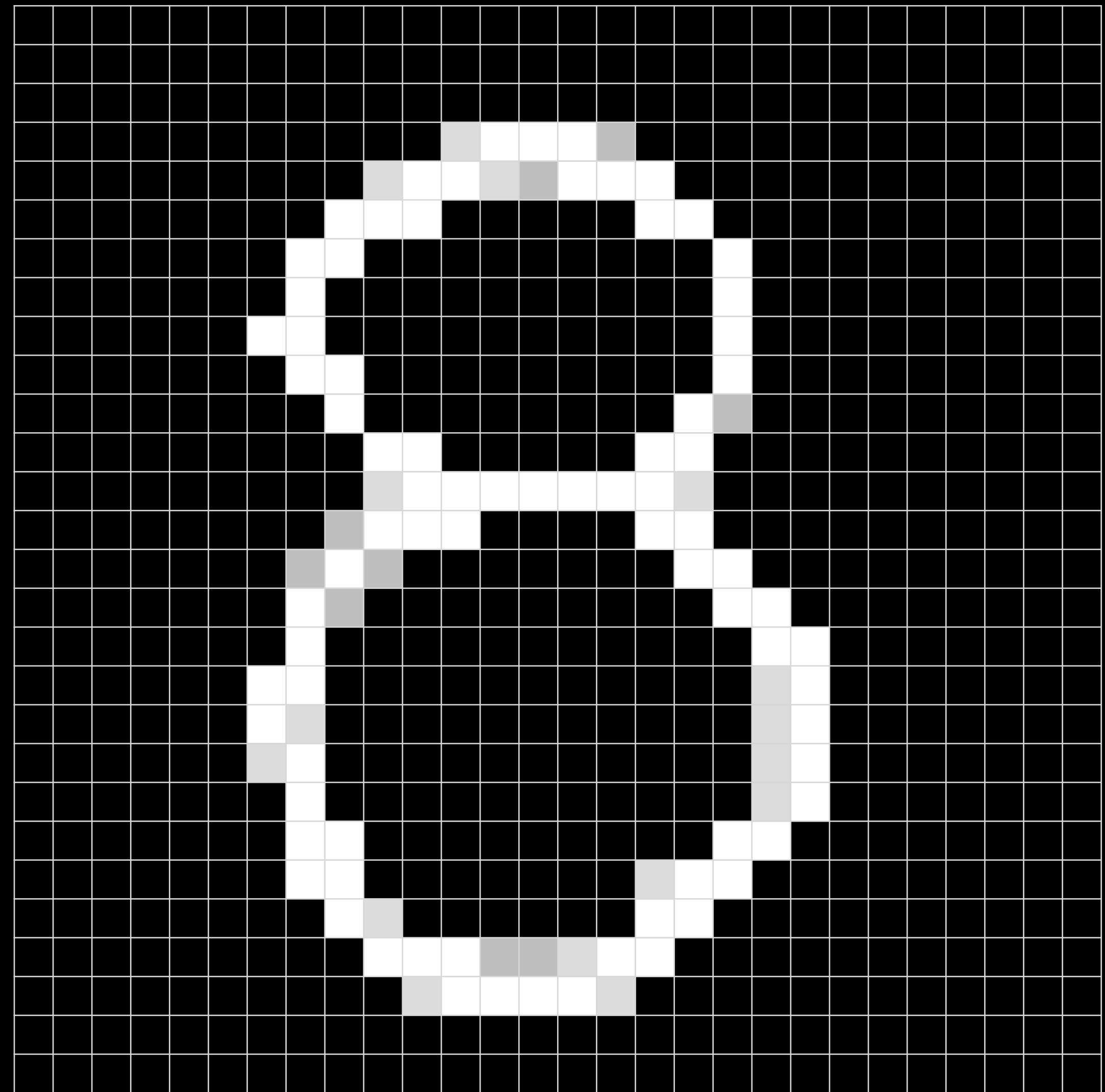
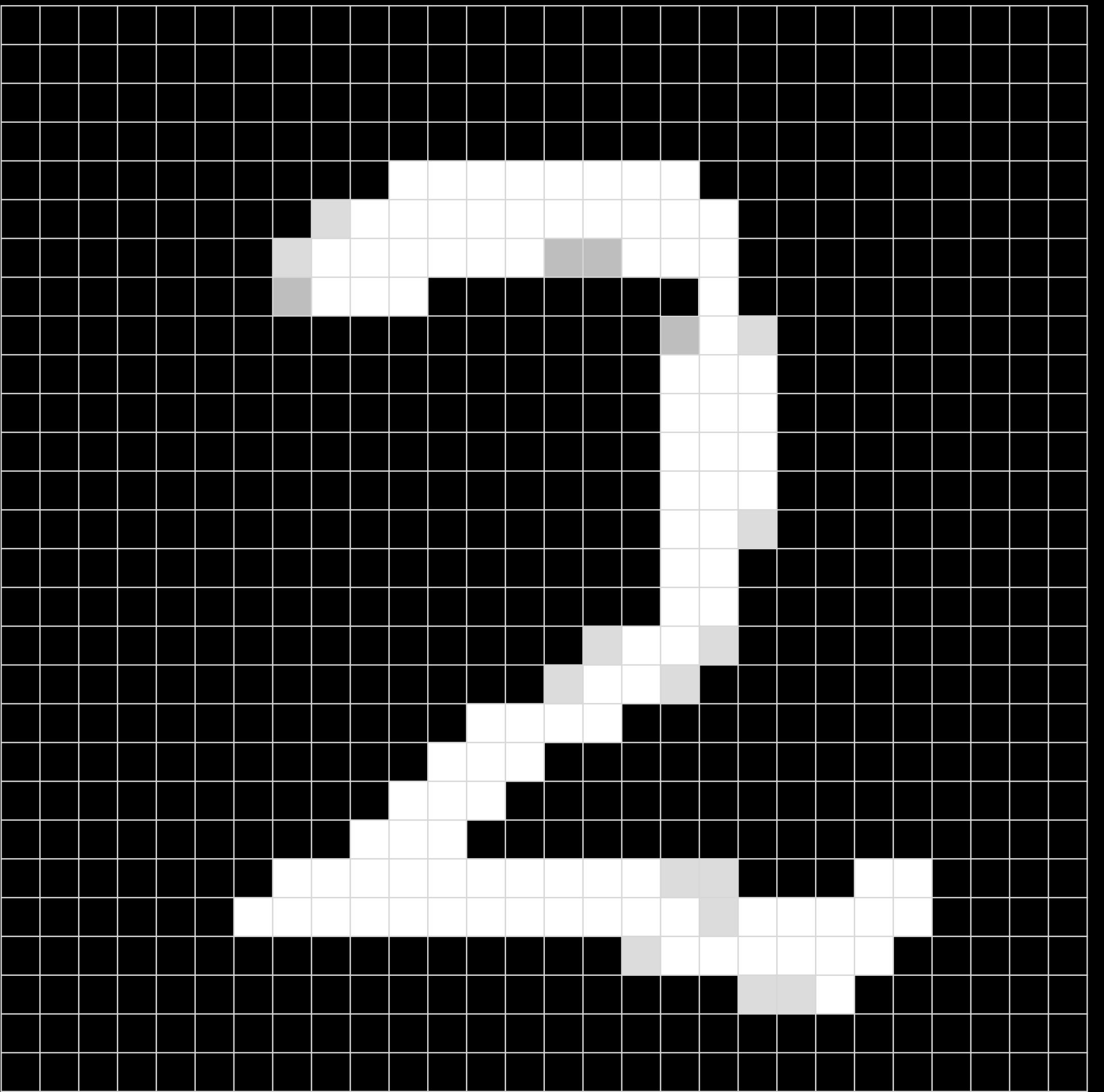
playground.tensorflow.org

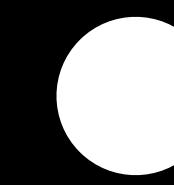
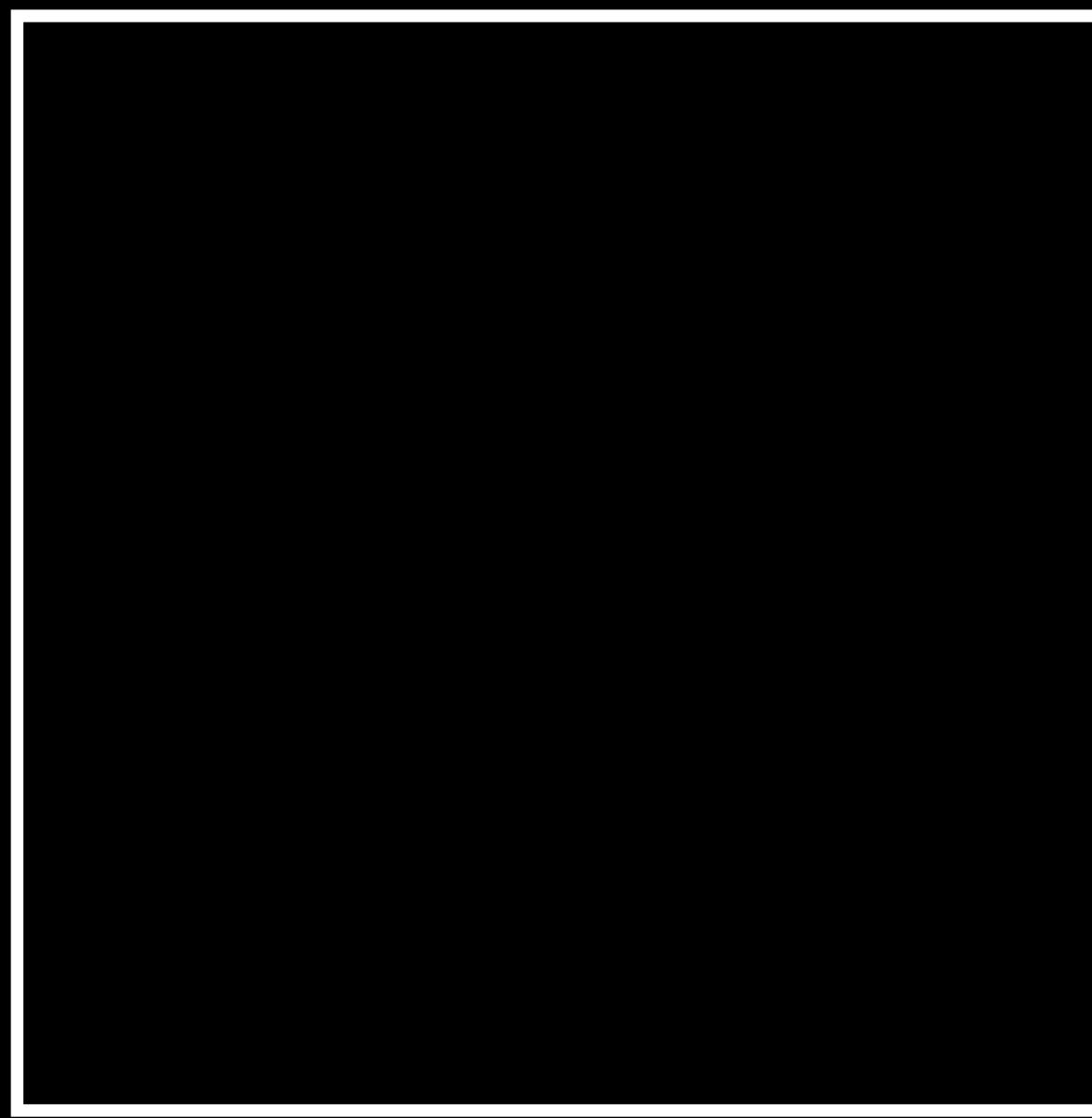
computer vision

computational methods for analyzing and
understanding digital images









0

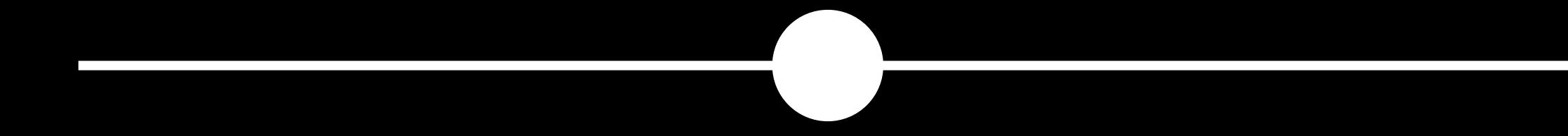
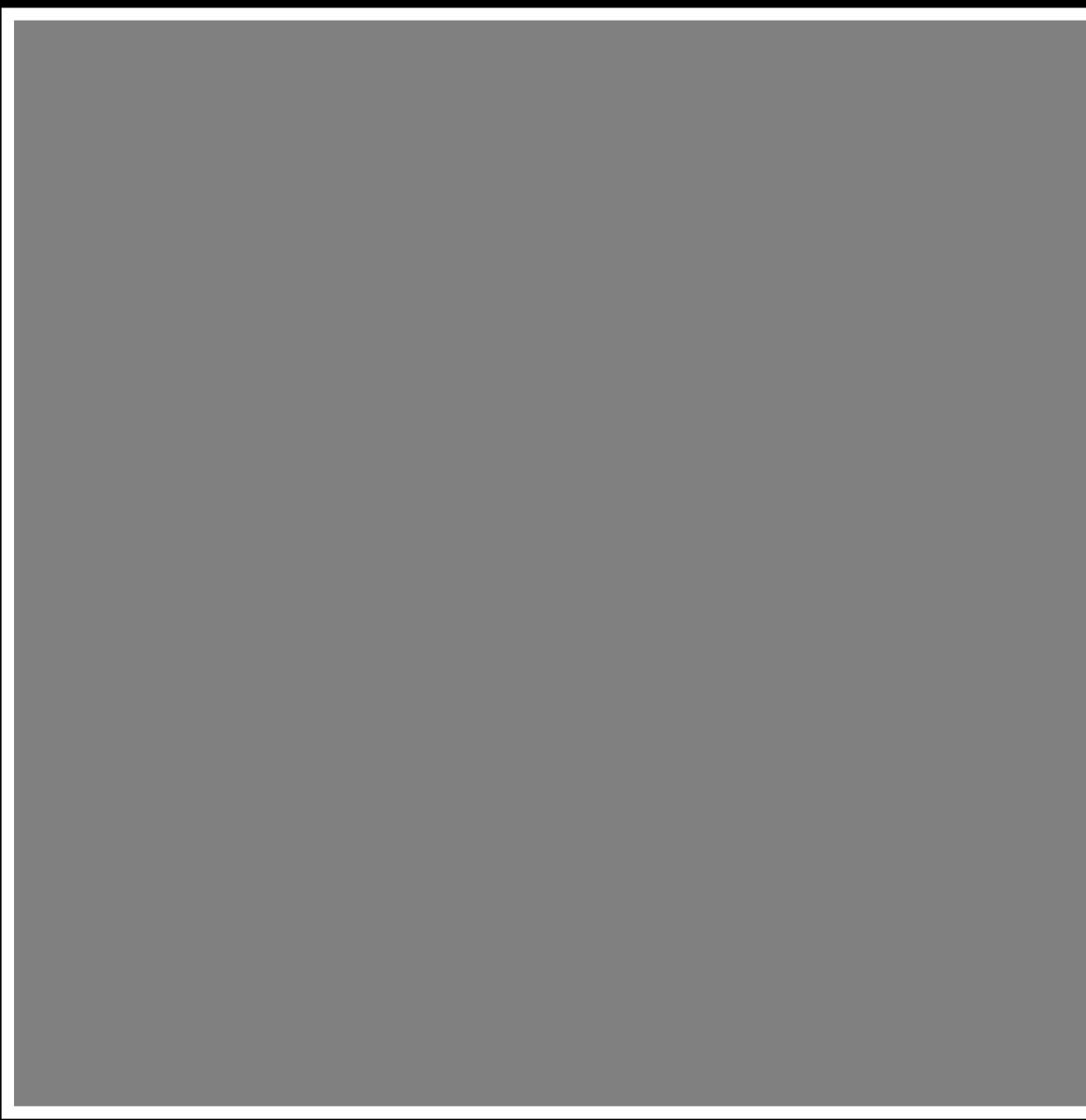
255



