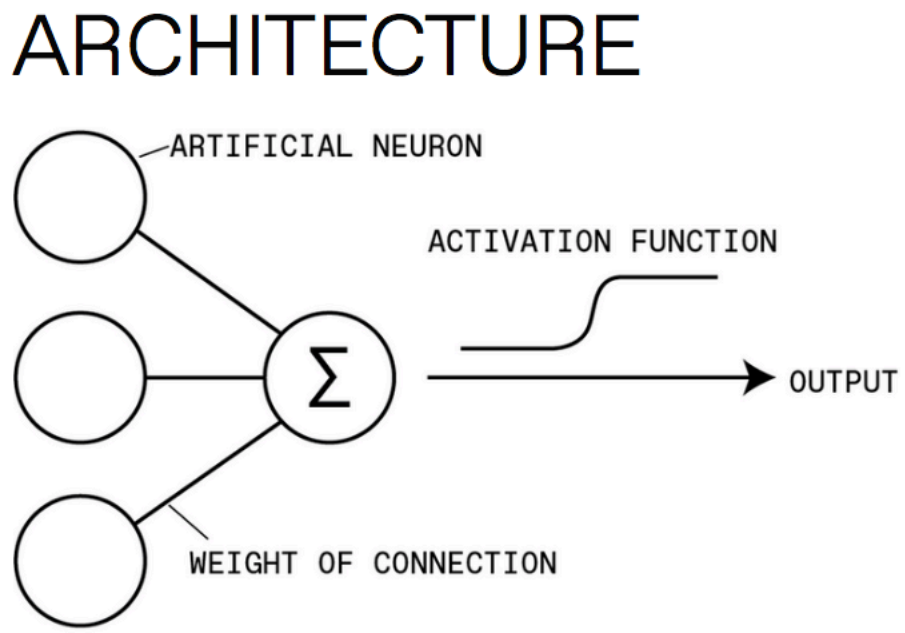
How Deep Learning Works

Based on Moore, S. K., D. Schneider, and E. Strickland (2021) How Deep Learning Works: Inside the Neural Networks that Power Today's AI, *IEEE Spectrum*, October 2021, pp. 32-33.

