**Connor Lambert**

**Neural Network to Recognize Handwritten Digits**

**CSC 492: Special Problems – Dr. Ali**

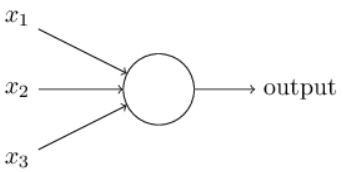
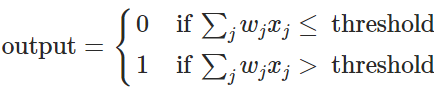
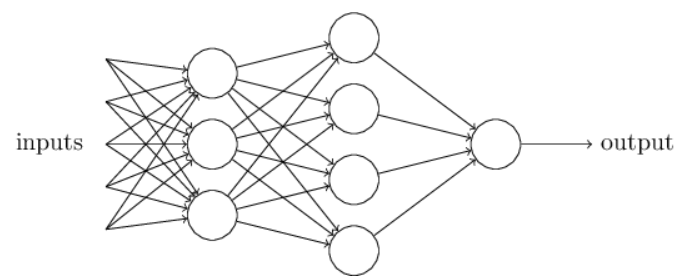
**Research Notes**

**Goal:** To create a neural network, that which can recognize handwritten digits.

**Background of Problem:** Image recognition using our eyes is deceptively easy, as it has evolved across hundreds of millions of years into a system of visual cortices working in conjunction to perceive our surrounding environment. When we view a number written by a human, we can use visual patterns to almost instantaneously know what number we are looking at – “an 8 has two circles sitting on top of one another”. The concept of visual pattern recognition is easy when we do it but becomes extremely complicated when attempting to express in the form of an algorithm. Using precise rules to express this unconscious pattern recognition can result in becoming lost in a series of cases and exceptions.

**Solution:** Develop a system that which can take many handwritten digits (training data) and create rules to assist in digit recognition.

**Notes on Methods Etc.:**

* **Perceptrons**
  + Developed by Frank Rosenblatt in the 50’s and 60’s.
  + Not commonly used in present day compared to Sigmoid Neurons
  + Perceptrons take several binary inputs and produce a single binary output:
  + 
  + Computation of the output is achieved by pairing each input with something called a *weight* – a real number that expresses the importance of the respective inputs to the output. Once paired, each input is multiplied by its weight and summed with the other products. If this summation is <= some threshold value, the Output is 0, otherwise if it is greater than this threshold value, the Output is 1.
  + Formula for Output: 
  + We can abstract network of perceptrons into more layers, giving the network an ability to engage in more sophisticated decision-making
  + We can see from the above graph, layer 1 of perceptrons are making three “simple” decisions based off the weight of the given input. From that point, layer 2 perceptrons make decisions by weighing up the decisions from layer 1. The idea is, the latter layers can make decisions at more complex and abstract levels than perceptrons in previous layers.
  + **Simplifying the Formula**
    - The notation, **∑jwjxj > threshold**, is cumbersome and can be simplified.
    - First, we can rewrite “**∑jwjxj**” as a dot product, ***w* ∙ *x* ≡ ∑jwjxj**. *W* and *X* are vectors whose components are the weights and inputs respectively.
    - Second, we can move the **threshold** to the other side of the inequality, and to replace it with what is more popularly used, the perceptron’s ***bias, b* ≡ -threshold**. The bias is the measure of how easy it is for the perceptron to “fire”. If the bias is large, it’s extremely easy to receive an output of 1, the same is true for a small bias and an output of 0.
    - The final form of the new formula can be written as:

**Sources**

* Nielsen, Michael (2018). *Neural Networks and Deep Learning.* Retrieved from http://neuralnetworksanddeeplearning.com