0

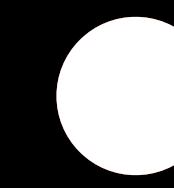
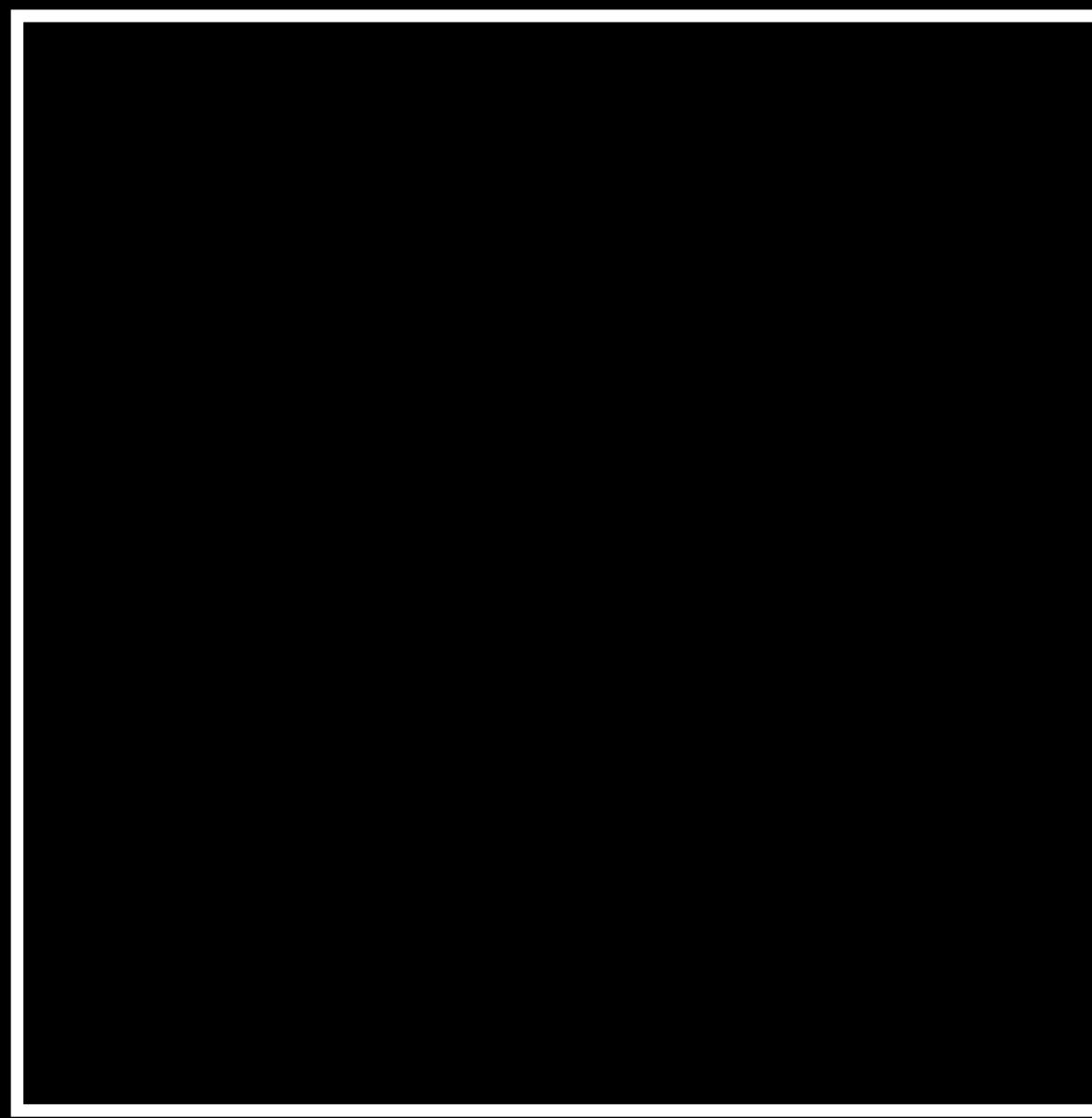