https://spectrum.ieee.org/what-is-deep-learning/backpropagation

Architecture

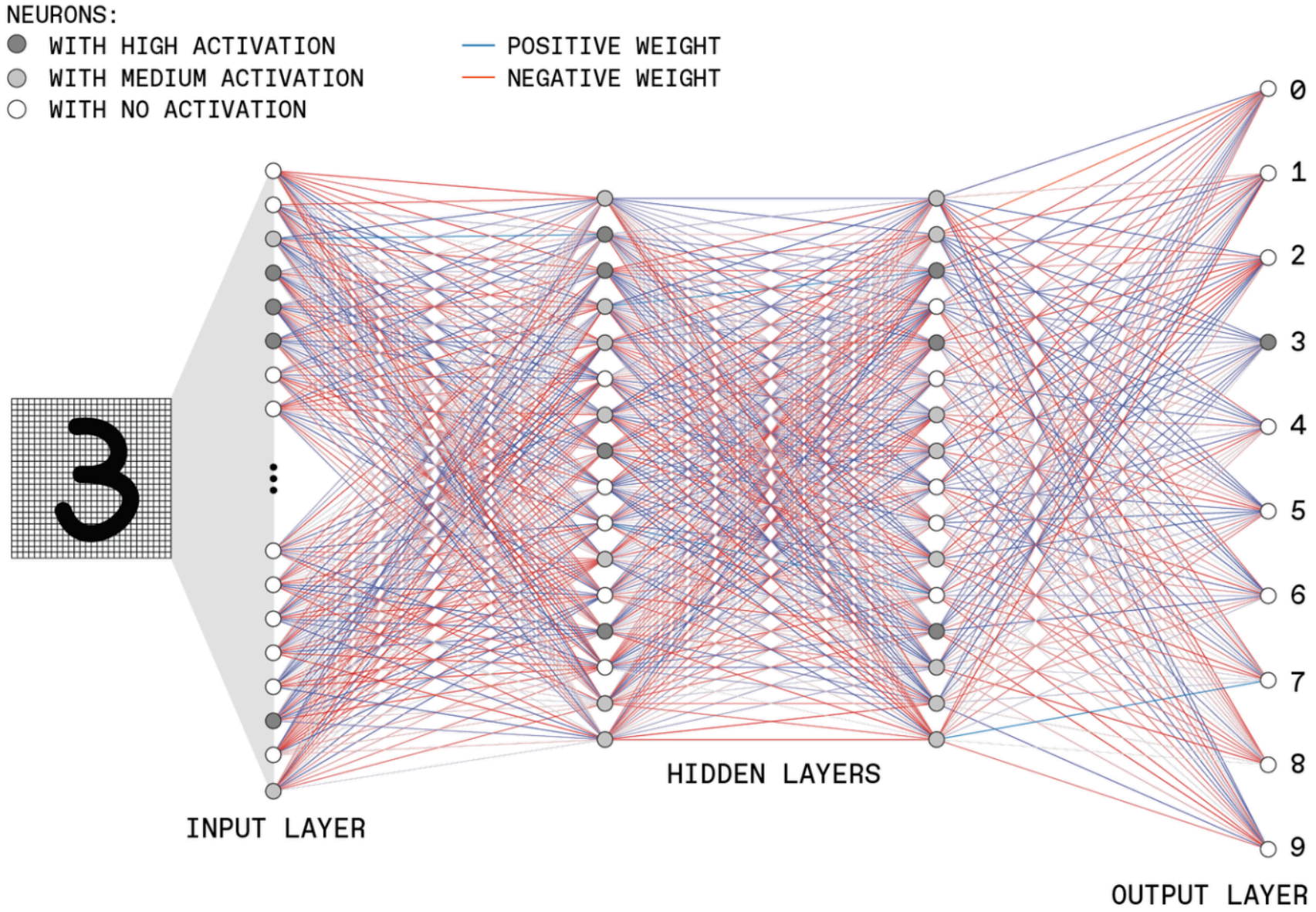
Neuron in an Artificial Neural Network



Each neuron in an artificial neural network sums its inputs and applies an activation function to determine its output. This architecture was inspired by what goes on in the brain, where neurons transmit signals between one another via synapses.

Architecture

Network Structure



DAVID SCHNEIDER

Architecture

How the structure works

- A hypothetical feed-forward deep neural network for recognizing handwritten digits as one of the 10 possible numerals
  - "deep" because it contains multiple hidden layers
- The **input layer** contains many neurons, each of which has an **activation** set to the gray-scale value of one pixel in the image
- These input neurons are **connected** to neurons in the next layer, passing on their **activation** levels after they have been multiplied by a certain value, called a **weight**
- Each neuron in the **second layer sums** its many inputs and applies an activation function to determine its output, which is fed forward in the same manner

