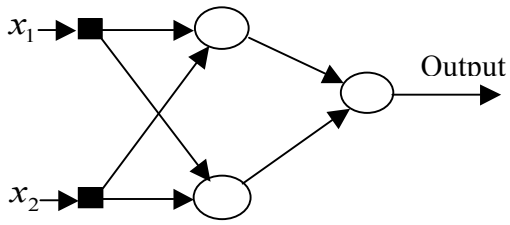
The Johns Hopkins University

JHU ENGINEERING FOR PROFESSIONALS PROGRAM

NEURAL NETWORKS

Module 7 Programming Assignment

For this assignment, create a neural network program using a language of your choice that implements the feedforward, back propagation (FFBP) algorithm for a network with the following topology like the one in the lecture:



You must use your own code --- no libraries or neural network packages can be used except ones that you have written yourself. In other words, you are to write your own implementation of the FFBP algorithm in a neural network **from scratch** and using only code you have written from previous programming assignments**.**

Use the following parameters and assumptions:

- Set the initial weights for both nodes in the first layer are 0.3 and for the second layer as 0.8.
- Set $\eta = 1.0$ and all biases to 0.
- Use inputs {1,2} for the two input values and use the desired output value of 0.7 as was done in class.
- For this part of the assignment, disable routines that update the biases.
- **Verify that your network performs correctly** by checking the results for the first iteration as was done in the lecture videos for this network and shown in the slides (Module 5.2-An Example). Correct any mistakes you find in your code until your answers match those in the slides.
- Once you have ensured that your program works correctly, do the following exercise **after you re-enable routines that update the bias**:
- Train the network with the following two input/output pairs using the two different approaches described below and the initial weights, bias and η values noted above:

  (1, 1) → 0.9;
  (-1, -1) → 0.05;

Before you run your code to perform each of the methods described below, make sure you use the proper initial values for the weights and biases as noted above, otherwise your final computations will be off.

Method 1: For each cycle of this training procedure, present the first input/output pair, perform the back propagation technique to update the weights, then present the second input/output pair and again perform the back propagation technique to update the weights. This constitutes a single cycle. Perform 15 such cycles and determine the errors E for each input/output pair. This method essentially updates the weights by alternately presenting each input/output pair … the first pair, and then the second pair and so on. **After** the 15^{th} training cycle, present the input values to the network and print out the total Error (Big E) and the final weights associated with each input/output pair.

Method 2: In this method, we update weights for **one** input/output pair for 15 iterations of the FFBP algorithm, then present the second input/output pair and run the FFBP algorithm to update weights for another 15 iterations. Thus, for each cycle, present the first input/output pair, run the FFBP for 15 iterations, then present the second input/output pair and run the FFBP for another 15 iterations. This second set of iterations therefore begins updating the weights from the values obtained after the first 15 iterations with the first input/output pair. **After** the training of the second input/output pair, present the input of the first pair, print out the total Error (Big E) and the final weights associated with it, then present the input of the second pair and print out the total Error (Big E).

Answer the following questions:
Run your finished program and answer the online questions. Before answering the online questions, ensure that the outputs of your network have been produced **after all the training cycles**.

Provide contributions to the Forum as to how the two methods compare? Is one method superior to the other? What do you think might happen if we approached the problem in the same way as above but with different initial values for the weights?

After you have finished your online questions, submit your code for review. This code will be checked using numerous databases, so make sure it's your own code!