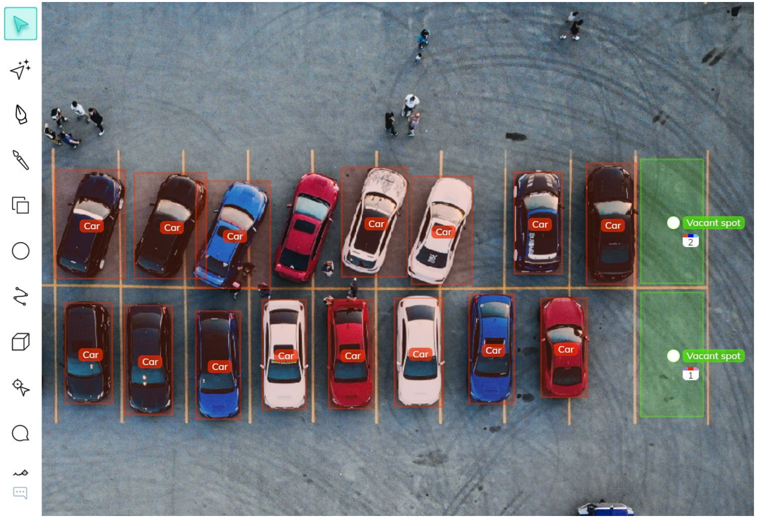
Computer Vision

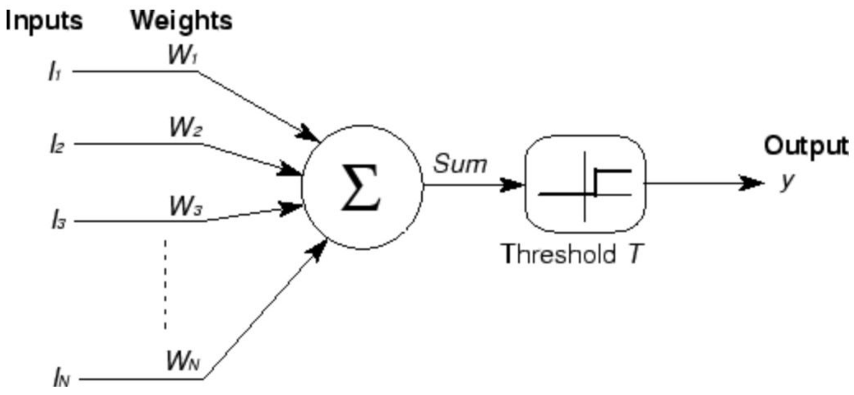
What is Computer Vision?

- Field of AI
- Enables computers to derive meaningful information from visual inputs
- Images, videos, etc.
- We can then do cool stuff
- Cancer Detection, Cell Classification, Digital Pathology, Crop and Yield Monitoring, and much more



History of Neural Networks

- Mcculloch and Pitts
- Threshold Logic
- Hebbian Learning



A McCulloch-Pitts neuron

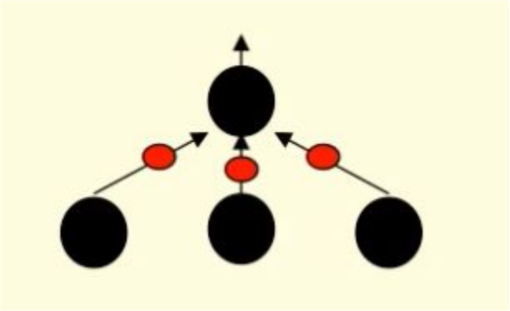
Why do we care?

- Computers have always been good at things humans are bad at
  - multiplying large numbers
  - processing large, sequential programs very quickly
- Computers have always been bad at things humans are good at
  - recognizing objects
  - understanding language
  - using many simple facts/operations to build a large, robust, belief system

Using the brain as inspiration can lead to mathematical models that can do those things that computers haven’t been able to