255



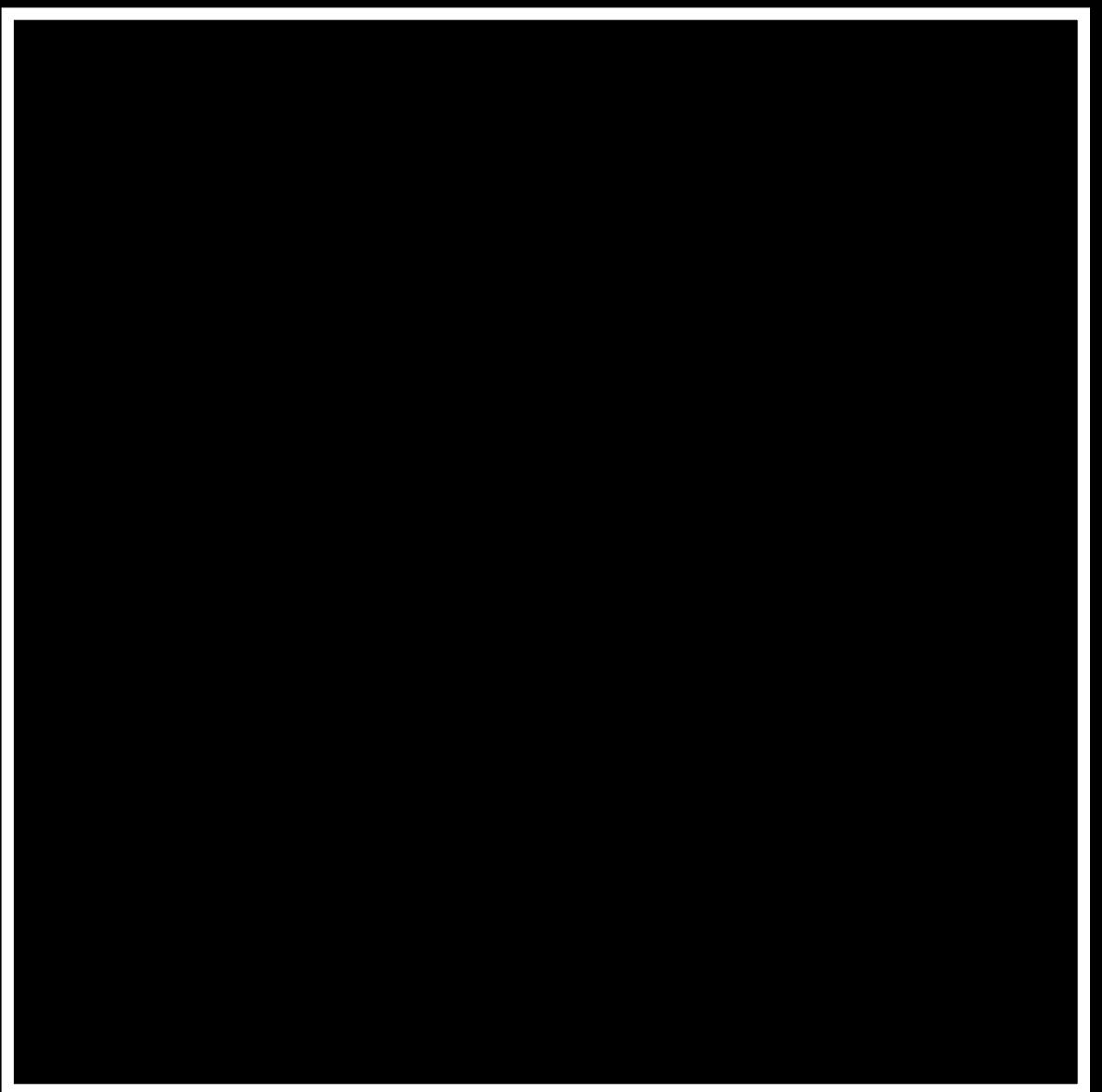
0

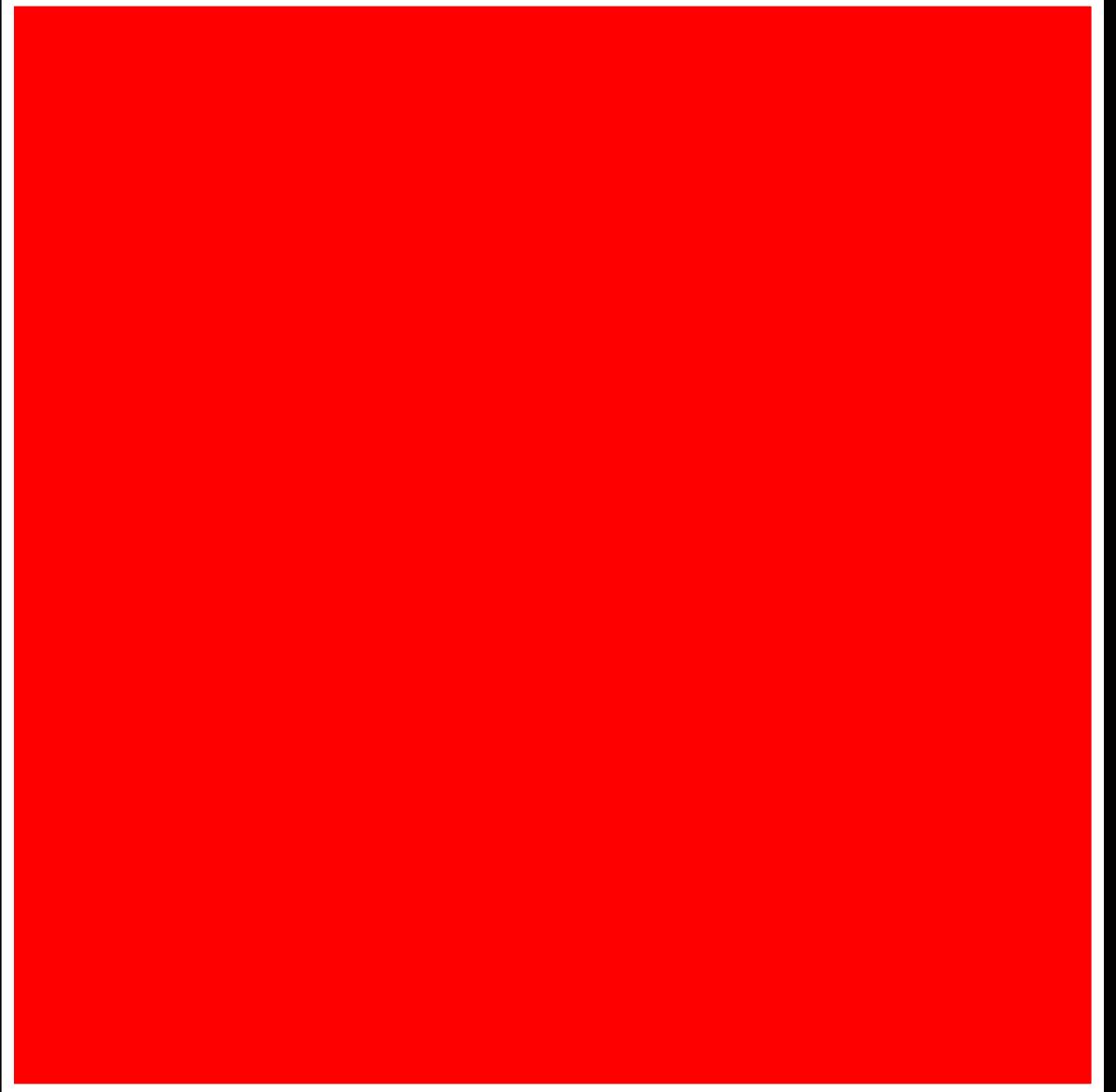
255

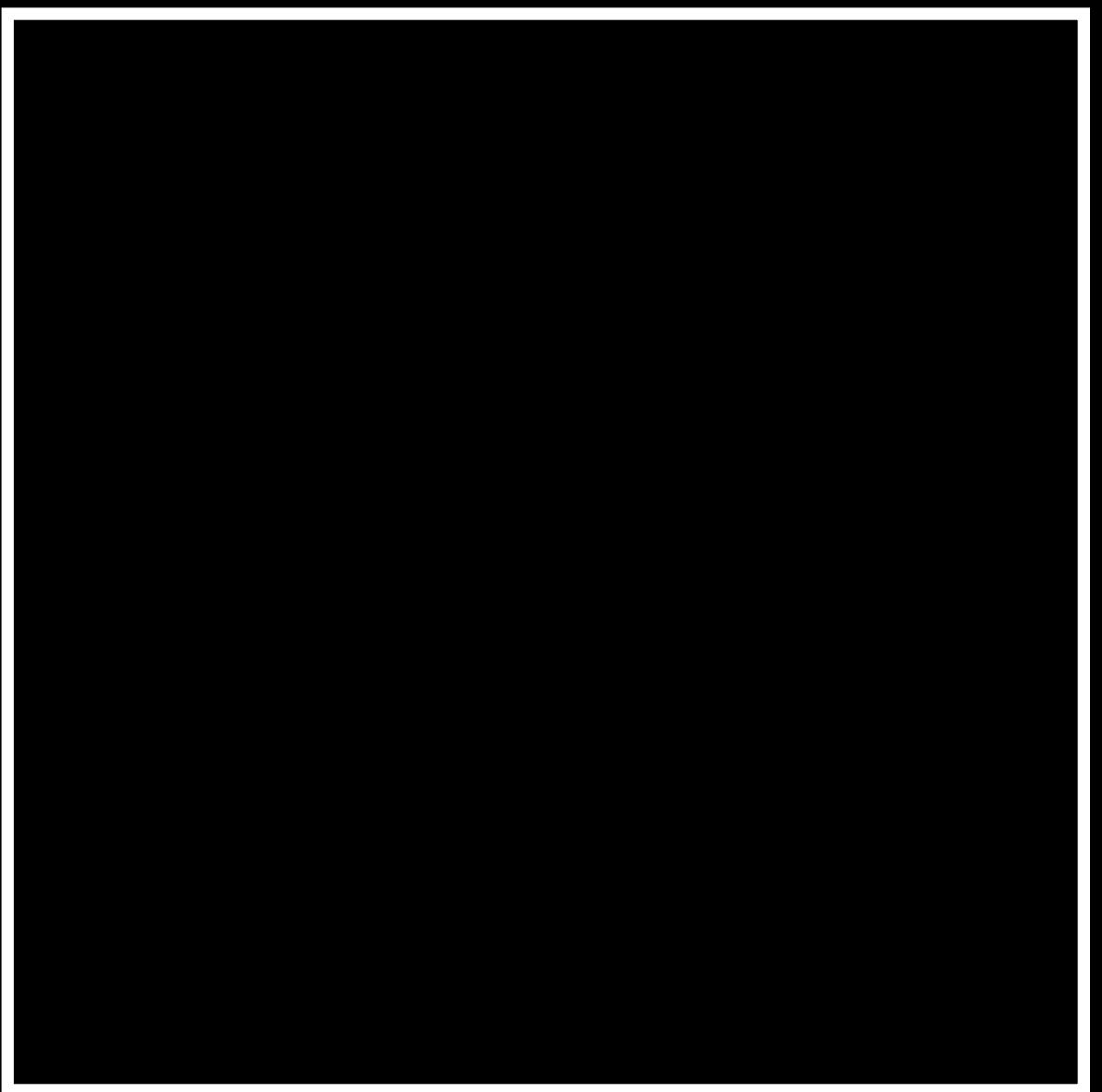


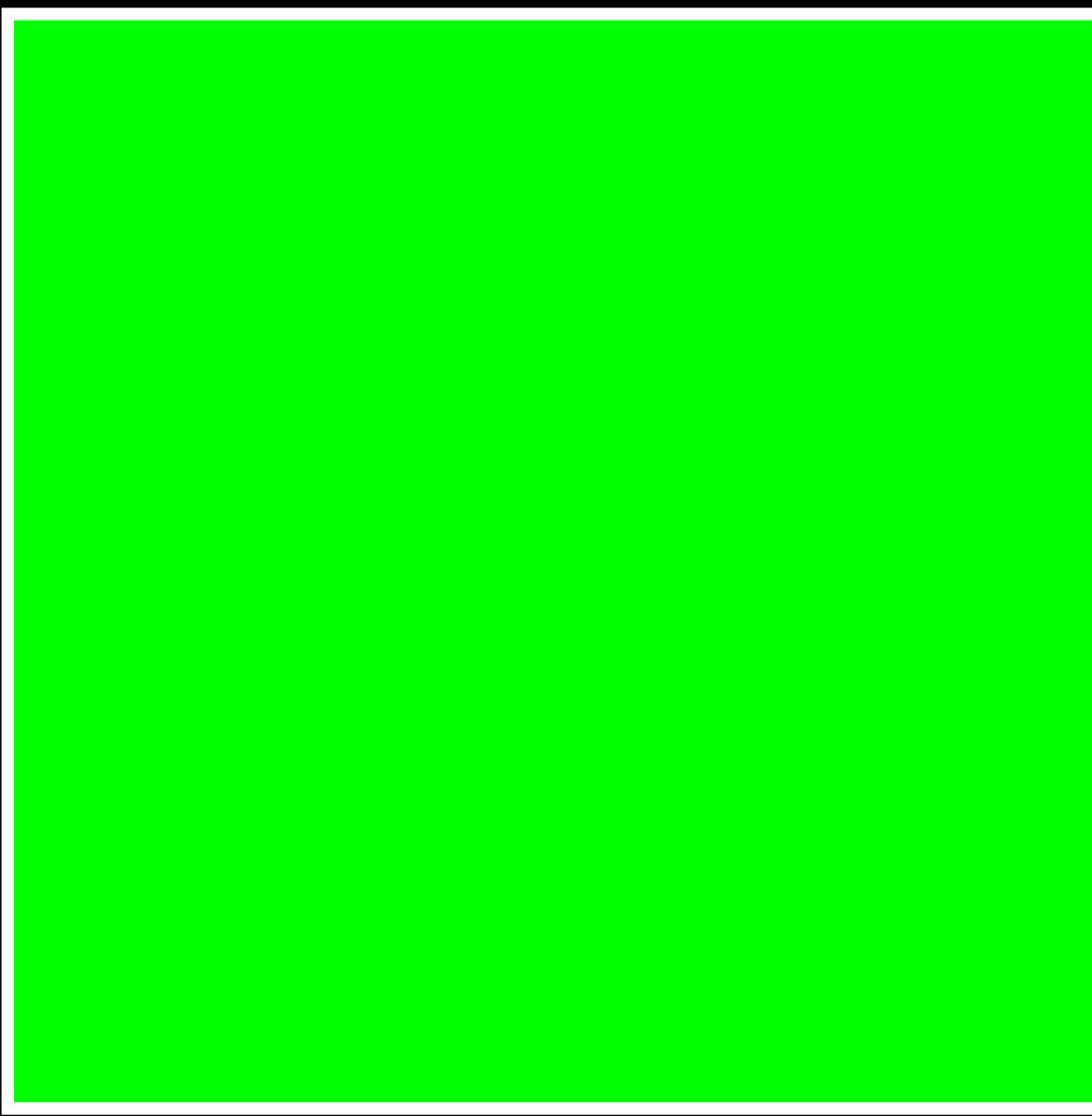
0