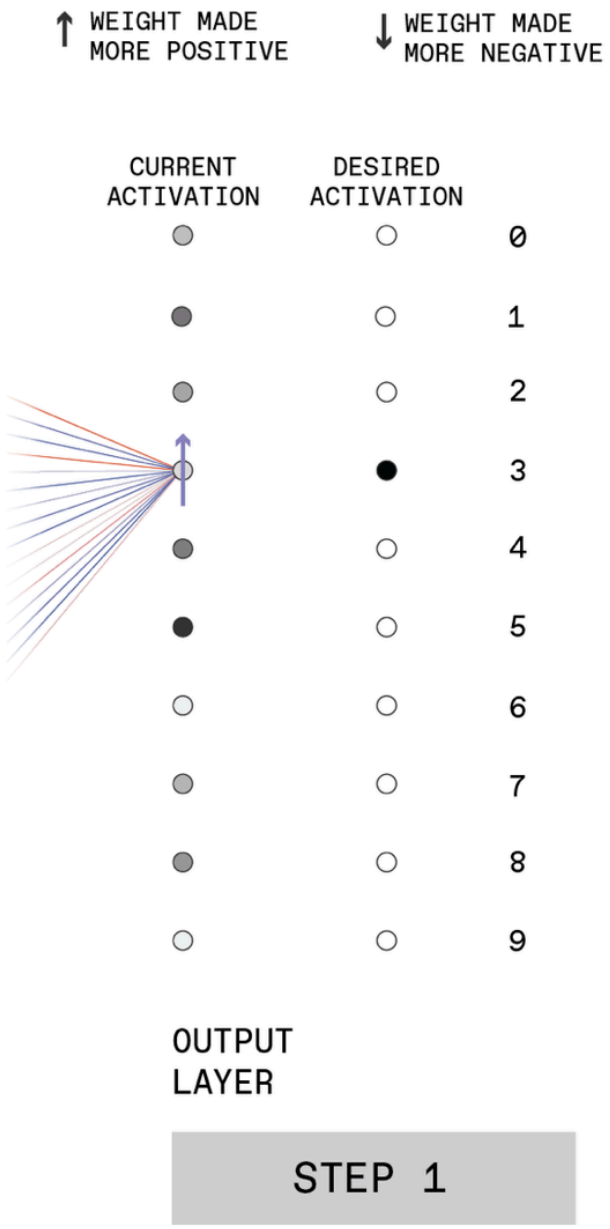
Training

Back-Propagation

- Trained by calculating the difference between the actual output and the desired output
  - The mathematical optimization problem here has as many dimensions as there are adjustable parameters in the network—primarily the weights of the connections between neurons, which can be positive [blue lines] or negative [red lines].
- Training the network → finding a minimum of this multidimensional "loss" or "cost" function.
- Done iteratively over many training runs, incrementally changing the network's state
  - Making many small adjustments to the network's weights based on the outputs that are computed for a random set of input examples, each time starting with the weights that control the output layer and moving backward through the network
  - This backpropagation process is repeated over many random sets of training examples until the loss function is minimized, and the network then provides the best results it can for any new input

