CSC 578 Homework 2, Fall 2021

- **Graded out of 5 points.**
- Create a single pdf file containing all of your answers.
- Do all questions
- **DO NOT include the question** in your submission. Just number your answers.

---

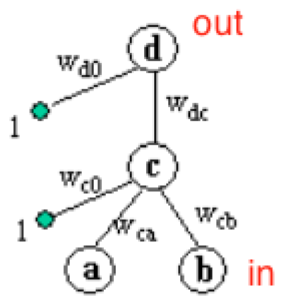
1 Weight updates

Consider a two-layer feedforward ANN with two inputs a and b one hidden unit c, and one output unit d. This network has five weights (w_{ca}, w_{cb}, w_{c0}, w_{dc}, w_{d0}), where w_{x0} represents the threshold weight for unit x. 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 $\eta = 0.3$, momentum $\alpha = 0.9$, incremental weight updates, and the following training examples:

a	b	d
1	0	1
0	1	0

Comments:

- When the question refers to the "threshold weight", interpret that as in this diagram (where the inputs are at the bottom and the output is at the top):



  You can treat the nodes in green as units with a fixed value of 1. Weights w_{c0} and w_{d0} are the threshold weights, AKA biases.

- Also assume the weight update for the 0th iteration (before the first pattern) is 0 for all weights. That means for the first iteration/pattern, the value of the momentum term is 0,

while for the second iteration/pattern, the momentum terms uses the weight update from the first iteration/pattern.
- Use the **sigmoid** activation / transfer function.

2 Changing Backprop

Revise the `Backpropagation` algorithm from Table 4.2 so that it operates on units using the squashing function $tanh$ in place of the sigmoid function. That is, assume the output of a single unit is $o = tanh(\vec{w} \cdot \vec{x})$. Give the weight update rule for output layer weights and hidden layer weights. Hint: $tanh'(x) = 1 - tanh^2(x)$.

Comments:

- This question is similar to Question 2 from HW1 in that it is asking you to change the neuron's activation function. But for this question, the activation function is $tanh$, and the algorithm is `Backpropagation` (BP), shown in Table 4.2 (p. 98).
- (Note: The non-linear BP algorithm is formulated differently from the algorithm for the linear unit in Table 4.1, so your description should be more like the one in Table 4.2, but with differences in how you propagate the error backward.)
- Before giving the new algorithm/steps, SHOW YOUR DERIVATION of the weight update rule for Δw_i. Then show how it comes into play in the algorithm with δ_i.