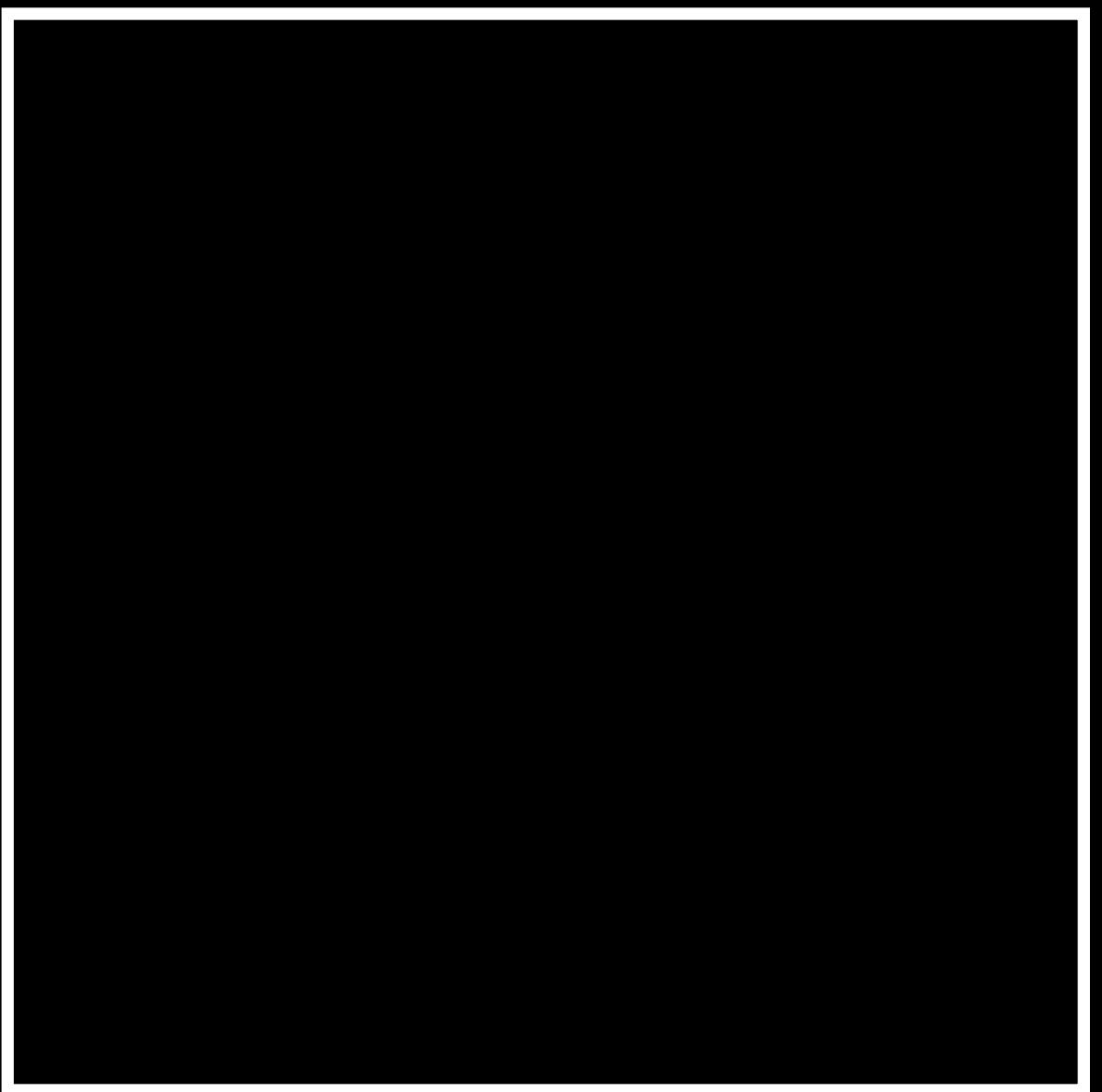
255

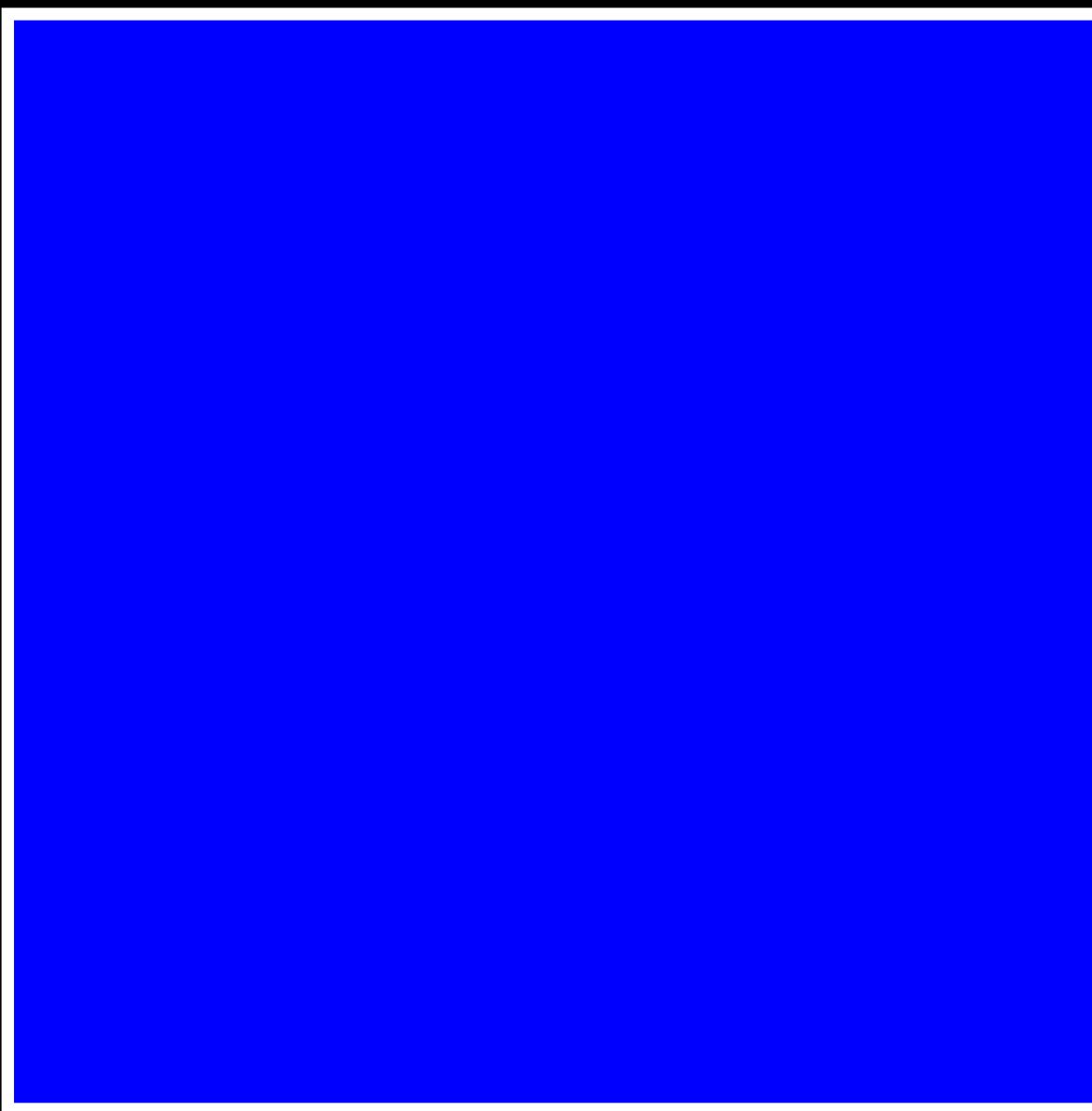


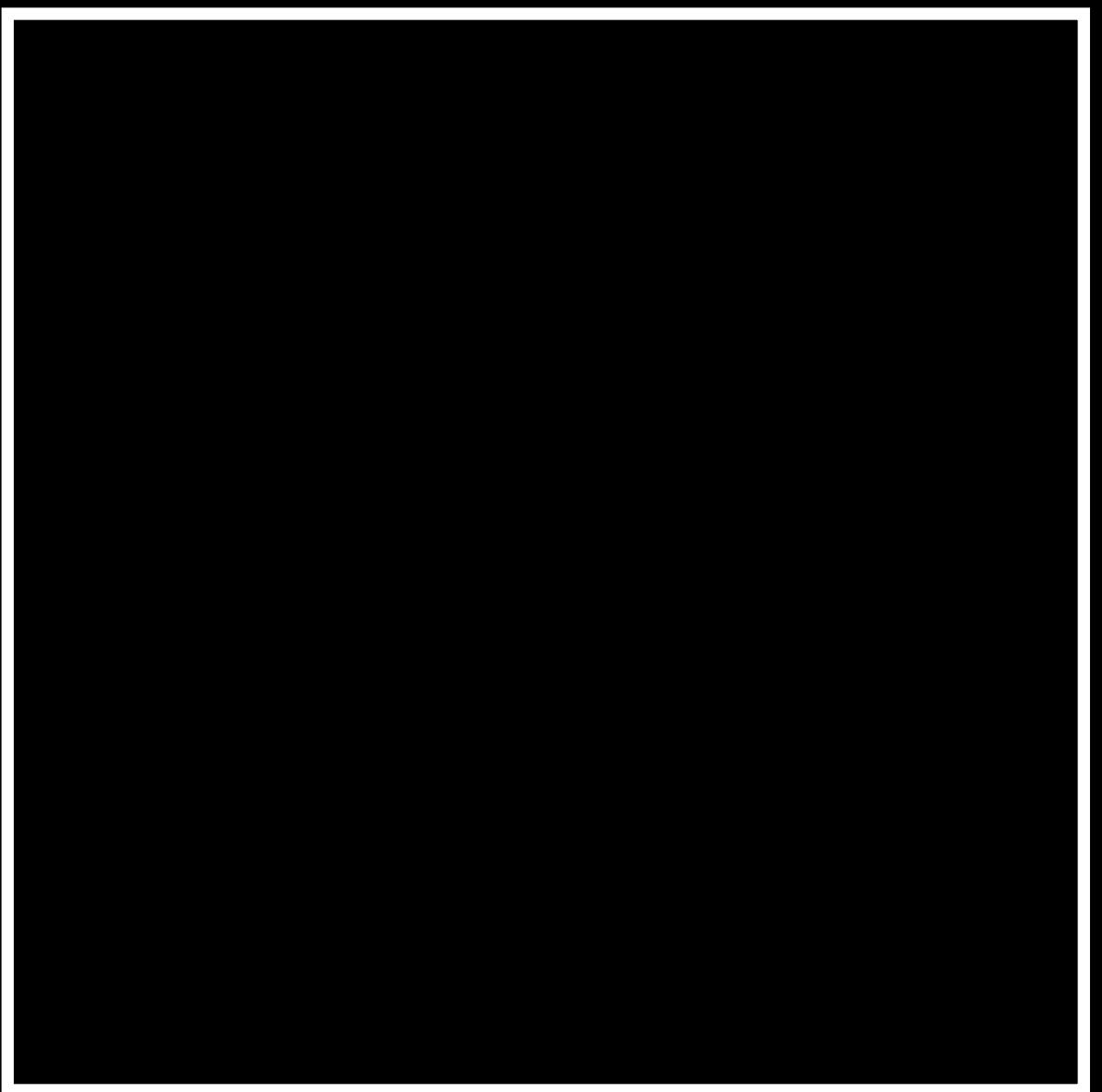


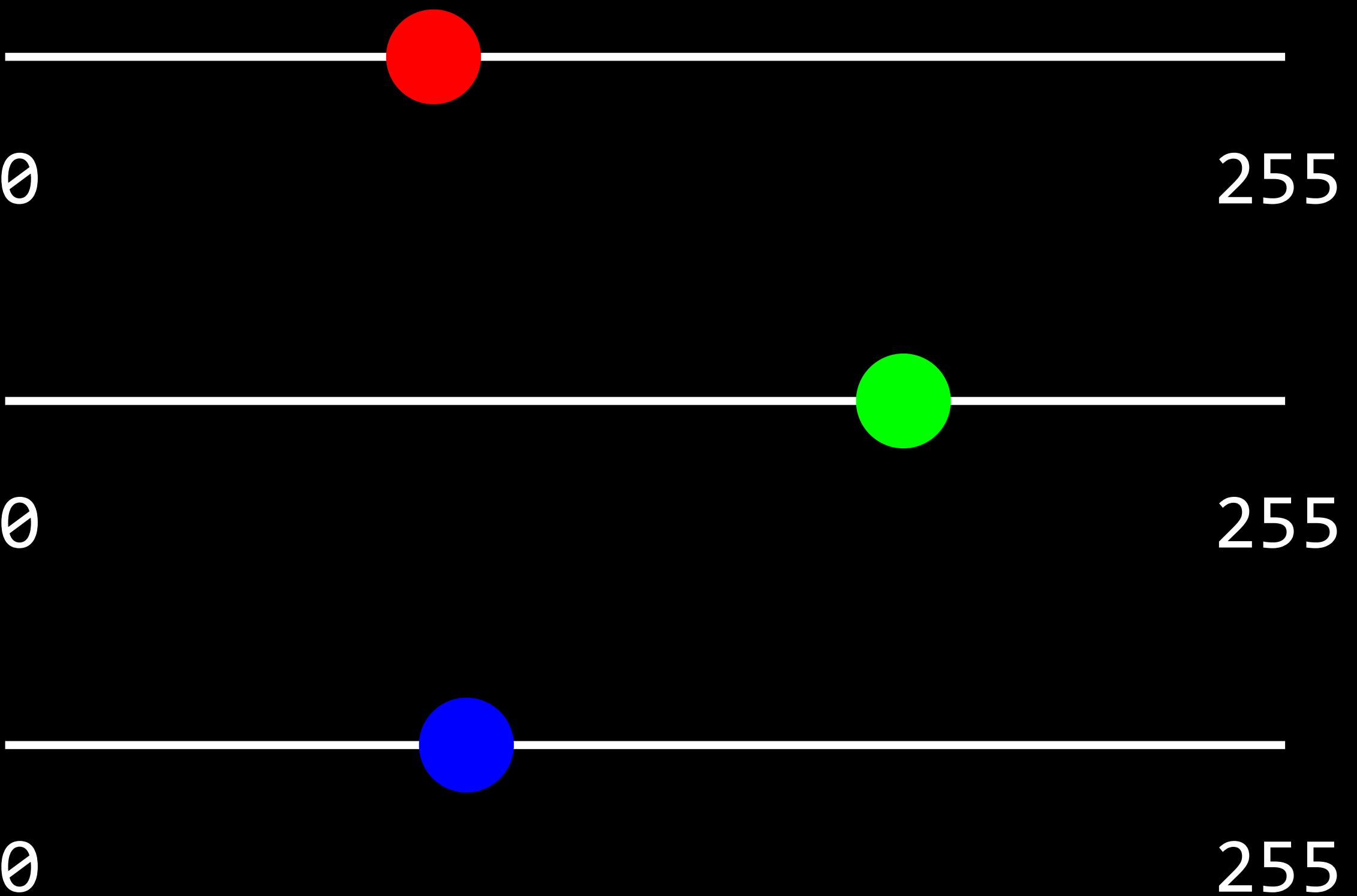
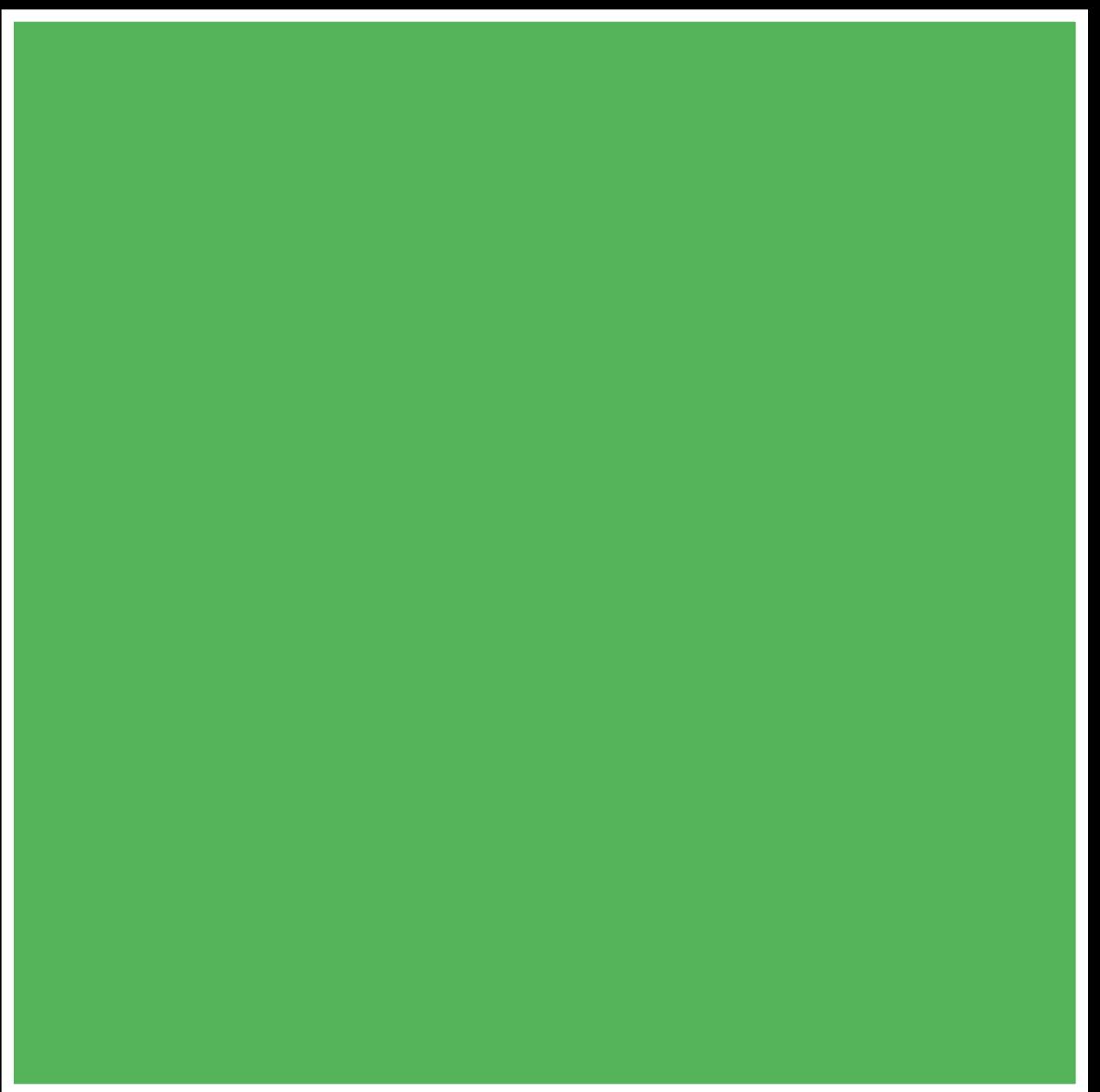