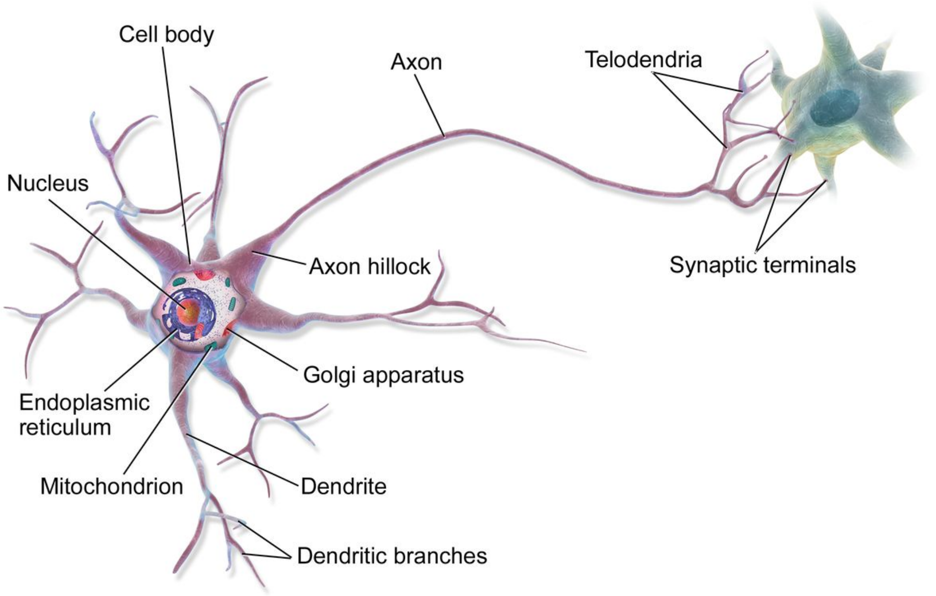
How the Brain Works – a little deeper look

- Each neuron receives inputs from other neurons
- The effect of each input line into the neuron is controlled by a synaptic weight
- The synaptic weights *adapt* so that the whole network learns to perform useful computations
  - eg. recognize objects, understand language, social interactions



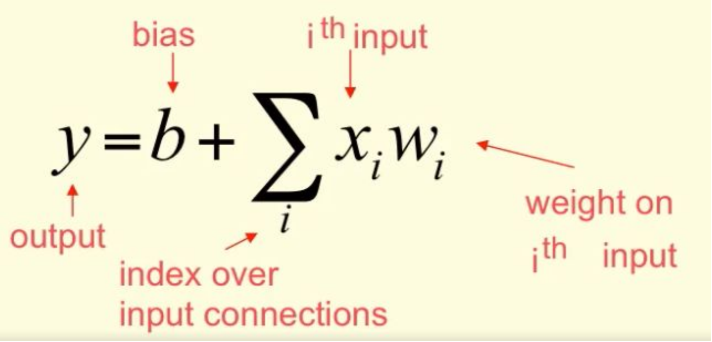
Neurons - What are they? What do they contain?

- In the brain, they store information in the form of a ‘signal’
- In math - we want to **idealize** this definition of a
- This allows us to apply mathematics to the model, and make comparisons and analogies to other familiar systems

