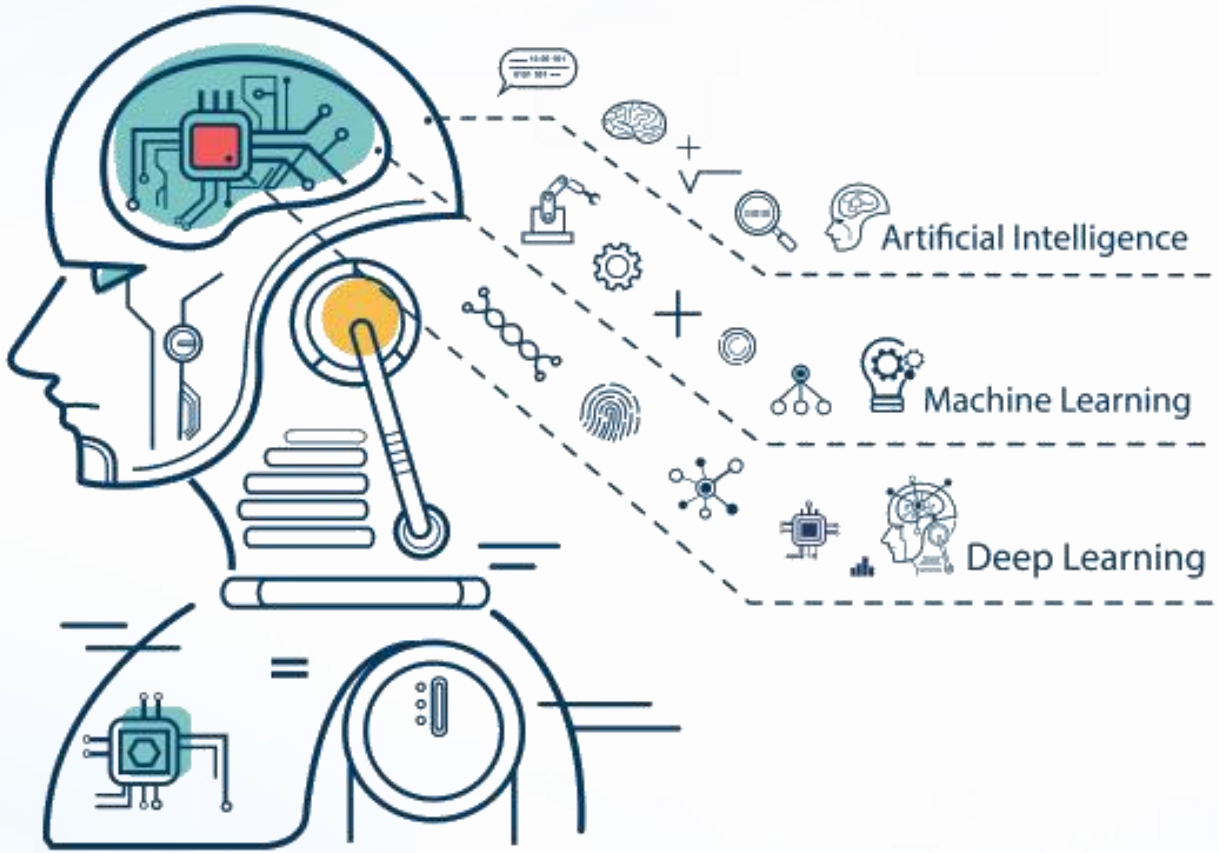
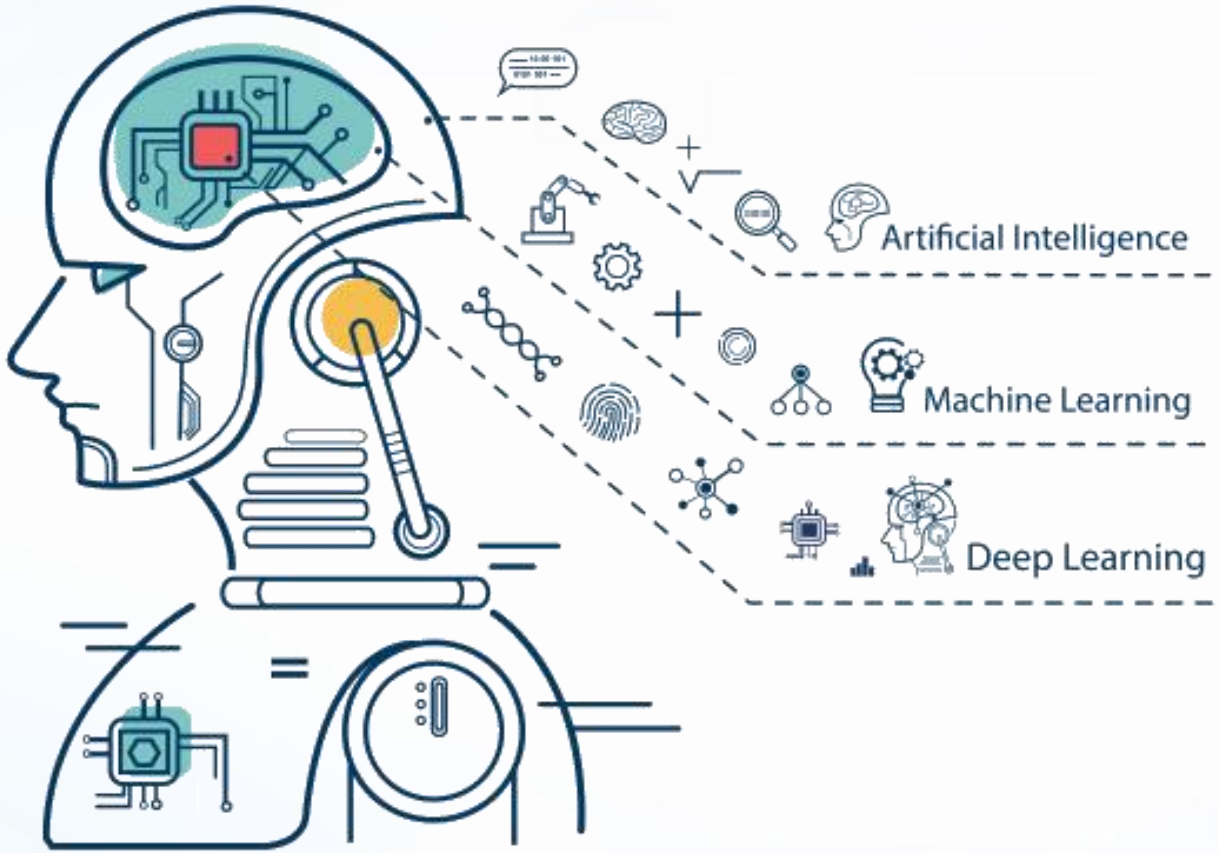


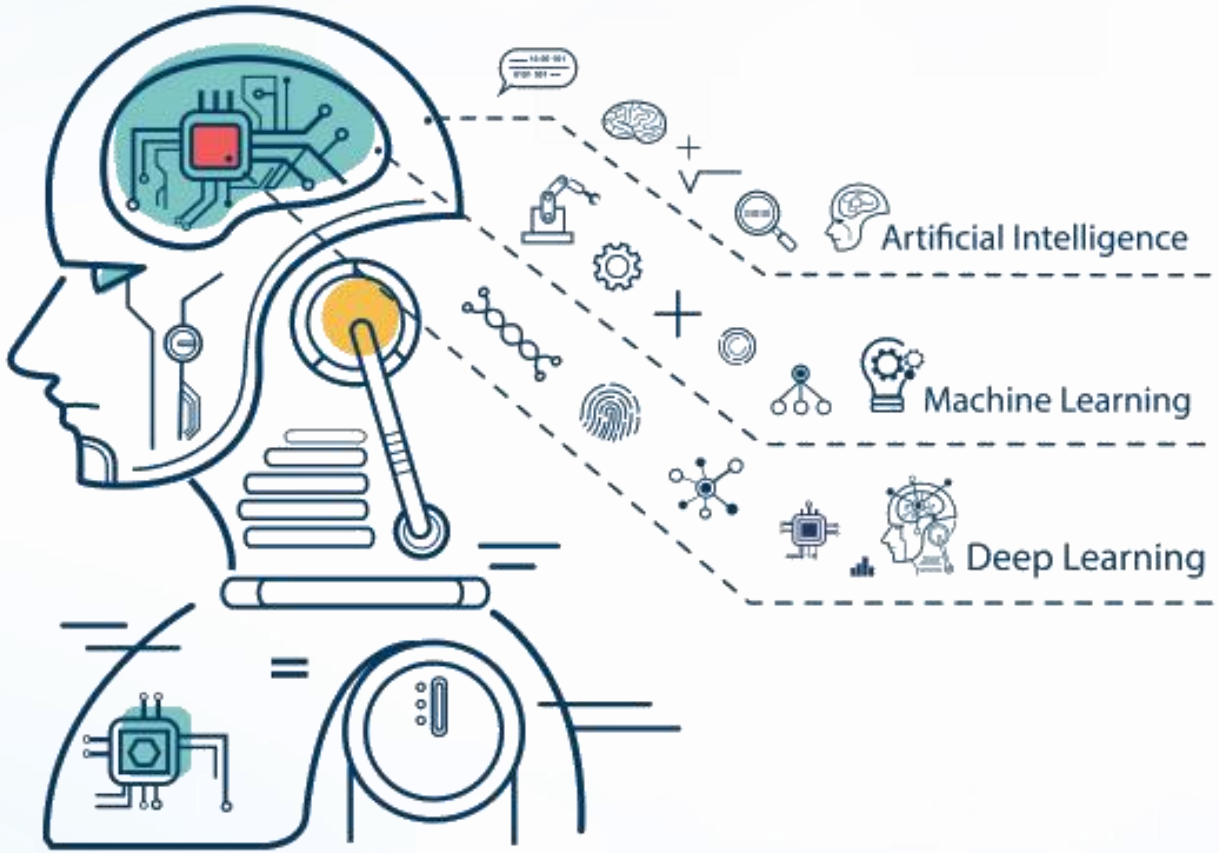
Deep Learning for Dummies

Don't model the World.
Model the Mind.



AGENDA

- AI and ML – How are they **related**?
- **Types** of AI and Machine Learning
- **What** is Neural Network?
- How is the Neural Network **learning**?
  - Gradient Descent
  - Back Propagation
- **When** and **where** can or should you use Deep Learning?



AGENDA

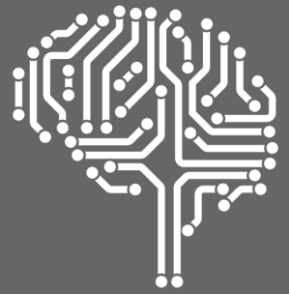
- **AI and ML – How are they related**?
- **Types** of AI and Machine Learning
- **What** is Neural Network?
- How is the Neural Network **learning**?
  - Gradient Descent
  - Back Propagation
- **When** and **where** can or should you use Deep Learning?