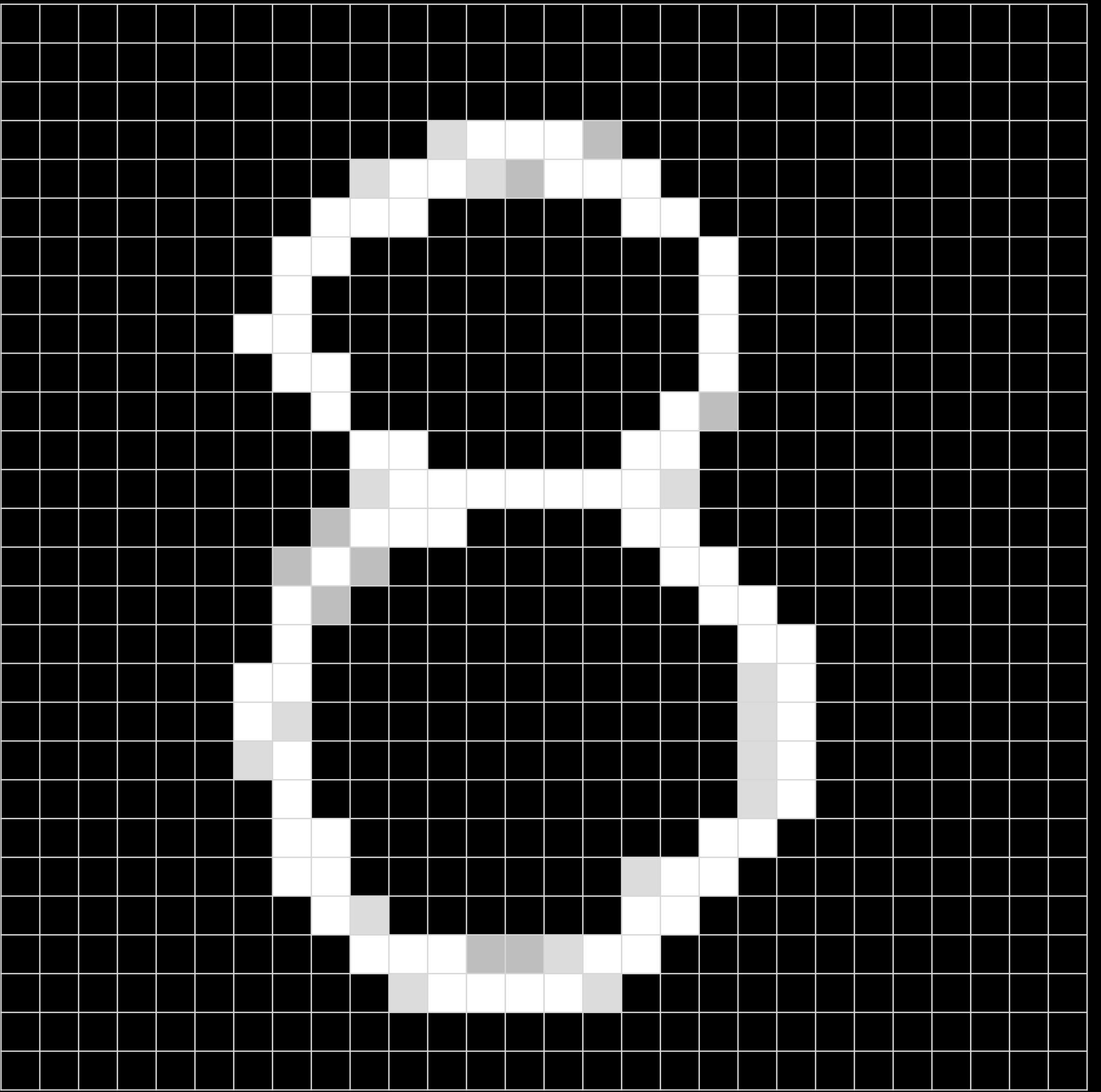
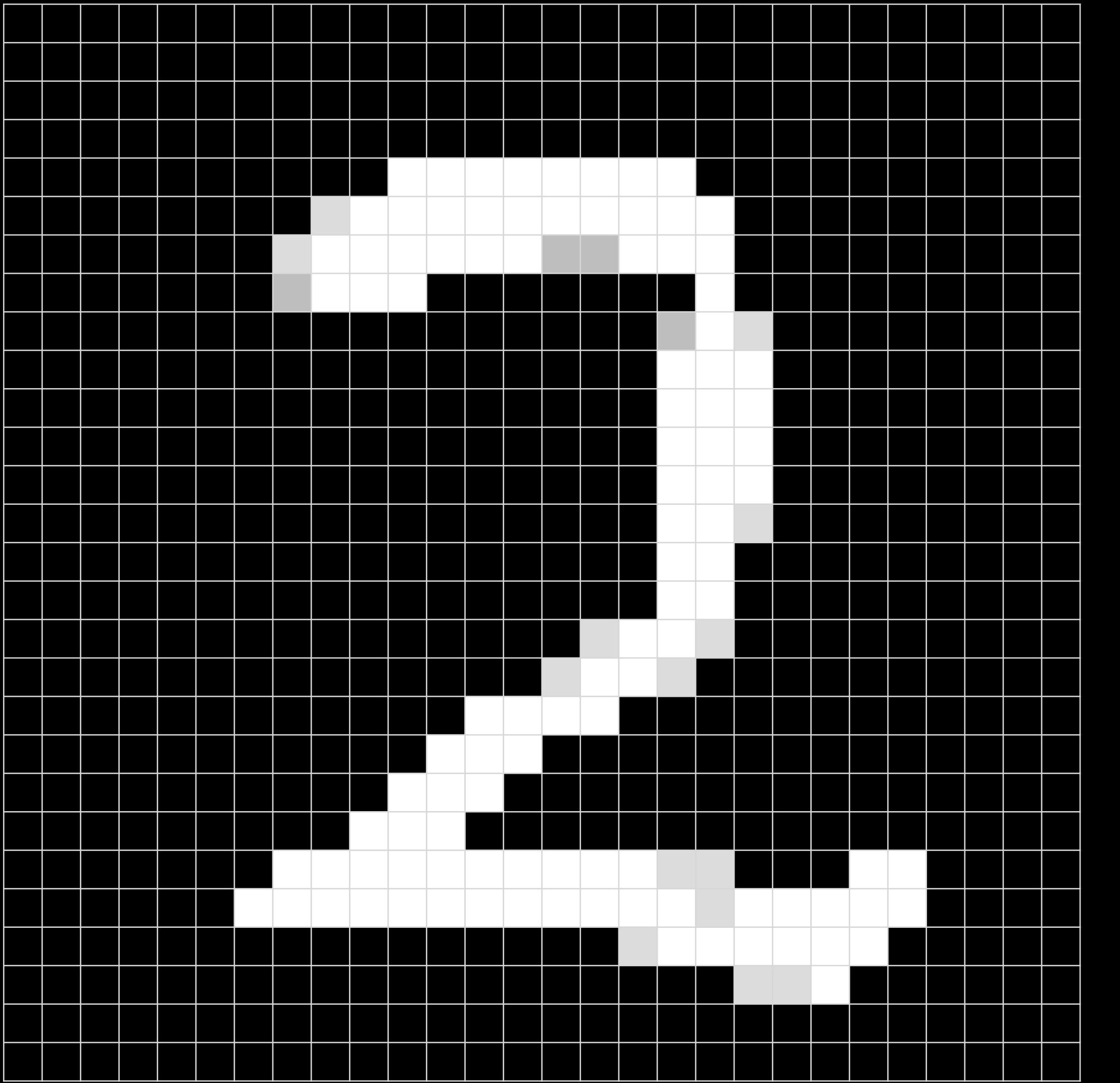












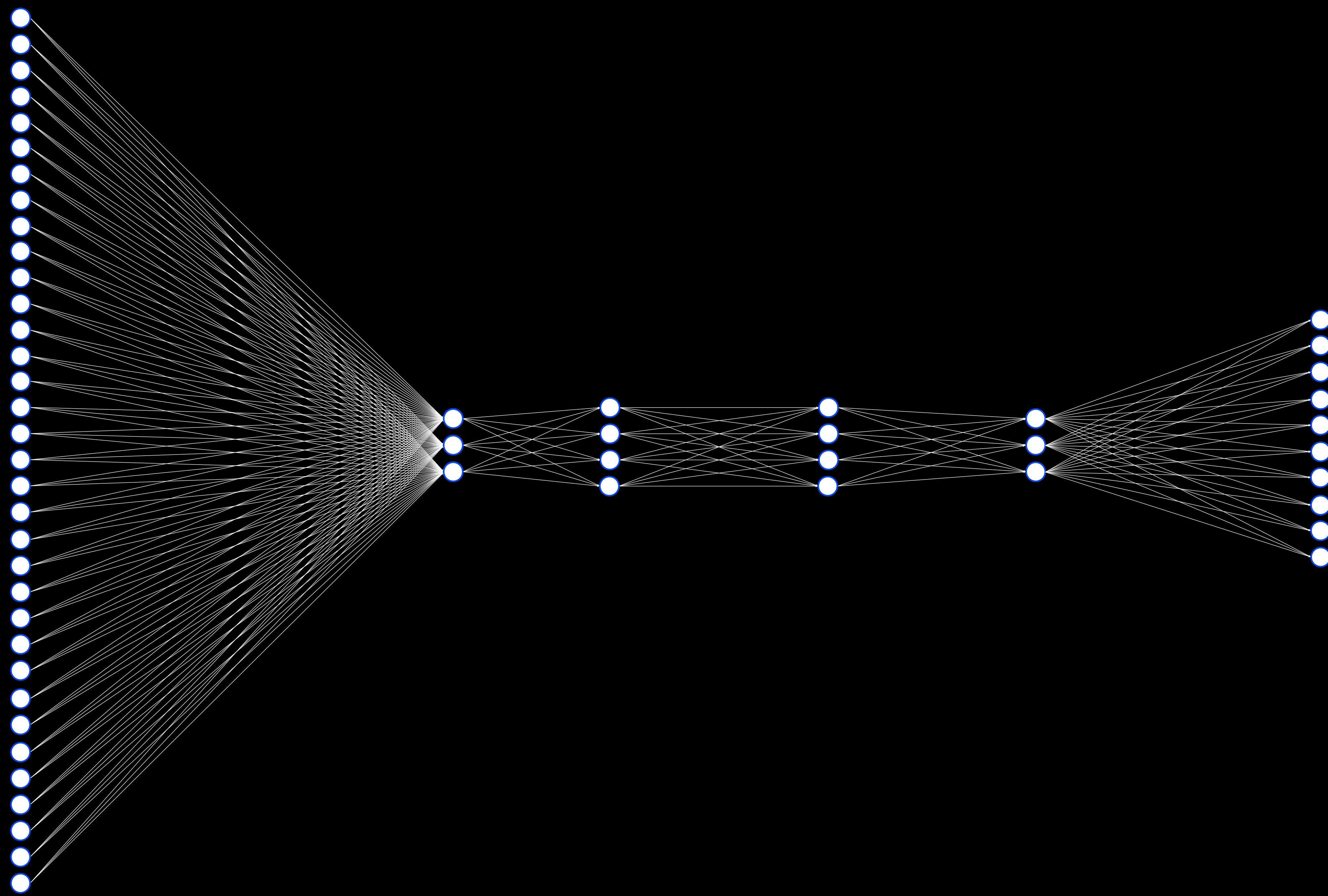


image convolution

applying a filter that adds each pixel value
of an image to its neighbors, weighted
according to a kernel matrix

0	-1	0
-1	5	-1
0	-1	0

10	20	30	40
10	20	30	40
20	30	40	50
20	30	40	50

0	-1	0
-1	5	-1
0	-1	0

10	20	30	40
10	20	30	40
20	30	40	50
20	30	40	50

0	-1	0
-1	5	-1
0	-1	0

10	20	30	40
10	20	30	40
20	30	40	50
20	30	40	50

0	-1	0
-1	5	-1
0	-1	0

10	20	30	40
10	20	30	40
20	30	40	50
20	30	40	50

0	-1	0
-1	5	-1
0	-1	0

10

10	20	30	40
10	20	30	40
20	30	40	50
20	30	40	50

0	-1	0
-1	5	-1
0	-1	0

10

10	20	30	40
10	20	30	40
20	30	40	50
20	30	40	50

0	-1	0
-1	5	-1
0	-1	0

10	20

10	20	30	40
10	20	30	40
20	30	40	50
20	30	40	50

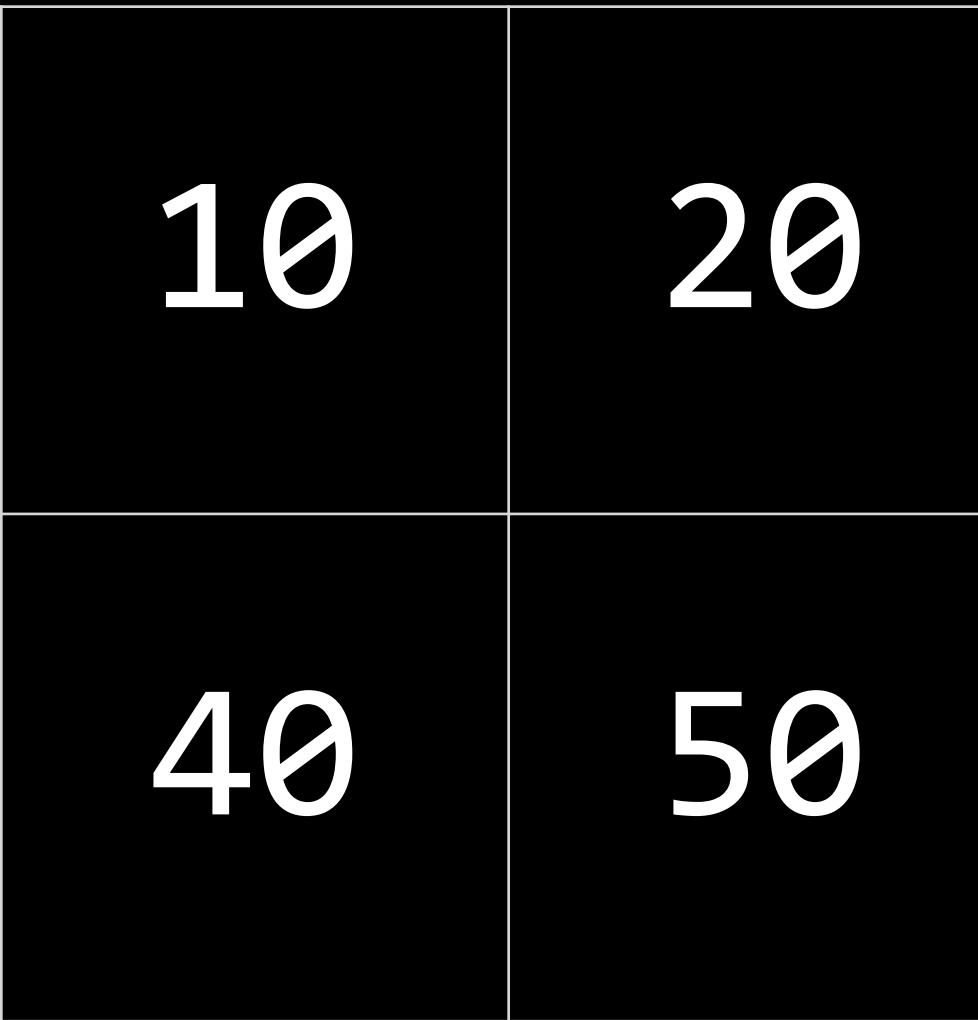
0	-1	0
-1	5	-1
0	-1	0

10	20
40	

10	20	30	40
10	20	30	40
20	30	40	50
20	30	40	50

0	-1	0
-1	5	-1
0	-1	0

10	20
40	



10	20	30	40
10	20	30	40
20	30	40	50
20	30	40	50

0	-1	0
-1	5	-1
0	-1	0

10	20
40	50

-1	-1	-1
-1	8	-1
-1	-1	-1

20	20	20
20	20	20
20	20	20

-1	-1	-1
-1	8	-1
-1	-1	-1

$$\begin{aligned} & (20)(-1) + (20)(-1) + (20)(-1) \\ & + (20)(-1) + (20)(8) + (20)(-1) \\ & + (20)(-1) + (20)(-1) + (20)(-1) \end{aligned}$$

