Arduino Basics

An Arduino tutorial blog. Free Arduino tutorials for everyone !

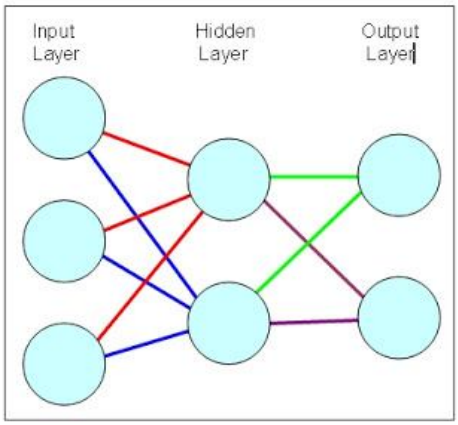
11 August 2011

Neural Network (Part 1) - The Connection

Introduction

In this tutorial, I am going to walk you through my interpretation of a neural network. I will use terminology that makes sense to me, hoping that Neural Network enthusiasts don't get offended by my novice approach.

This is what a feed-forward Neural network normally looks like:



The input layer receives input from the outside world, and passes this value to the hidden layer.

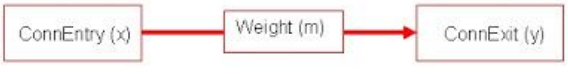
The value that reaches the hidden layer depends on the connection between the layers.

Each connection has a weight. This weight multiplier can either increase or decrease the value coming from the input layer. Like water coming out of a tap, you can make it come out faster or slower (or not at all). This weight can even be negative, which would mean that the water is being sucked up, rather than pouring out.

The hidden layer then processes the values, and pass them onto the output layer. The connections between the hidden layer and the output layer also have weights. The values in the output layer are processed and produce a final set of results. The final results can be used to make yes/no decisions, or can be used to make certain classifications etc etc. In my case, I would like to receive sensor data from the arduino, pass this info to the neural network, and get it to classify the data into 4 different classes (Red, Yellow, Green or Ambient), but we will get to see this in another tutorial. For now, we are just going to design a neural network that can be applied to any of your arduino projects... so lets start from the ground up.

I have programmed this neural network in the Processing language, and have decided to break the neural network into smaller bits. Each structural component of the neural network is a class (as you will soon discover).

The connection



ConnEntry (x) : is the value being fed into the connection.
Weight (m) : Is the value that either amplifies or weakens the ConnEntry value.
ConnExit (y): Is the output value of the connection.

The relationship between y and x is linear, and can be described by the following formula.

$$y = mx$$

In other words, you multiply the ConnEntry value by the Weight to get the ConnExit value.

Here is the code for the Connection class:

processing code Connection Class

```
/* ---------------------------------------------------------------------------
A connection determines how much of a signal is passed through to the neuron.
---------------------------------------------------------------------------*/

class Connection{
  float connEntry;
  float weight;
  float connExit;

  /*This is the default constructor for a Connection  */
  Connection(){
    randomiseWeight();
  }

  /*A custom weight for this Connection constructor  */
  Connection(float tempWeight){
    setWeight(tempWeight);
  }

  /*Function to set the weight of this connection  */
  void setWeight(float tempWeight){
    weight=tempWeight;
  }

  /*Function to randomise the weight of this connection  */
  void randomiseWeight(){
    setWeight(random(2)-1);
  }

 /*Function to calculate and store the output of this Connection  */
  float calcConnExit(float tempInput){
    connEntry = tempInput;
    connExit = connEntry * weight;
    return connExit;
  }
}
```

code formatter

When a connection class is constructed, it automatically randomises the weight for you.
Here are some of the other functions of this class:

setWeight() : allows you to specifically set the weight of this connection.

randomiseWeight() : can be used to wipe the "memory" of the connection if required.

calcConnExit() function is the main function of this class. It is the one that multiplies the connEntry value with the Weight to produce a connExit value as described above.

Up next: Neural Network (Part 2): The Neuron

To go back to the table of contents click here

Posted by Scott C at 23:16

Labels: ArduinoBasics, best arduino blog, code, Connection, Feed Forward, Neural Network, Processing.org,

project, tutorial, Weight

1 comment:

Anonymous 20 July 2012 at 11:52

I love your blog!
really useful!

:)

Reply

Enter your comment...

Comment as: Unknown (Goo ▼ Sign out

Publish Preview Notify me

Feel free to leave a comment about this tutorial below.
Any questions about your particular project should be asked in the ArduinoBasics forum.

Comments are moderated due to large amount of spam.

Links to this post

Create a Link

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)

This Week's Most Popular Posts

433 MHz RF module with Arduino Tutorial 1

HC-SR04 Ultrasonic Sensor

Simple Arduino Serial Communication

Relay Module

433 MHz RF module with Arduino Tutorial 2

Most Recent Posts

Get Arduino Data over the internet using jQuery and AJAX

Two Million Views

Generosity Campaign Update - Day 3

Generosity Campaign Update - Day 2

Generosity campaign - Day 1

Recent Posts Widget by Helplogger