Machine Learning is a technique of artificial intelligence where systems can learn from data, identify patterns and make decisions with minimal human intervention

RELATION BETWEEN AI, ML & DL

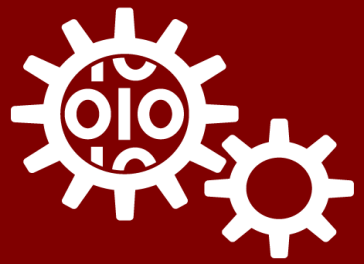
ARTIFICIAL INTELLIGENCE

THE SCIENCE OF ENABLING COMPUTERS TO MIMIC HUMAN BEHAVIOR



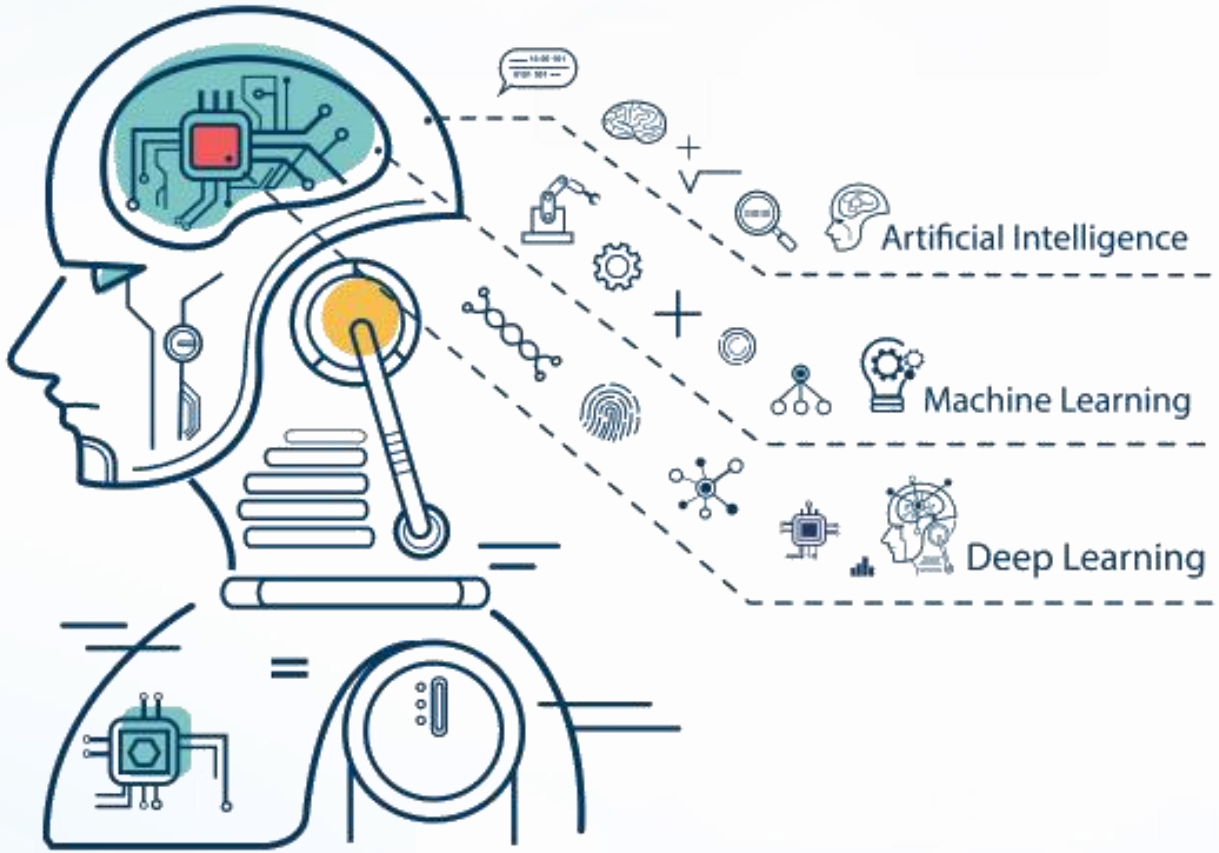
MACHINE LEARNING

AI TECHNIQUE TO GIVE COMPUTER THE ABILITY TO LEARN WITHOUT EXPLICIT PROGRAMMING



DEEP LEARNING

A METHOD TO PERFORM HIGH DIMENSIONAL MATRIX COMPUTATION TO EXTRACT FEATURES FROM DATA

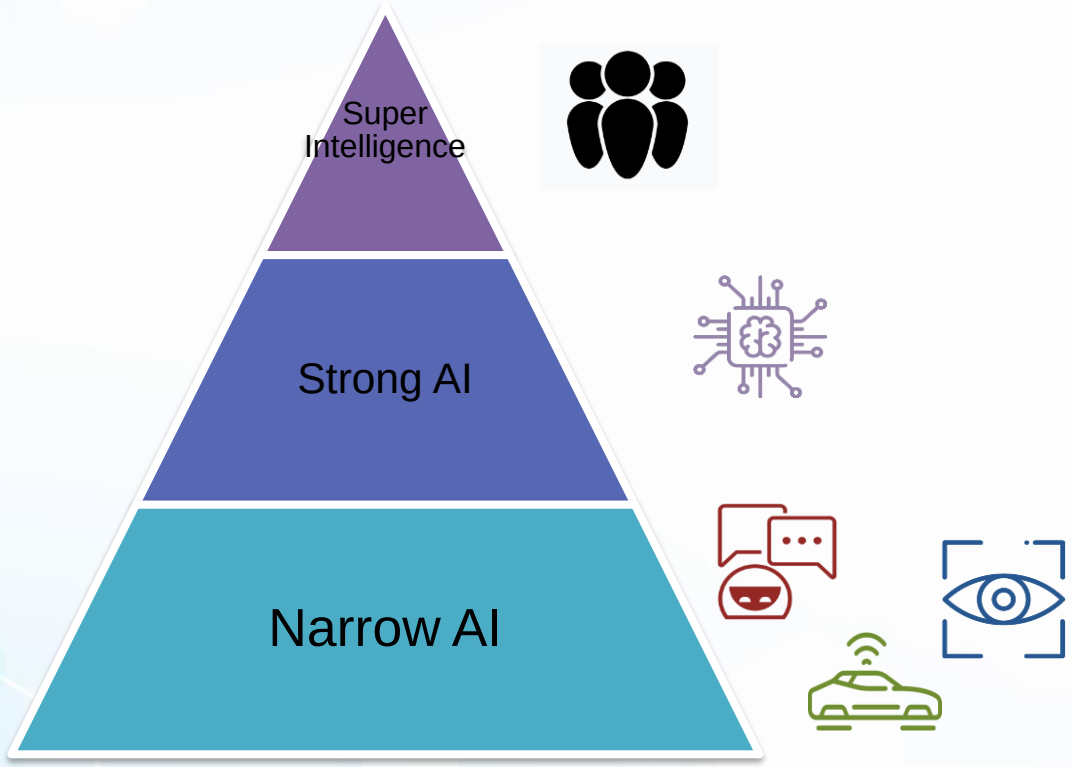


AGENDA

- AI and ML – How are they **related**?
- **Types of AI and Machine Learning**
- **What** is Neural Network?
- How is the Neural Network **learning**?
  - Gradient Descent
  - Back Propagation
- **When** and **where** can or should you use Deep Learning?

AI can be classified based on level of intelligence and functionality

Level of Intelligence



Level of Functionality

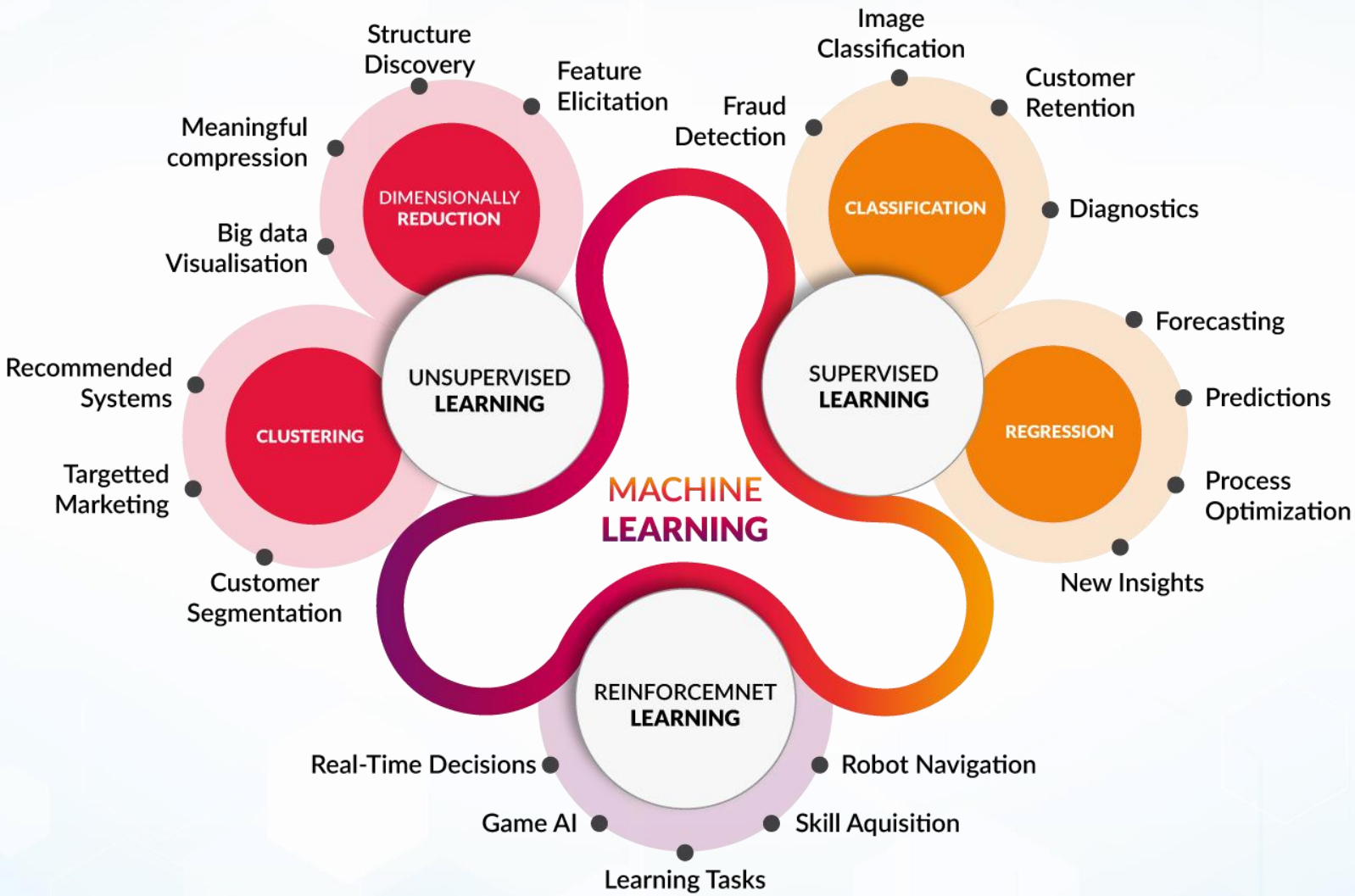
Reactive Machine
Doesn't have past memory and cannot use past information to information for the future actions

