The introduction of artificial neural networks

1.Purpose

This article illustrates the artificial neural network briefly,to tell the beginner who have no previous knowledge about ANNs,how dose it works.

2.Biological neural networks

Each neuron has a wire-like thread called an axon ,which is used to transmit signals to other neurons,and a multitude of other smaller threads (called *dendrites*) branching in every direction.The neurons exchange signals using an electrochemical process. Incoming signals are received at the junctions where the synaptic terminals and the dendrites meet.These junctions are known as the synapses. The neuron sums all the incoming signals from the synapses, and if the total signal exceeds a threshold value, the neuron fires and an electrical signal is sent shooting down the axon. If the total is less than the threshold, the neuron doesn’t fire.

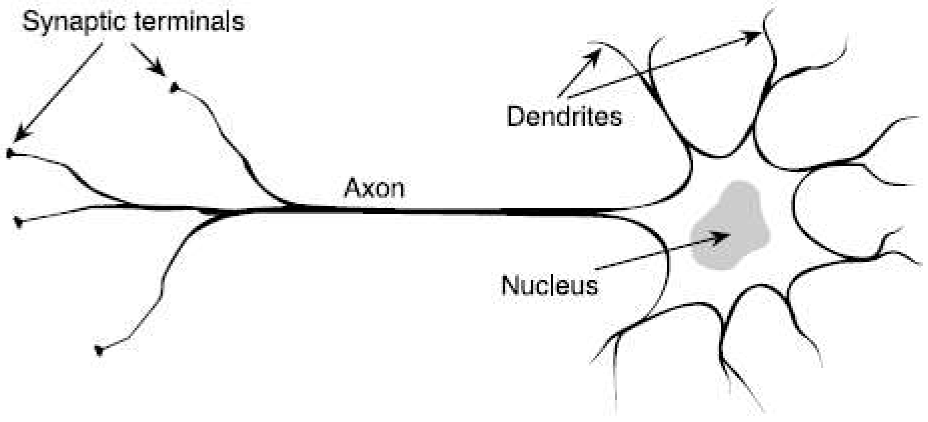


Figure 1

3.Artificial neural network

ANNs are built the same way as biological neural networks in that they use many artificial neurons.check out Figure 2,it depicts one way of representing an artificial neuron.The w’s in the gray circles represent floating-point numbers called *weights*. Each input into the artificial neuron has a weight associated with it and it’s these weights that determine the overall activity of the neural network. For the moment, imagine that all these weights are set to small random values—let’s say between -1.0 and 1.0. Because a weight can be either positive or negative, it can exert an excitory influence over the input it’s associated with, or it can exert an *inhibitory* influence. As the inputs enter the neuron, they are multiplied by their respective weights. A function in the nucleus—the *activation function*—then sums all these new, weight-adjusted input values to give the *activation value* (again a floating-point number, which can be negative or positive). If this activation value is above a certain threshold, let’s use the number one as an example, the neuron outputs a signal and will output a one. If the activation is less than one, the artificial neuron outputs a zero. This is one of the simplest types of activation functions found in artificial neurons and it's called a step function(look at Figure 3).

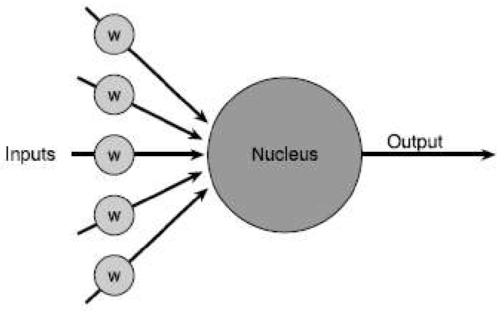


Figure 2

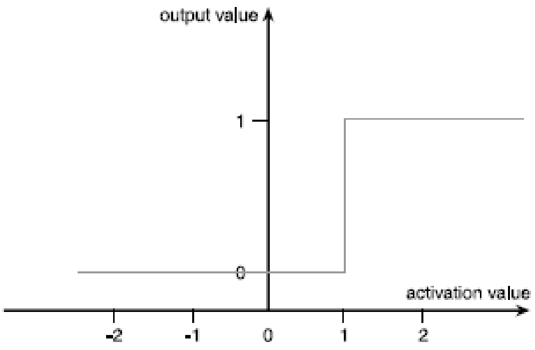


Figure 3

An artificial neuron can have any number of inputs numbered from one to n—where n is the total number of inputs. Each input can be expressed mathematically as:

X1,X2,X3,X4….Xn

The weights can be expressed similarly as:

W1,W2,W3,W4….Wn

Remember, the activation is the sum of all the weights × inputs. This can now be written as:

A = X1W1+X2W2+X3W3+X4W4….+XnWn

Figure 4 represents the equations as a diagram. Remember, if the activation exceeds the threshold, the neuron outputs a one; if the activation is below the threshold, the neuron outputs a zero. This is equivalent to a biological neuron firing or not firing. Imagine a neuron with five inputs, and all its weights initialized to random values (-1 < *w* < 1).

