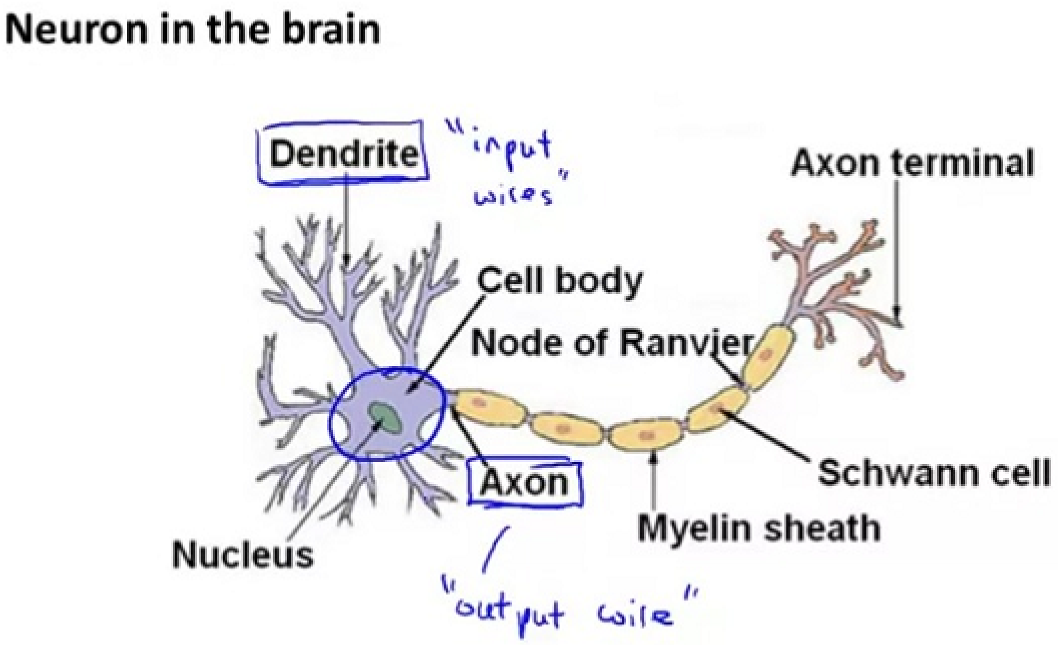
~ Model Representation I ~ ¶

How do we represent neural networks (NN's)?

Neural networks were developed as simulating neurons or networks of neurons in the brain. The neuron has a cell body, and moreover, the neuron has a number of input wires, and these are called the **dendrites**.

We think of them as input wires, and these receive inputs from other locations. And a neuron also has an output wire called an **Axon**, and this output wire is what it uses to send signals to other neurons, so to send messages to other neurons.

So, at a simplistic level what a neuron is, is a computational unit that gets a number of inputs through it input wires and does some computation and then it says outputs via its axon to other nodes or to other neurons in the brain.



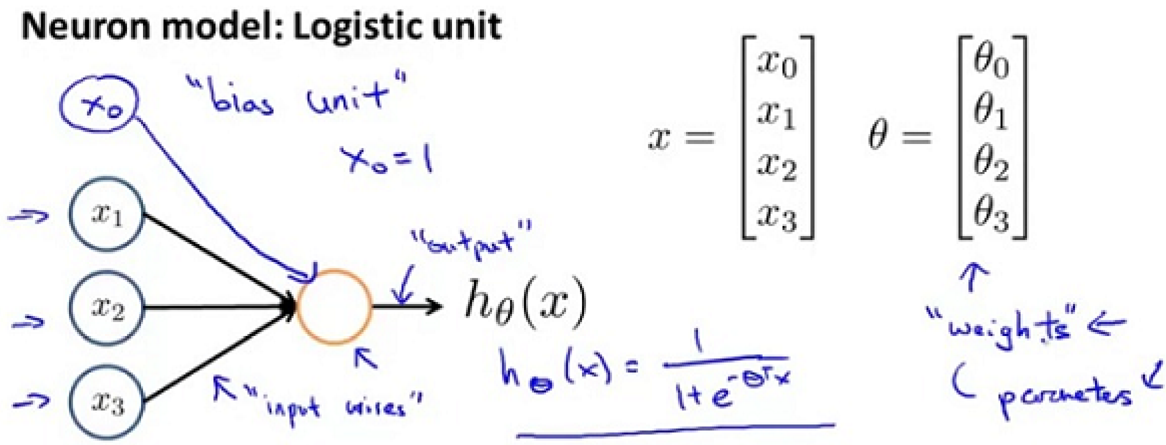
The way that neurons communicate with each other is with little pulses of electricity, they are also called spikes but that just means pulses of electricity. This is the process by which all human thought happens. It's these Neurons doing computations and passing messages to other neurons as a result of what other inputs they've got.