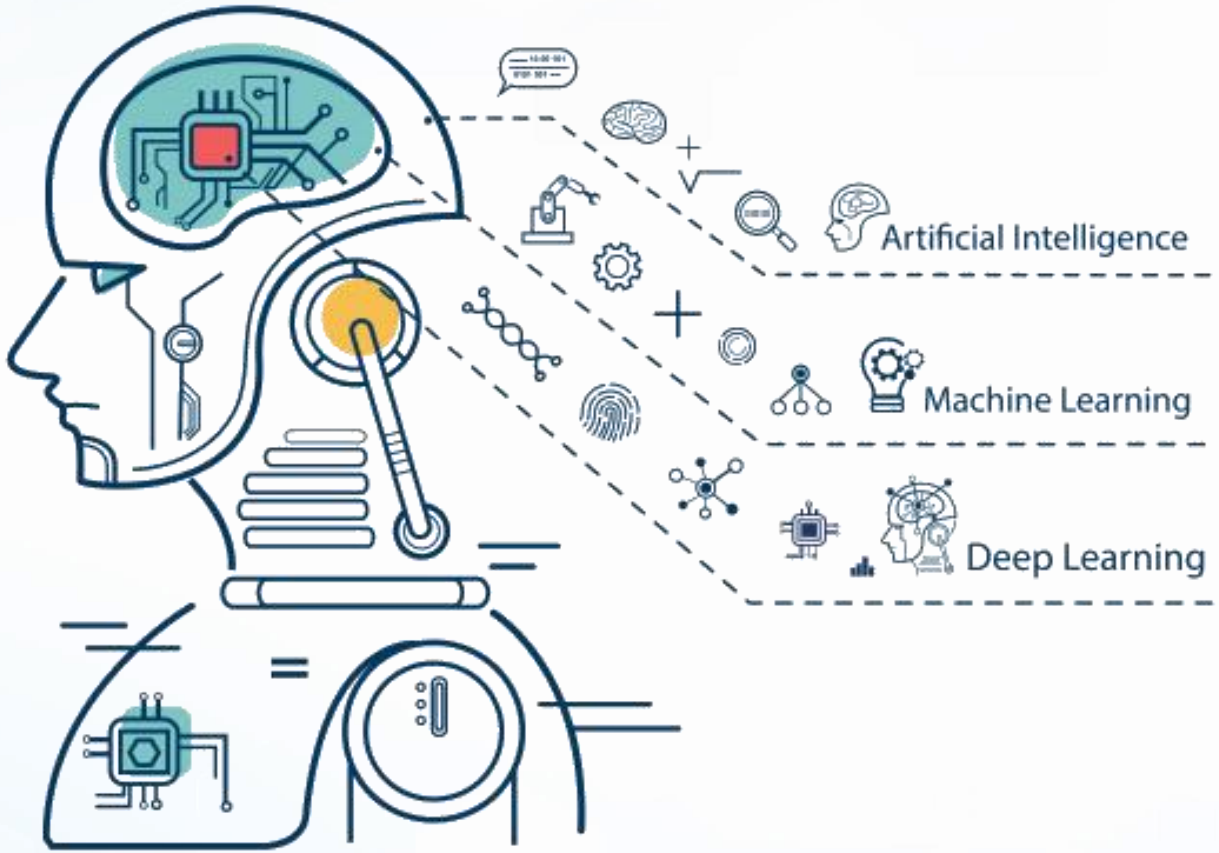
Limited Memory
Use short-term past experiences to drive decisions in the not so distant future

Theory of Mind
Understand people's emotion, belief, thoughts, expectations and be able to interact socially

Self Awareness
Extension of Theory of Mind, having self-consciousness, awareness, sentiment and intelligence

Machine Learning Algorithms are classified based their purpose

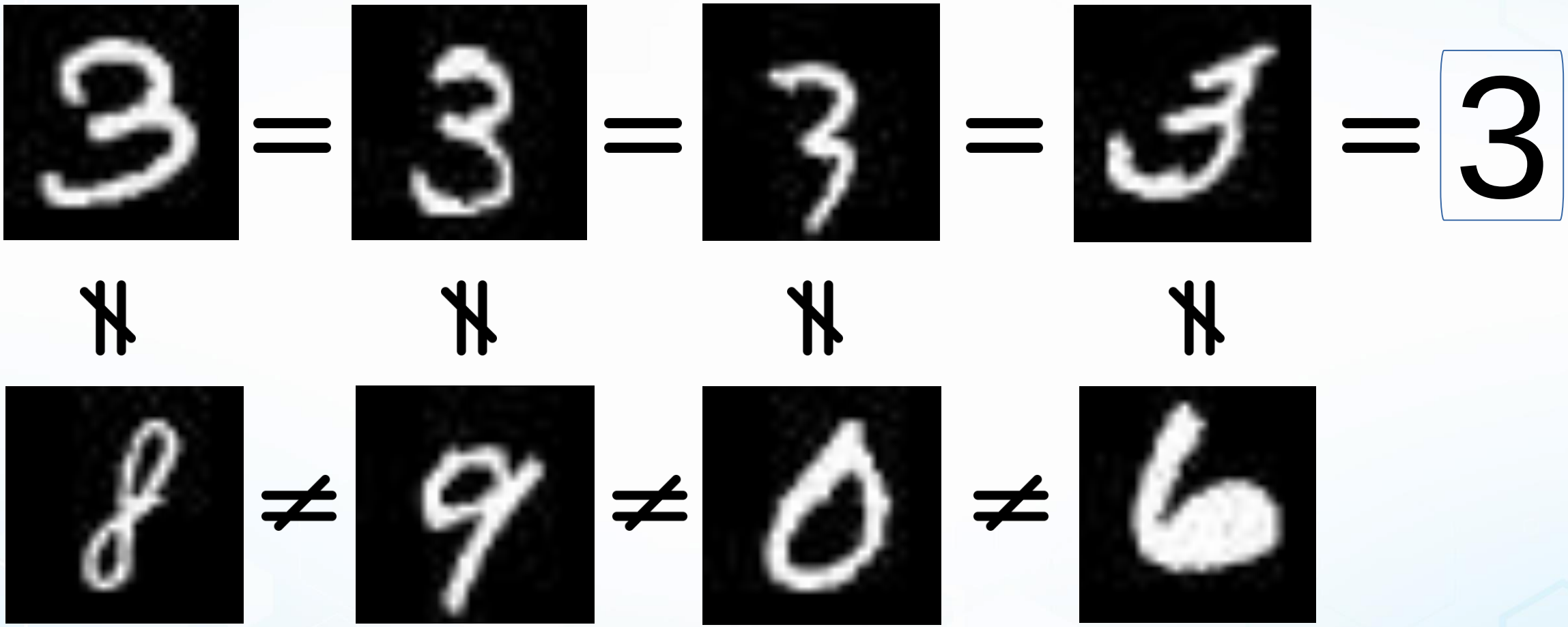




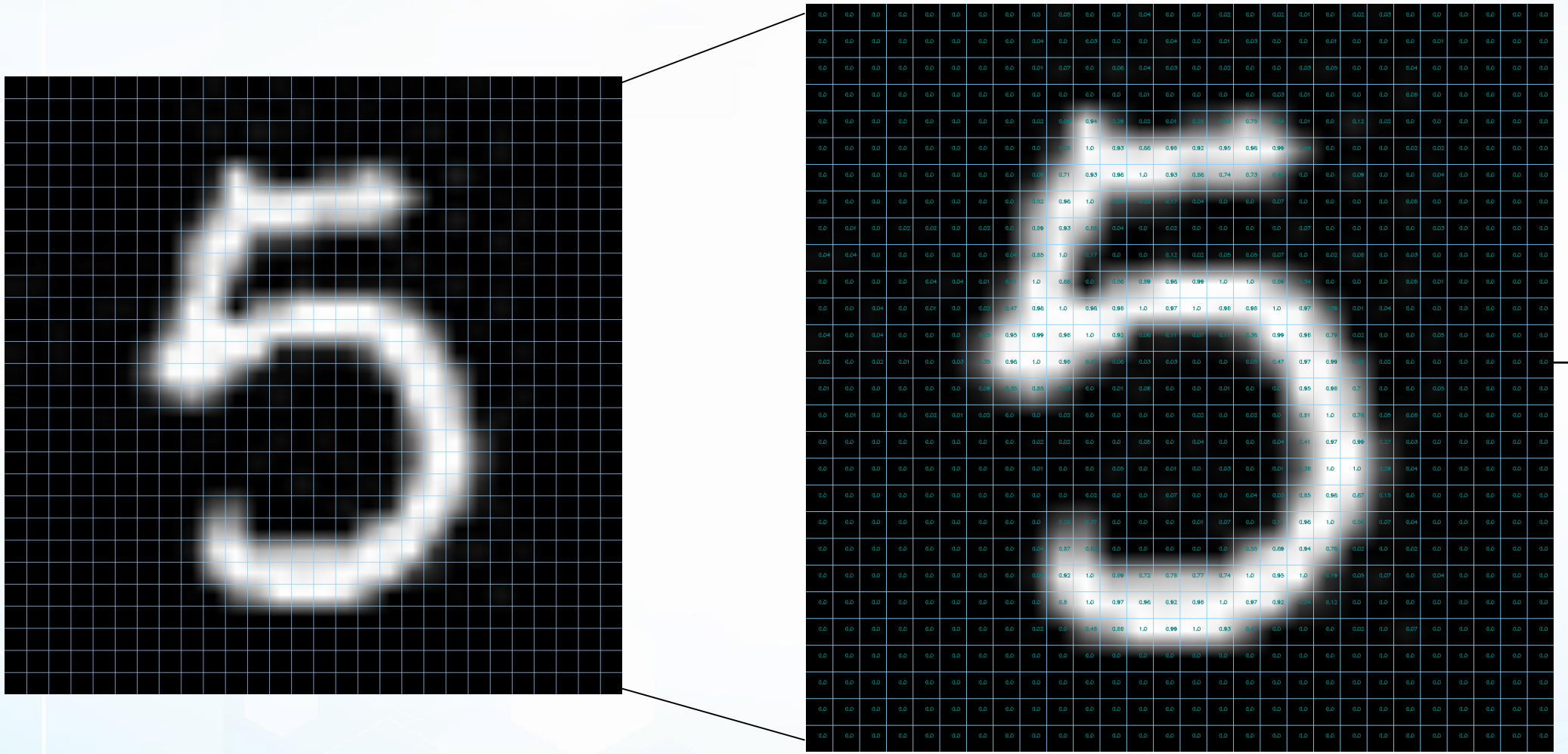
AGENDA

- AI and ML – How are they **related**?
- **Types** of AI and Machine Learning
- **What is Neural Network?**
- How is the Neural Network **learning**?
  - Gradient Descent
  - Back Propagation
- **When** and **where** can or should you use Deep Learning?

Our brain is one of the most complex system known to mankind developed over millions of years of evolution



Computer visualizes image as a 2D matrix of pixel values and hence image processing is nothing but intense matrix computation



