ML@LSE - Bootcamp 5: Feedforward Neural Networks

Introduction: What are neural networks used for?

We've studied methods of supervised and unsupervised learning. Neural networks can perform both of these tasks. It also belongs Another class of ML: deep learning.
Within supervised learning, it can be used for classification and regression.

Neural networks are very polyvalent!

There are used in:

- Image recognition
- Speech recognition
- Credit risk modeling
- Time-series forecasting
- Algorithmic trading
- etc…

The intuition of NN

A- The structure of Neural Networks

1. **An intuitive example of a Neural Network**

a. Should I go to Brighton this weekend?
b. **Framing this with a perceptron**

2. **Feedforward Neural Network structure**

a. The sigmoid function
b. FNN is a network of perceptrons

B- Fitting the network

1. **Minimizing the Cost Function**

a. The Cost function of a network
b. Gradient descent

2. **Fitting the network using backpropagation**

a. Closer look at the gradient
b. Backpropagation

Should I go to Brighton this weekend?
➢ How do you make such a decision? What parameters do you consider?

1. How many friends are willing to go? → x_1
2. Will it be sunny? → x_2
3. How much work have I got to do? → x_3
4. How much money do I have → x_4

How to make the decision? Weight each parameter and decide!

Go if: $w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 \geq b$

$$w \cdot x \geq b$$

$$y(w \cdot x - b) = \begin{cases} 1 \text{ if } w \cdot x - b \geq 0 \\ 0 \text{ if } w \cdot x - b \leq 0 \end{cases}$$ This is called the activation function

