34 input nodes

* 16-bit first number
* 16-bit second number
* 2-bit function indicator

1156 weights

34 node layer

1156 weights

34 node layer

1088 weights

32 node output

1 bias per non-input node – 100 biases total

Add all weights including bias then sigmoid that sum

Methods:

Build one with random weights

Build many with random weights

Test one

Test many

Tweak one

Tweak many

Tweak, test, tweak, test, …

Class Node

* inInputLayer
  + setInInputLayer(boolean)
    - inInputLayer = Boolean
  + getInInputLayer()
    - return inInputLayer
* Bias
  + If inInputLayer
    - Bias = 0
  + setBias
  + getBias
* Input
  + setInput(pinput)
    - input = pinput
    - setOutput()
  + getInput()
    - return input
* Function
  + If inInputLayer
    - Assign step function
  + Else
    - Assign sigmoid function
* Output
  + setOutput()
    - Output = function(input + bias)
  + getOutput()
    - return output

Class Network

* Node arrays to be layers
  + 4 distinct arrays
    - firstLayer (input) = 34 nodes
    - secondLayer = 34 node
    - thirdLayer = 34 nodes
    - fourthLayer (output) = 32 nodes
* Weight arrays for each non-input node
  + Each element corresponding to the weight of a previous layer node on it
    - 34 weight array per secondLayer node
    - 34 weight array per thirdLayer node
    - 34 weight array per fourthLayer node
* Sum [weights (move along a row)]\*[previous layer outputs (move down previous layer)] to be each current layer node input
  + For x in range(0, len(currLayer))
    - cnode = currLayer[x]
    - cnodeWeightedSum = 0
    - For y in range(0, len(prevLayer))
      * pnode = prevLayer[y]
      * cnodeWeightedSum += pnode.getOutput \* currLayerWeights[x][y]
    - cnode.setInput(weightedSum)
* CheckOutput
  + showInput
    - input = []
    - for node in firstLayer
      * input.append(node.getOutput)
  + showExpectedOutput
  + ShowOutput
    - Output = []
    - For node in fourthLayer
      * Output.append(node.getOutput)

Class Builder

Class Tester

Class Tweaker

Class Trainer