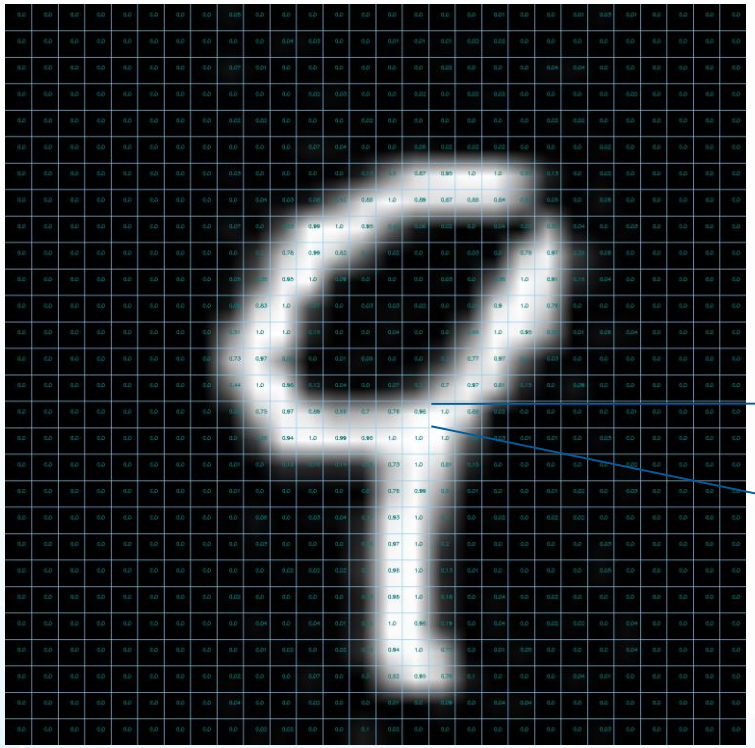
0
1
2 ?
3
4
5
6
7
8
9

Handwritten digit recognition is like the “hello world” for Neural Network

Container holding
a **number**

0.75 1.0 0.3

0 to 1



28

28

28 X 28 = 784

0.98

Activation

Neural Network

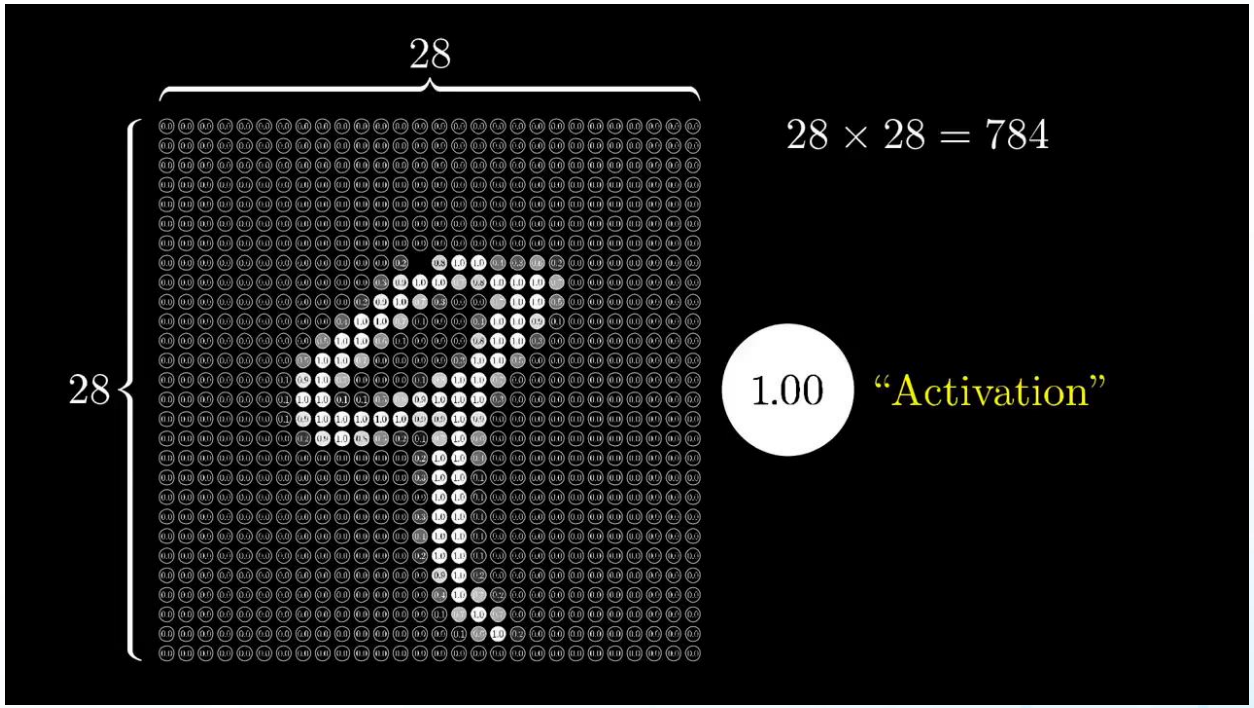
What are the Neurons?

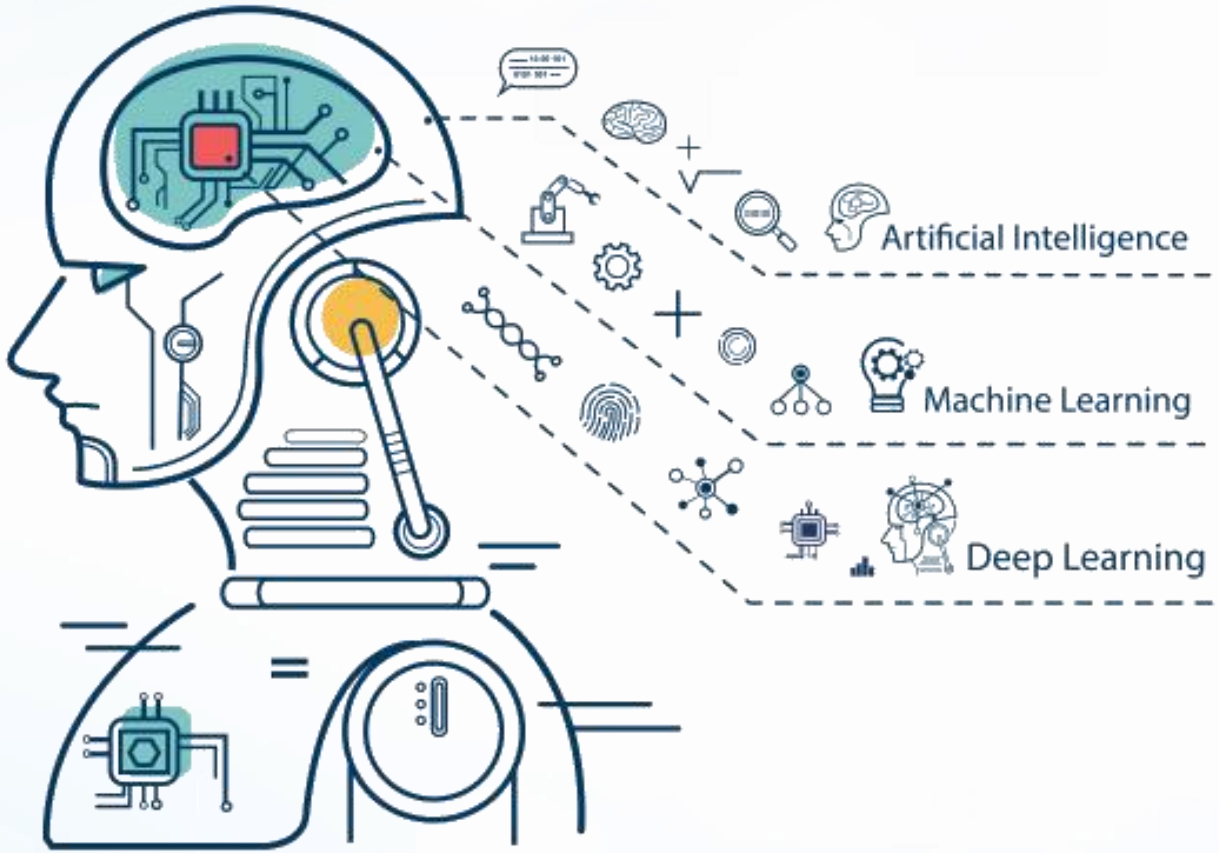
How are they connected?