The intuition of NN

A- The structure of Neural Networks

1. **An intuitive example of a Neural Network**

a. Should I go to Brighton this weekend?
b. **Framing this with a perceptron**

2. **Feedforward Neural Network structure**

a. The sigmoid function
b. FNN is a network of perceptrons

B- Fitting the network

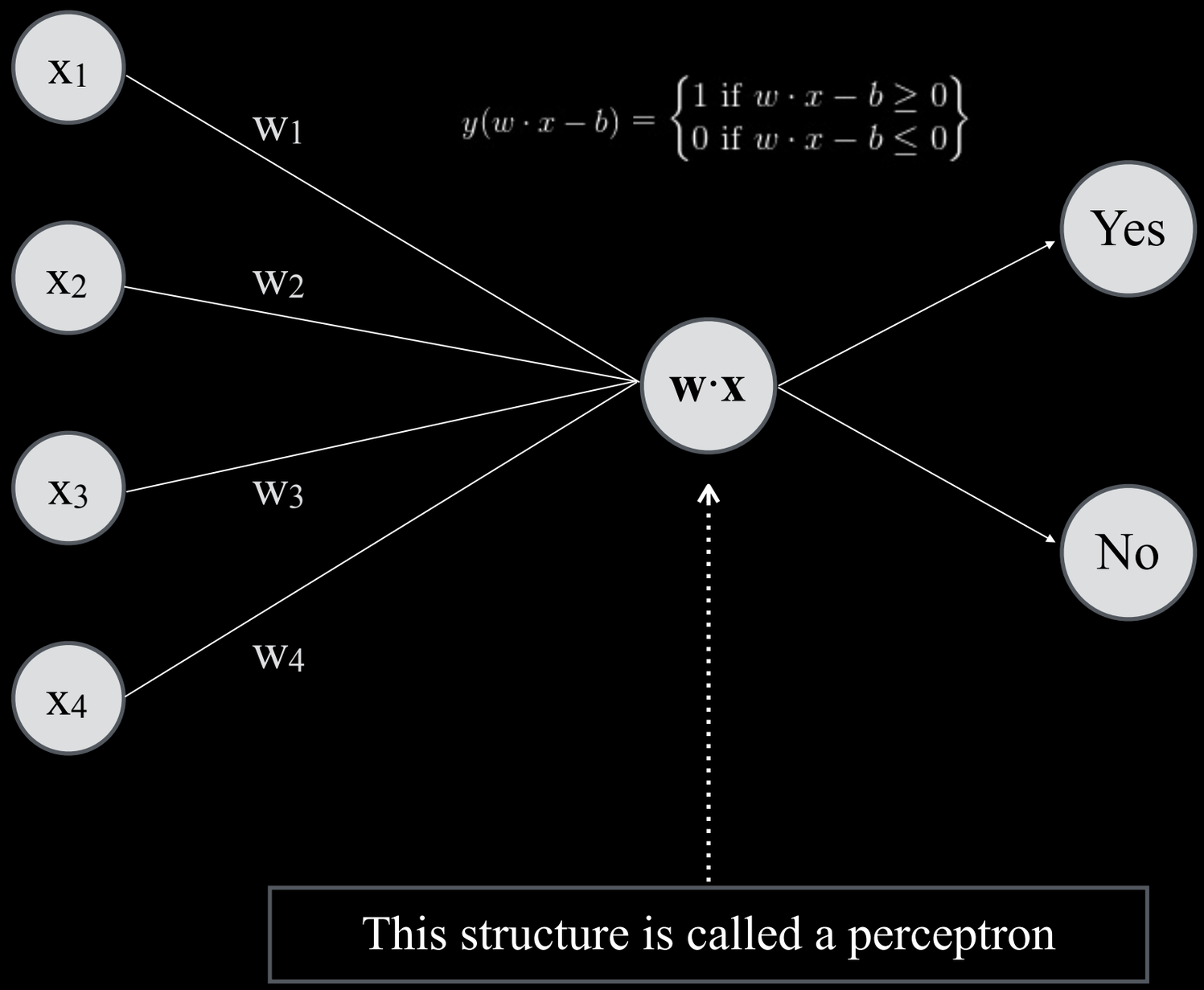
1. **Minimizing the Cost Function**

a. The Cost function of a network
b. Gradient descent

2. **Fitting the network using backpropagation**

a. Closer look at the gradient
b. Backpropagation

We can represent this schematically:

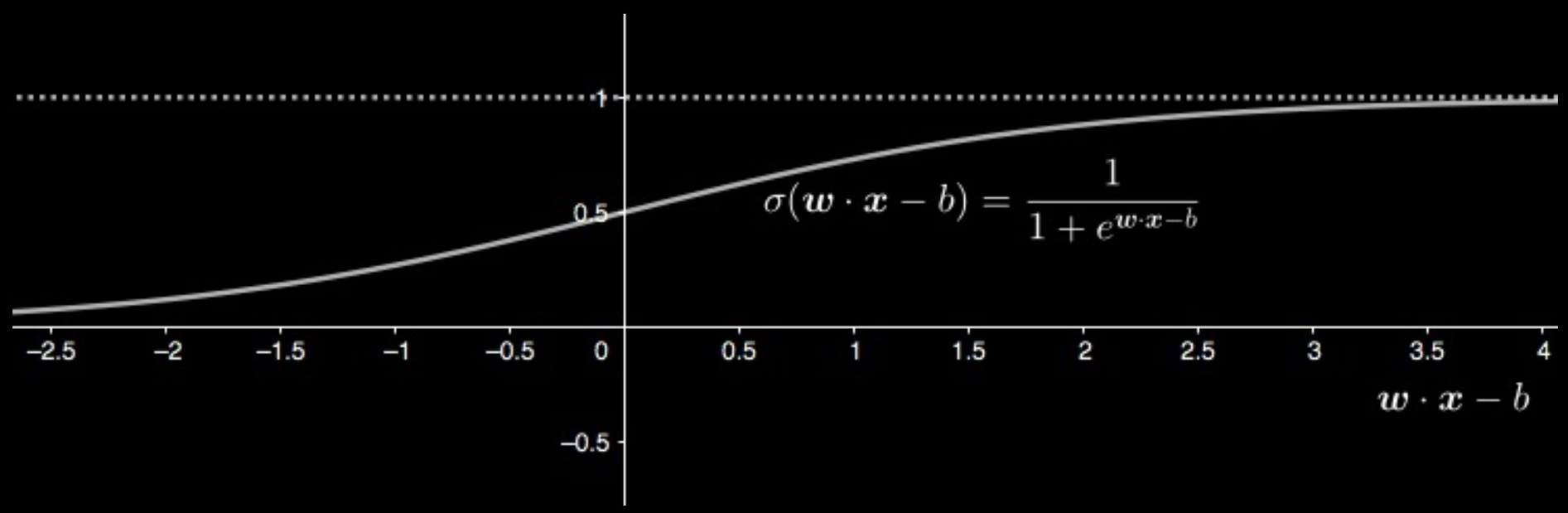


The sigmoid activation function

The activation function $y(w \cdot x - b)$ is binary (returns either 0 or 1).

In the following we'll use a more subtle activation function that returns any number between 0 and 1. You can think of these numbers as probabilities (if the activation function returns a number close to one, the probability that I will have a good time in Brighton is high).