Training

Step 1

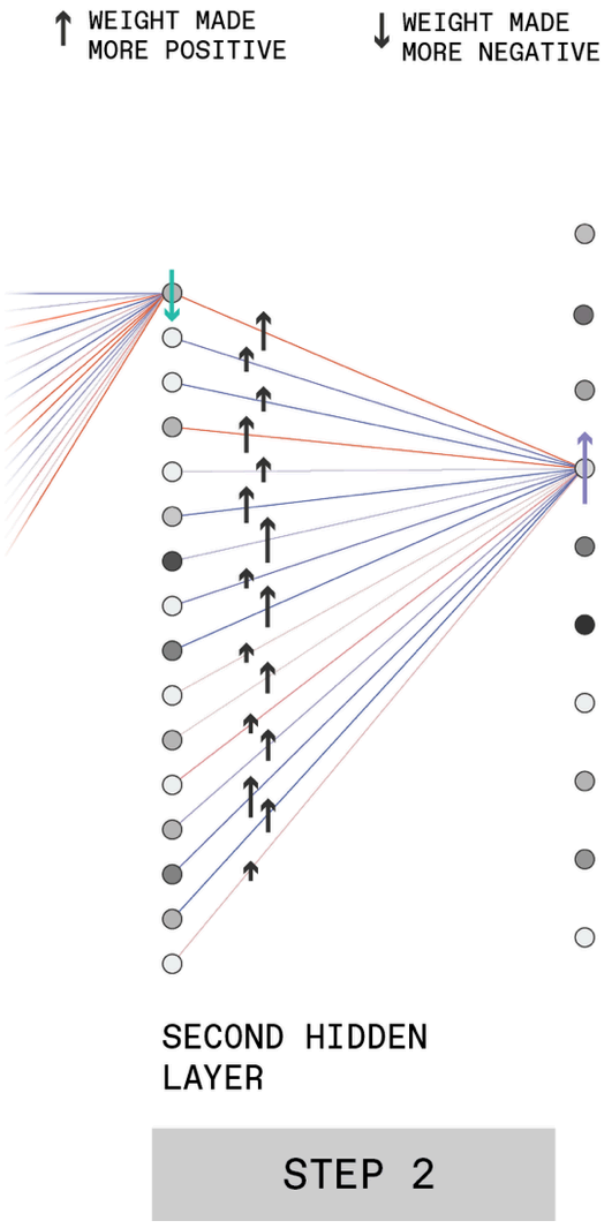


STEP 1

When presented with a handwritten "3" at the input, the output neurons of an untrained network will have random activations. The desire is for the output neuron associated with 3 to have high activation [dark shading] and other output neurons to have low activations [light shading]. So the activation of the neuron associated with 3, for example, must be increased [purple arrow].

Training

Step 2

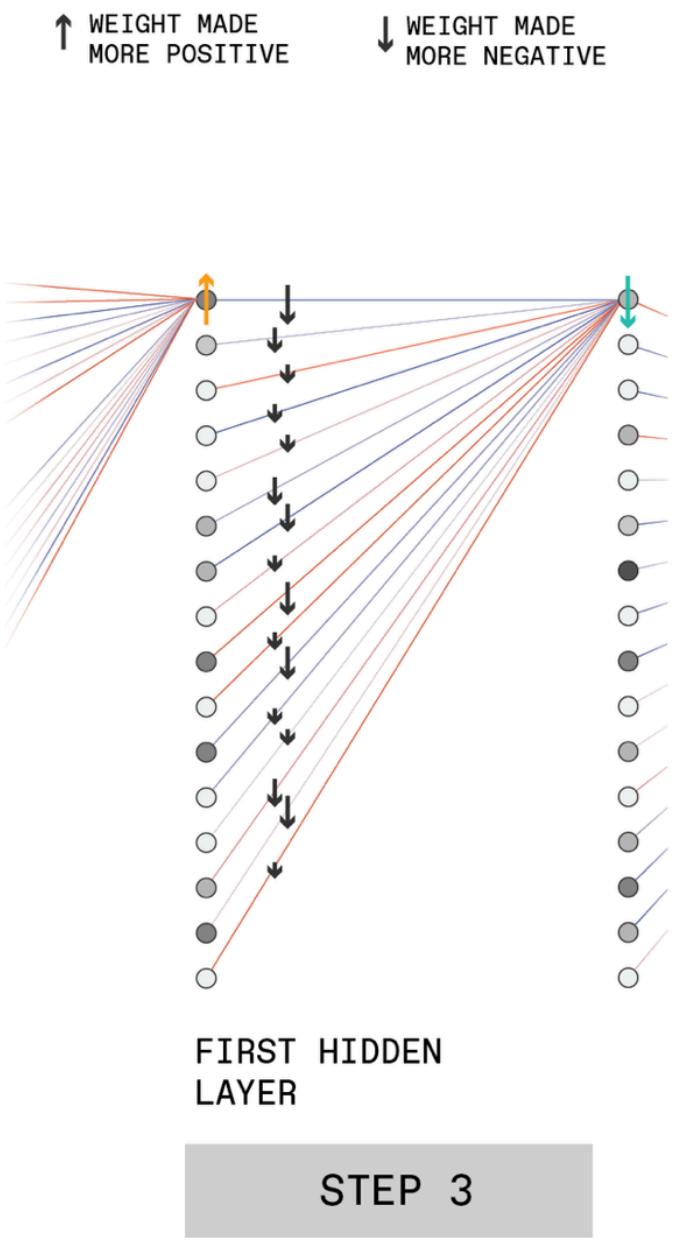


STEP 2

To do that, the weights of the connections from the neurons in the second hidden layer to the output neuron for the digit "3" should be made more positive [black arrows], with the size of the change being proportional to the activation of the connected hidden neuron.