Input Layer

Hidden Layer

Output Layer

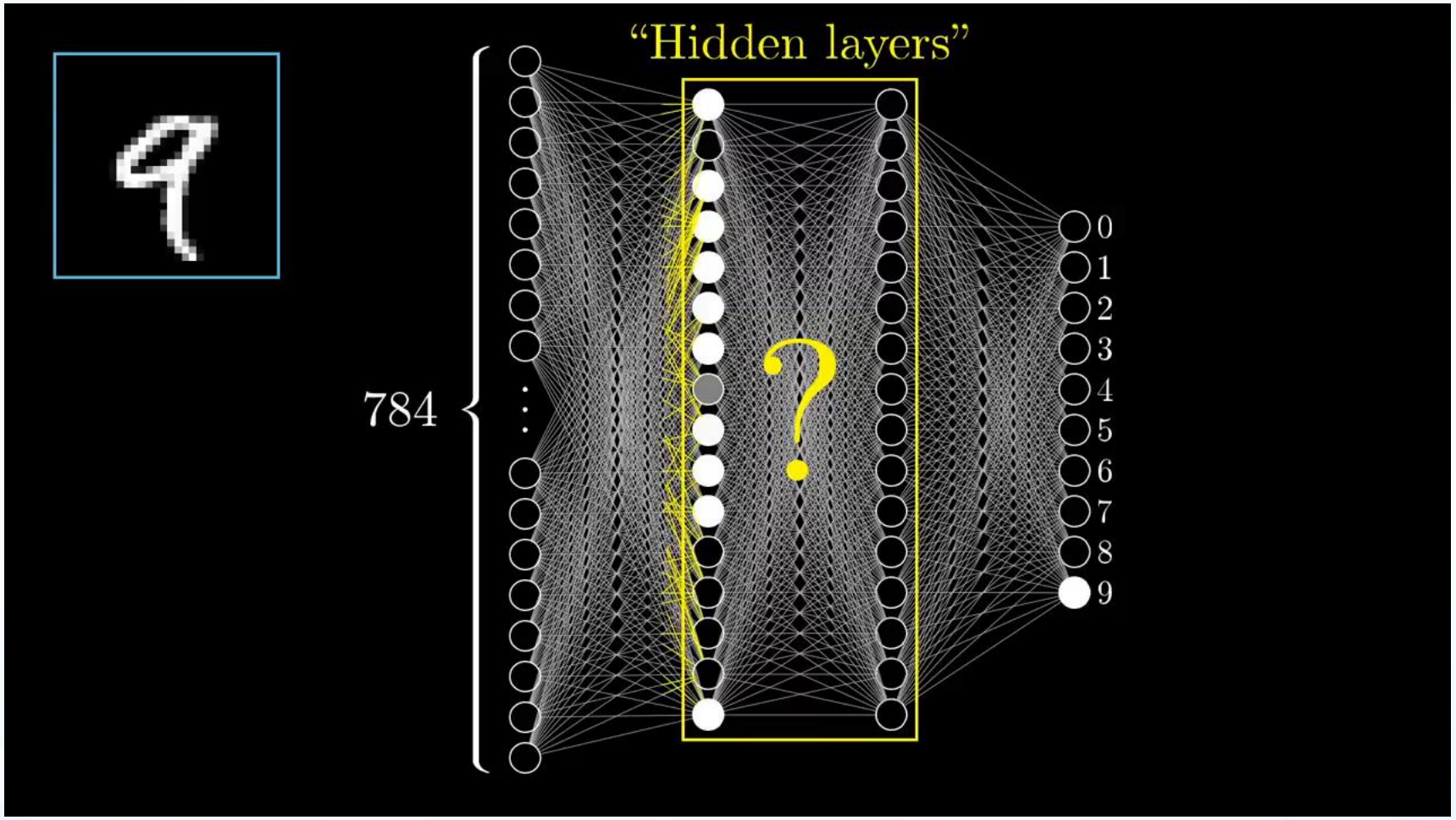




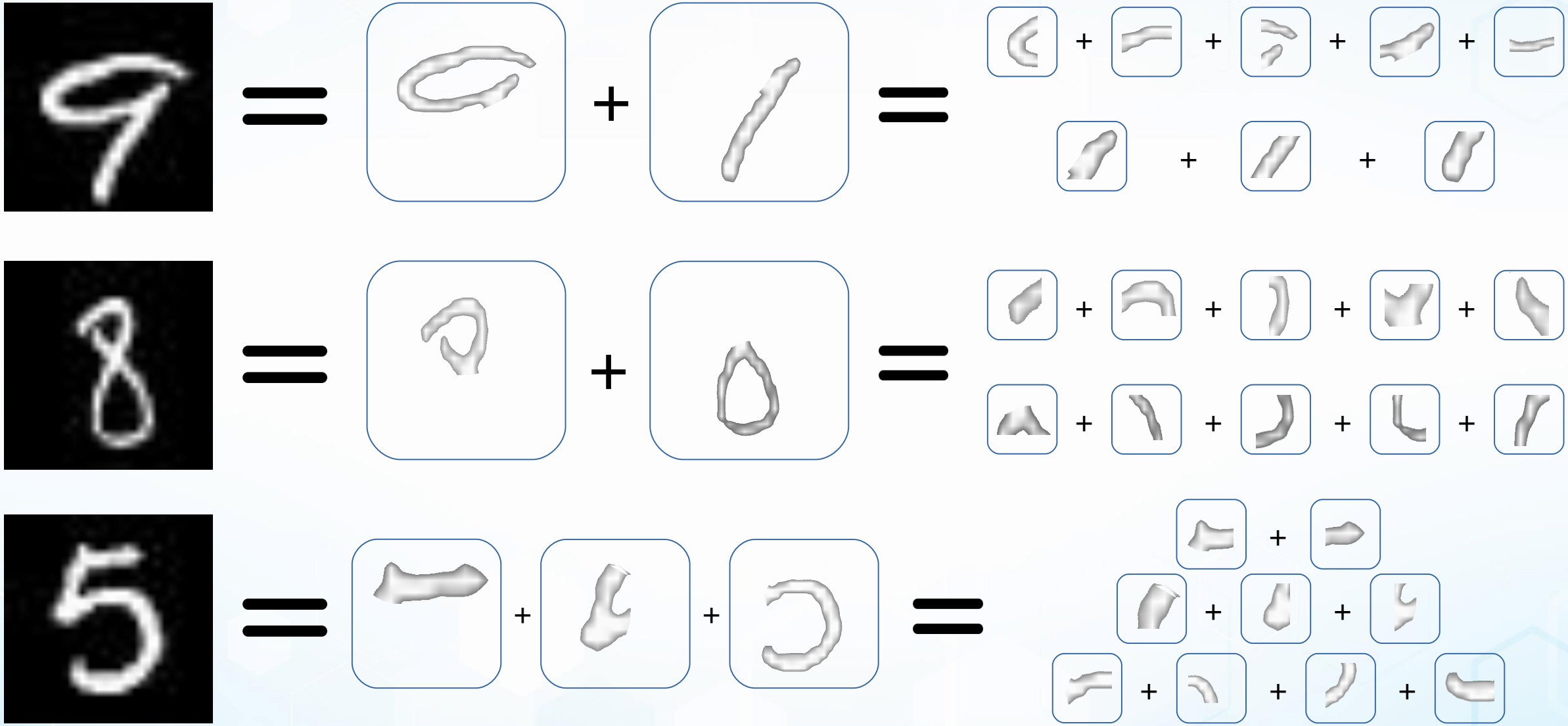
AGENDA

- AI and ML – How are they **related**?
- **Types** of AI and Machine Learning
- **What** is Neural Network?
- **How is the Neural Network learning?**
  - **Gradient Descent**
  - **Back Propagation**
- **When** and **where** can or should you use Deep Learning?

The network operates by taking activations from one layer to determine the activation of the next layer

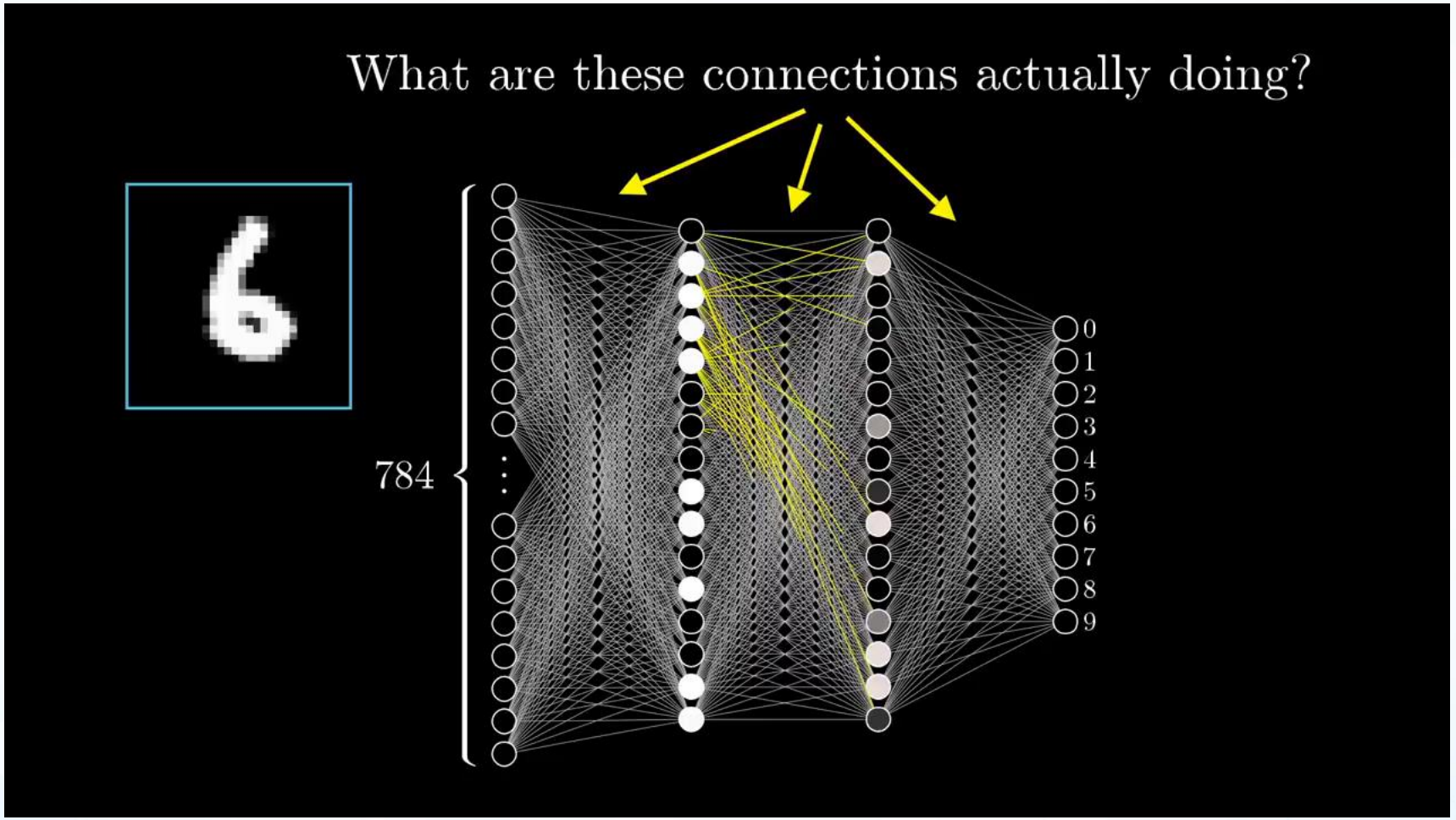


Why the hidden layers in the Neural Network?



Many recognition tasks breaks down into the layers of absraction

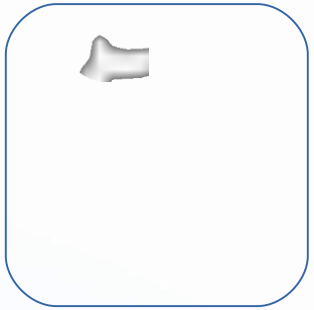
The goal is to have mechanism that could, **MAYBE,** combine pixels into edges, edges into patterns, and patterns into digits



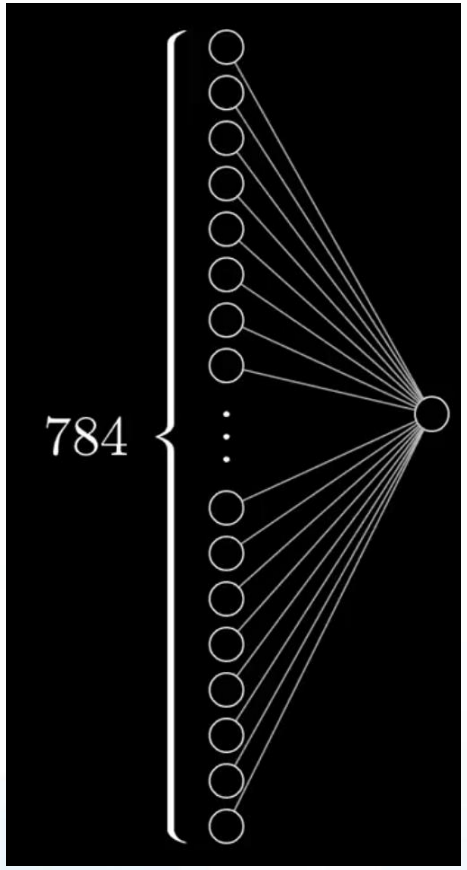
Many recognition tasks breaks down into the layers of abstraction

But how? What's the math behind it?

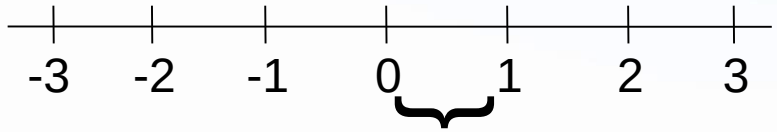
What parameters should the network have that you can tweak so that it's expressive enough to potentially identify this pattern

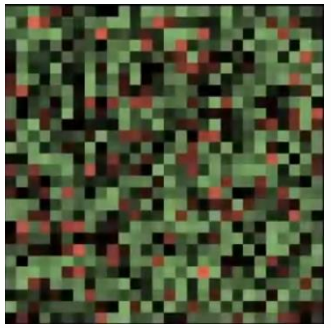


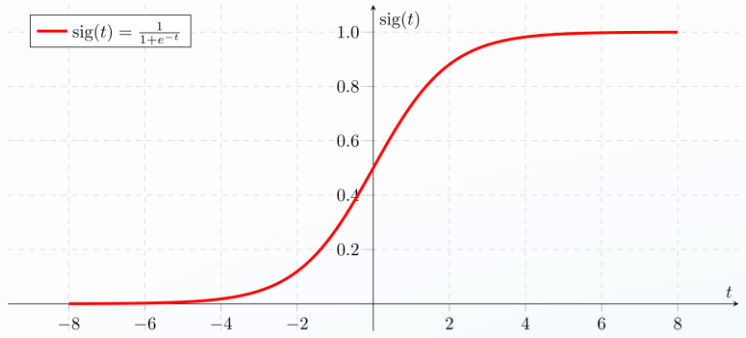
One particular Neuron in the 2nd layer picks up whether or not the image has an edge in this region



$$w_1a_1 + w_2a_2 + w_3a_3 + w_4a_4 .... + w_na_n$$





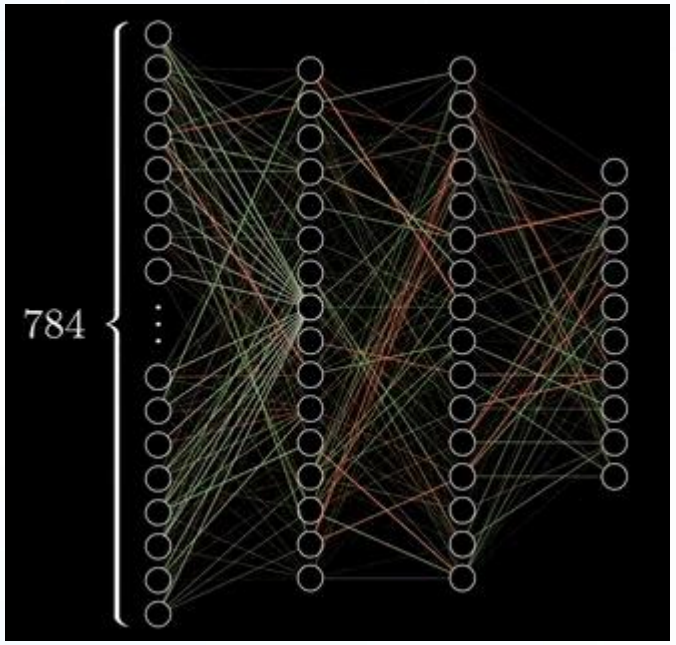


$$\sigma\,(w_1a_1 + w_2a_2 + w_3a_3 + w_4a_4 .... + w_na_n + b)$$

So, activation of a Neuron is a measure of how positive the relevant weight is, but you need to activate it only if the sum crosses a threshold.