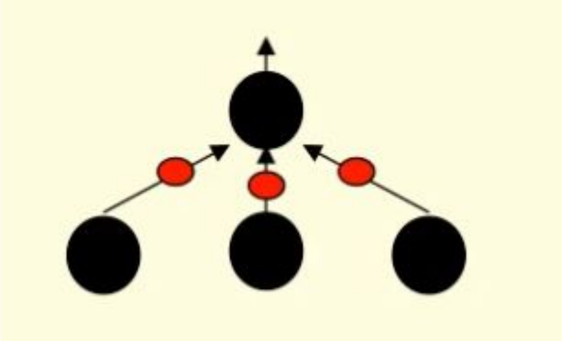


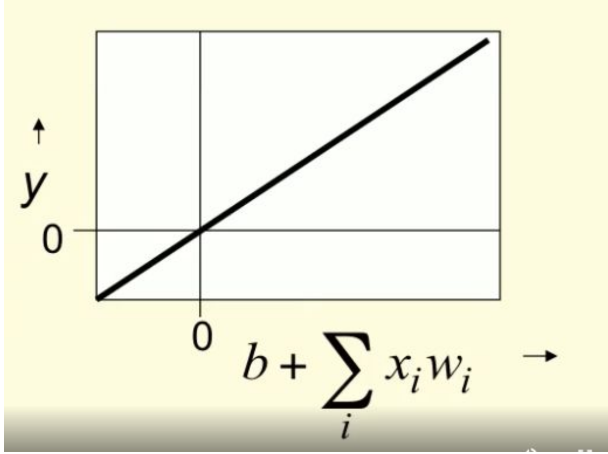
Different Types of Neurons

- Linear Neurons - simple, but computationally limited



$$y = b + \sum_i x_i w_i$$

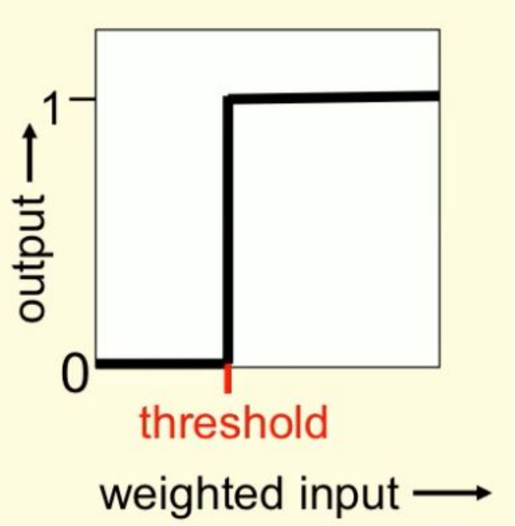




Different Types of Neurons

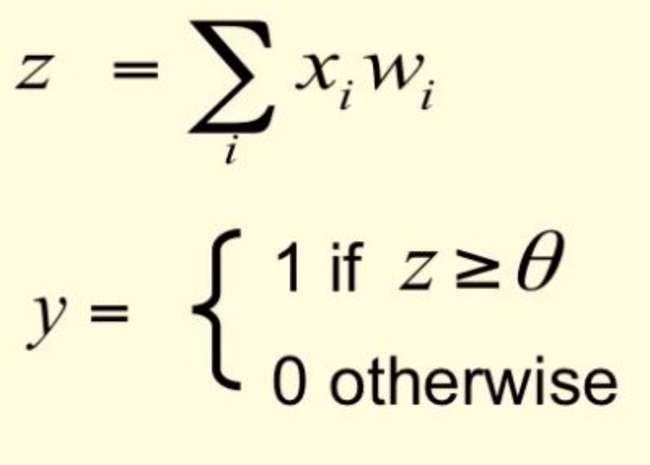
Binary Threshold Neurons -

- First, compute the weighted sum of the inputs
- Then send out a fixed size spike of activity if the weighted sum exceeds some threshold value
- It is thought that the spike is likely the truth value of some proposition, and each neuron then combines truth values of smaller propositions to attain truth values of larger propositions.



Different Types of Neurons

Binary Threshold Neuron



$$z = \sum_i x_i w_i$$

$$y = \begin{cases} 1 \text{ if } z \geq \theta \\ 0 \text{ otherwise} \end{cases}$$

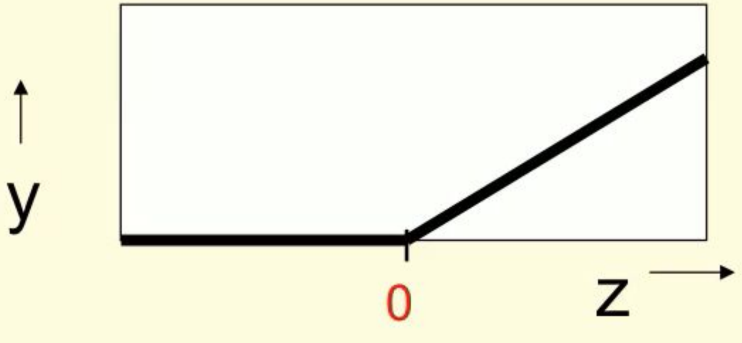
Different Types of Neurons

Rectified Linear Neurons -

- They compute a *linear* weighted sum of their inputs
- The output is a *non-linear* transformation of that total input

$$z = b + \sum_i x_i w_i$$

$$y = \begin{cases} z & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$$



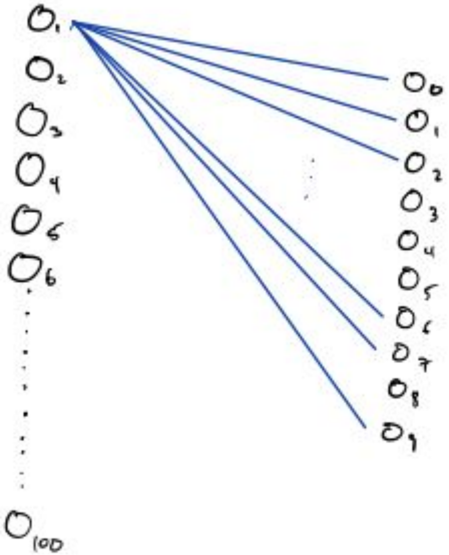
A Simple Computer Vision Problem with a Neural Network

