Can you write a program that can recognize a number given a picture of it?

Can you write a program that can recognize a word given a audio file of it?

Can you write a program that can recognize a face from a given picture of it?

The answer to all these questions is likely no, especially not without any help. However, these problem have been attacked for the past few decades, and the way they have been attack is by using neural networks.

*What is a neural network?*

A neural network is a computer program, modeled after the brain. A neural network consists of nodes called neurons, arranged into layers, in which a neuron from one layer has a connection to neurons of the next layer.

Furthermore, each neuron has an *activation level* between 0 and 1, corresponding to a real neuron’s state (“firing or not firing”).

The activation levels of one layer of neurons completely determine the activation levels of the next layer of neurons.

Let’s try to tackle to problem of recognizing digits (0 – 9) using a neural network.

Program: Input = Picture of a digit -------- Output = what digit is in the picture

We have 4 layers:

Input layer – 784 Nodes, each of whose activation corresponds with the brightness of a pixel in the picture

2 Hidden layer – 16 Nodes each, activations are dependent on the activations of the layer before it (has a connection between every neuron it in the previous layer).

Output layer – 10 Nodes, each corresponding to a number 0-9. Node with highest activation corresponds with number program “decides” is in the picture.

Idea:

Break up a large problem into smaller level details.

Recognize a 9 <-> recognizing a loop and a line <-> recognizing four lines arrange straight and in a circle etc.

*How do we recognize these smaller level details?*

Say for example we are trying to find a line within the image, and we have a neuron in the second layer represent whether there is a line in the image or not. Then we could sum up the activation levels of every neuron corresponding to a pixel where we think we will find the line this sum is positive, we know these pixels are white, and therefore there is a line there. We can selected these four pixels by assigning weights to the connections between all the neurons in the input layer and this first neuron of the second layer. If the node corresponds to this section, it has weight of 1, else 0. However we also want to make sure that space surrounding the line is dark, so let’s assign a negative weight to the connections between this first neuron of the second layer, and the neurons of the first layer around the region we are interested in. Taking the net (sum), we can have the activation of this 2nd layer neuron to represent whether there is a line in the image or not. However, activations are between 0 and 1, while the net could be super high or low value. Therefore we apply a sigmoid “squishification” function to the net, to condense it between -1 and 1. A final element not to forget about is called *bias*, and it tells a neuron in the second layer when the *net* is significant enough for that neuron to be activated.

As there is a large number of linear calculations, this entire process of transforming activations from one layer to another layer is generally represented by an equation like this:

(A1) = W(A0) – b

Where,

A1 = activations of first hidden layer

A0 = activations of input layer

W = matrix representing weighted connections between every neuron in the first to the second

B = Bias vector containing biases for each neuron in 2nd layer

Big Ideas:

A neural network is a computer program modeled after the function of the brain. It is a great example of biomimicry, by studying and understanding the function of some animal or plant or insect, we can use that same principle to create a machine that can perform a similar function. A neural network models the brain’s ability to recognize similiariy among a diverse set of things.