Week 5-1: Neural Networks

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: $\Delta_{ij}^{(2)} := \Delta_{ij}^{(2)} + \delta_i^{(3)} \times \left(a^{(2)}\right)_j$ for every i, j. Which of the following is a correct vectorization of this step?

   $\boxed{\Delta^{(2)} := \Delta^{(2)} + \delta^{(3)} \times \left(a^{(2)}\right)^T}$

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?

   Theta1 will have 15 elements and Theta2 will have 24 elements, therefore thetaVec will have $15 + 24 = 39$ elements. To recover Theta2, we will need the last 24 elements of the unrolled vector.

   ```
   reshape(thetaVec(16:39), 4, 6)
   ```

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

   Plug in $(\theta + \epsilon)$ for θ and then plug in θ and ϵ accordingly.

   $$\frac{J(\theta+\epsilon) - J(\theta-\epsilon)}{2\epsilon} \Rightarrow \frac{\left[3(\theta+\epsilon)^4 + 4\right] - \left[3(\theta-\epsilon)^4 + 4\right]}{2\epsilon} = \frac{3(1.01)^4 - 3(0.99)^4}{2\epsilon} = \boxed{12.0012}$$

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

   (a) Using gradient checking can help verify if one's implementation of backpropogation is bug-free.

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

   (a) Suppose we have a correct implementation of backpropogation and are training a neural network using gradient descent. Suppose we plot $J(\Theta)$ as a function of the number of iterations and find that it is **increasing** rather than decreasing. One possible cause of this is that the learning rate α is too large.

   (b) If we are training a neural network using gradient descent, one reasonable "debugging" step is 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.