We want to recognize handwritten digits

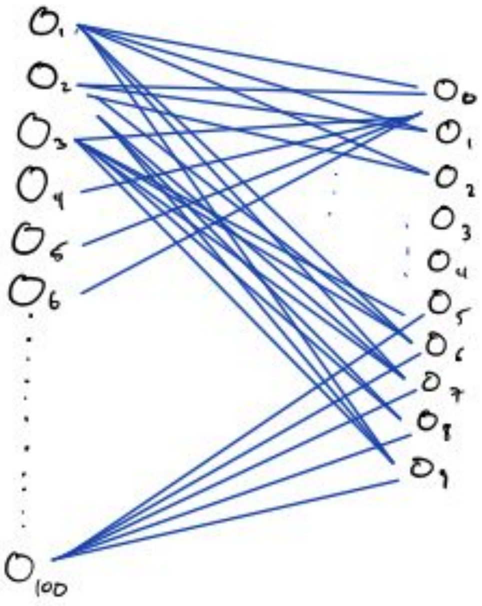
Consider a neural network with two layers of neurons

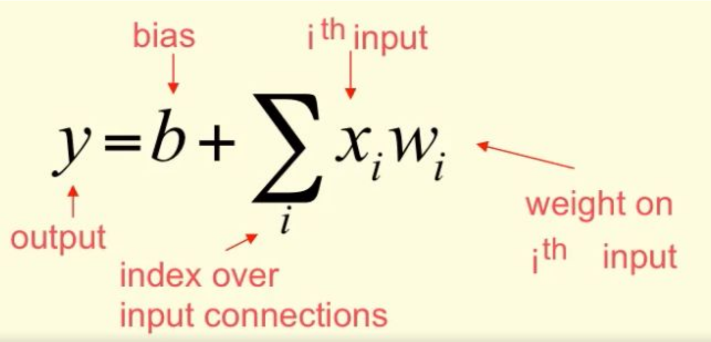
- neurons in the first layer represent pixel intensities
- neurons on the second layer represent known shapes