This activation function is the sigmoid function: $\sigma(\mathbf{w} \cdot \mathbf{x} - b) = \dfrac{1}{1 + e^{-(\mathbf{w} \cdot \mathbf{x} - b)}}$



FNN as a network of percetrons

A feedforward neural network can be seen as a network of percetrons:

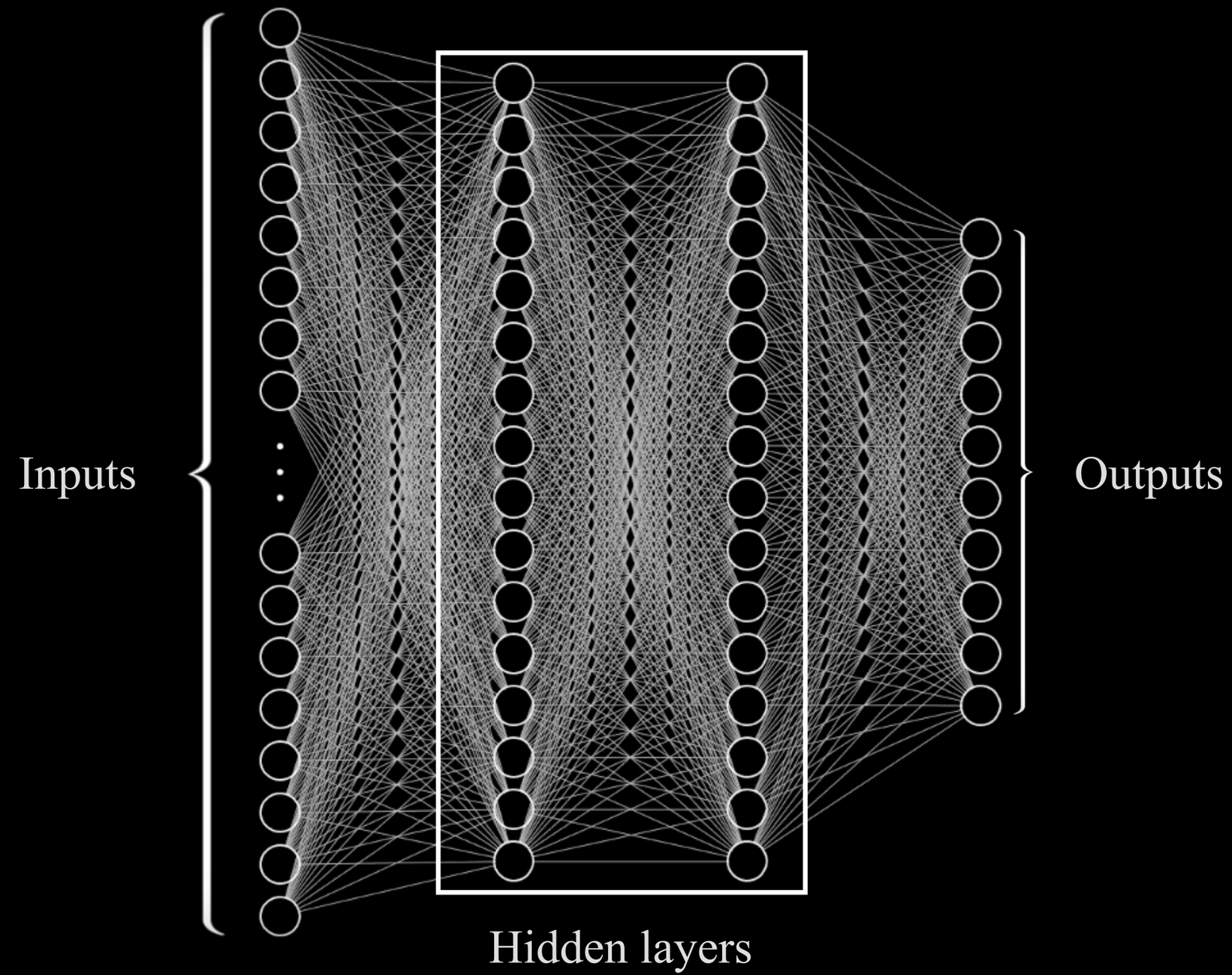


Image source: https://www.youtube.com/watch?v=aircAruvnKk&t=990s

FNN as a network of percetrons

A- The structure of Neural Networks

1. **An intuitive example of a Neural Network**

a. Should I go to Brighton this weekend?
b. Framing this with a perceptron

2. **Feedforward Neural Network structure**

a. The sigmoid function
b. **FNN is a network of perceptrons**

B- Fitting the network

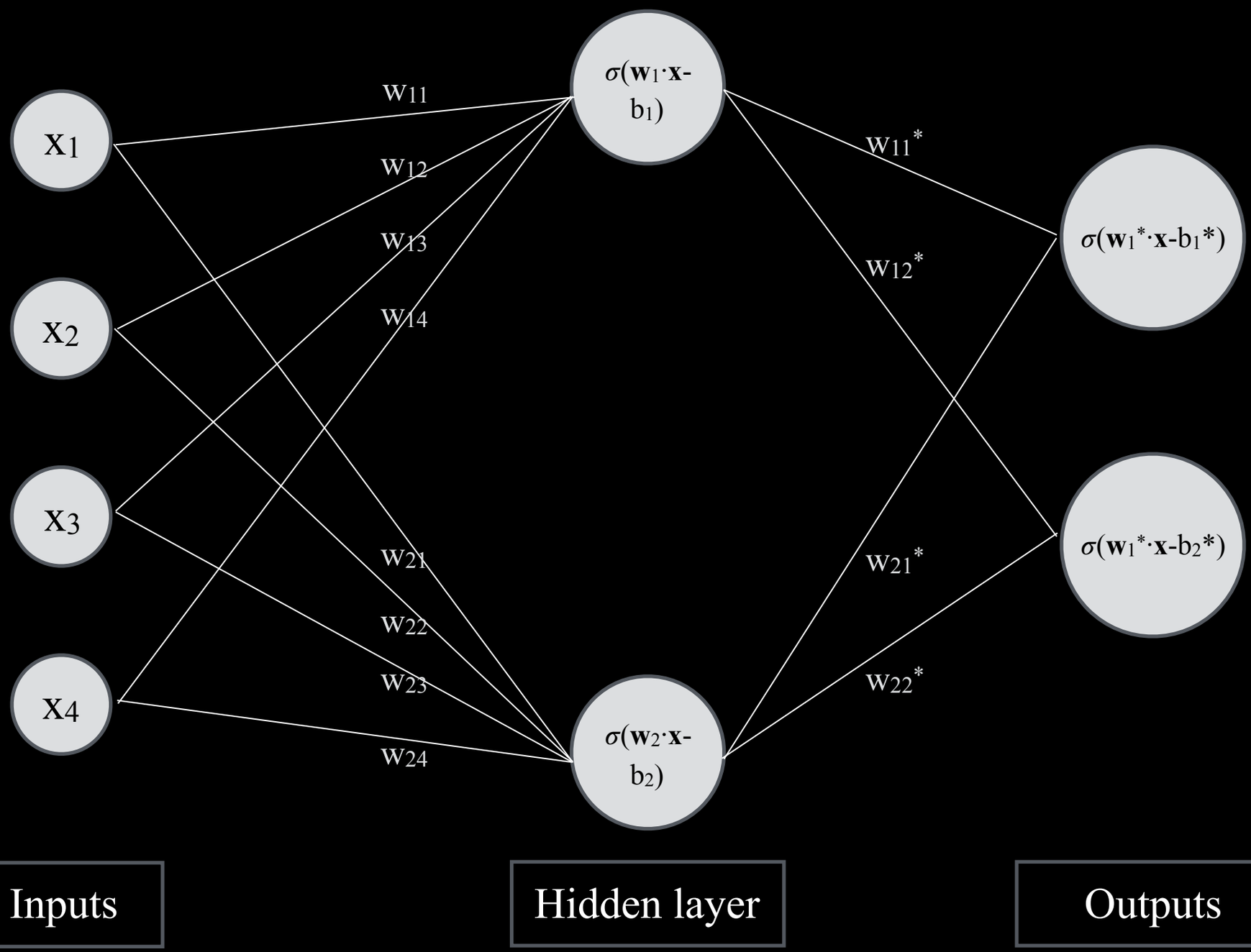
1. **Minimizing the Cost Function**

a. The Cost function of a network
b. Gradient descent

2. **Fitting the network using backpropagation**

a. Closer look at the gradient
b. Backpropagation

A simple neural network with one hidden layer:



FNN as a network of percetrons

Hence a network is a sort of function f(**x**) with inputs and outputs.
(The true mathematical term is a correspondence, as the network can have several outputs)

Importantly, this f(**x**) has many parameters, which are?

➢ The weights and biases are the parameters of the network!

As we've seen many times now, machine learning oftentimes consists in shaping a function properly, using the training data, so that this function has some predictive power.
With a neural network, we will "shape the function" by adjusting the weights and biases properly.

A practical example

A- The structure of Neural Networks

1. **An intuitive example of a Neural Network**

a. Should I go to Brighton this weekend?
b. Framing this with a perceptron

2. **Feedforward Neural Network structure**

a. The sigmoid function
b. **FNN is a network of perceptrons**

B- Fitting the network

1. **Minimizing the Cost Function**

a. The Cost function of a network
b. Gradient descent

2. **Fitting the network using backpropagation**

a. Closer look at the gradient
b. Backpropagation

Let’s look at a practical example:

➢ How can we use our neural network to classify these shapes?

What is the output?

- Probabilities assigned to each digit 0, 1, …, 9

What is the input?

- These numbers are represented by a grid of 28 by 28 pixels, whose brightness varies from 0 to 1. Hence the input will be the brightness of each of the 784 pixels.

