Congratulations! You passed!

Grade received 80%

Latest Submission Grade 80%

To pass 80% or higher

Go to next item

1. What is stored in the 'cache' during forward propagation for latter use in backward propagation? **1 / 1 point**

- $b^{[l]}$
- $Z^{[l]}$
- $A^{[l]}$
- $W^{[l]}$

Expand

Correct
Yes. This value is useful in the calculation of $dW^{[l]}$ in the backward propagation.

2. Which of the following are “parameters” of a neural network? (Check all that apply.) **0 / 1 point**

- [x] $b^{[l]}$ the bias vector.

Correct
Correct. The weight matrices and the bias vectors are the parameters of the network.

- [x] $g^{[l]}$ the activation functions.

This should not be selected
Incorrect. This is a hyperparameter. The weight matrices and the bias vectors are the parameters of the network.

- [ ] L the number of layers of the neural network.
- [x] $W^{[l]}$ the weight matrices.

Correct
Correct. The weight matrices and the bias vectors are the parameters of the network.

Expand

Incorrect

Incorrect

You chose the extra incorrect answers.

3. Considering the intermediate results below, which layers of a deep neural network are they likely to belong to? **0 / 1 point**

- Later layers of the deep neural network.
- Early layers of the deep neural network.
- Middle layers of the deep neural network.
- Input layer of the deep neural network.

Expand

Incorrect

Incorrect. The deep layers of a neural network are typically computing more complex features such as the ones shown in the figure.

4. Vectorization allows us to compute $a^{[l]}$ for all the examples on a batch at the same time without using a for loop. True/False? **1 / 1 point**

- True
- False

Expand

Correct

Correct. Vectorization allows us to compute the activation for all the training examples at the same time, avoiding the use of a for loop.

5. Assume we store the values for $n^{[l]}$ in an array called layer_dims, as follows: layer_dims = [n_x, 4,3,2,1]. So layer 1 has four hidden units, layer 2 has 3 hidden units and so on. Which of the following for-loops will allow you to initialize the parameters for the model?

1 / 1 point

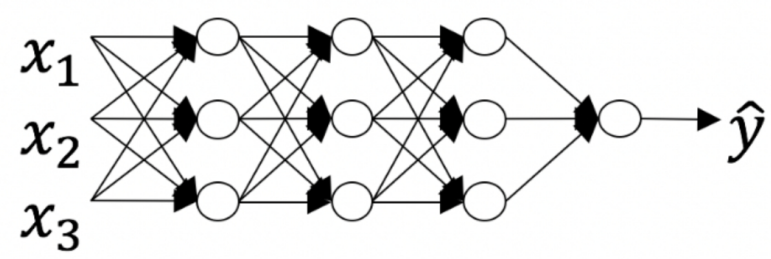
- (•) for i in range(1, len(layer_dims)):
  parameter['W' + str(i)] = np.random.randn(layer_dims[i], layer_dims[i-1]) * 0.01
  parameter['b' + str(i)] = np.random.randn(layer_dims[i], 1) * 0.01
- ( ) for i in range(1, len(layer_dims)):
  parameter['W' + str(i)] = np.random.randn(layer_dims[i-1], layer_dims[i]) * 0.01
  parameter['b' + str(i)] = np.random.randn(layer_dims[i], 1) * 0.01
- ( ) for i in range(1, len(layer_dims)/2):
  parameter['W' + str(i)] = np.random.randn(layer_dims[i], layer_dims[i-1]) * 0.01
  parameter['b' + str(i)] = np.random.randn(layer_dims[i], 1) * 0.01
- ( ) for i in range(1, len(layer_dims)/2):
  parameter['W' + str(i)] = np.random.randn(layer_dims[i], layer_dims[i-1]) * 0.01
  parameter['b' + str(i)] = np.random.randn(layer_dims[i-1], 1) * 0.01

Expand

✓ Correct

6. Consider the following neural network.

1 / 1 point



How many layers does this network have?