A pixel gets to vote if it has ink on it



A Simple Computer Vision Problem with a Neural Network





$$y = b + \sum_i x_i w_i$$

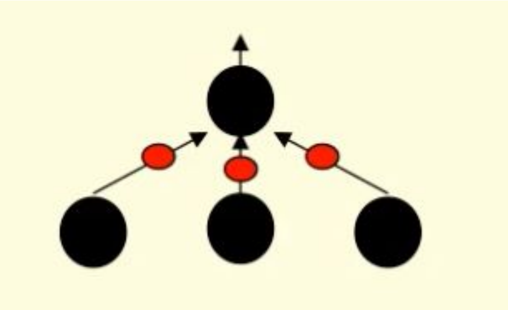
bias

i th input

output

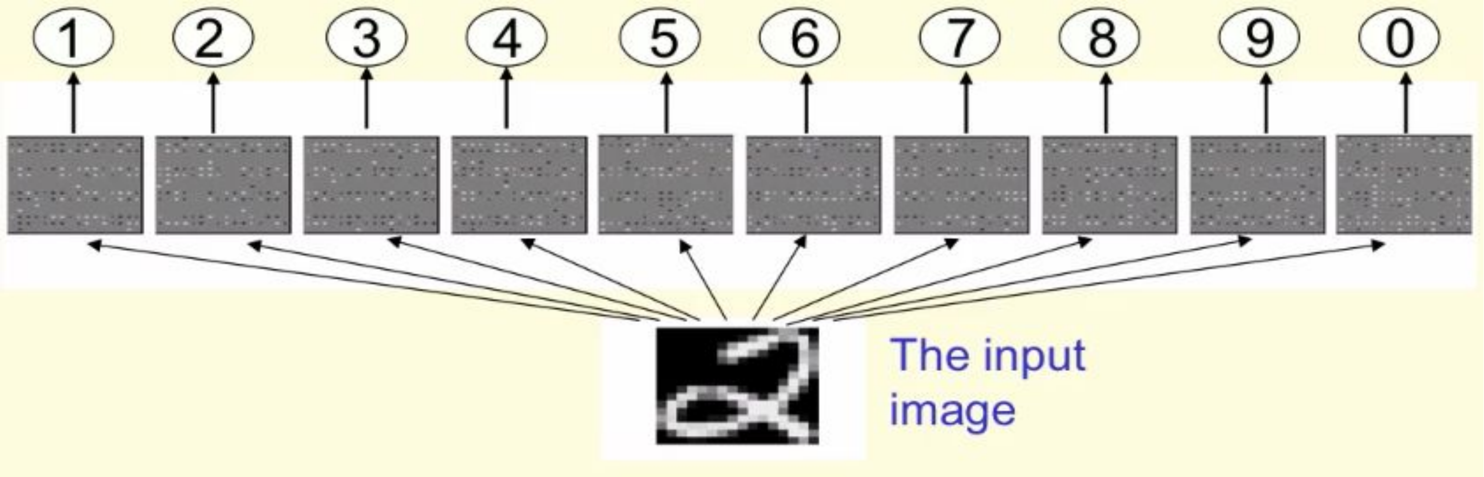
index over input connections

weight on i th input



A Simple Computer Vision Problem with a Neural Network

A different view -
How to display the weights in this diagram?

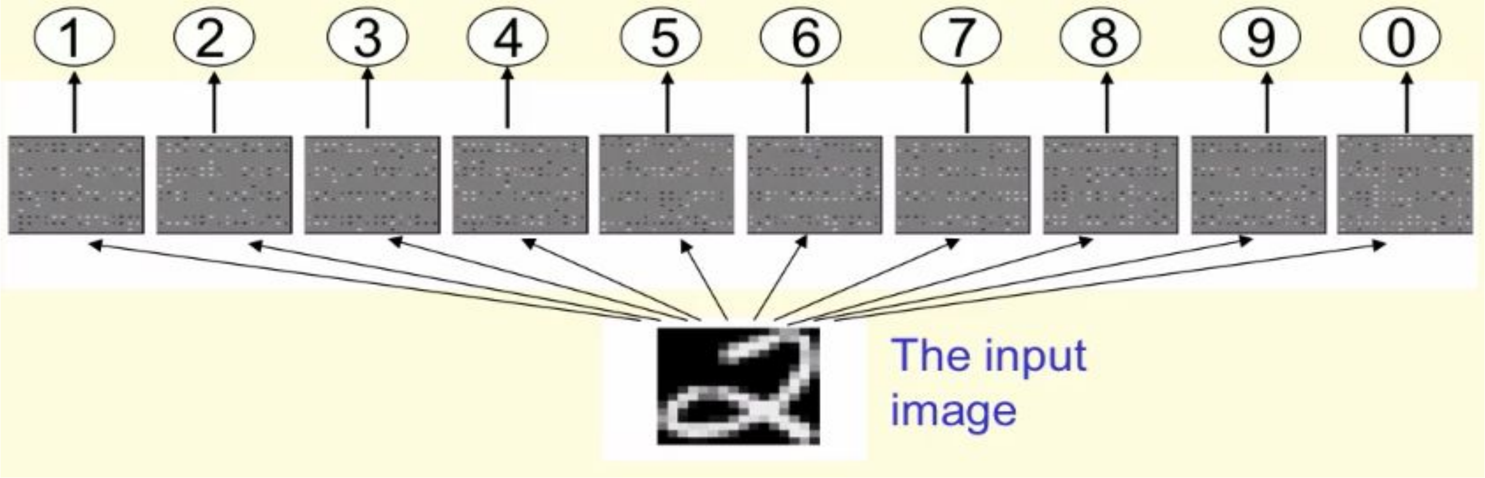


Give each output unit its own “map” of the input image and display the weight coming from each pixel in the location of that pixel in the map

A Simple Computer Vision Problem with a Neural Network

How are we going to learn the weights?