A practical example

A- The structure of Neural Networks

1. **An intuitive example of a Neural Network**

a. Should I go to Brighton this weekend?
b. Framing this with a perceptron

2. **Feedforward Neural Network structure**

a. The sigmoid function
b. **FNN is a network of perceptrons**

B- Fitting the network

1. **Minimizing the Cost Function**

a. The Cost function of a network
b. Gradient descent

2. **Fitting the network using backpropagation**

a. Closer look at the gradient
b. Backpropagation

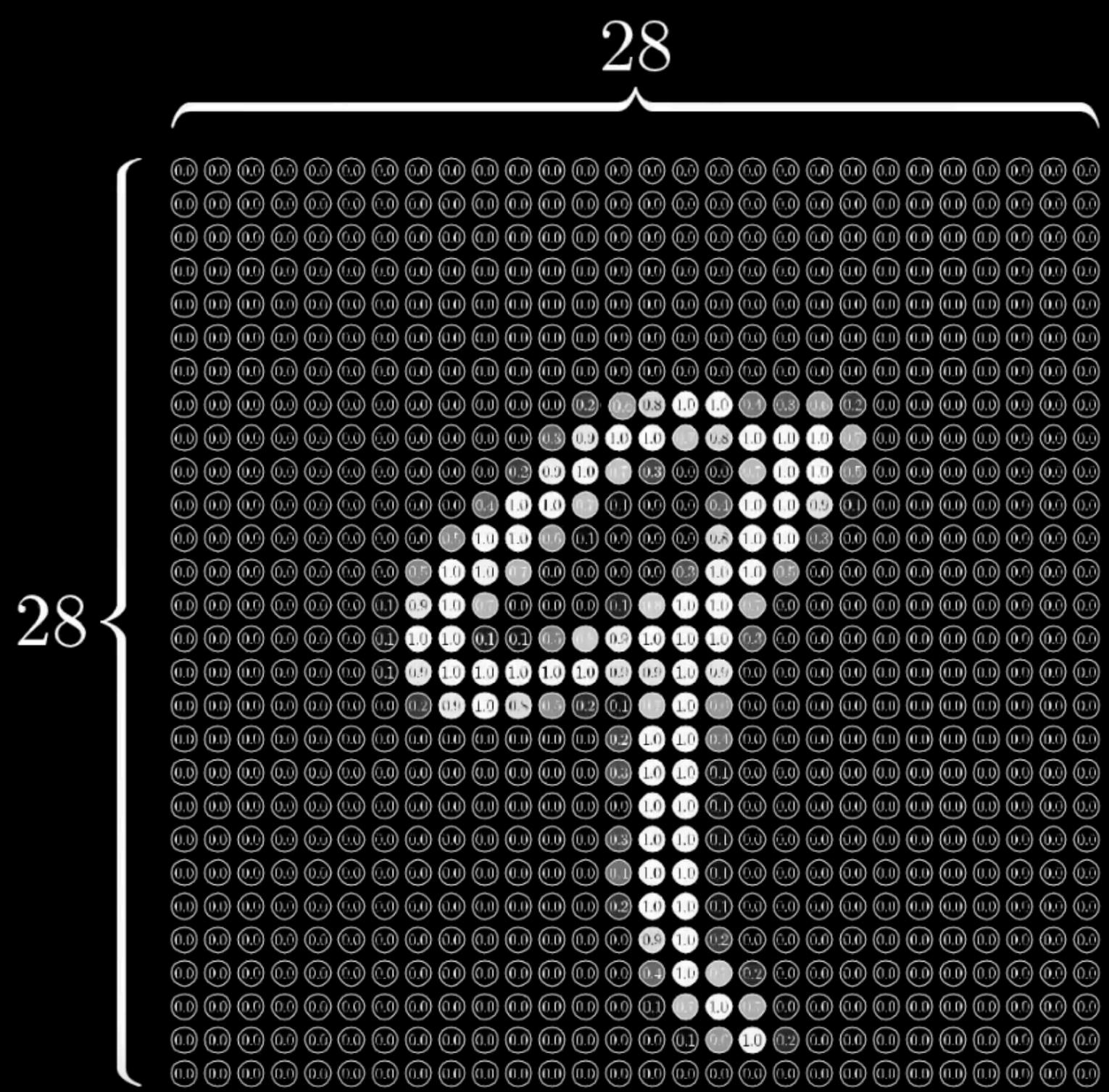


Image source: https://www.youtube.com/watch?v=aircAruvnKk&t=990s

A practical example

