Chapter 15: Associative Learning

Brandon Morgan

2/1/2021

E15.1

We are given the following single layer *hardlim* network with initial weights $w^0 = 1$,$w = 0$, and bias -0.8; as well as the training sets:

$$p^0 = 1, p = 1$$

$$p^0 = 0, p = 1$$

We need to train using Hebb rule with decay, $\alpha = 0.3$ and $\gamma = 0.1$.

1

With decay, Hebb's rule becomes:

$$W(q) = W(q-1) + \alpha a(q)p^T(q) - \gamma W(q-1)$$

$$a(1) = hardlim(w^0p^0(1) + w(0)p(1) - 0.8) = hardlim(1(1) + 0(1) - 0.8) = 1$$

Now we update our weights:

$$w(1) = w(0) + \alpha a(1)p^T(1) - \gamma W(0) = 0 + 0.3(1)(1) - 0.1(0) = 0.3$$

Now with the next input set:

$$a(2) = hardlims(w^1p^0(2) + w(1)p(2) - 0.8) = hardlims(1(0)) + 0.3(1) - 0.8) = 0$$

Now we update our weights:

$$w(2) = w(1) + \alpha a(2)p^T(2) - \gamma W(1) = 0.3 + 0.3(0)(1) - 0.1(0.3) = 0.27$$

We can repeat this process through the code below:

```
hebb = function(weight, input, bias, learning_rate, decay_rate, maxIter, mask) {
  size = length(input)
  index = 1
  all = 0
  iter = 0
  while(all<size) {
    if(index > size) {
      index = 1
    }
    p = input[[index]]
    n = weight%*%p+bias
    hardlim = function(x) {
      if (x<0) {
        return(0)
      }
      return(1)
    }
    a = hardlim(n)
    weight[1,2] = weight[1,2]+learning_rate*a*p[2,1]-decay_rate*weight[1,2]
    index = index + 1
    if(iter > maxIter-1){
       print(sprintf("   MAX ITER TAKEN"))
      print("Final Weights: ")
      print(weight)
      return(weight)
    }
    if(!mask) {
      print(sprintf("   ITERATION: %d", iter+1))
      print(sprintf("n: %f", n))
      print(sprintf("a: %f", a))
      print(sprintf("New weights:"))
      print(weight)
    }
    if(a==1) {
      all = all+1
    }
    if(iter %% size == 0 && iter > 0 && all == 0) {
      all = 0
    }

    iter = iter + 1
  }
  print(sprintf("   ALGORITHM CONVERGED"))
  print(sprintf("Iter taken: %d", iter))
  print("Final Weights: ")
  print(weight)
  return(weight)
}

weight = matrix(c(1,0),nrow=1)
input1 = matrix(c(1,1),nrow=2)
input2 = matrix(c(0,1),nrow=2)
```

```
w=hebb(weight, list(input1, input2), -0.5, 0.3, 0.1, 30, 0)
```

```
## [1] "   ITERATION: 1"
## [1] "n: 0.500000"
## [1] "a: 1.000000"
## [1] "New weights:"
##      [,1] [,2]
## [1,]    1  0.3
## [1] "   ITERATION: 2"
## [1] "n: -0.200000"
## [1] "a: 0.000000"
## [1] "New weights:"
##      [,1] [,2]
## [1,]    1 0.27
## [1] "   ITERATION: 3"
## [1] "n: 0.770000"
## [1] "a: 1.000000"
## [1] "New weights:"
##      [,1]  [,2]
## [1,]    1 0.543
## [1] "   ALGORITHM CONVERGED"
## [1] "Iter taken: 3"
## [1] "Final Weights: "
##      [,1]  [,2]
## [1,]    1 0.543
```

We can see here it took five iterations for the network to learn the classifications.