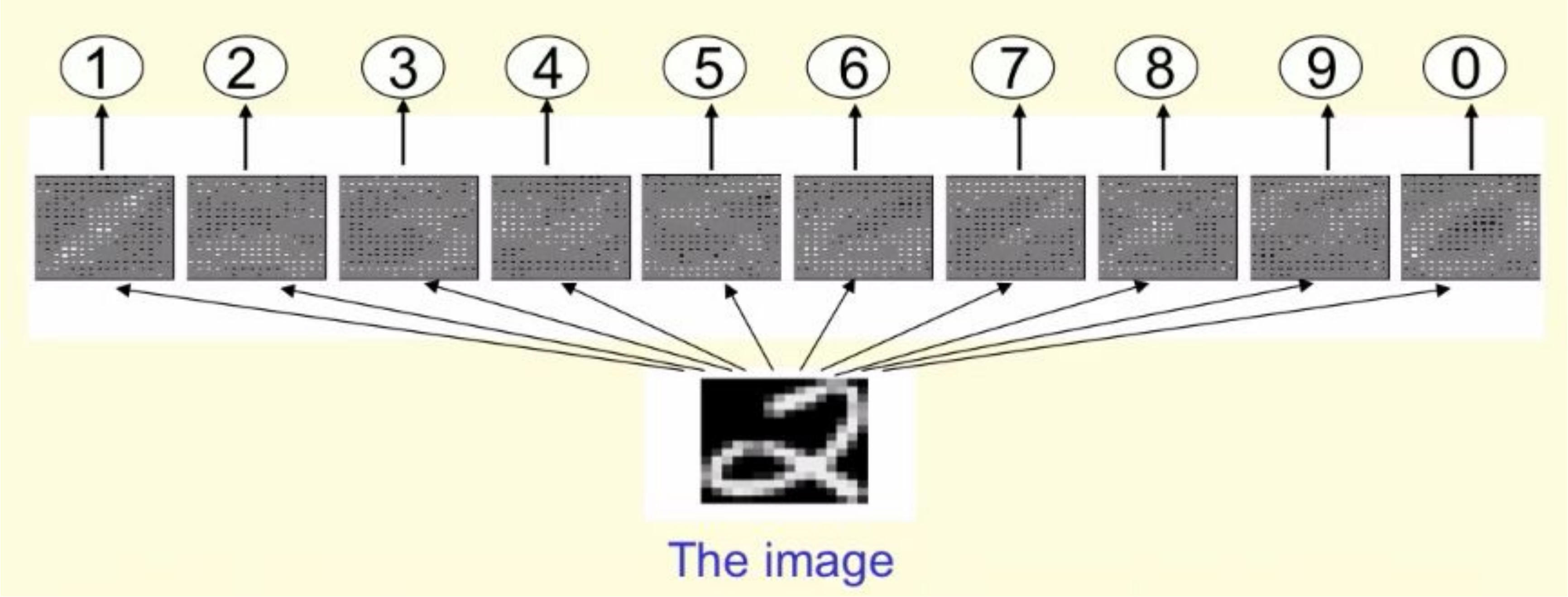
- Show the network an image and *increment* the weights coming from active pixels to the correct class
- Also *decrement* the weights from the active pixels to whatever class the network guesses



