Quiz: Neural Networks: Learning

Question 1:

1. You are training a three layer neural network and would like to use backpropagation to compute the gradient of the cost function. In the backpropagation algorithm, one of the steps is to update

$$\Delta_{ij}^{(2)} := \Delta_{ij}^{(2)} + \delta_i^{(3)} * (a^{(2)})_j$$

for every i, j. Which of the following is a correct vectorization of this step?

$$\bigcap \Delta^{(2)} := \Delta^{(2)} + \delta^{(2)} * (a^{(2)})^T$$

$$\bigcap \Delta^{(2)} := \Delta^{(2)} + (a^{(3)})^T * \delta^{(3)}$$

$$\bigcirc \hspace{0.15cm} \Delta^{(2)} := \Delta^{(2)} + \delta^{(3)} * (a^{(2)})^T$$

$$\Delta^{(2)} := \Delta^{(2)} + \delta^{(3)} * (a^{(3)})^T$$



✓ Correcto

This version is correct, as it takes the "outer product" of the two vectors $\delta^{(3)}$ and $a^{(2)}$ which is a matrix such that the (i,j)-th entry is $\delta_i^{(3)}*(a^{(2)})_j$ as desired.

Results: Correct

Question 2:

- 2. Suppose Theta1 is a 5x3 matrix, and Theta2 is a 4x6 matrix. You set thetaVec = [Theta1(:); Theta2(:)]. Which of the following correctly recovers Theta2?
 - reshape(thetaVec(16:39), 4, 6)
 - \bigcirc reshape(thetaVec(15:38),4,6)
 - reshape(thetaVec(16:24), 4, 6)
 - reshape(thetaVec(15:39), 4, 6)
 - O reshape(thetaVec(16:39), 6, 4)



This choice is correct, since Theta1 has 15 elements, so Theta2 begins at index 16 and ends at index 16 + 24

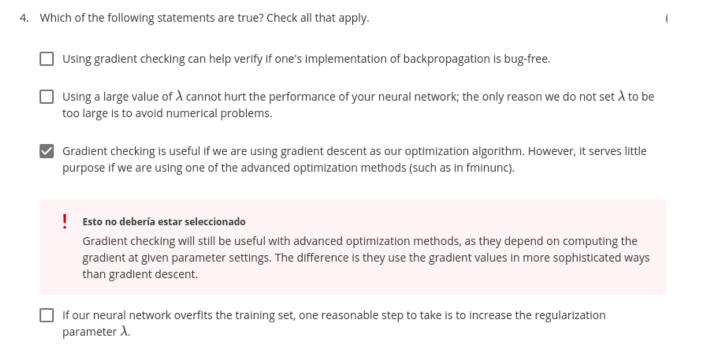
Results: Correct

Question 3:

3.	Let $J(\theta)=3\theta^4+4$. Let $\theta=1$, and $\epsilon=0.01$. Use the formula $\frac{J(\theta+\epsilon)-J(\theta-\epsilon)}{2\epsilon}$ to numerically compute an approximation to the derivative at $\theta=1$. What value do you get? (When $\theta=1$, the true/exact derivative is $\frac{dJ(\theta)}{d\theta}=12$.)
	O 12
	O 11.9988
	O 6
	12.0012
	\checkmark Correcto We compute $\frac{(3(1.01)^4+4)-(3(0.99)^4+4)}{2(0.01)}=12.0012.$

Results: Correct

Question 4



Results: Incorrect

Question 5

- 5. Which of the following statements are true? Check all that apply.
 - If we are training a neural network using gradient descent, one reasonable "debugging" step to make sure it is working is to plot $J(\Theta)$ as a function of the number of iterations, and make sure it is decreasing (or at least non-increasing) after each iteration.

✓ Correcto

Since gradient descent uses the gradient to take a step toward parameters with lower cost (ie, lower $J(\Theta)$), the value of $J(\Theta)$ should be equal or less at each iteration if the gradient computation is correct and the learning rate is set properly.

- If we initialize all the parameters of a neural network to ones instead of zeros, this will suffice for the purpose of "symmetry breaking" because the parameters are no longer symmetrically equal to zero.
- Suppose you have a three layer network with parameters $\Theta^{(1)}$ (controlling the function mapping from the inputs to the hidden units) and $\Theta^{(2)}$ (controlling the mapping from the hidden units to the outputs). If we set all the elements of $\Theta^{(1)}$ to be 0, and all the elements of $\Theta^{(2)}$ to be 1, then this suffices for symmetry breaking, since the neurons are no longer all computing the same function of the input.
- Suppose you are training a neural network using gradient descent. Depending on your random initialization, your algorithm may converge to different local optima (i.e., if you run the algorithm twice with different random initializations, gradient descent may converge to two different solutions).

✓ Correcto

The cost function for a neural network is non-convex, so it may have multiple minima. Which minimum you find with gradient descent depends on the initialization.

Results: Correct