**Parabéns! Você foi aprovado!**

**PARA SER APROVADO** 80% ou superior

Continue aprendendo

NOTA
80%

# Neural Networks: Learning

NOTA DO ENVIO MAIS RECENTE

80%

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

1 / 1 ponto

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

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

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

✓ **Correto**

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.

2. Suppose $\texttt{Theta1}$ is a 5x3 matrix, and $\texttt{Theta2}$ is a 4x6 matrix. You set $\texttt{thetaVec} = [\texttt{Theta1}(:); \texttt{Theta2}(:)]$. Which of the following correctly recovers $\texttt{Theta2}$?

1 / 1 ponto

- ◉ $\texttt{reshape}(\texttt{thetaVec}(16:39), 4, 6)$
- ○ $\texttt{reshape}(\texttt{thetaVec}(15:38), 4, 6)$
- ○ $\texttt{reshape}(\texttt{thetaVec}(16:24), 4, 6)$
- ○ $\texttt{reshape}(\texttt{thetaVec}(15:39), 4, 6)$
- ○ $\texttt{reshape}(\texttt{thetaVec}(16:39), 6, 4)$

✓ **Correto**

This choice is correct, since **Theta1** has 15 elements, so **Theta2** begins at index 16 and ends at index 16 + 24 - 1 = 39.

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 $\frac{dJ(\theta)}{d\theta} = 12$.)

1 / 1 ponto

- ○ 12
- ○ 11.9988
- ○ 6
- ◉ 12.0012

✓ **Correto**

We compute $\frac{(3(1.01)^4+4)-(3(0.99)^4+4)}{2(0.01)} = 12.0012.$

4. Which of the following statements are true? Check all that apply.

1 / 1 ponto

- [x] Using gradient checking can help verify if one's implementation of backpropagation is bug-free.

✓ **Correto**

If the gradient computed by backpropagation is the same as one computed numerically with gradient checking, this is very strong evidence that you have a correct implementation of backpropagation.

- [ ] Gradient checking is useful if we are using one of the advanced optimization methods (such as in fminunc) as our optimization algorithm. However, it serves little purpose if we are using gradient descent.
- [x] For computational efficiency, after we have performed gradient checking to

verify that our backpropagation code is correct, we usually disable gradient checking before using backpropagation to train the network.

**Correto**

Checking the gradient numerically is a debugging tool: it helps ensure a corre

ct implementation, but it is too slow to use as a method for actually computing gradients.

- [ ] Computing the gradient of the cost function in a neural network has the same efficiency when we use backpropagation or when we numerically compute it using the method of gradient checking.

5. Which of the following statements are true? Check all that apply. **0 / 1 ponto**

- [ ] 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.

- [x] 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.

**! Não deve ser selecionado**

The trouble with initializing the parameters to all zeros is not the specific value of zero but instead that every unit in the network will get the same update after backpropagation. Initializing the parameters to all ones has the same difficulty.

- [x] 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.

**Correto**

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.

- [ ] 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).