0

20	20	20
50	50	50
50	50	50

-1	-1	-1
-1	8	-1
-1	-1	-1

$$\begin{aligned} & (20)(-1) + (20)(-1) + (20)(-1) \\ & + (50)(-1) + (50)(8) + (50)(-1) \\ & + (50)(-1) + (50)(-1) + (50)(-1) \end{aligned}$$

90





pooling

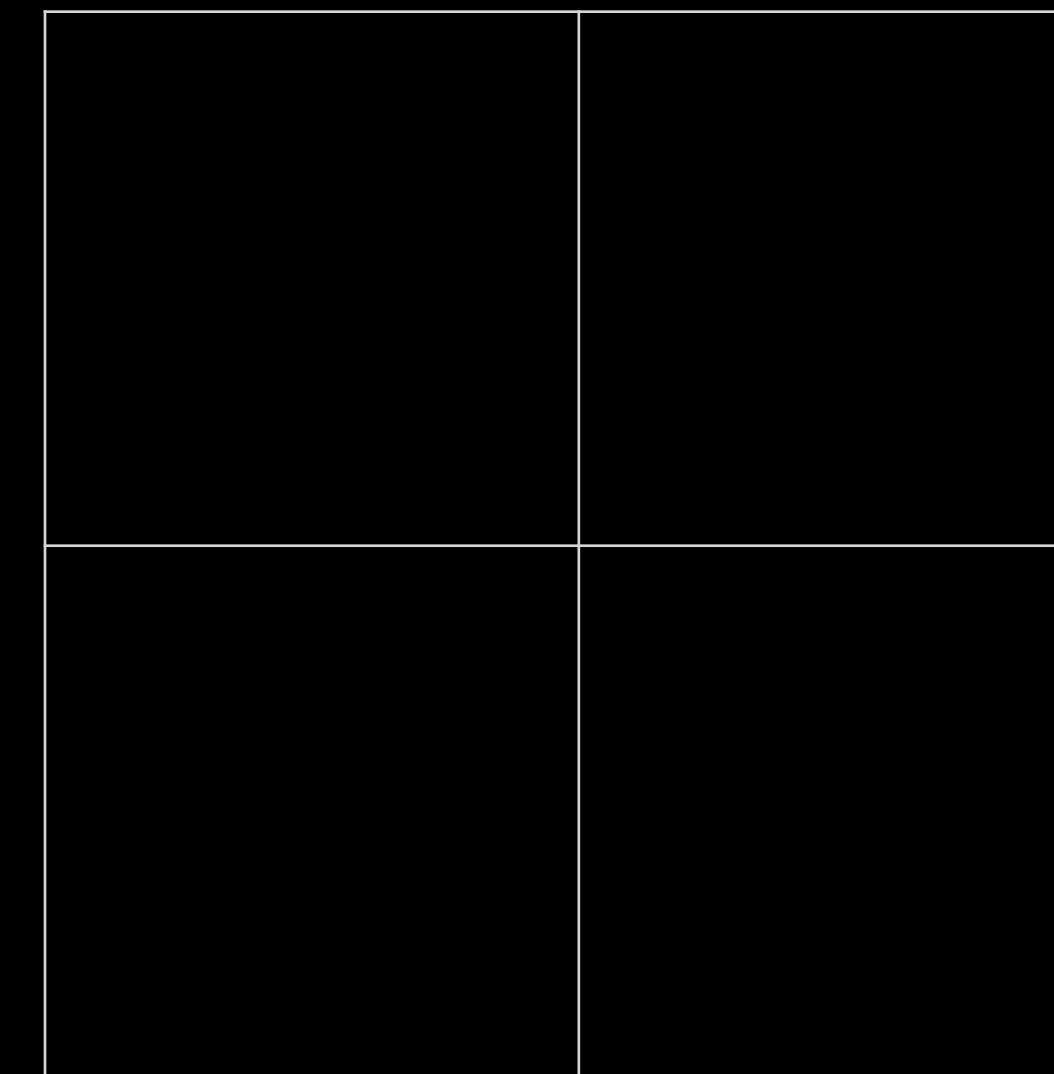
reducing the size of an input by sampling
from regions in the input

max-pooling

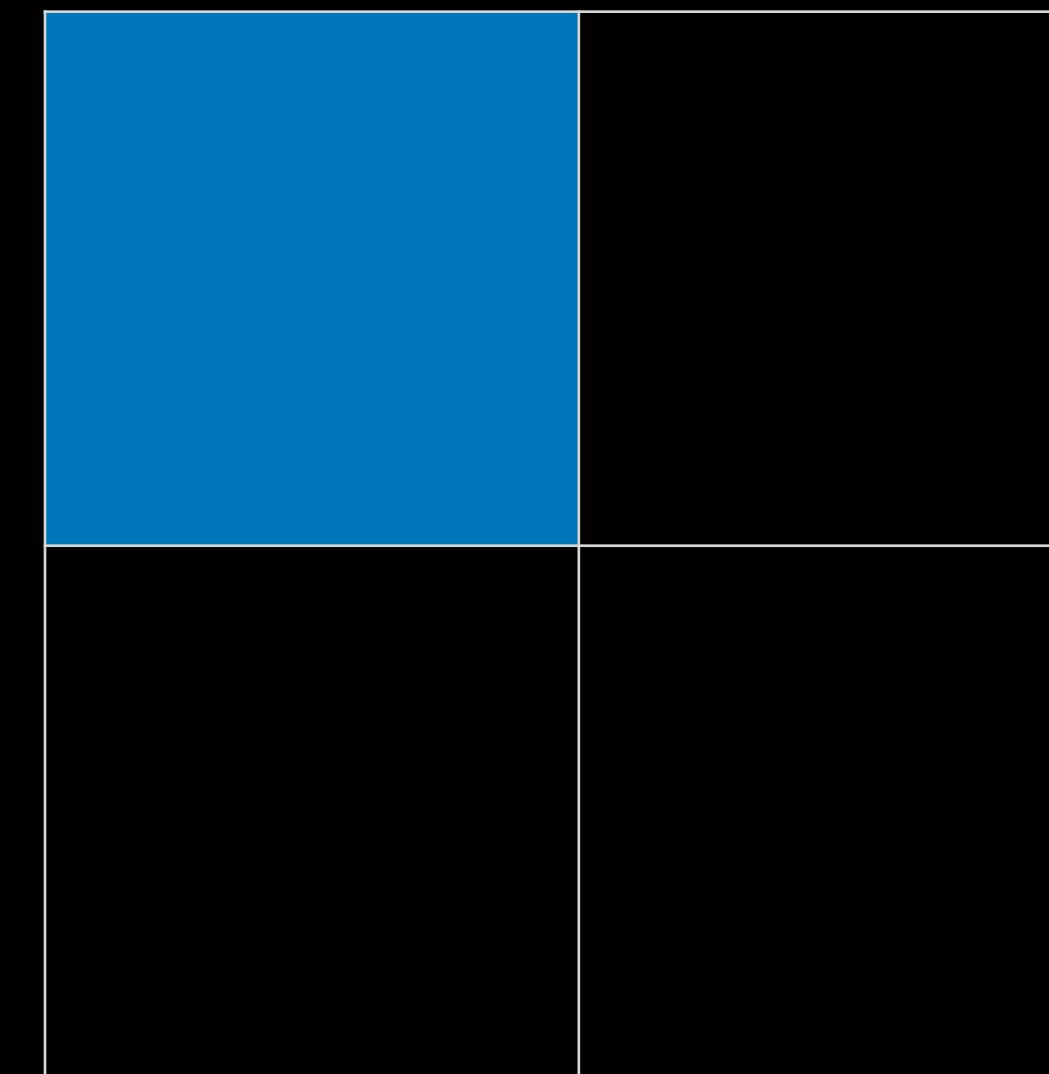
pooling by choosing the maximum value in each region

30	40	80	90
20	50	100	110
0	10	20	30
10	20	40	30

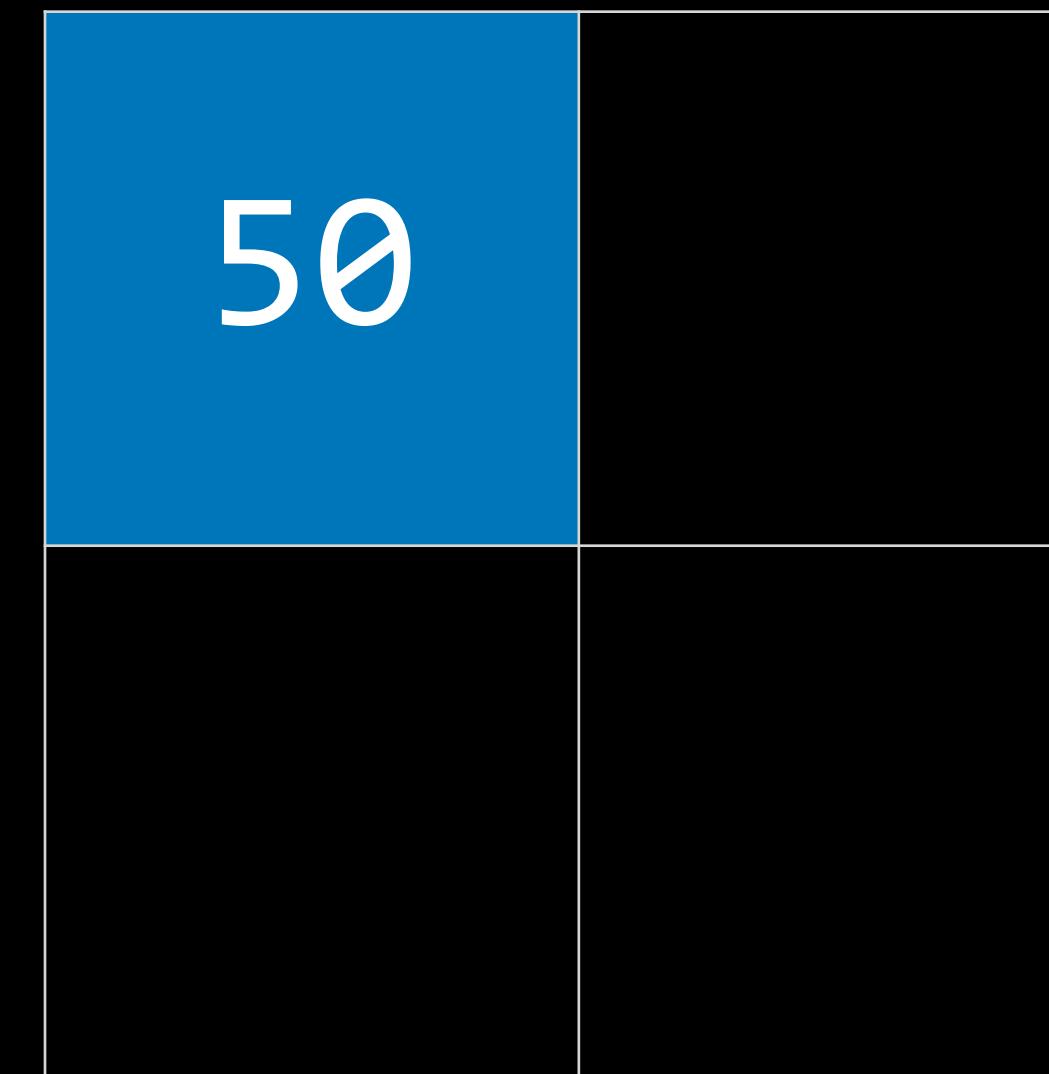
30	40	80	90
20	50	100	110
0	10	20	30
10	20	40	30



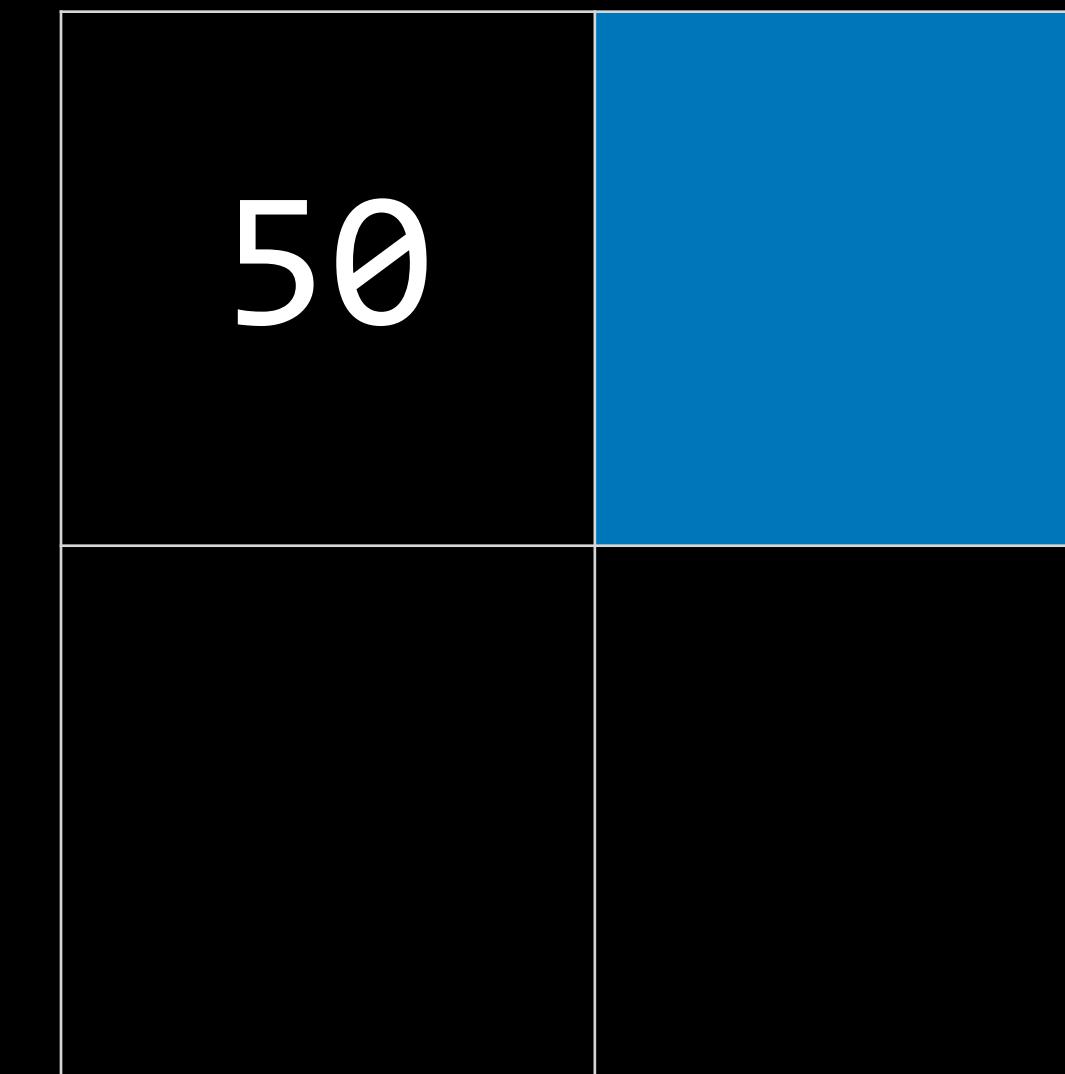
30	40	80	90
20	50	100	110
0	10	20	30
10	20	40	30



