Congratulations! You passed!

Grade received 90% | Latest Submission Grade 90% | 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**

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

Expand

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

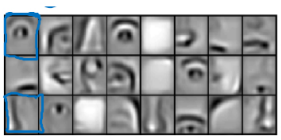
2. Among the following, which ones are "hyperparameters"? (Check all that apply.) **1 / 1 point**

- [x] number of iterations
  - Correct
- [x] learning rate α
  - Correct
- [ ] weight matrices $W^{[l]}$
- [ ] bias vectors $b^{[l]}$
- [x] size of the hidden layers $n^{[l]}$
  - Correct
- [x] number of layers L in the neural network
  - Correct
- [ ] activation values $a^{[l]}$

Expand

Correct
Great, you got all the right answers.

3. Which of the following is more likely related to the early layers of a deep neural network? **1 / 1 point**



Expand

Correct
Yes. The early layer of a neural network usually computes simple features such as edges and lines.

4. We can not use vectorization to calculate $da^{[l]}$ in backpropagation, we must use a for loop over all the examples. True/False? **1 / 1 point**

- True
- False (selected)

Expand

Correct
Correct. We can use vectorization in backpropagation to calculate $dA^{[l]}$ for each layer. This computation is done over all the training examples.

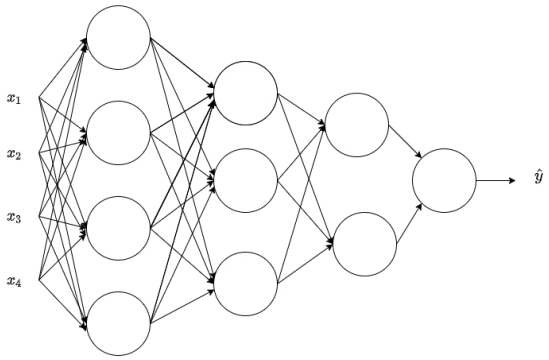
5. Assume we store the values for $n^{[l]}$ in an array called layer_dims, as follows: layer_dims = [n_x,4,3,2,1]. So layer 1 has four hidden units, layer 2 has 3 hidden units, and so on. Which of the following for-loops will allow you to initialize the parameters for the model? **1 / 1 point**

- for i in range(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
- **(selected)** 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
- 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)):
  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 $(n^{[l]}, n^{[l-1]})$.

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



What are all the values of $n^{[0]}, n^{[1]}, n^{[2]}, n^{[3]}$ and $n^{[4]}$?

- 4, 3, 2
- 4, 4, 3, 2
- 4, 3, 2, 1
- **(selected)** 4, 4, 3, 2, 1

Expand

Correct
Yes. The $n^{[l]}$ are the number of units in each layer, notice that $n^{[0]} = n_x$.

7. If L is the number of layers of a neural network then $dZ^{[L]} = A^{[L]} - Y$. True/False? **0 / 1 point**

- True
  Yes. The gradient of the output layer depends on the difference between the value computed during the forward propagation process and the target values.
- **(selected)** False
  No. The gradient of the output layer depends on the difference between the value computed during the forward propagation process and the target values.

Expand

Incorrect

8. There are certain functions with the following properties: **1 / 1 point**

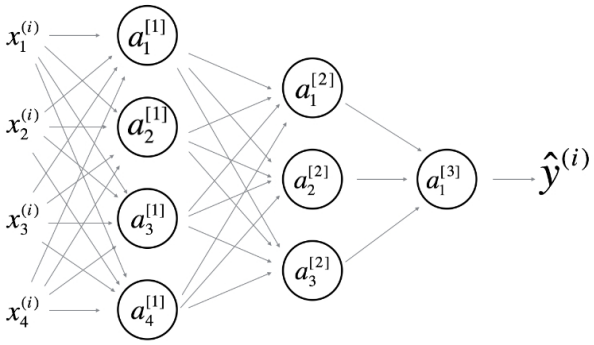
(i) To compute the function using a shallow network circuit, you will need a large network (where we measure size by the number of logic gates in the network), but (ii) To compute it using a deep network circuit, you need only an exponentially smaller network. True/False?

- **(selected)** True
- False

Expand

Correct

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



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

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

  Correct
  Yes. More generally, the shape of $b^{[l]}$ is $(n^{[l]}, 1)$.
- [x] $b^{[1]}$ will have shape (4, 1)

  Correct
  Yes. More generally, the shape of $b^{[l]}$ is $(n^{[l]}, 1)$.
- [x] $b^{[3]}$ will have shape (1, 1)

  Correct
  Yes. More generally, the shape of $b^{[l]}$ is $(n^{[l]}, 1)$.
- [x] $W^{[2]}$ will have shape (3, 4)

  Correct
  Yes. More generally, the shape of $W^{[l]}$ is $(n^{[l]}, n^{[l-1]})$.
- [ ] $W^{[1]}$ will have shape (3, 4)
- [x] $W^{[3]}$ will have shape (1, 3)

  Correct
  Yes. More generally, the shape of $W^{[l]}$ is $(n^{[l]}, n^{[l-1]})$.
- [ ] $b^{[2]}$ will have shape (1, 1)
- [ ] $W^{[2]}$ will have shape (3, 1)
- [ ] $b^{[1]}$ will have shape (3, 1)
- [x] $W^{[1]}$ will have shape (4, 4)

  Correct
  Yes. More generally, the shape of $W^{[l]}$ is $(n^{[l]}, n^{[l-1]})$.
- [ ] $W^{[3]}$ will have shape (3, 1)

Expand

Correct
Great, you got all the right answers.

10. In the general case if we are training with m examples what is the shape of $A^{[l]}$? 1 / 1 point

- ○ $(m, n^{[l+1]})$
- ◉ $(n^{[l]}, m)$
- ○ $(m, n^{[l]})$
- ○ $(n^{[l+1]}, m)$

Expand

Correct
Yes. The number of rows in $A^{[l]}$ corresponds to the number of units in the l-th layer.