Bias for inactivity

Weight tells you the pixel pattern getting picked up by a Neuron in the next layer and bias tells you how high weighted sum need to be before the Neuron gets activated



Weights: 784X16 + 16X16 + 16X10 = 12960

Biases: 16 + 16 + 10 = 42

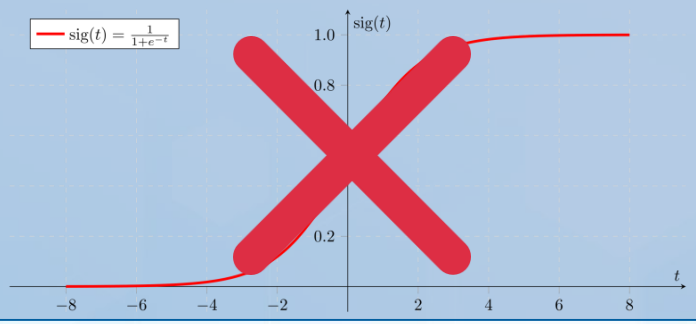
13002 parameters

which can be tweaked and turned to make the network behave in different ways

Learning ⟶ Finding the right weight and biases

Function ⟵ Neuron ⟶ ~~Number~~

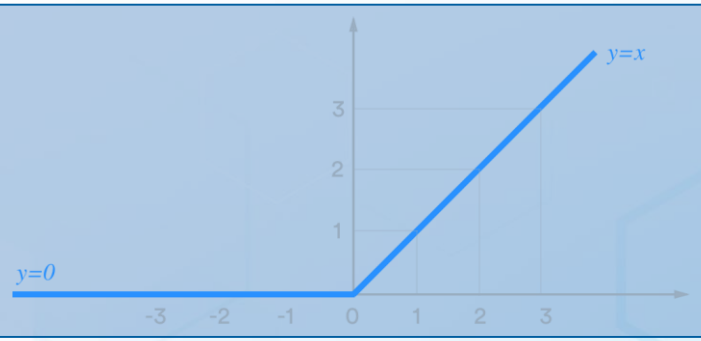
$$f(a_0, \ldots, a_{783}) = \begin{bmatrix} y_0 \\ \vdots \\ y_9 \end{bmatrix}$$



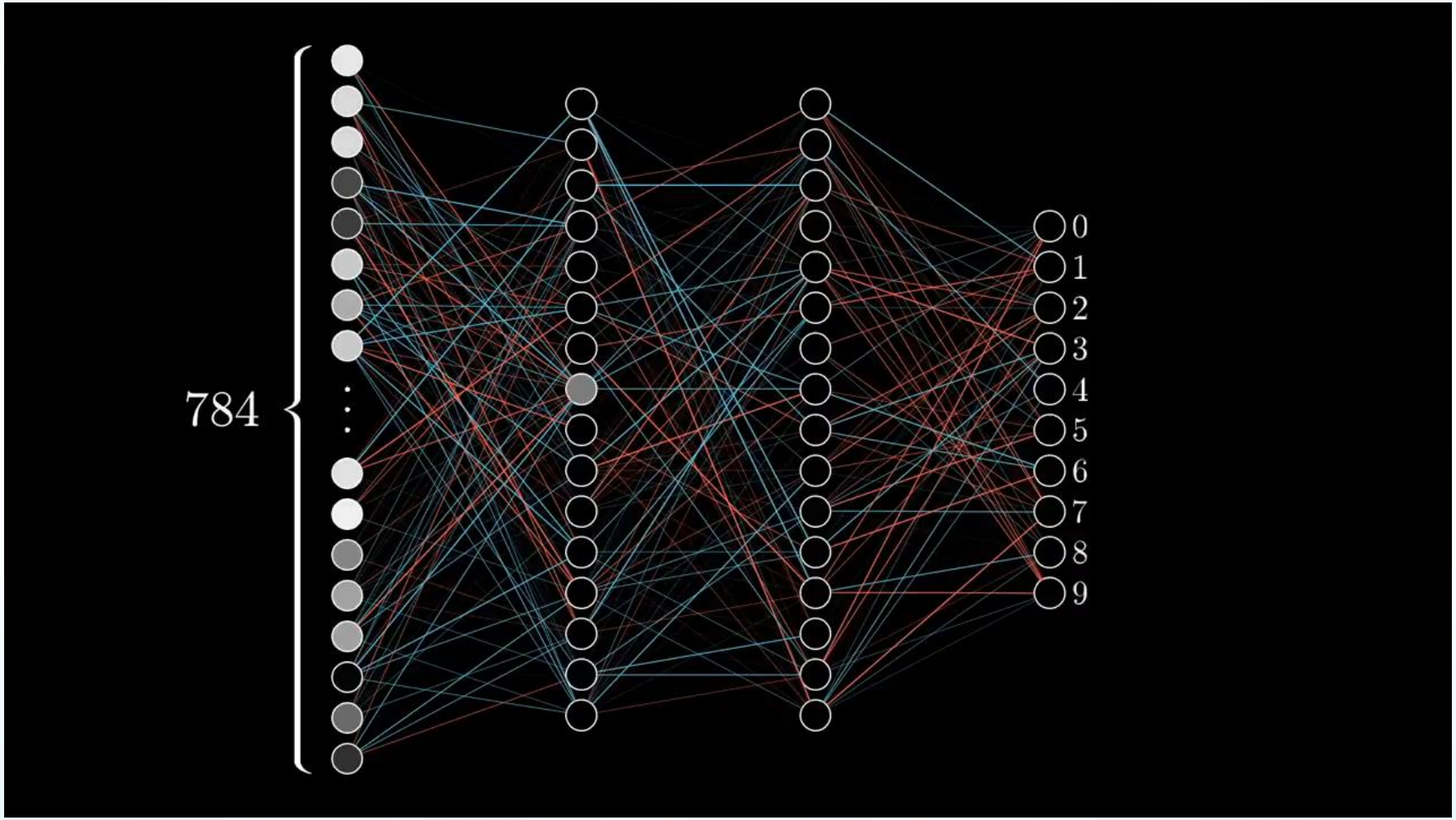
Rectified **L**inear **U**nit

y = x if x > 0

0 otherwise



The network starts predicting "rubbish" output, but if trained well, it can provide accuracy of about 99.79%



The cost measures how lousy your network is!

$$3.37 \left\{ \begin{array}{rl} 0.1863 \leftarrow & (0.43 - 0.00)^2 + \\ 0.0809 \leftarrow & (0.28 - 0.00)^2 + \\ 0.0357 \leftarrow & (0.19 - 0.00)^2 + \\ 0.0138 \leftarrow & (0.88 - 1.00)^2 + \\ 0.5242 \leftarrow & (0.72 - 0.00)^2 + \\ 0.0001 \leftarrow & (0.01 - 0.00)^2 + \\ 0.4079 \leftarrow & (0.64 - 0.00)^2 + \\ 0.7388 \leftarrow & (0.86 - 0.00)^2 + \\ 0.9817 \leftarrow & (0.99 - 0.00)^2 + \\ 0.3998 \leftarrow & (0.63 - 0.00)^2 \end{array} \right.$$

COST

VERY LOUSY → VERY SMART

AVERAGE OF COST OF ALL THE TRAINING IMAGES

COMPLICATED!

$$0.03 \left\{ \begin{array}{rl} 0.0006 \leftarrow & (0.02 - 0.00)^2 + \\ 0.0007 \leftarrow & (0.03 - 0.00)^2 + \\ 0.0039 \leftarrow & (0.06 - 0.00)^2 + \\ 0.0009 \leftarrow & (0.97 - 1.00)^2 + \\ 0.0055 \leftarrow & (0.07 - 0.00)^2 + \\ 0.0004 \leftarrow & (0.02 - 0.00)^2 + \\ 0.0022 \leftarrow & (0.05 - 0.00)^2 + \\ 0.0033 \leftarrow & (0.06 - 0.00)^2 + \\ 0.0072 \leftarrow & (0.08 - 0.00)^2 + \\ 0.0018 \leftarrow & (0.04 - 0.00)^2 \end{array} \right.$$

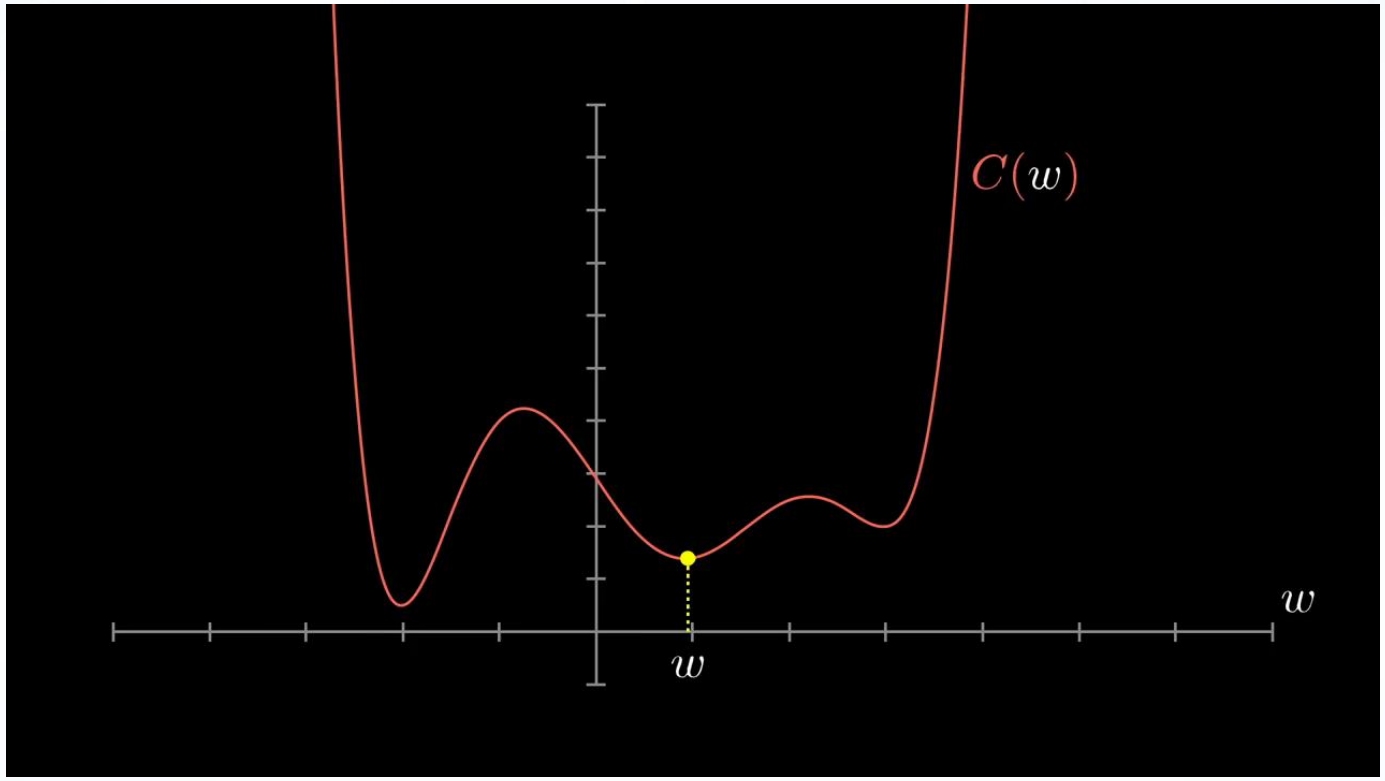
Neural Network Function		Cost Function
784 numbers (pixels)	INPUT	13002 weights/biases
10 numbers	OUTPUT	1 number (cost)
13002 weights/biases	PARAMETERS	Thousands of training images

You need an approach that minimises the cost function most quickly!