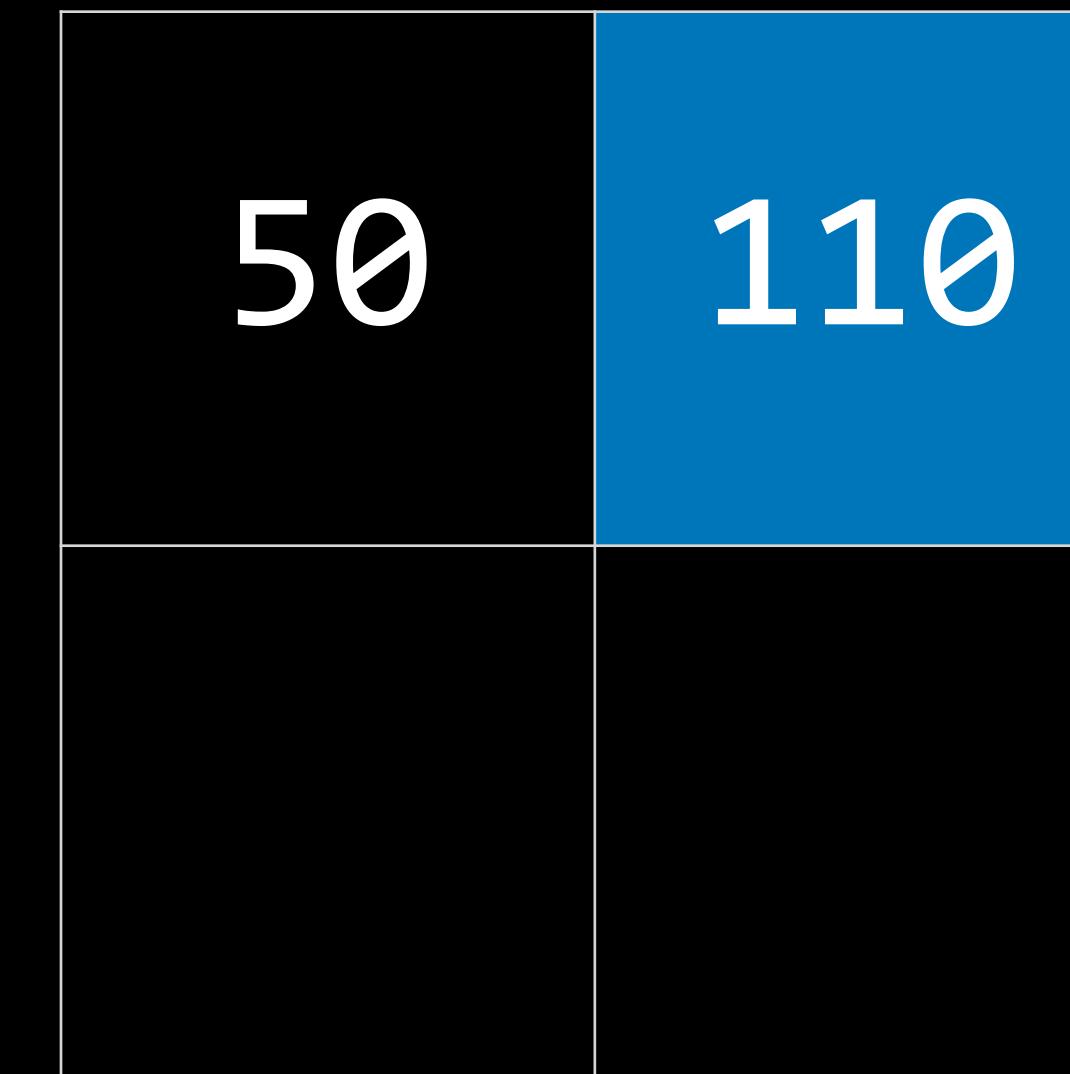
30	40	80	90
20	50	100	110
0	10	20	30
10	20	40	30



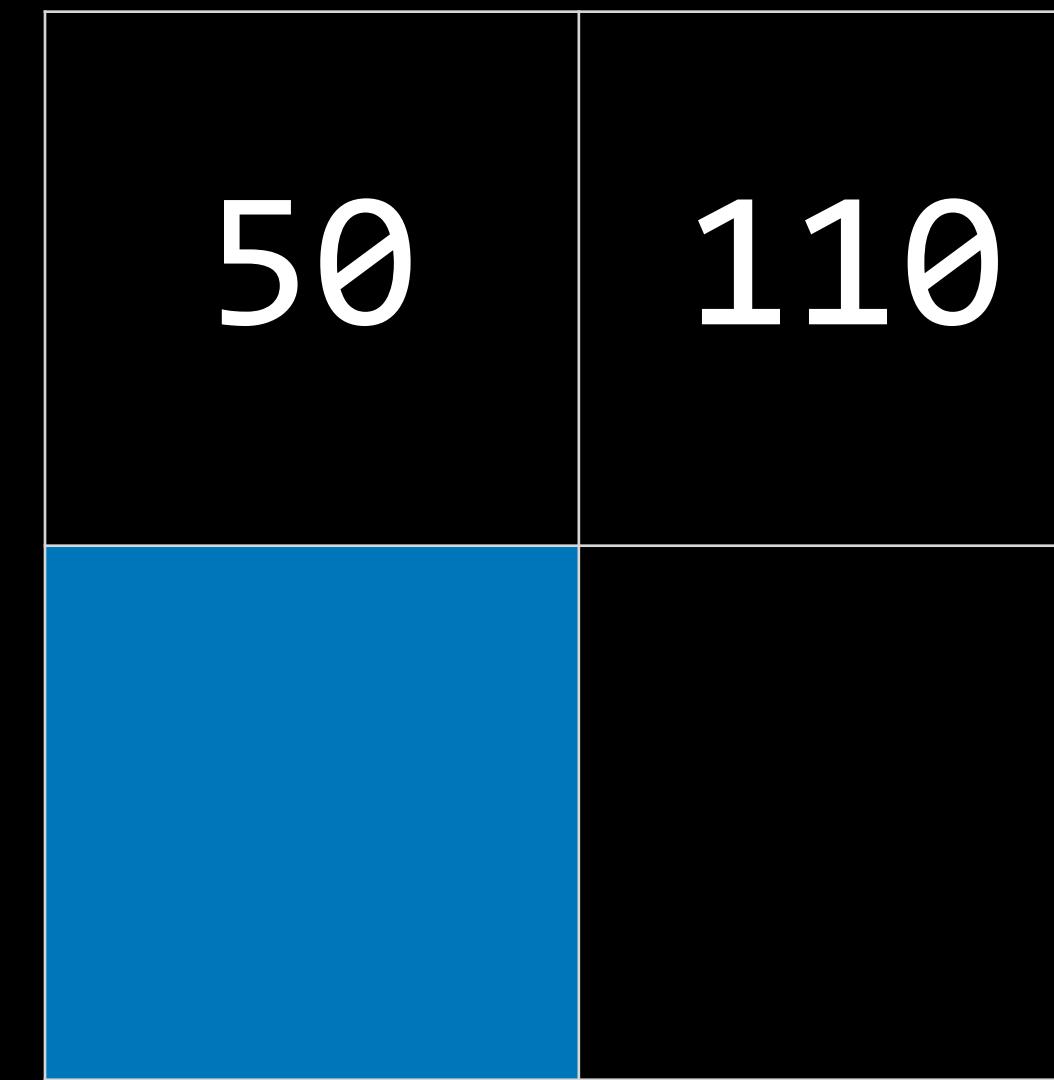
30	40	80	90
20	50	100	110
0	10	20	30
10	20	40	30



30	40	80	90
20	50	100	110
0	10	20	30
10	20	40	30



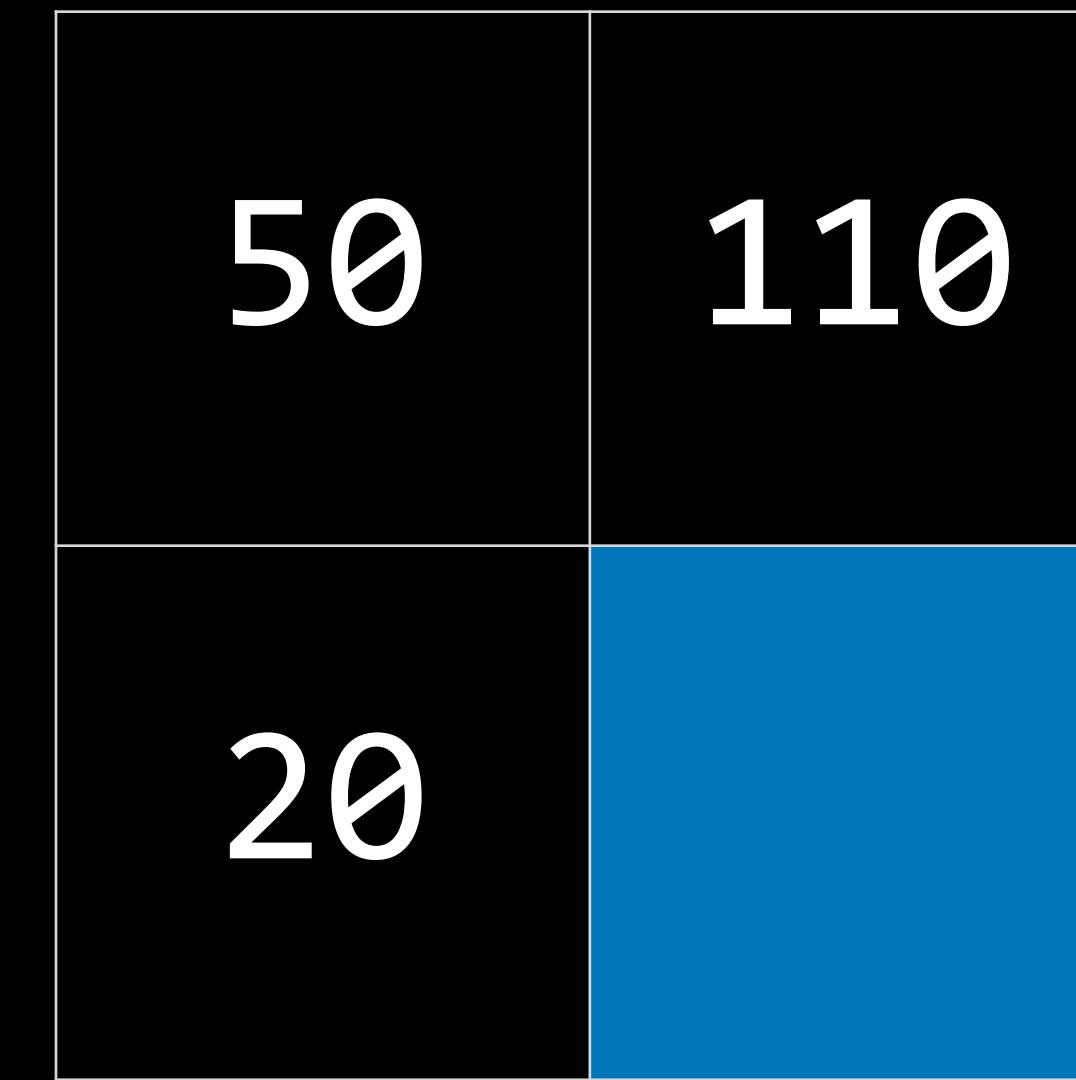
30	40	80	90
20	50	100	110
0	10	20	30
10	20	40	30



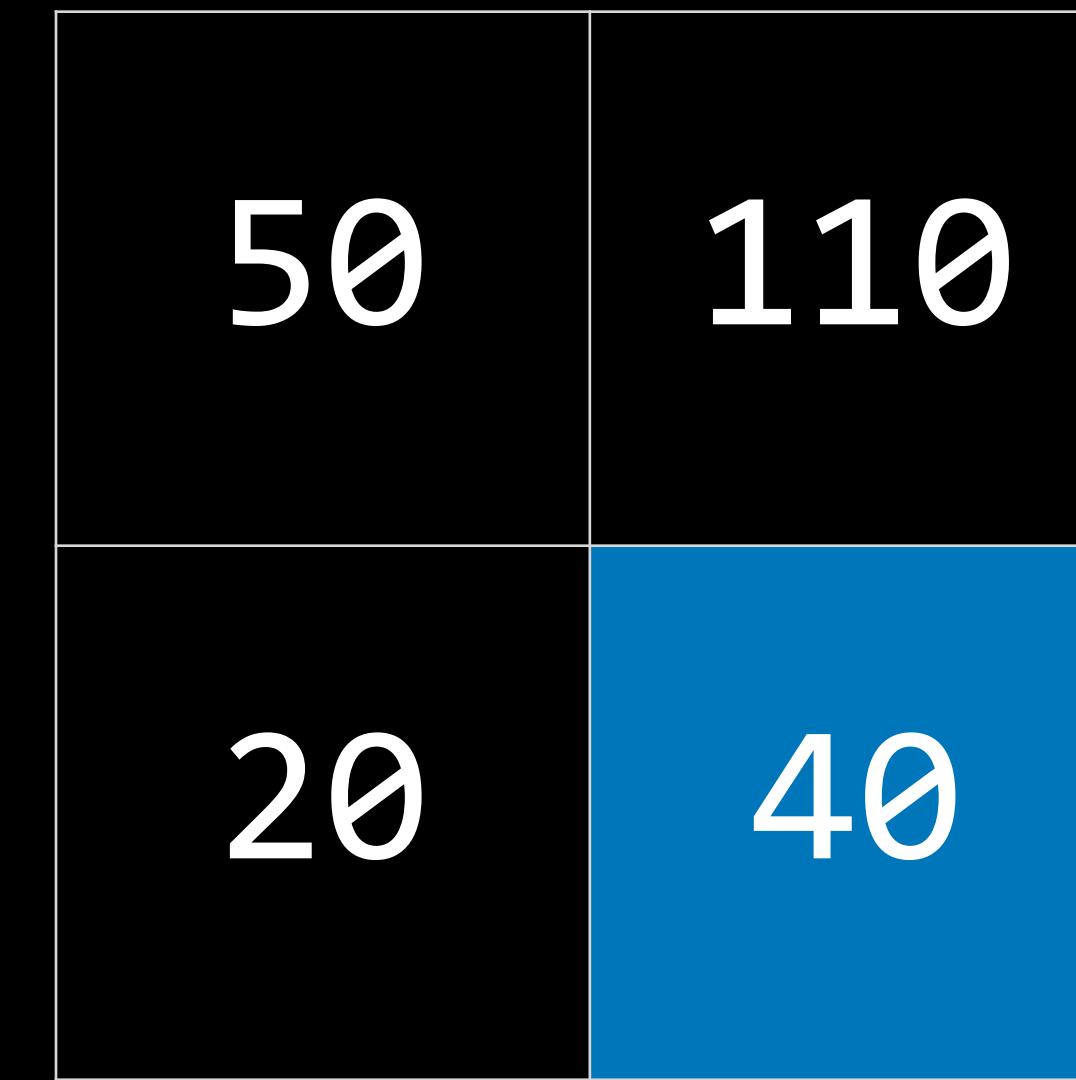
30	40	80	90
20	50	100	110
0	10	20	30
10	20	40	30

50	110
20	

30	40	80	90
20	50	100	110
0	10	20	30
10	20	40	30



30	40	80	90
20	50	100	110
0	10	20	30
10	20	40	30



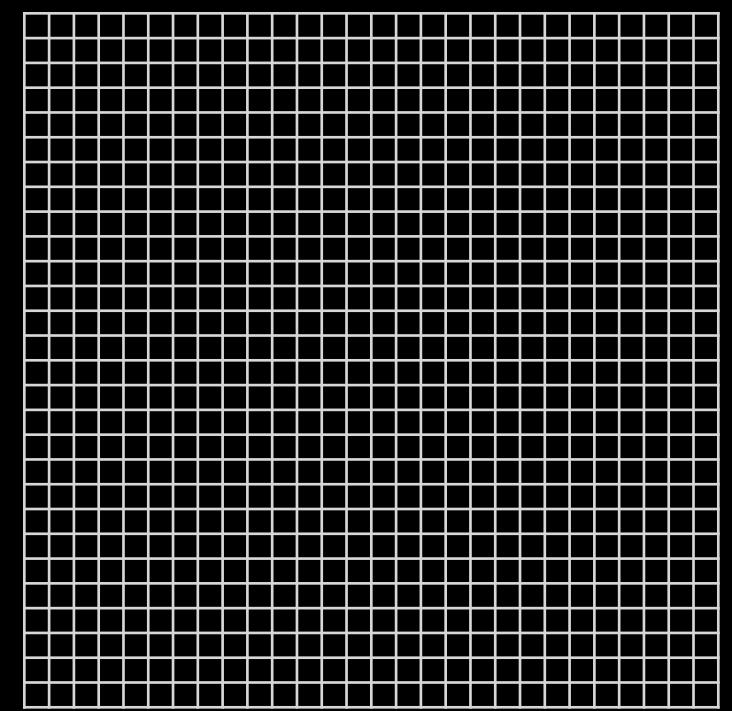
30	40	80	90
20	50	100	110
0	10	20	30
10	20	40	30

