A Simple Introduction to Neural Networks

Jack Adam Collins

This paper will discuss what neural networks are (part I), why they work (part II), and how they are trained (part III). The central aim of this post is two-fold. Firstly, to reinforce what I have learned about neural nets via research and development within the machine learning team. Secondly, to share this knowledge more widely for the purposes of finding an appropriate application for this methodology to work within Quantum. This introduction is written to serve as an accessible presentation with the first two sections requiring only minimal mathematical notation.

I

Neural networks describe a class of machine learning predictors which take their influence from the biological architecture that constitutes the animal brain[1].

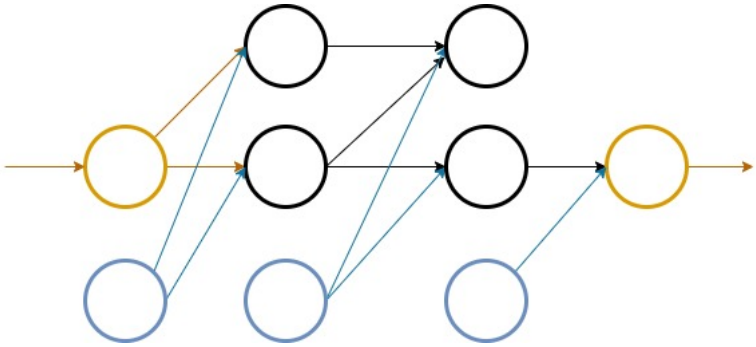

Figure 1: Schematic for a basic feed-forward neural network

The neural network in the above schematic serves as primary example throughout, beginning by defining the specific components visualised in the diagram. The circles represent individual *neurons*, and the interconnecting arrows are *edges*. In combination, they define the *graph* underlying the neural network – a cluster of nodes with linking arrows between them. If the graph has no cycles, that is, edges looping around in a circle, then the neural net is considered *feed-forward*, which for the purposes of simplicity will be the focus of the neural networks presented. In the following image, the green neuron is the *input neuron*, the yellow neuron is the *output neuron*, and the blue neurons are *bias neurons*.

[1] Warren S. McCulloch; Walter Pitts. "A Logical Calculus of the Idea Immanent in Nervous Activity"

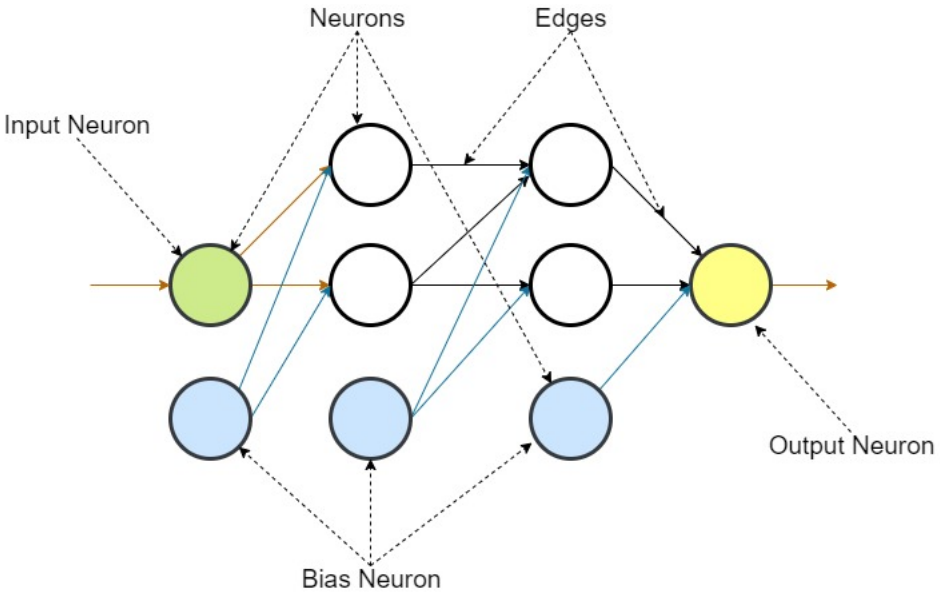

Figure 2: Labelled schematic for the feed-forward neural network

If the possibility exists of dividing the entire set of neurons into an ordered list of subsets, such that each neuron *only* points towards neurons in the subset next in line, then the network is considered *layered*. While the focus here is layered networks, non-layered networks do exist. For example, a neural net with the given edges:

$$A \to B, \tag{1}$$

$$A \to C, \tag{2}$$

$$B \to C \tag{3}$$

cannot be divided into layers.

The graph underlying the neural network illustrates the way in which each neuron points only towards neurons within the next layer. This depiction is simplified but can be expanded upon by incorporating adequate notation:

In the schematic, *weights* are given by $w_{i,j}$. The weights determine the relative importance of the value of the incoming edge. *Functions* are given by σ_x, in which functions are considered in the classical sense: a value is taken as input before being output as something else. If we are to discuss, say, the 3^{rd} neuron, then we shall herein use the notation σ_3. Referring specifically to the previous diagram we can now walk through the particular operations of the network. An input value x is given to the input neuron, σ_1, which forwards it to neurons three, σ_3, and four, σ_4. Synchronously, σ_1 (the first bias neuron) sends a constant value to neurons three and four (sending the number 1 is all that bias neurons ever do). At this point, the third neuron has received two values – it now multiplies them with the weights on the respective edges, so it multiplies the value x input from the first neuron with $w_{1,3}$ and the number 1 coming from the