- ( ) The number of layers L is 5. The number of hidden layers is 4.
- ( ) The number of layers L is 3. The number of hidden layers is 3.
- (•) The number of layers L is 4. The number of hidden layers is 3.
- ( ) The number of layers L is 4. The number of hidden layers is 4.

Expand

Correct
Yes. As seen in lecture, the number of layers is counted as the number of hidden layers + 1. The input and output layers are not counted as hidden layers.

7. During forward propagation, in the forward function for a layer l you need to know what is the activation function in a layer (sigmoid, tanh, ReLU, etc.). During backpropagation, the corresponding backward function also needs to know what is the activation function for layer l, since the gradient depends on it. True/False? **1 / 1 point**

- False
- True

Expand

Correct
Yes, as you've seen in week 3 each activation has a different derivative. Thus, during backpropagation you need to know which activation was used in the forward propagation to be able to compute the correct derivative.

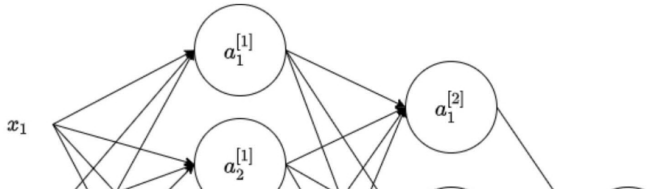
8. For any mathematical function you can compute with an L-layered deep neural network with N hidden units there is a shallow neural network that requires only $\log N$ units, but it is very difficult to train. **1 / 1 point**

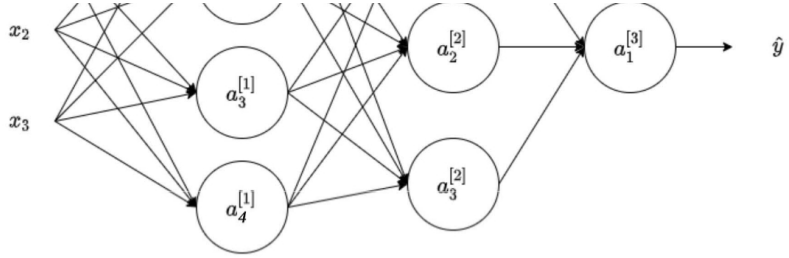
- False
- True

Expand

Correct
Correct. On the contrary, some mathematical functions can be computed using an L-layered neural network and a given number of hidden units; but using a shallow neural network the number of necessary hidden units grows exponentially.

9. Consider the following 2 hidden layers neural network: **1 / 1 point**





Which of the following statements is true? (Check all that apply).

- [x] $b^{[1]}$ will have shape (4, 1)

  Correct
  Yes. More generally, the shape of $b^{[l]}$ is $(n^{[l]}, 1)$.

- [ ] $b^{[1]}$ will have shape (3, 1)
- [x] $W^{[1]}$ will have shape (4, 3)

  Correct
  Yes. More generally, the shape of $W^{[l]}$ is $(n^{[l]}, n^{[l-1]})$.

- [ ] $b^{[1]}$ will have shape (1, 4)
- [ ] $W^{[2]}$ will have shape (1, 3)
- [ ] $W^{[2]}$ will have shape (3, 1)
- [ ] $W^{[2]}$ will have shape (4, 3)
- [ ] $W^{[1]}$ will have shape (3, 4)
- [x] $W^{[2]}$ will have shape (3, 4)

  Correct
  Yes. More generally, the shape of $W^{[l]}$ is $(n^{[l]}, n^{[l-1]})$.

Expand

Correct
Great, you got all the right answers.

10. Whereas the previous question used a specific network, in the general case what is the dimension of W^{[l]}, the weight matrix associated with layer l? **1 / 1 point**

- (•) $W^{[l]}$ has shape $(n^{[l]}, n^{[l-1]})$
- ( ) $W^{[l]}$ has shape $(n^{[l+1]}, n^{[l]})$

- $W^{[l]}$ has shape $(n^{[x]}, n^{[x-1]})$
- $W^{[l]}$ has shape $(n^{[l-1]}, n^{[l]})$

Expand

Correct
True