Feedback — IX. Neural Networks: Learning

Help

You submitted this quiz on **Fri 22 Nov 2013 7:06 PM PST**. You got a score of **5.00** out of **5.00**.

Question 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?

Your Answer		Score	Explanation
$\Delta^{(2)} := \Delta^{(2)} + (a^{(3)})^T * \delta^{(3)}$			
$\Delta^{(2)} := \Delta^{(2)} + \delta^{(3)} * (a^{(2)})^T$	✔	1.00	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.
$\Delta^{(2)} := \Delta^{(2)} + \delta^{(3)} * (a^{(3)})^T$			
$\Delta^{(2)} := \Delta^{(2)} + \delta^{(2)} * (a^{(3)})^T$			
Total		1.00 / 1.00	

Question 2

Suppose `Theta1` is a 6x2 matrix, and `Theta2` is a 2x7 matrix. You set `thetaVec = [Theta1(:) ; Theta2(:)]`. Which of the following correctly recovers `Theta2`?

Your Answer		Score	Explanation
`reshape(thetaVec(11:24), 2, 7)`			
`reshape(thetaVec(13:26), 2, 7)`	✔	1.00	This choice is correct, since `Theta1` has 12 elements, so `Theta2` begins at index 13 and ends at index 13 + 14 - 1 = 26.
`reshape(thetaVec(12:25), 2, 7)`			
`reshape(thetaVec(12:26), 2, 7)`			
Total		1.00 / 1.00	

Question 3

Let $J(\theta) = 3\theta^3 + 2$. 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 derivati ve is $\frac{dJ(\theta)}{d\theta} = 9$.)

Your Answer		Score	Explanation
11			
8.9997			
9			
9.0003	✔	1.00	We compute $\frac{(3(1.01)^3+2)-(3(0.99)^3+2)}{2(0.01)} = 9.0003$.
Total		1.00 / 1.00	

Question 4

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

Your Answer		Score	Explanation
☑ 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.	✔	0.25	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.
☐ Gradient checking is useful if we are using gradient descent as our optimization algorithm. However, it serves little purpose if we are using one of the advanced optimization methods (such as in fminunc).	✔	0.25	Gradient checking will still be useful with advanced optimization methods, as they depend on computing the gradient at given parameter settings. The difference is they use the gradient values in more sophisticated ways than gradient descent.
☐ Using a large value of λ cannot hurt the performance of your neural network; the only reason we do not set λ to be too large is to avoid numerical problems.	✔	0.25	A large value of λ can be quite detrimental. If you set it too high, then the network will be underfit to the training data and give poor predictions on both training data and new, unseen test data.

☑ If our neural network overfits the training set, one reasonable step to take is to increase the regularization parameter λ.	✔	0.25	Just as with logistic regression, a large value of λ will penalize large parameter values, thereby reducing the changes of overfitting the training set.
Total		1.00 / 1.00	

Question 5

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

Your Answer		Score	Explanation
☑ 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).	✔	0.25	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.
☐ Suppose we are using gradient descent with learning rate α. For logistic regression and linear regression, $J(\theta)$ was a convex optimization problem and thus we did not want to choose a learning rate α that is too large. For a neural network however, $J(\Theta)$	✔	0.25	Even when $J(\Theta)$ is not convex, a learning rate that is too large can prevent gradient descent from converging.

may not be convex, and thus choosing a very large value of α can only speed up convergence.			
☐ Suppose that the parameter $\Theta^{(1)}$ is a square matrix (meaning the number of rows equals the number of columns). If we replace $\Theta^{(1)}$ with its transpose $(\Theta^{(1)})^T$, then we have not changed the function that the network is computing.	✔	0.25	$\Theta^{(1)}$ can be an arbitrary matrix, so when you compute $a^{(2)} = g(\Theta^{(1)} a^{(1)})$, replacing $\Theta^{(1)}$ with its transpose will compute a different value.
☑ Suppose we have a correct implementation of backpropagation, 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.	✔	0.25	If the learning rate is too large, the cost function can diverge during gradient descent. Thus, you should select a smaller value of α.
Total		1.00 / 1.00	