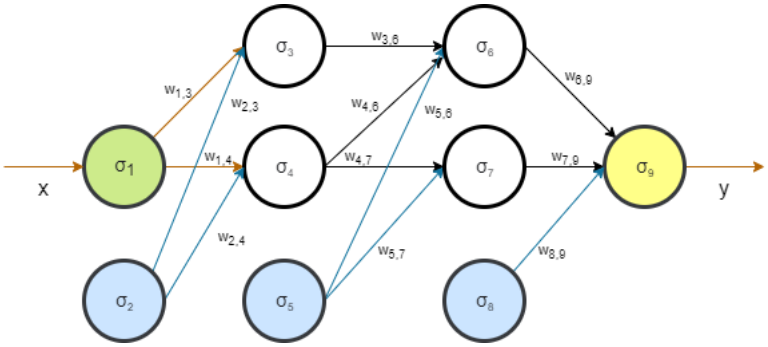

Figure 3: The feed-forward neural network with the layers depicted

bias neuron with $w_{2,3}$. The values are then added together and the function σ_3 is then applied to them. The result of this process is:

$$\sigma_3(w_{1,3} \cdot x + w_{2,3} \cdot 1) \tag{4}$$

The fourth neuron also receives two values, performing the same operation (using the weights $w_{1,4}$ and $w_{2,4}$), leading to the term:

$$\sigma_4(w_{1,4} \cdot x + w_{2,4} \cdot 1) \tag{5}$$

The third neuron outputs its value to the sixth neuron, the fourth outputs its value to both the sixth and the seventh, and the fifth neuron (the bias neuron) outputs the number 1 to both the sixth and the seventh neuron. Their weights are then applied, then their functions, and so on. Eventually, the ninth neuron receives a value, applies σ_9 to it, and outputs the result, which is the resulting output of the entire network.

Recall that the blue neurons depicted in the schema are named bias neurons – having no incoming edges. Instead of computing something, they always output a given number and thus bias the weighted sum, which arrives at the neurons in the next layer, by a constant factor. Each hidden layer has exactly one bias neuron as does the input layer. In this way, each neuron, excluding the input neurons has an incoming edge from a bias neuron. Also, note that the input neuron doesn't apply any function to the input it receives (but all other neurons do).

To describe this process more expansively, additional notation is required in order to express the value incoming a given neuron, in particular, the weighted sum of the values on the incoming edges, and the final value which comes out after the application of the function. Examining the sixth neuron, σ_6 in particular:

IMAGE NEURON 6 GOES HERE

Evidently, the input values are received, the weighted sum a_6 is computed, then σ_6 is applied to it, resulting in the output value z_6. Thus, we can generalise:

$$z_6 = \sigma_6(a_6) = z_6(w_{3,6} \cdot z_3 + w_{4,6} \cdot z_4 + w_{5,6}) \tag{6}$$

Hence, the output of a given neuron is expressed in terms of the outputs of the neurons in the previous layer; thus, the equation can give a description of the output

of any neuron within the network. Note, the value z_5 is missing – since σ_6 is a bias neuron with the constant value 1, therefore $w_{4,5} \cdot z_5$ is equivalent to $w_{4,5}$. Furthermore, note that the image shows the weights of the incoming edges, but not the weight of the outgoing edge: the weights of an edge should always be thought of as belonging to the neuron the edge acts as an input to.

Recalling that our neural network can be divided into layers, a useful process for purposes of evaluating the network, we can analyse further:

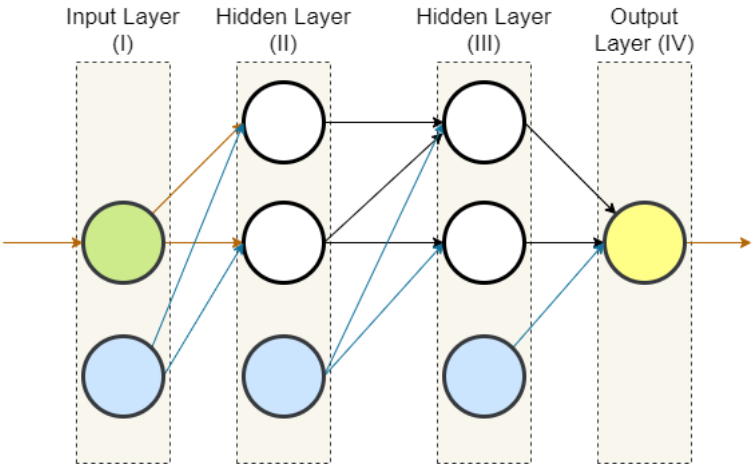

Figure 4: The feed-forward neural network with the respective layers depicted

To begin, the network is fed the input value, or values, x_k, setting:

$$z_k = x_k \tag{7}$$

for each neuron, k, in the input layer. Assuming all output values for some layer, n, are known, it is possible to then compute the output values for each neuron, k, in layer $n + 1$ by setting

$$z_k = \sigma_k \left(\sum_{i \in I_k} w_{i,k} z_i \right) \tag{8}$$

where I_k is the set of indices of neurons with an edge pointing towards neuron k. As the equation applies for any layer, the entire network can be evaluated, a single layer at a time. The output values z_k of the final layers will be the output values y_k of the network.

II

What are neural networks good for? Why do they work so well?

If all values[2], are real numbers, then the total network *implements a function*:

[2]That is, input, output, and the values sent between neurons