Congratulations! You passed!

Grade received 100% | Latest Submission Grade 100% | To pass 80% or higher

Retake the assignment in **23h 43m**

Go to next item

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

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

Expand

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

2. Among the following, which ones are "hyperparameters"? (Check all that apply.) **1 / 1 point**

- activation values $a^{[l]}$
- number of iterations (selected)

Correct

- weight matrices $W^{[l]}$
- size of the hidden layers $n^{[l]}$ (selected)

Correct

- bias vectors $b^{[l]}$
- learning rate α (selected)

Correct

- number of layers L in the neural network (selected)

Correct

Expand

Expand

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

2. Among the following, which ones are "hyperparameters"? (Check all that apply.) **1 / 1 point**

- activation values $a^{[l]}$
- number of iterations (selected)

Correct

- weight matrices $W^{[l]}$
- size of the hidden layers $n^{[l]}$ (selected)

Correct

- bias vectors $b^{[l]}$
- learning rate α (selected)

- The earlier layers of a neural network are typically computing more complex features of the input than the deeper layers.
- The deeper layers of a neural network are typically computing more complex features of the input than the earlier layers. (selected)

Expand

Correct

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

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

Expand

Correct
Yes. This iterates over 0, 1, 2, 3 and assigns to $W^{[l]}$ the shape $(n^{[l]}, n^{[l-1]})$.

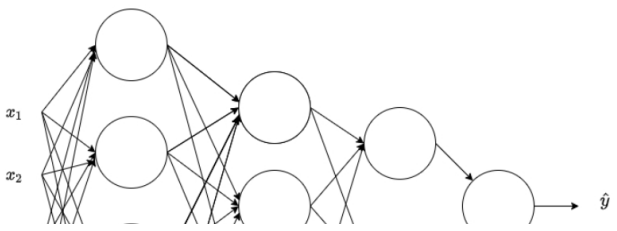
- ( ) for i in range(len(layer_dims)-1):
  parameter['W' + str(i+1)] = np.random.randn(layer_dims[i], layer_dims[i+1]) * 0.01
  parameter['b' + str(i+1)] = np.random.randn(layer_dims[i+1], 1) * 0.01
- ( ) for i in range(len(layer_dims)):
  parameter['W' + str(i+1)] = np.random.randn(layer_dims[i+1],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

Expand

Correct
Yes. This iterates over 0, 1, 2, 3 and assigns to $W^{[l]}$ the shape $(n^{[l]}, n^{[l-1]})$.

Correct
Yes. This iterates over 0, 1, 2, 3 and assigns to $W^{[l]}$ the shape $(n^{[l]}, n^{[l-1]})$.

6. Consider the following neural network: 1 / 1 point



Expand

Correct

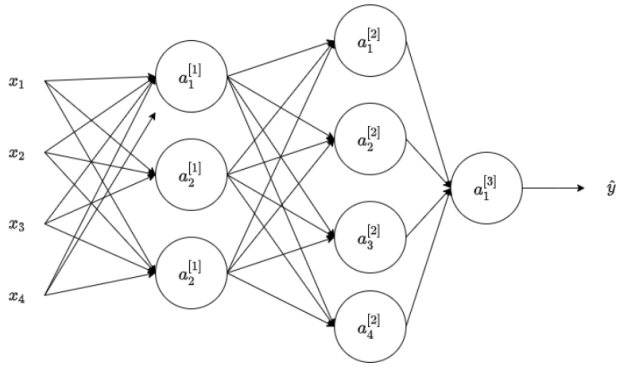
8. A shallow neural network with a single hidden layer and 6 hidden units can compute any function that a neural network with 2 hidden layers and 6 hidden units can compute. True/False? **1 / 1 point**

- ○ True
- ◉ False

Expand

Correct
Correct. As seen during the lectures there are functions you can compute with a "small" L-layer deep neural network that shallower networks require exponentially more hidden units to compute.

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



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

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

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

- ☑ $W^{[2]}$ will have shape (4, 3)

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

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

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

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

Expand

Correct
True