Learning Algorithm Explained

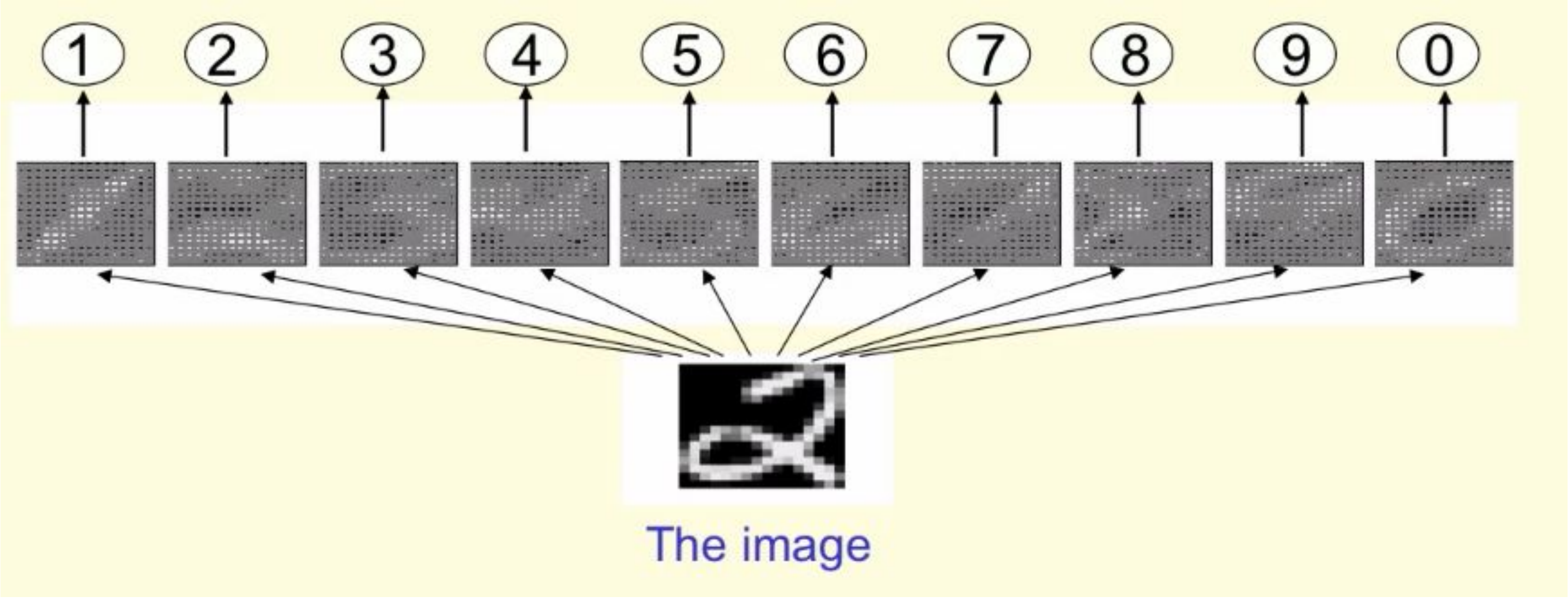
A Simple Computer Vision Problem with a Neural Network

Show the model a few 100 examples



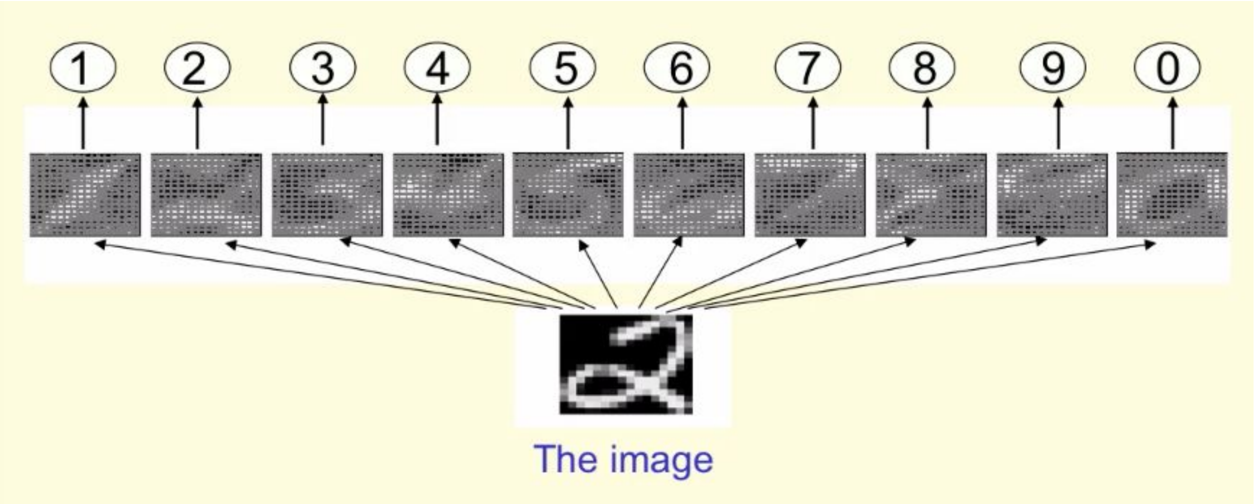
A Simple Computer Vision Problem with a Neural Network

Show the model a few more 100 examples.



A Simple Computer Vision Problem with a Neural Network

Show the model a few more 100 examples.



The weights now look like templates of the digits.

Why the simple learning algorithm is insufficient

- A two layer network with a single winner in the last layer is equivalent to having a rigid template for each shape
- The winner is the template that has the biggest overlap with the input

- The ways in which handwritten digits vary are much too complicated to be captured by simple template matches of the whole shapes
    - INSTEAD...in order to capture all the allowable variations of a digit, we first need to learn the features that is is composed of
    - and then we can look at arrangements of those features

This gives the model much more flexibility, and power.