And, this is how our senses and our muscles work as well. If we want to move one of our muscles the way that where else in our neuron may send this electricity to our muscle and that causes our muscles to contract and our eyes, some senses like our eye must send a message to our brain while it does it senses hosts electricity entity to a neuron in our brain like so.

Neuron model: Logistic unit

In a neuron network, or rather, in an artificial neuron network that we've implemented on the computer, we're going to use a very simple model of what a neuron does we're going to model a neuron as just a **logistic unit**.

The way we feed the neuron is with a few inputs who's various dendrites or input wiles, and the neuron does some computation, and output some value on this output wire (the output wire in the biological neuron is the axon).

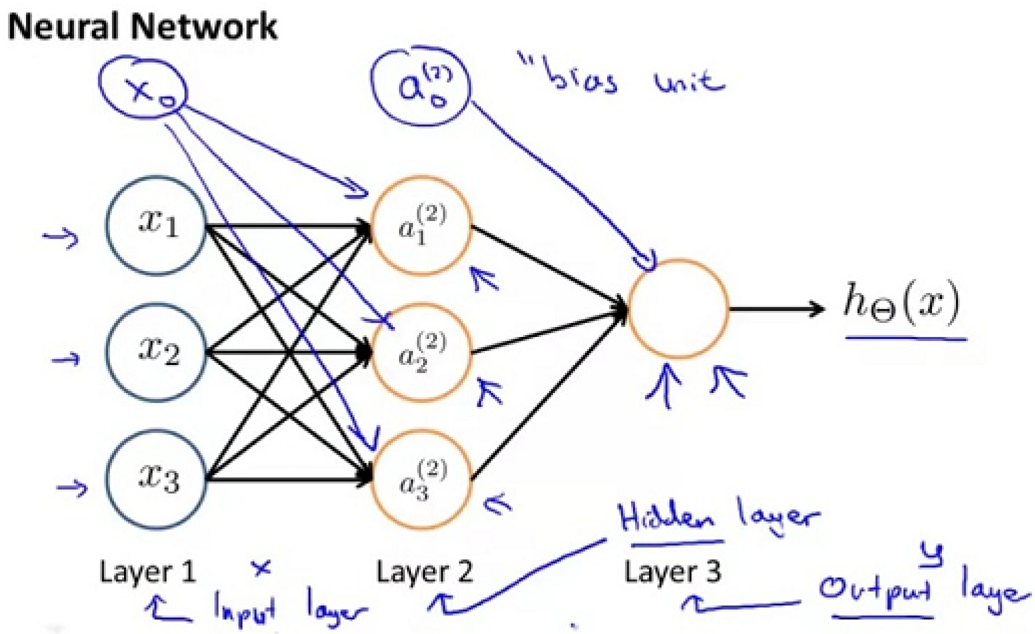


This is a very simple, maybe a vastly oversimplified model, of the computations that the neuron does, where it gets a number of inputs, x_0, x_1, x_2, x_3 and it outputs some value computed (x_0 sometimes is called the bias unit or the bias neuron, because x_0 is always equal to 1).