A- The structure of Neural Networks

1. **An intuitive example of a Neural Network**

a. Should I go to Brighton this weekend?
b. Framing this with a perceptron

2. **Feedforward Neural Network structure**

a. The sigmoid function
b. **FNN is a network of perceptrons**

B- Fitting the network

1. **Minimizing the Cost Function**

a. The Cost function of a network
b. Gradient descent

2. **Fitting the network using backpropagation**

a. Closer look at the gradient
b. Backpropagation

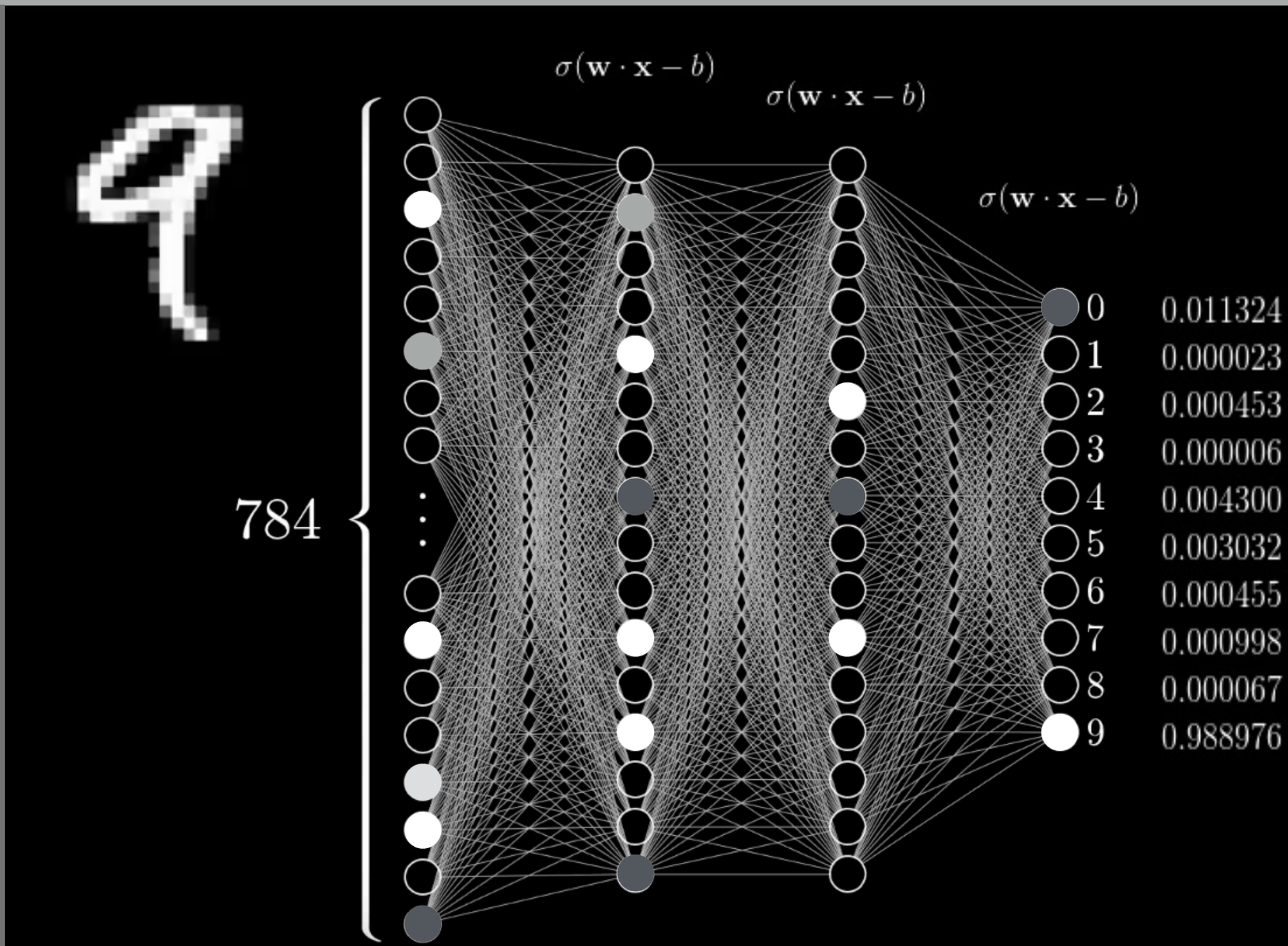


Image source: https://www.youtube.com/watch?v=aircAruvnKk&t=990s

Cost

How do we fit the parameters?