50	110
20	40

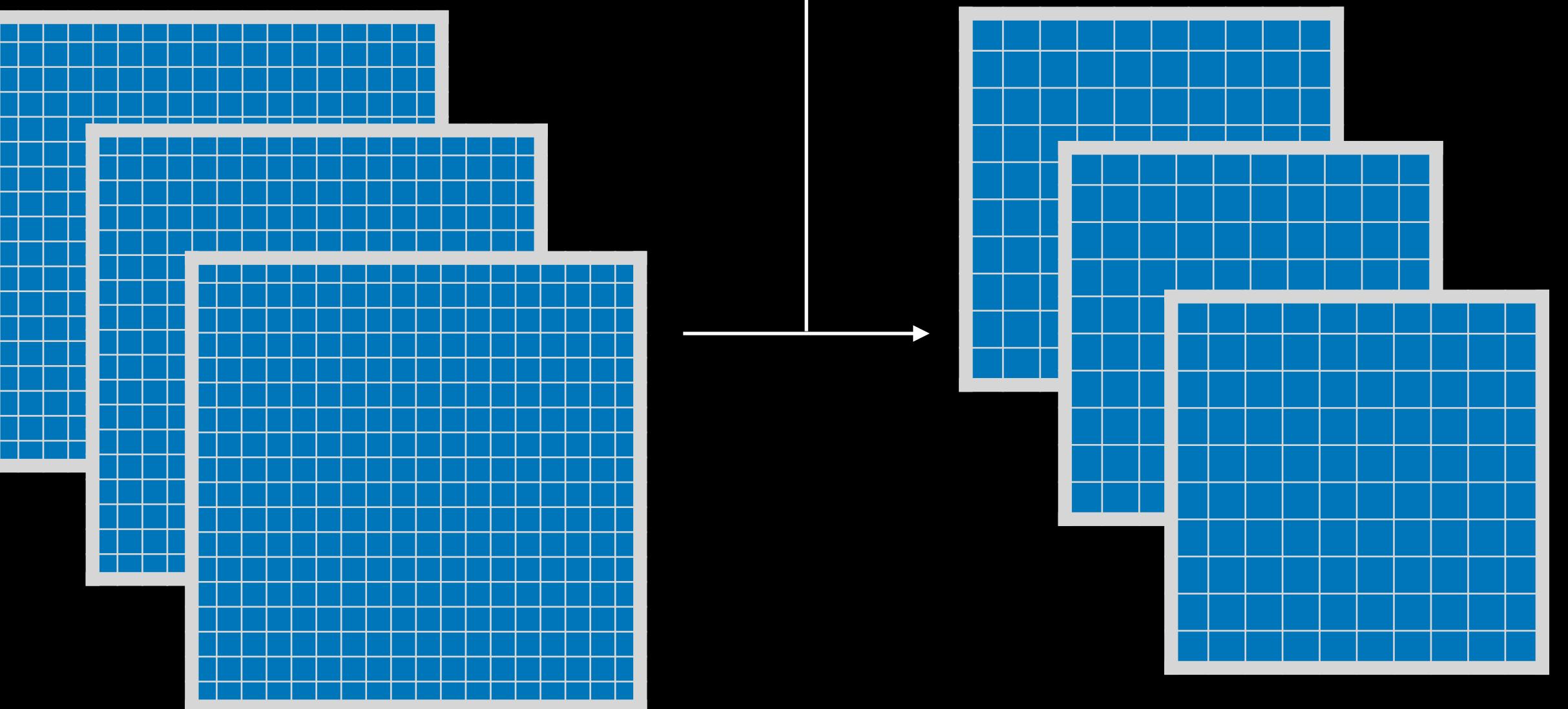
convolutional neural network

neural networks that use convolution,
usually for analyzing images

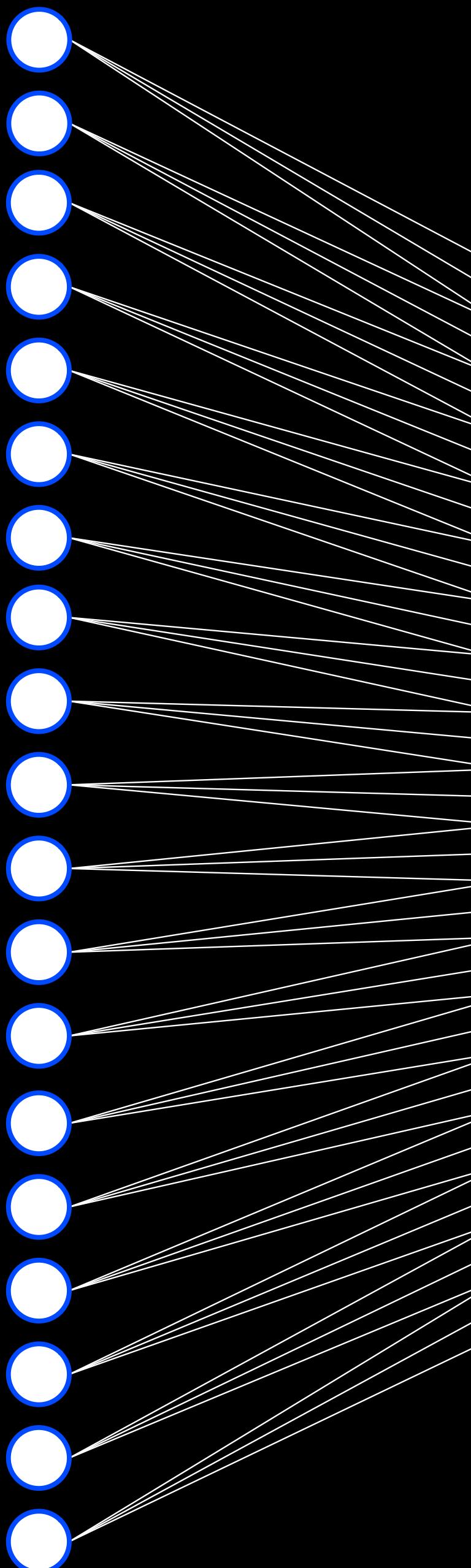
convolution

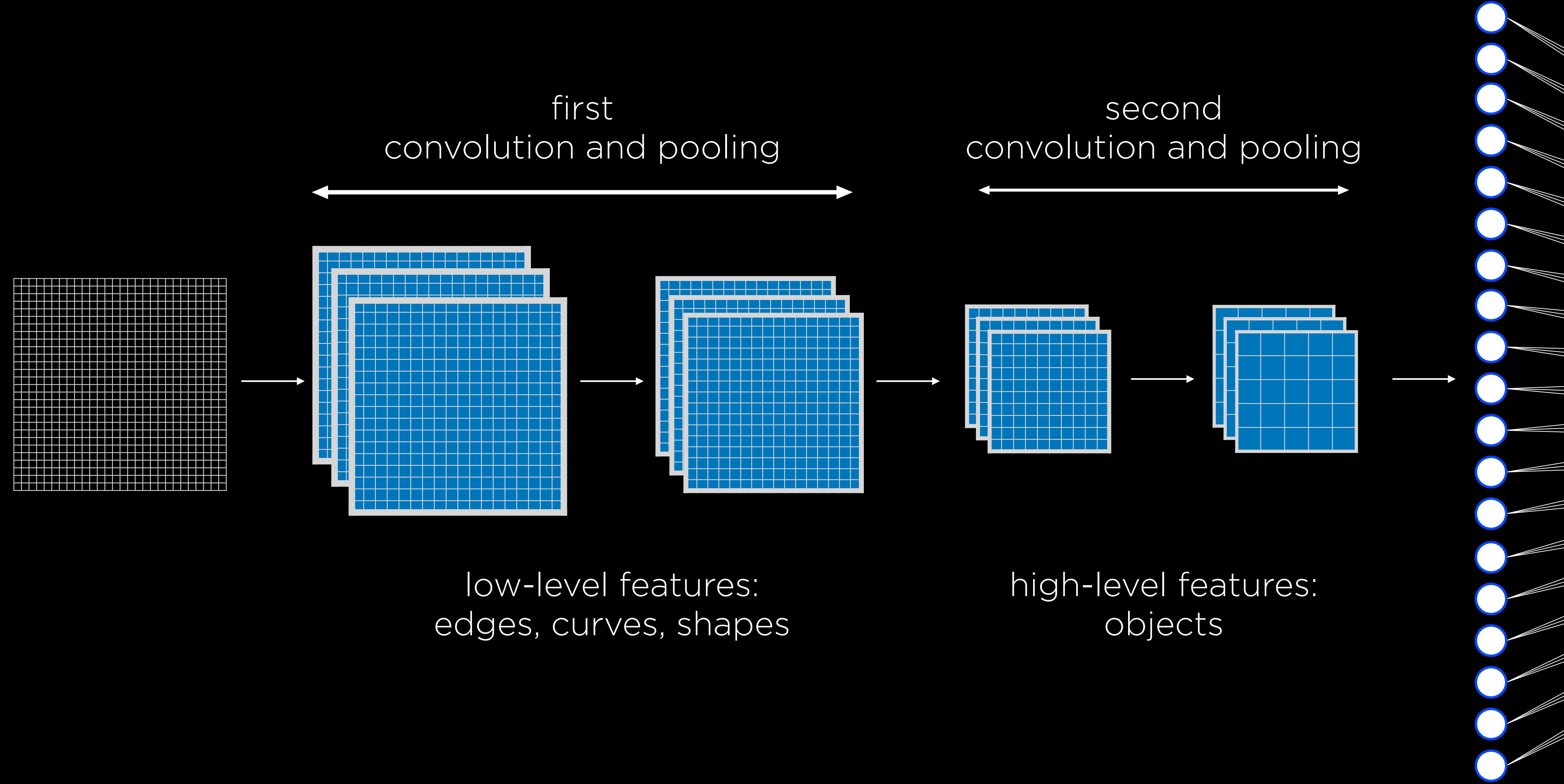


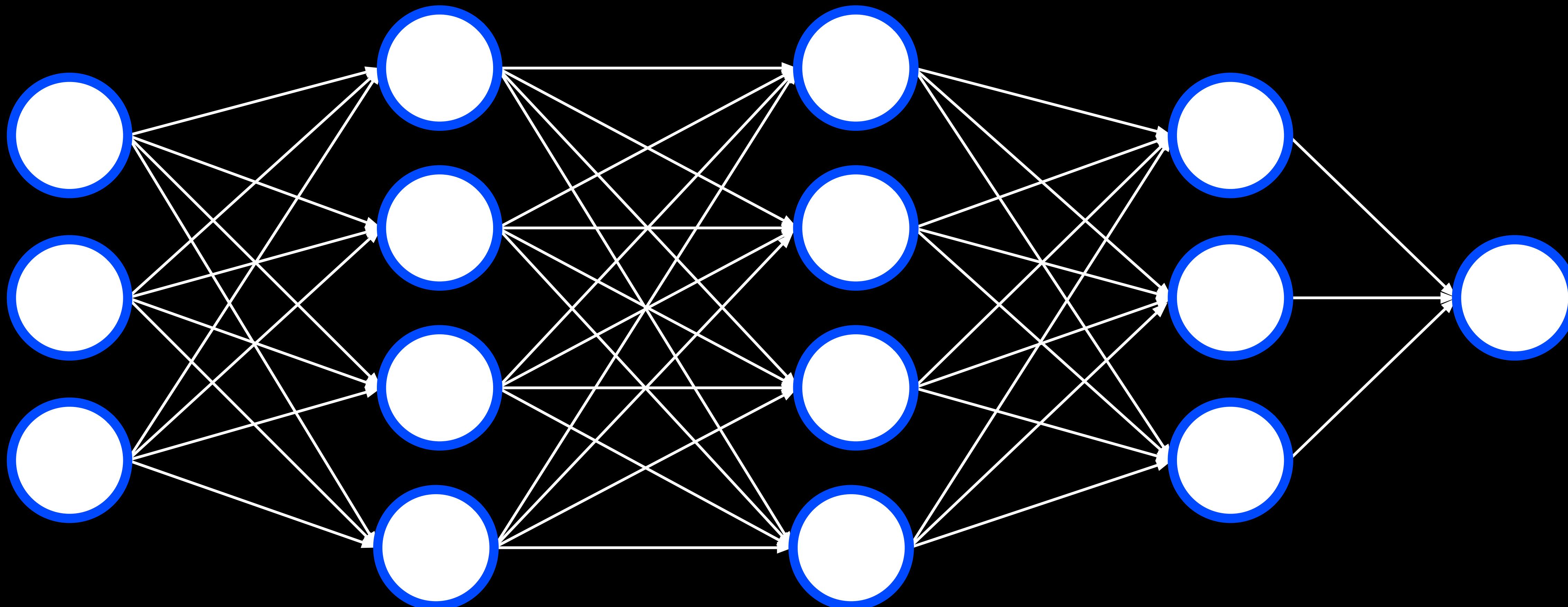
pooling

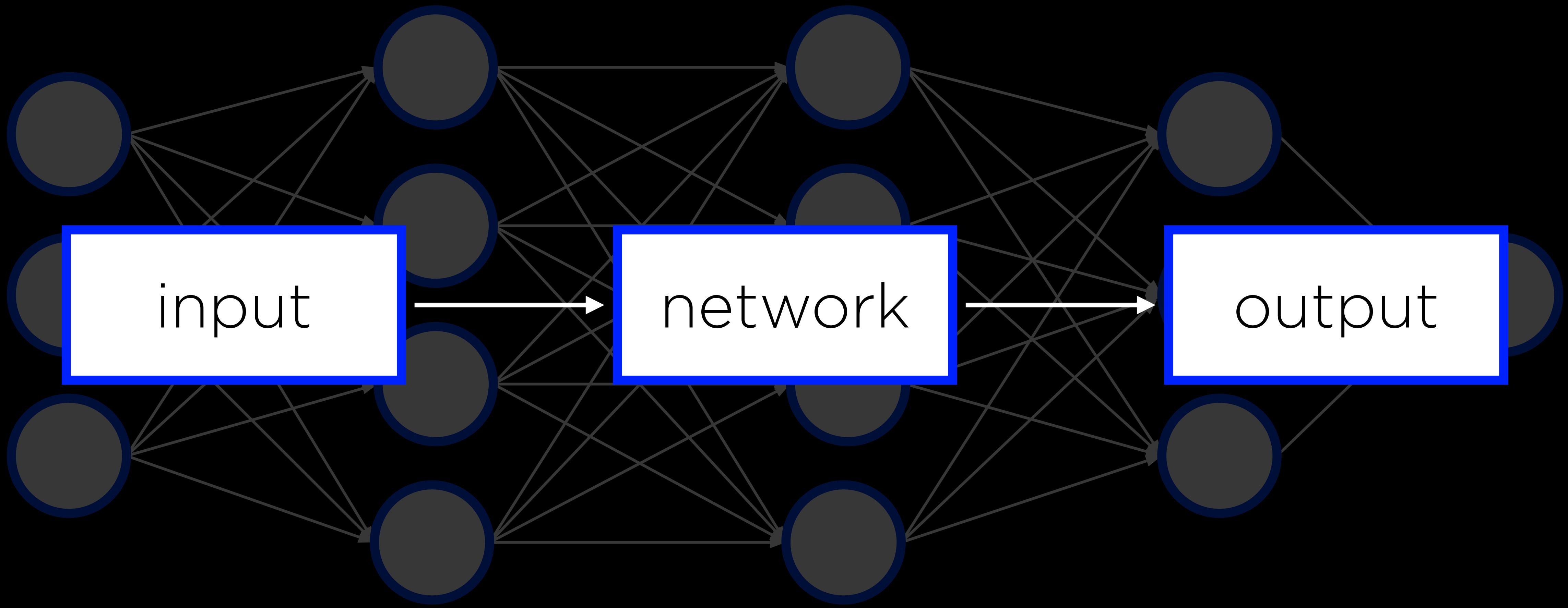


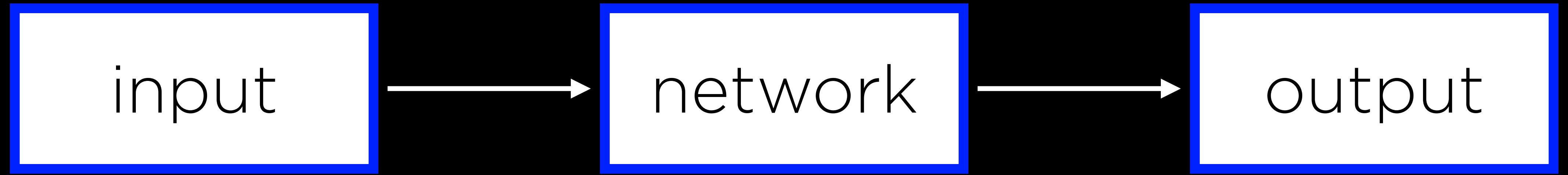
flattening









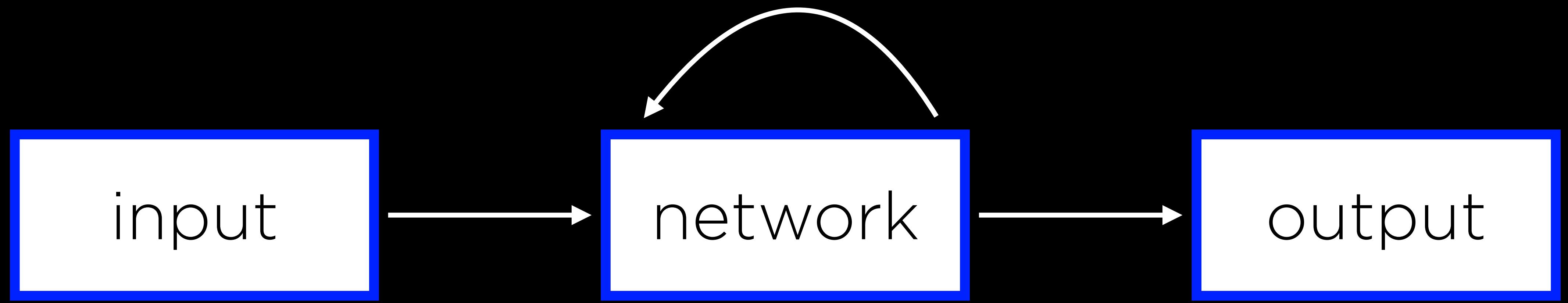


feed-forward neural network

neural network that has connections only in one direction

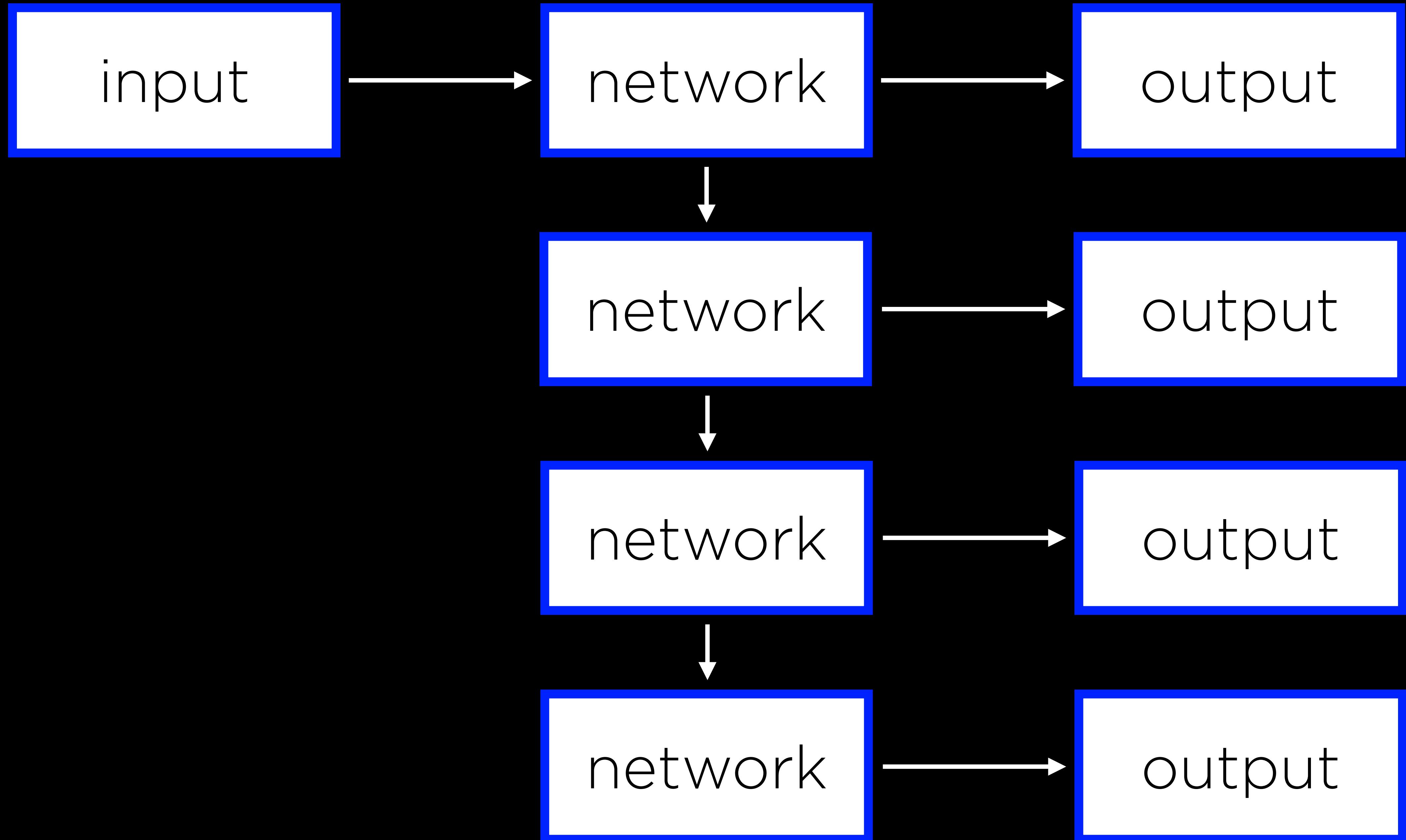
recurrent neural network

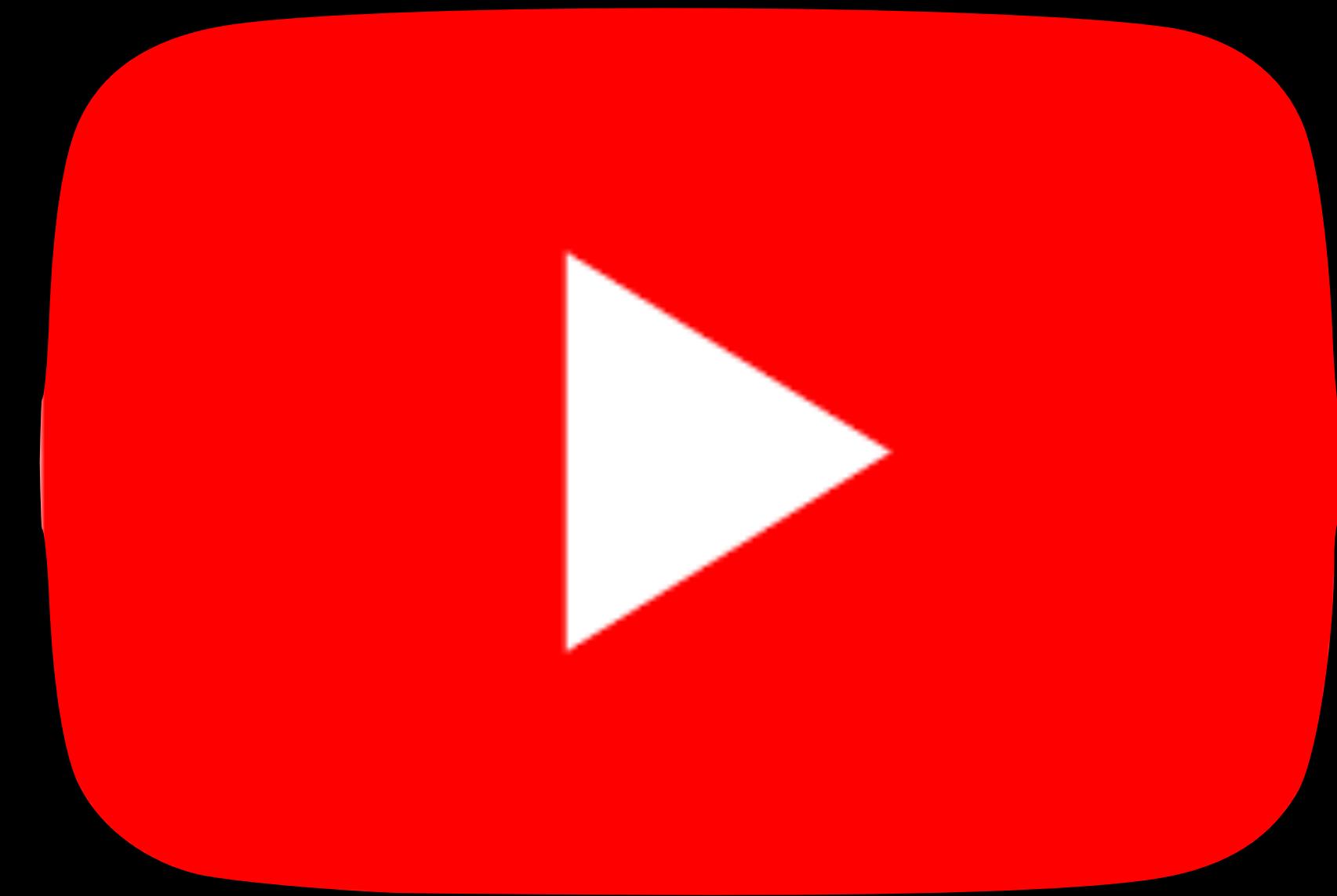
neural network that generates output that feeds back into its own inputs

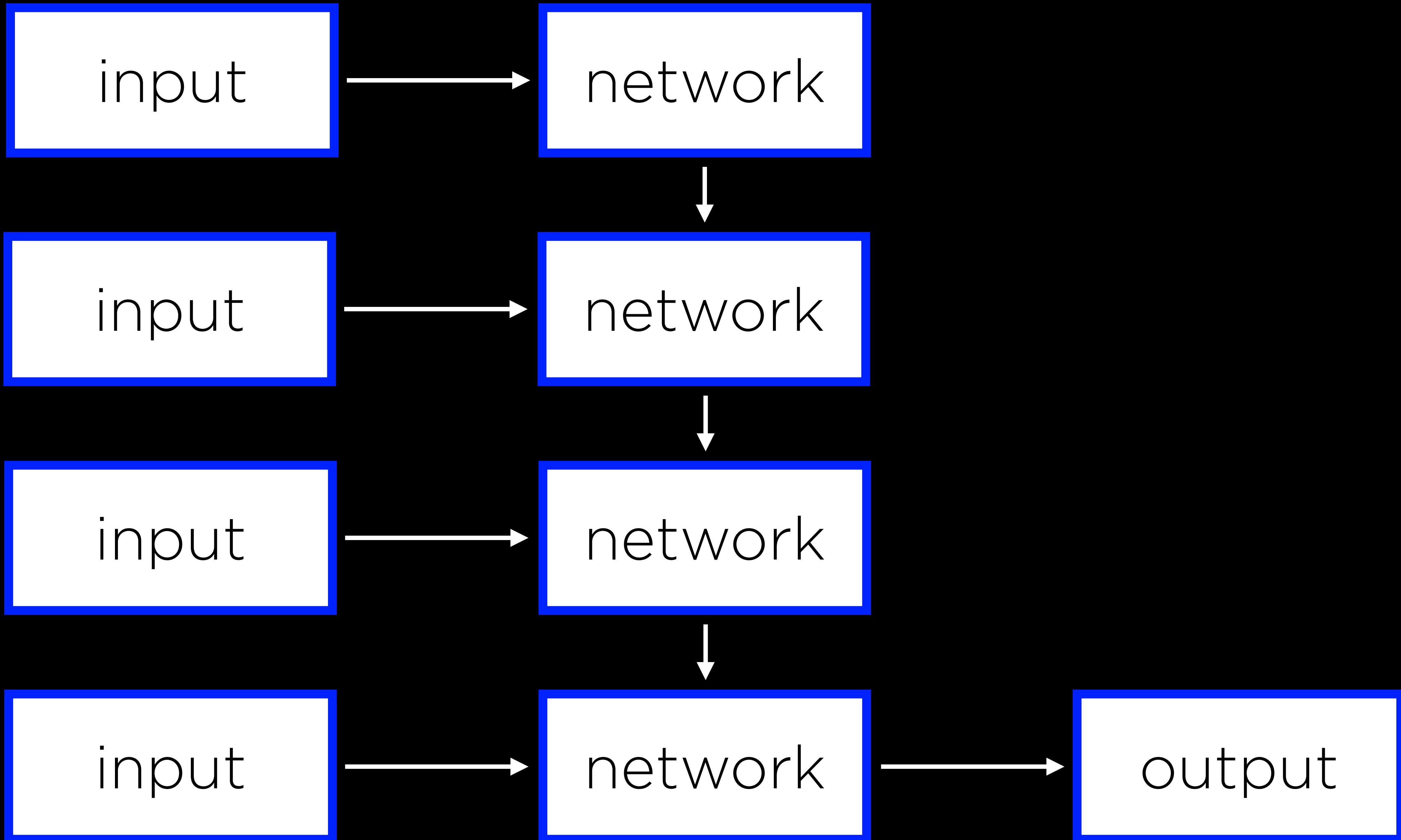




group of people walking in front of a building





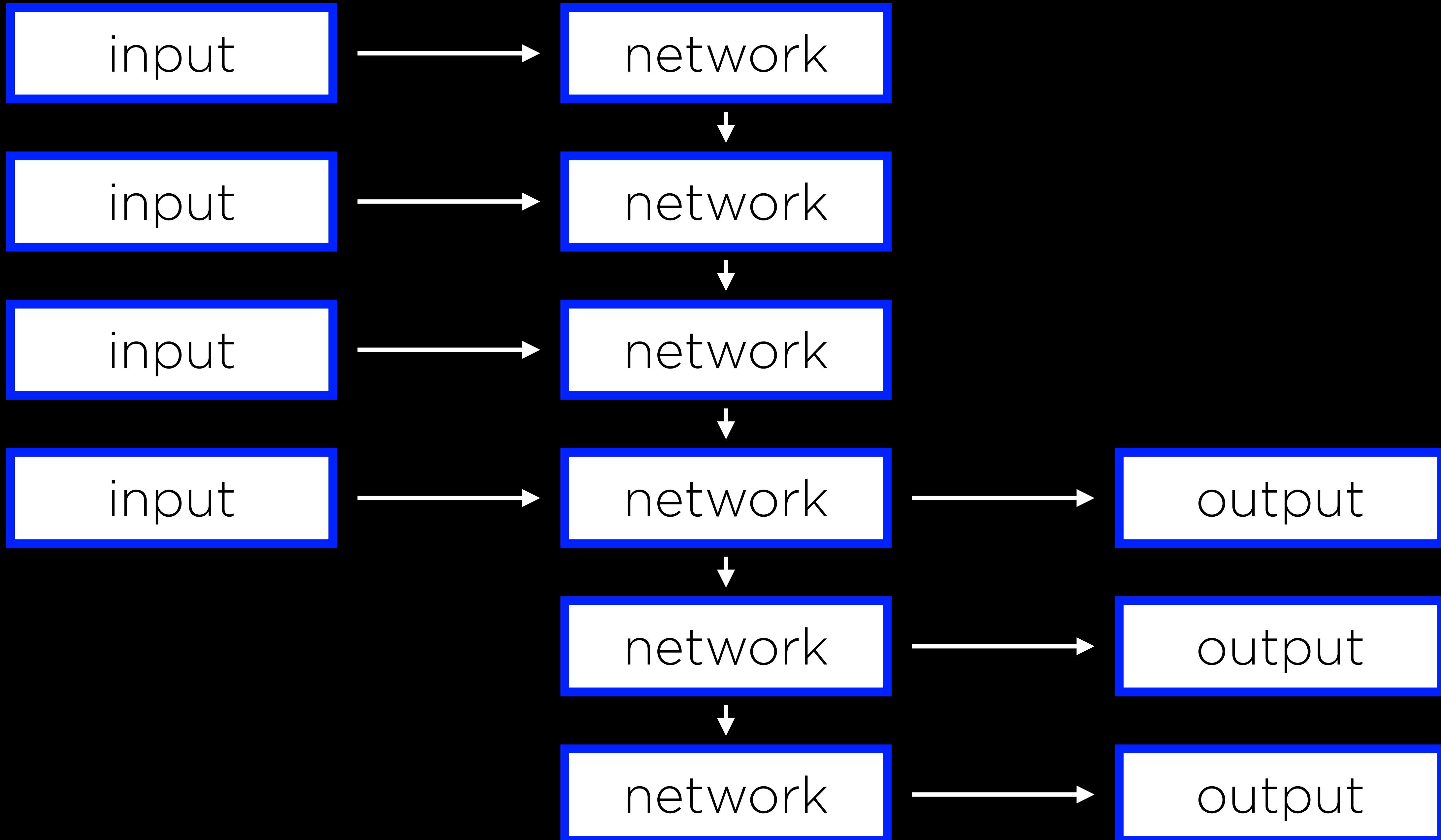




She is in the library.



她在圖書館



Neural Networks

Introduction to
Artificial Intelligence
with Python