When we talk about neural networks, sometimes we'll say that this is a neuron or an artificial neuron with a **sigmoid** or **logistic activation function** (in the neural network terminology) and finally, the parameters (theta values) sometimes are called the weights of the model.

Neuronal Network

What a neural network is, is just a group of this different neurons strong together. In a neural network, the first layer, this is also called the **input layer** because this is where we input our features, x_1, x_2, x_3. The final layer is also called the **output layer** because that layer has a neuron, that outputs the final value computed by a hypothesis.



And then, layer 2 in between, is called the **hidden layer** (the hidden layer name is because, we don't observe the values processed in there). Basically, anything that isn't an input layer and isn't an output layer is called a hidden layer).

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \rightarrow \begin{bmatrix} a_0^{(2)} \\ a_1^{(2)} \\ a_2^{(2)} \\ a_3^{(2)} \end{bmatrix} \rightarrow h_\theta(x)$$

To understand these specific computations represented by a neural network, we are going to use the following notation:

$a_i^{(j)}$ = "activation" of unit i in layer j
$\Theta^{(j)}$ = matrix of weights controlling function mapping from layer j to layer $j+1$

The computations are represented by the following diagram:

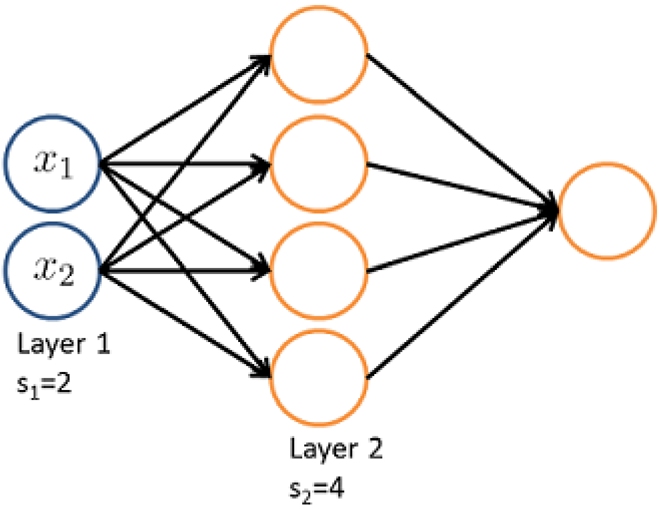
$$
\begin{aligned}
a_1^{(2)} &= g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3) \\
a_2^{(2)} &= g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3) \\
a_3^{(2)} &= g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3) \\
h_\Theta(x) = a_1^{(3)} &= g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})
\end{aligned}
$$

Looking at the Θ matrix

- Column length is the number of units in the following layer
- Row length is the number of units in the current layer + 1 (because we have to map the bias unit)

Concretely if network has s_j units in layer j, s_{j+1} units in layer $j+1$, then $\Theta^{(j)}$ will be of dimensions $s_{j+1} \times (s_j + 1)$.

Video Question: Consider the following neural network:



What is the dimension of $\Theta^{(1)}$ (Hint: add a bias unit to the input and hidden layers)?

- 2×4
- 4×2
- 3×4

4×3

Summary

Let's examine how we will represent a hypothesis function using neural networks. At a very simple level, neurons are basically computational units that take inputs (**dendrites**) as electrical inputs (called "spikes") that are channeled to outputs (**axons**). In our model, our dendrites are like the input features $x_1 \cdots x_n$, and the output is the result of our hypothesis function. In this model our x_0 input node is sometimes called the "bias unit." It is always equal to 1. In neural networks, we use the same logistic function as in classification, $\frac{1}{1+e^{-\theta^T x}}$, yet we sometimes call it a sigmoid (logistic) **activation** function. In this situation, our "theta" parameters are sometimes called "weights".