➢ As usual, using a cost function:

Predicted	Actual
0.011324	0
0.000023	0
0.000453	0
0.000006	0
0.004300	0
0.003032	0
0.000455	0
0.000998	0
0.000067	0
0.988976	1

Take the sum of square differences:

$$(0.011324 - 0)^2 + \ldots + (0.988976 - 1)^2$$

$$= \sum_{n=1}^{N} (\hat{y}_n - y_n)^2$$

Suppose we have sample of S written digits

$$Cost = \frac{1}{S} \sum_{s=1}^{S} \left(\sum_{n=1}^{N} (\hat{y}_{sn} - y_{sn}) \right)$$

$$Cost = C(\mathbf{w}, \mathbf{b})$$

We want to solve:

$$\min_{\mathbf{w},\mathbf{b}} \quad C(\mathbf{w}, \mathbf{b})$$

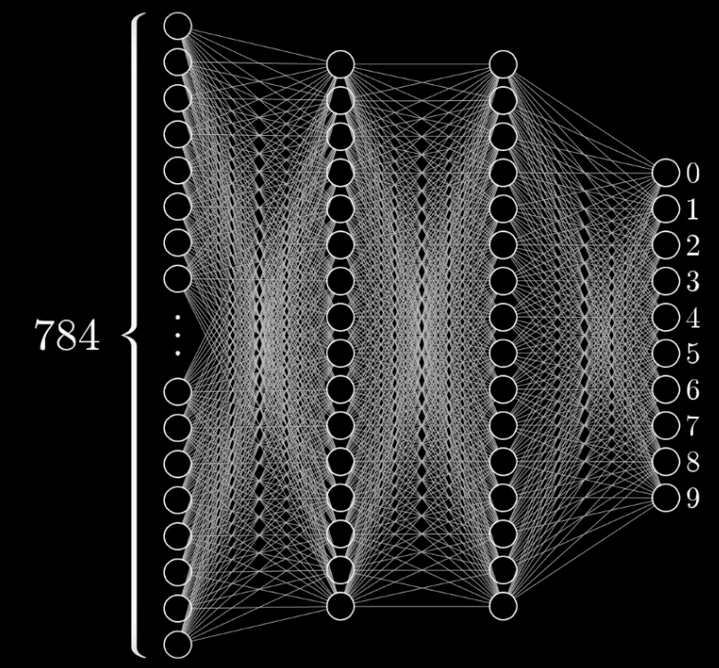
Cost is a complicated function!

How to solve $\min_{\mathbf{w},\mathbf{b}} C(\mathbf{w},\mathbf{b})$??

$$\frac{\partial C(\mathbf{w},\mathbf{b})}{\partial \mathbf{w}} = \mathbf{0}, \frac{\partial C(\mathbf{w},\mathbf{b})}{\partial \mathbf{b}} = \mathbf{0}$$

Maybe not, C has too many parameters!

What solution?



Let’s use an analogy: imagine you’re in Scotland on the top of a hill. You want to reach a valley but there’s fog and you can’t see further than 10 meters away. What is your best strategy?

Gradient descent

Gradient descent:

Let **W** represent the set of both weights **w** and biases **b**, i.e. **W** is the set of all parameters in the network that we want to optimize.

- Start with a random $\mathbf{W}_0$, and then update it (go downhill) according to:

$$\mathbf{W}_{n+1} = \mathbf{W}_n - \eta \nabla C(\mathbf{W}_n), \qquad \eta > 0$$

This equation means: "wherever you are now, go down as steep as you can."

An approximation of this method, stochastic graduate descent (SGD), is usually use.

3D model

$$\nabla C(\mathbf{W}) = \begin{bmatrix} \frac{\partial C}{\partial w_1} \\ \frac{\partial C}{\partial w_2} \\ \cdots \\ \frac{\partial C}{\partial w_m} \\ \frac{\partial C}{\partial b_1} \\ \cdots \\ \frac{\partial C}{\partial b_k} \end{bmatrix}$$