Training

Step 3

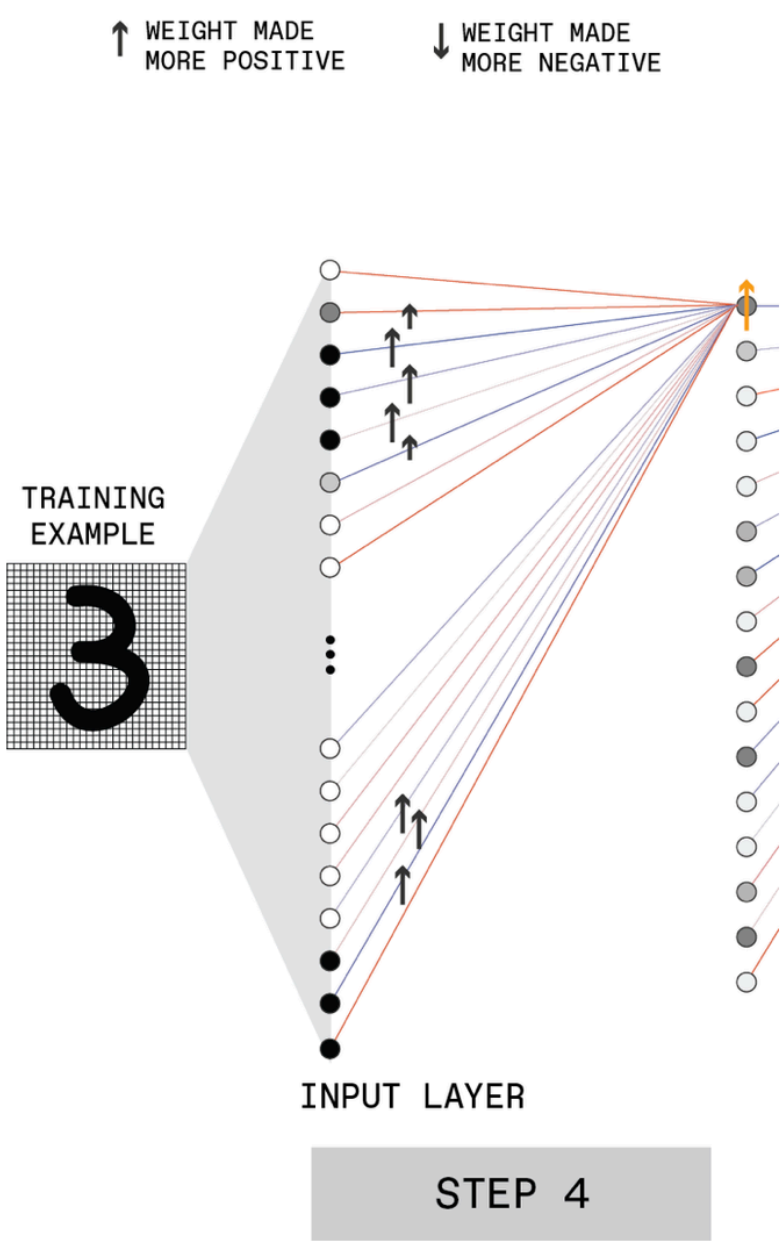


STEP 3

A similar process is then performed for the neurons in the second hidden layer. For example, to make the network more accurate, the top neuron in this layer may need to have its activation reduced [green arrow]. The network can be pushed in that direction by adjusting the weights of its connections with the first hidden layer [black arrows].

Training

Step 4



STEP 4

The process is then repeated for the first hidden layer. For example, the first neuron in this layer may need to have its activation increased [orange arrow].