Q1: Derive a gradient descent training rule for a single unit with output o, where

**Solution**

First, the error function is defined as:

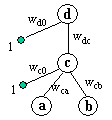
The update rule is the same, namely:

For ,

Thus,

For

Q2: Consider a two-layer feedforward ANN with two inputs and , one hidden unit ,  
and one output unit . This network has five weights , where  
 represents the threshold weight for unit . Initialize these weights to the values  
(.1, .1, .1, .1, .1), then give their values after each of the first two training iterations of  
the BACKPROPAGATION algorithm. Assume learning rate = .3, momentum = 0.9,  
incremental weight updates, and the following training examples:



|  |  |  |
| --- | --- | --- |
| a | b | d |
| 1 | 0 | 1 |
| 0 | 1 | 0 |

With the use of "threshold weight for unit x", you can assume the network as above, where the nodes in green are threshold units and their values are 1.

**Solution**

Training example 1 <<1, 0>, 1>

*Step 1: Propagate forward*: compute the activation of the nodes, noting that and :

*Step 2: Propagate backward*

First compute the error at each node, noting that :

Compute the correction terms as follows, noting that , and :

and the new weights become:

Training example 2 <<0, 1>, 0>

*Step 1: Propagate forward*: compute the activation of the nodes, noting that and :

*Step 2: Propagate backward*

First compute the error at each node, noting that

Compute the correction terms as follows, noting that , , and :

and the new weights become:

Q3: Revise the BACKPROPAGATION algorithm in Table 4.2 so that it operates on units

using the squashing function in place of the sigmoid function. That is, assume

the output of a single unit is . Give the weight update rule for output

layer weights and hidden layer weights. Hint: .

**Solution**

*Propagate the input forward through the network:*

1. Input the instance to the network and compute the output of every unit *u* in the network.

*Propagate the error backward through the network:*

1. For each network output unit *k*, calculate its error term
2. For each hidden unit *h*, calculate its error term
3. Update each network weight

where