Just as biological neurons in the brain connect to other neurons, these artificial neurons are connected together in some way to create the neural network. The most widely used is by connecting the neurons together in layers, as in Figure 5. This type of ANN is called a feedforward network. It gets its name from the way each layer of neurons feed their outputs into the next layer until an output is given. Each input is sent to every neuron in the hidden layer, and then the output from each neuron in the hidden layer is connected to every neuron in the next layer. There can be any number of hidden layers within a feedforward network, but one is usually enough to cope with most of the problems you will tackle. In fact, there can be any number of neurons in each layer(it all depends on the problem).

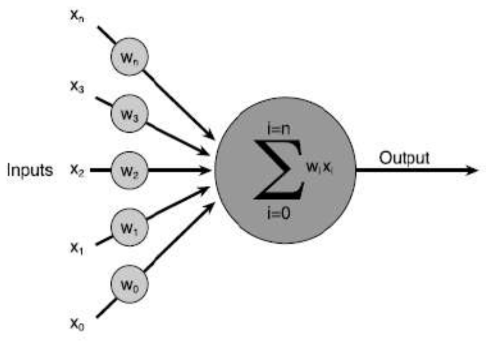


Figure 4

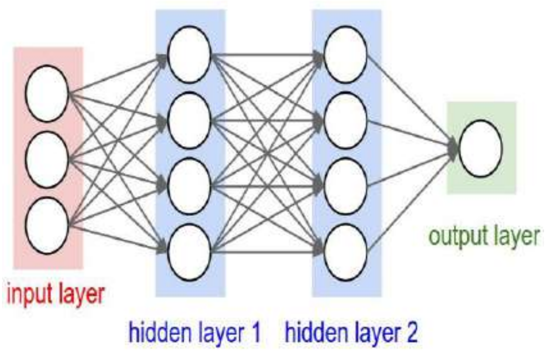


Figure 5

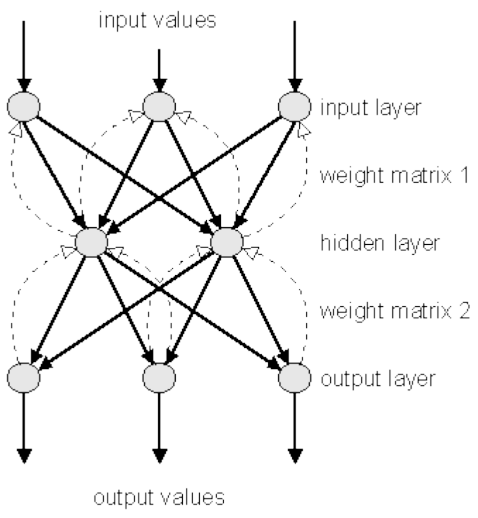
4.Back propagation algorithm

Backpropagation works like this: First create a network with one or more hidden layers and randomize all the weights—say to values between -1.0 and 1.0. Then present a pattern to the network and note its output. The difference between this value and the target output value is called the error value. This error value is then used to determine how the weights from the layer below the output layer are adjusted so if the same input pattern is presented again, the output will be a little closer to the correct answer. Once the weights for the current layer have been adjusted, the same thing is repeated for the previous layer and so on until the first hidden layer is reached and all the weights for every layer have been adjusted slightly. If done correctly, the next time the input pattern is presented, the output will be a little bit closer to the target output. This whole process is then repeated with all the different input patterns many times until the error value is within acceptable limits for the problem at hand. The network is then said to be trained.

5.The neural networks training

The training set required for an ANNs to learn would be a series of vectors like this:
This set of matched input/output patterns is used to train the network as follows:

1. Initialize weights to small random values.
2. For each pattern, repeat step a to step e
   - a. Present to the network and evaluate the output *o*.
   - b. Calculate the error between *o* and the target output value (*t*).
   - c. Calculate the error in the hidden layer.
   - d. Adjust the weights in the output layer.
   - e. Adjust the weights in the hidden layer.
3. Repeat Step 2 until the sum of all the errors in Step b is within an acceptable limit.



6.The formula of neural networks training

1.Calculate the Error for the Output Layer:

The output from a neuron *k*, will be given as O_k and target output from a neuron will be given as T_k. To begin with, the error value E_k, for each neuron is calculated.

$$E_k = (T_k - O_k) * O_k(1 - O_k)$$

2.Calculate the Error for the Hidden Layer:

A neuron j in a hidden layer, the error value is calculated like this.

$$E_j = O_j(1 - O_j) * (E_1 W_{j1} + E_2 W_{j2} + E_3 W_{j3} + \ldots + E_k W_{jk} + \ldots + E_n W_{jn})$$

(n is the number of units in the output layer)

3.Adjusting the Weights for the Output Layer:

To change the weight between a unit j in the hidden layer and an output unit k, use the following formula:

$$W_{jk} += L * E_k * O_j$$

(L is a small positive value known as the learning rate. The bigger the learning rate, the more the weight is adjusted. This figure has to be adjusted by hand to give the best performance)

4.Adjusting the Weights for the Hidden Layers:

the weight adjustment from the hidden unit j, to the input units i, can be made:

$$W_{ij} += L * E_j * O_i$$

5.This entire process is repeated until the error value over all the training patterns has been reduced to an acceptable level.

7. Further reading

About artifical neural networks:

https://en.wikipedia.org/wiki/Artificial_neural_network

About Backpropagation algorithm:

https://en.wikipedia.org/wiki/Backpropagation