Let us consider a single input cost function *C(w)*

In calculus, **derivative** of a function gives a **minima**, but it's not feasible for such complicated function

Consider getting the slope of the function and find the direction to move to make that output lower



There are many possible valleys where you might land

No guarantee that the local minima you landed is actually the global minima

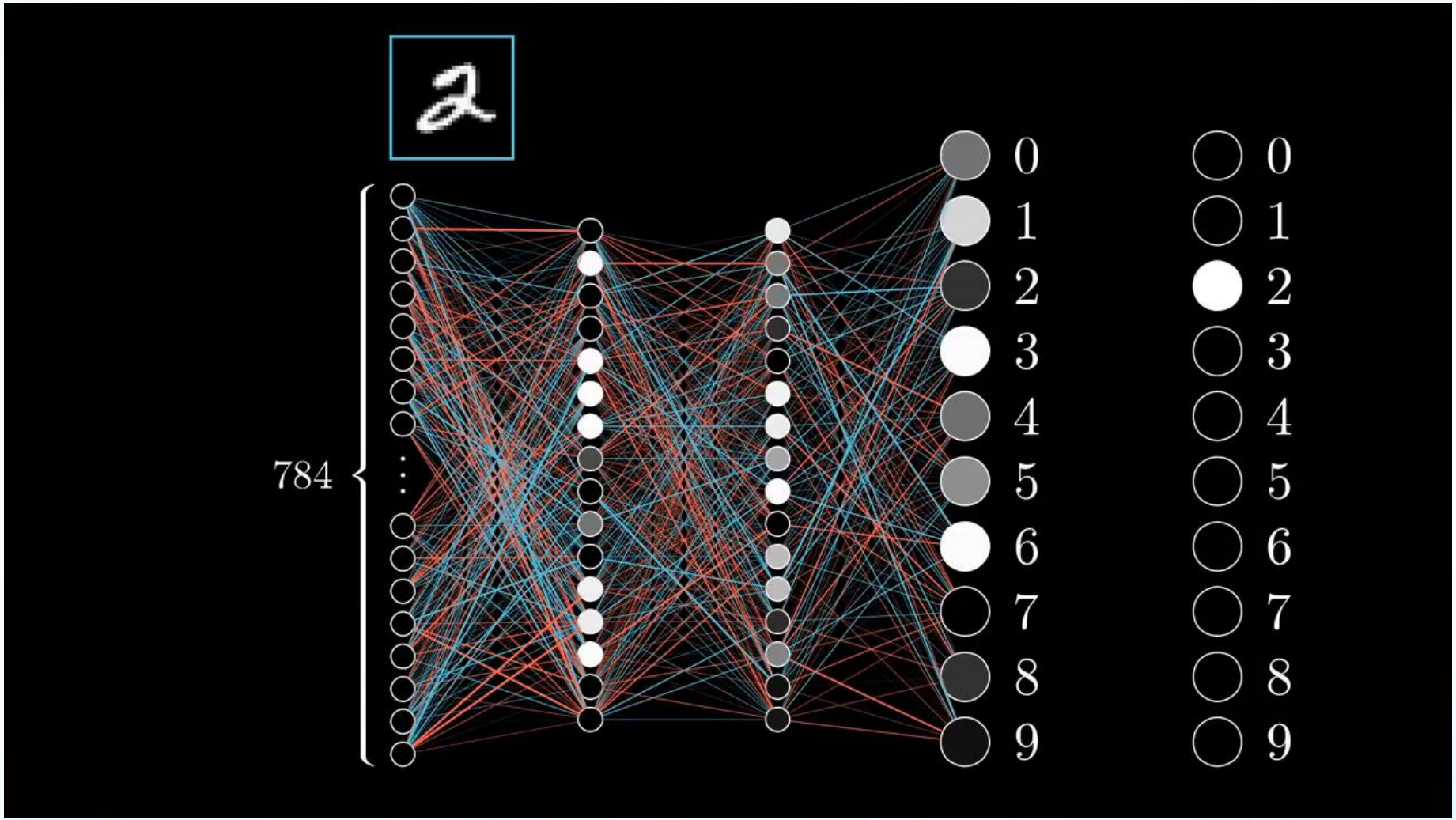
Which direction decreases cost most quickly?

Gradient of a function gives the direction of steepest ascent

GRADIENT DESCENT

The negative of that gradient gives the direction to step that decreases the function most quickly

The algorithm to compute this gradient efficiently is called Back Propagation



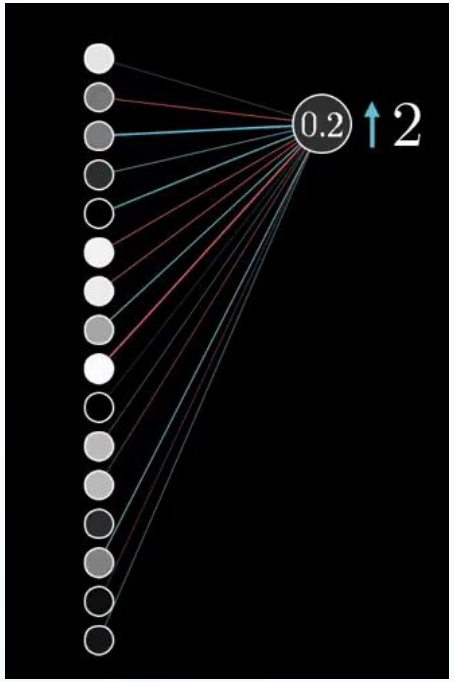
3 different avenues that can team up to increase the required activation

$$\sigma\left(w_1a_1 + w_2a_2 + w_3a_3 + w_4a_4 \ldots + w_na_n + b\right)$$

Changing $\mathbf{w_i}$

Changing **b**

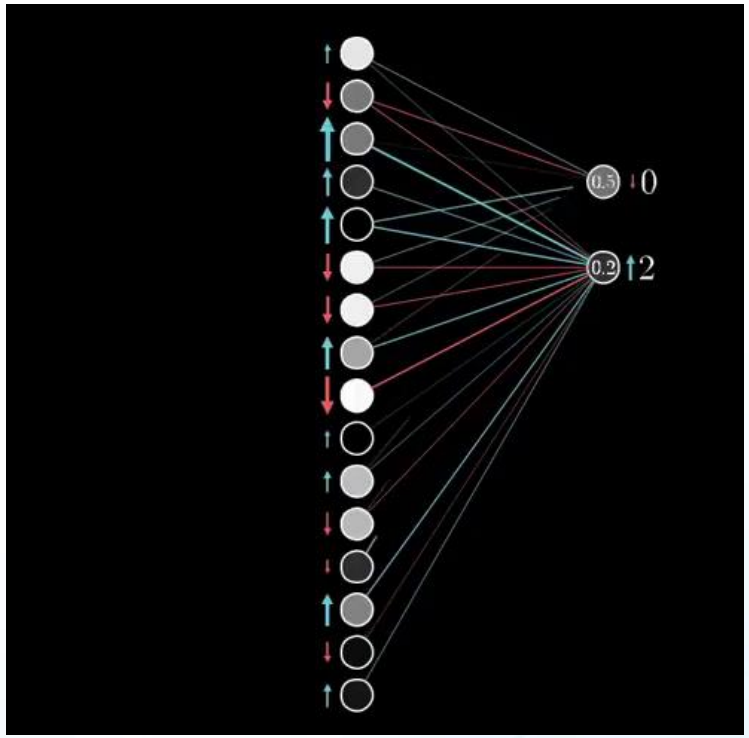
Changing $\mathbf{a_i}$



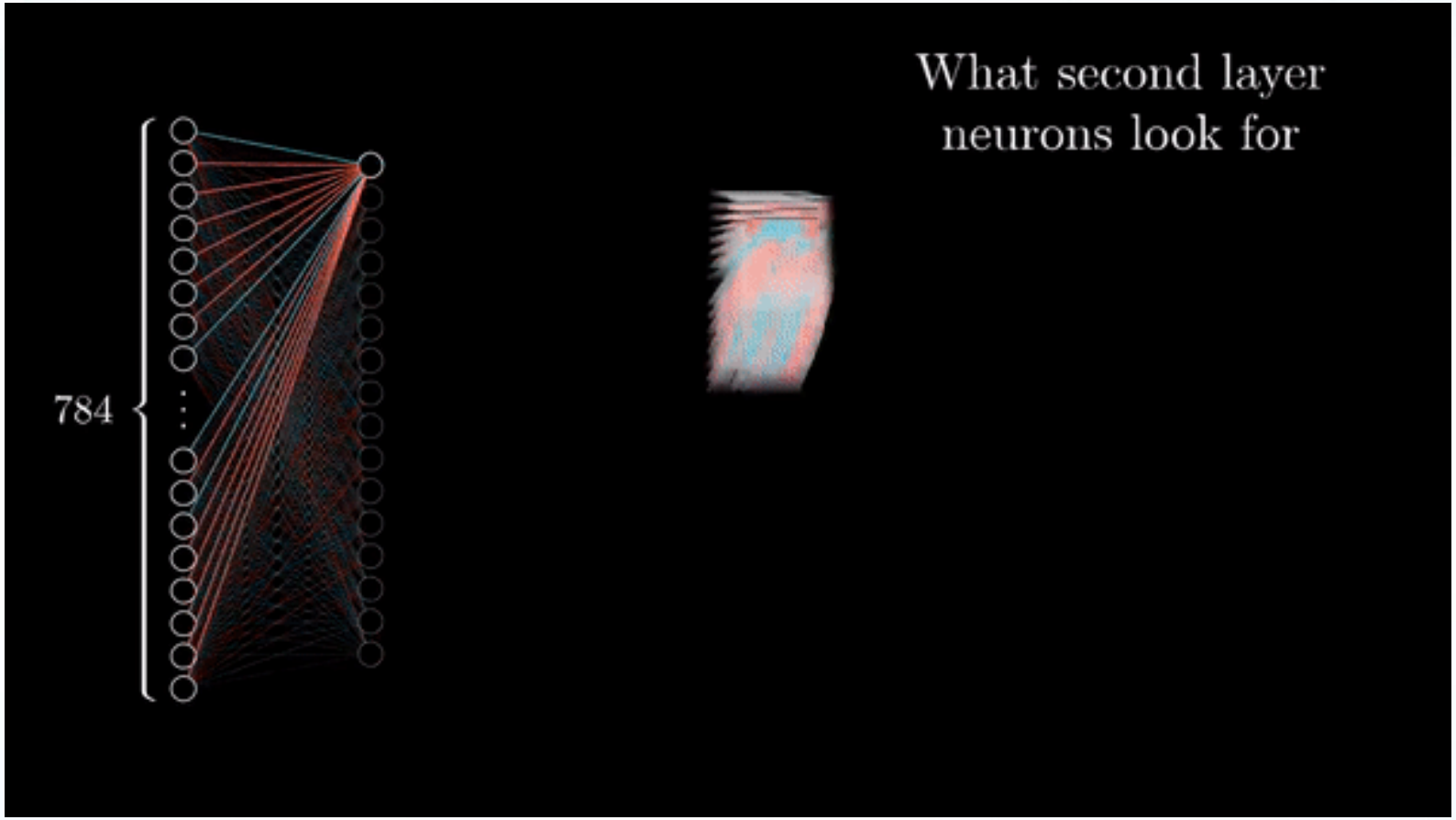
Connections with the **brightest Neurons** from the **preceding layer** have the **biggest** effect

If every Neuron connected with positive weight got brighter and those connected with negative weight got dimmer, then the required Neuron becomes **more active**

No direct influence on a_i, but we can control w_i and b



This particular network is not exactly learning edges or patterns, but we have come a long way!