Partial derivatives

A- The structure of Neural Networks

1. **An intuitive example of a Neural Network**

a. Should I go to Brighton this weekend?
b. Framing this with a perceptron

2. **Feedforward Neural Network structure**

a. The sigmoid function
b. FNN is a network of perceptrons

B- Fitting the network

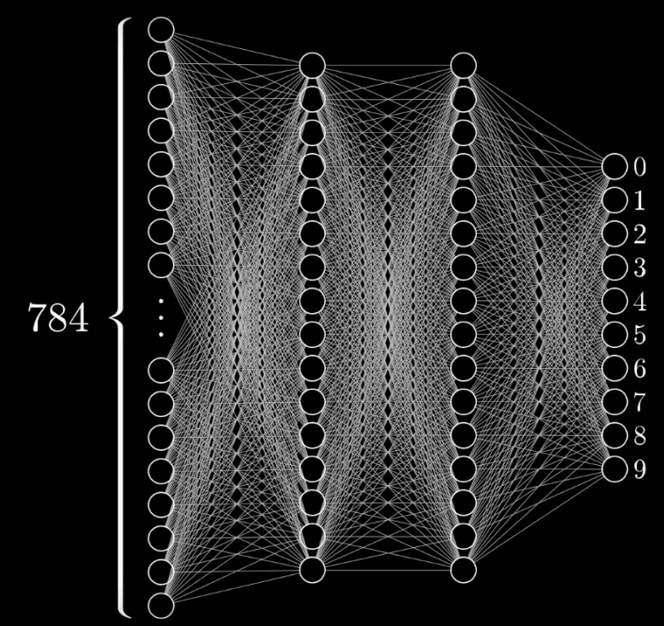
1. **Minimizing the Cost Function**

a. The Cost function of a network
b. Gradient descent

2. **Fitting the network using backpropagation**

a. Closer look at the gradient
b. Backpropagation

$$\nabla C(\mathbf{W}) = \begin{bmatrix} \frac{\partial C}{\partial w_1} \\ \frac{\partial C}{\partial w_2} \\ \cdots \\ \frac{\partial C}{\partial w_m} \\ \frac{\partial C}{\partial b_1} \\ \cdots \\ \frac{\partial C}{\partial b_k} \end{bmatrix}$$



Backpropagation

A- The structure of Neural Networks

---

1. **An intuitive example of a Neural Network**

a. Should I go to Brighton this weekend?
b. Framing this with a perceptron

2. **Feedforward Neural Network structure**

a. The sigmoid function
b. FNN is a network of perceptrons

B- Fitting the network

---

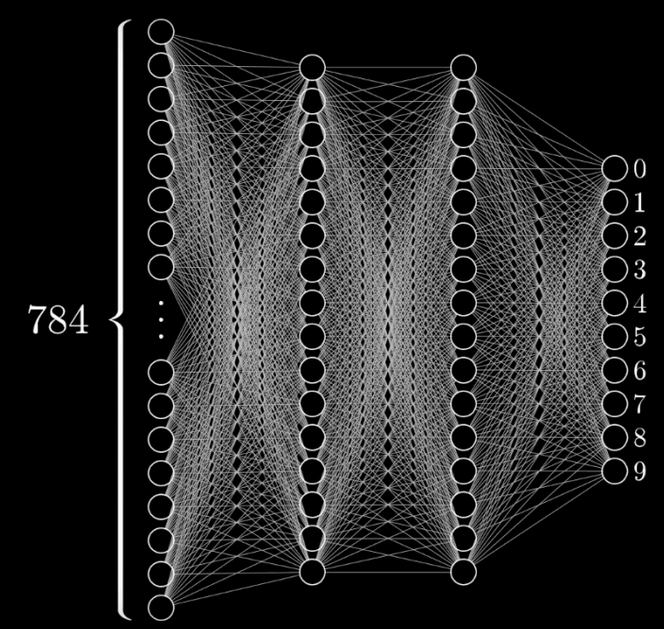
1. **Minimizing the Cost Function**

a. The Cost function of a network
b. Gradient descent

2. **Fitting the network using backpropagation**

a. Closer look at the gradient
b. **Backpropagation**

$$\nabla C(\mathbf{W}) = \begin{bmatrix} \frac{\partial C}{\partial w_1} \\ \frac{\partial C}{\partial w_2} \\ \cdots \\ \frac{\partial C}{\partial w_m} \\ \frac{\partial C}{\partial b_1} \\ \cdots \\ \frac{\partial C}{\partial b_k} \end{bmatrix}$$



$$\frac{d}{dx} f(g(x)) = \frac{df}{dg} \times \frac{dg}{dx}$$

$$\frac{\partial C}{\partial w_2} = \frac{\partial C}{\partial w_1} \times \frac{\partial w_1}{\partial w_2}$$

w_2

w_3

$\sigma(w_2+w_3-b)$

$\frac{\partial C}{\partial w_1}$

w_1

$\sigma(\mathbf{w}\cdot\mathbf{x}-b)$

$(\hat{y}_n - y_n)^2$

$C(\mathbf{W})$

$$\frac{\partial C}{\partial w_2} = \frac{\partial C}{\partial w_1} \times \frac{\partial w_1}{\partial w_2}$$

Backpropagation

A- The structure of Neural Networks

---

1. **An intuitive example of a Neural Network**

a. Should I go to Brighton this weekend?
b. Framing this with a perceptron

2. **Feedforward Neural Network structure**

a. The sigmoid function
b. FNN is a network of perceptrons

B- Fitting the network

---

1. **Minimizing the Cost Function**

a. The Cost function of a network
b. Gradient descent

2. **Fitting the network using backpropagation**

a. Closer look at the gradient
b. Backpropagation

http://www.emergentmind.com/neural-network

Thanks for coming!