Visually, a simplistic representation looks like:

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} \rightarrow \begin{bmatrix} \ \ \ \end{bmatrix} \rightarrow h_\theta(x)$$

Our input nodes (layer 1), also known as the "input layer", go into another node (layer 2), which finally outputs the hypothesis function, known as the "output layer".

We can have intermediate layers of nodes between the input and output layers called the "hidden layers."

In this example, we label these intermediate or "hidden" layer nodes $a_0^2 \cdots a_n^2$ and call them "activation units."

$$\begin{aligned} & a_i^{(j)} = \text{"activation" of unit } i \text{ in layer } j \\ & \Theta^{(j)} = \text{matrix of weights controlling function mapping from layer } j \text{ to layer } j+1 \end{aligned}$$

If we had one hidden layer, it would look like:

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \rightarrow \begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \\ a_3^{(2)} \end{bmatrix} \rightarrow h_\theta(x)$$

The values for each of the "activation" nodes is obtained as follows:

$$\begin{aligned} a_1^{(2)} &= g(\Theta_{10}^{(1)} x_0 + \Theta_{11}^{(1)} x_1 + \Theta_{12}^{(1)} x_2 + \Theta_{13}^{(1)} x_3) \\ a_2^{(2)} &= g(\Theta_{20}^{(1)} x_0 + \Theta_{21}^{(1)} x_1 + \Theta_{22}^{(1)} x_2 + \Theta_{23}^{(1)} x_3) \\ a_3^{(2)} &= g(\Theta_{30}^{(1)} x_0 + \Theta_{31}^{(1)} x_1 + \Theta_{32}^{(1)} x_2 + \Theta_{33}^{(1)} x_3) \\ h_\Theta(x) = a_1^{(3)} &= g(\Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)}) \end{aligned}$$

This is saying that we compute our activation nodes by using a 3×4 matrix of parameters. We apply each row of the parameters to our inputs to obtain the value for one activation node. Our hypothesis output is the logistic function applied to the sum of the values of our activation nodes, which have been multiplied by yet another parameter matrix $\Theta^{(2)}$ containing the weights for our second layer of nodes.

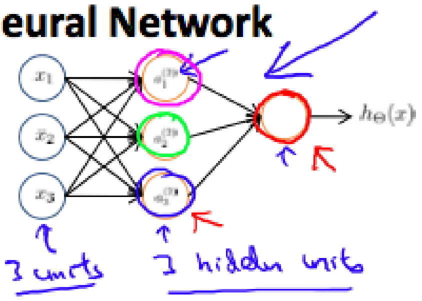
Each layer gets its own matrix of weights, $\Theta^{(j)}$.

The dimensions of these matrices of weights is determined as follows:

- If network has s_j units in layer j, s_{j+1} units in layer $j+1$, then $\Theta^{(j)}$ will be of dimensions $s_{j+1} \times (s_j + 1)$.

The +1 comes from the addition in $\Theta^{(j)}$ of the "bias nodes," x_0 and $\Theta_0^{(j)}$. In other words the output nodes will not include the bias nodes while the inputs will. The following image summarizes our model representation:

Neural Network



→ $a_i^{(j)}$ = "activation" of unit i in layer j

→ $\Theta^{(j)}$ = matrix of weights controlling function mapping from layer j to layer $j+1$

$\Theta^{(1)} \in \mathbb{R}^{3\times 4}$ $h_\Theta(x)$

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$

$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$

$\Theta^{(2)}$

$$a_3^{(2)} = g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$

$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$

→ If network has s_j units in layer j, s_{j+1} units in layer $j+1$, then $\Theta^{(j)}$ will be of dimension $\boxed{s_{j+1}} \times (\boxed{s_j} + 1)$.

$s_{j+1} \times (s_j + 1)$

Andrew N

Example: If layer 1 has 2 input nodes and layer 2 has 4 activation nodes. Dimension of $\Theta^{(1)}$ is going to be 4×3 where $s_j = 2$ and $s_{j+1} = 4$, so $s_{j+1} \times (s_j + 1) = 4 \times 3$.