Simple Neural Network(90's)

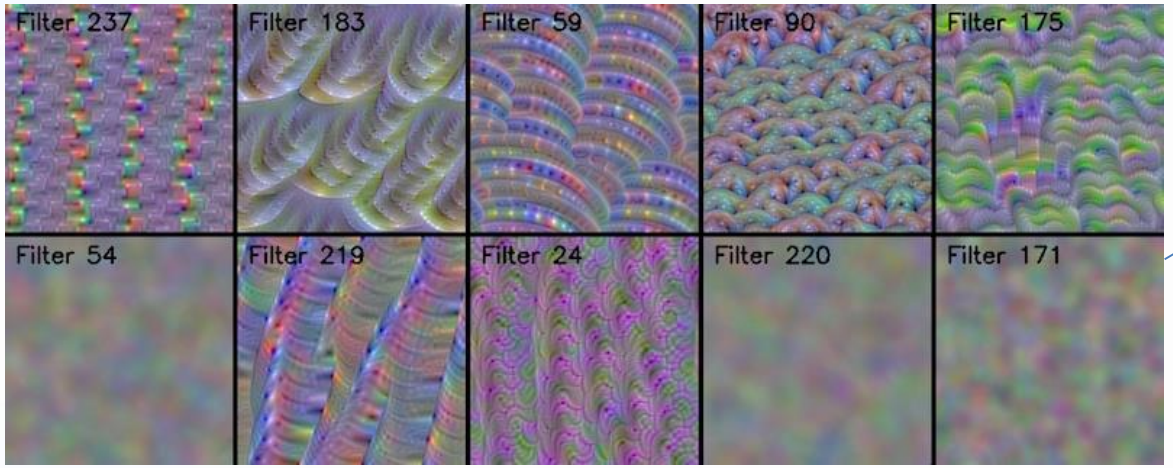
Convolutional NN

Long Short Term Memory(LSTM)

Generative Adverserial Network(GAN)

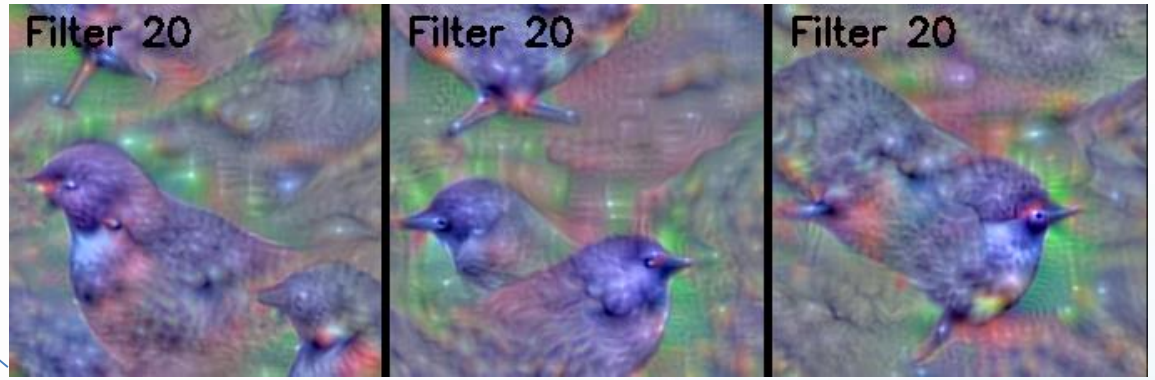
Approach to peer into the black box of the Neural Network to visualize how the network is working

keras-vis

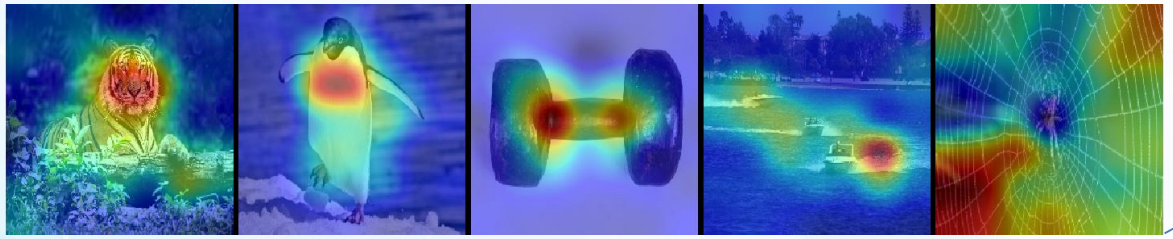


*Convolutional filters learn '**template matching**' filters that maximize the output when a similar template pattern is found in the input image.*

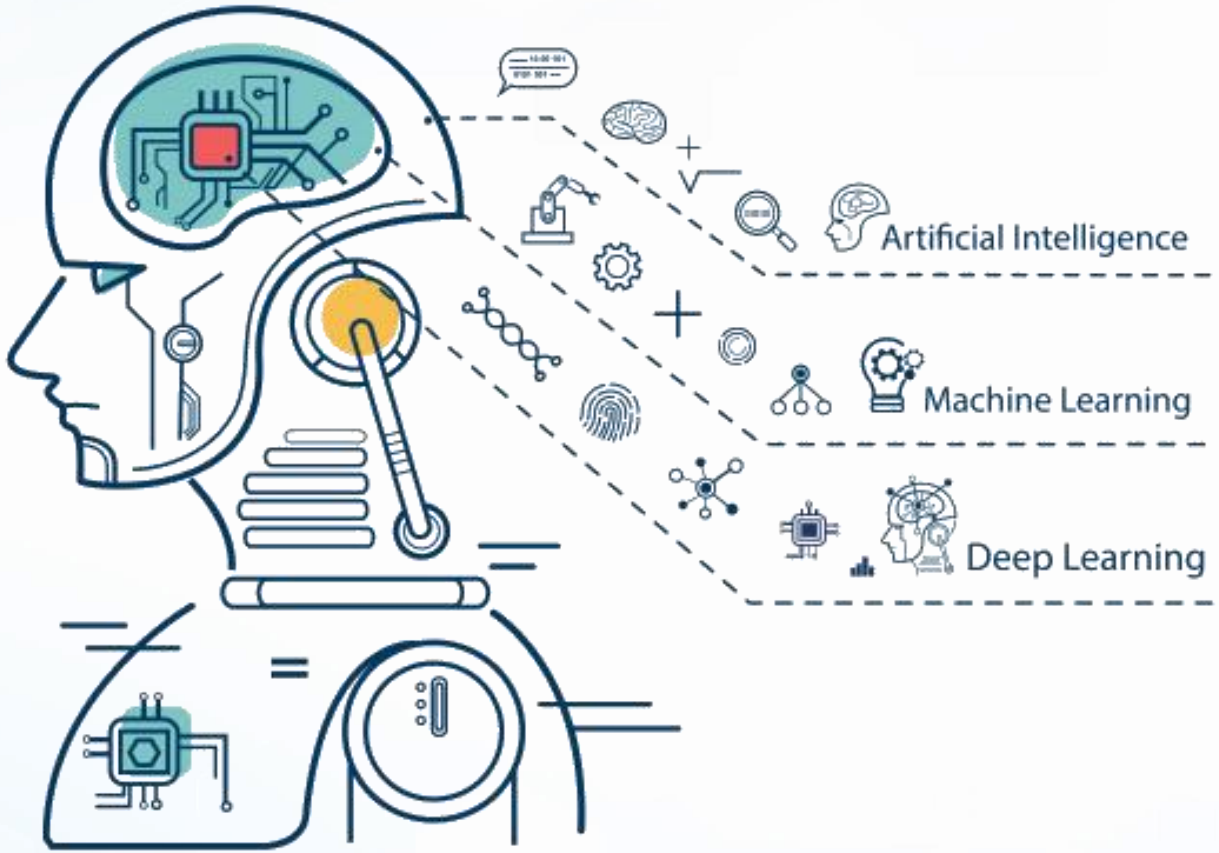
*Visualize those templates via **Activation Maximization**.*



*Assess whether a network is **over-fitting** or **under-fitting** or **generalizing** well*



*Assess whether a network is **attending** to correct parts of the image in order to generate a decision*



AGENDA

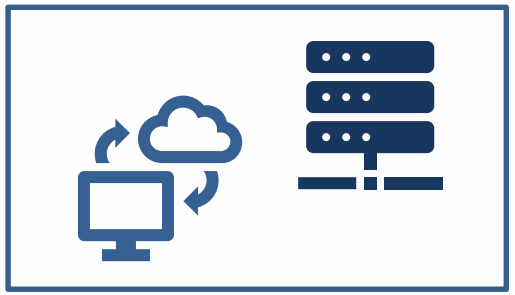
- AI and ML – How are they **related**?
- **Types** of AI and Machine Learning
- **What** is Neural Network?
- How is the Neural Network **learning**?
  - Gradient Descent
  - Back Propagation
- **When and where can or should you use Deep Learning?**

Key fact to keep in mind while working with ML problems – DON'T OVERKILL IT WITH DEEP LEARNING!

When there is ***lack of domain understanding for feature introspection***, **Deep Learning** techniques outshines other techniques as you have to worry less about feature engineering

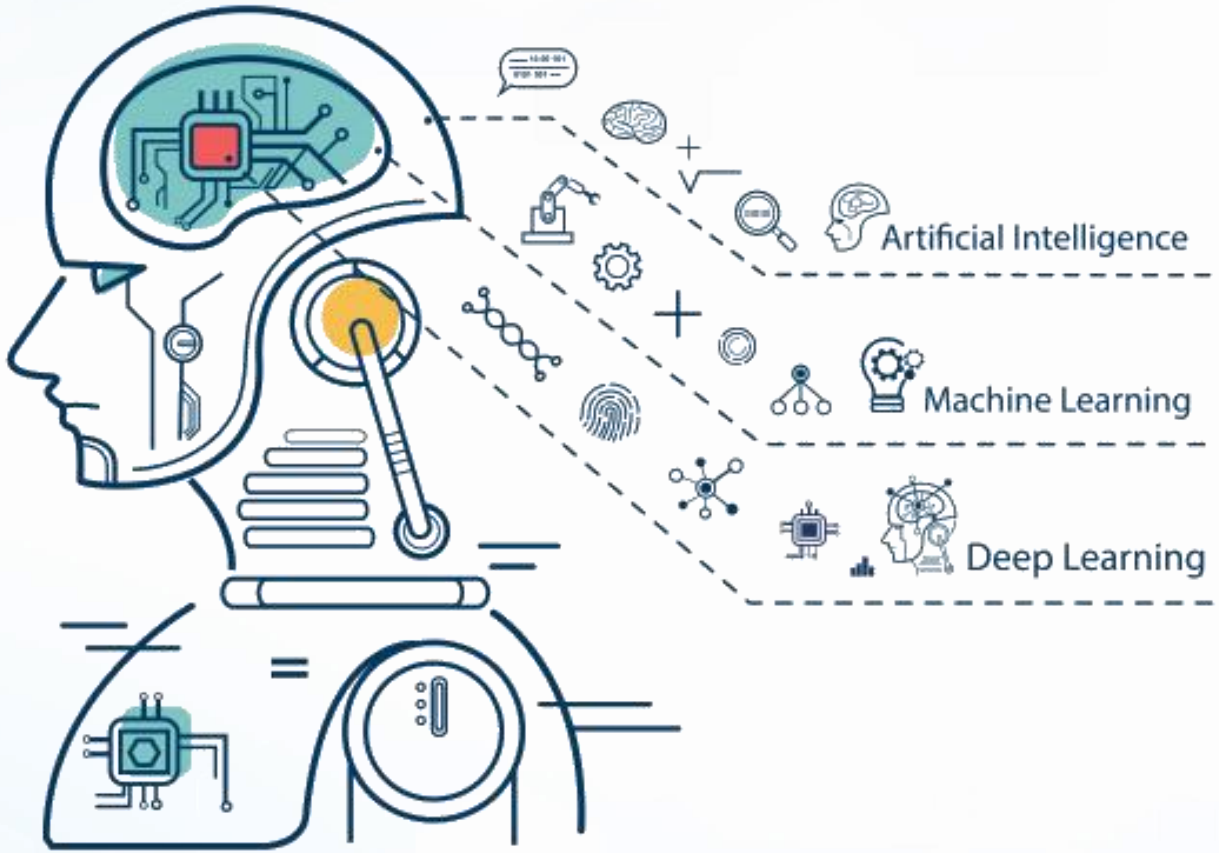
High predictive power, but LOW INTERPRETABILITY

LARGE DATA SET



HIGH-END INFRASTRUCTURE

EXECUTION TIME



THANK YOU