Key Concepts on Deep Neural Networks

Congratulations! You passed!

Grade received 100%

Latest Submission Grade 100%

To pass 80% or higher

Go to next item

1. We use the "cache" in our implementation of forward and backward propagation to pass useful values to the next layer in the forward propagation. True/False?

1 / 1 point

- (•) False
- ( ) True

Expand

Correct

Correct. The "cache" is used in our implementation to store values computed during forward propagation to be used in backward propagation.

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

- [x] number of layers L in the neural network

  ✓ **Correct**

- [ ] weight matrices $W^{[l]}$
- [x] learning rate α

  ✓ **Correct**

- [ ] bias vectors $b^{[l]}$
- [x] number of iterations

  ✓ **Correct**

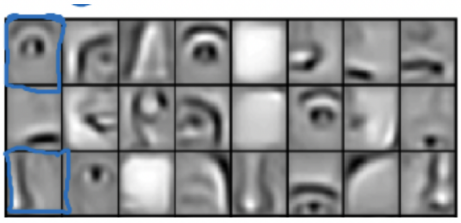
- [ ] activation values $a^{[l]}$
- [x] size of the hidden layers $n^{[l]}$

  ✓ **Correct**

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. Vectorization allows you to compute forward propagation in an L-layer neural network without an explicit for-loop (or any other explicit iterative loop) over the layers l=1, 2, …,L. True/False? **1 / 1 point**

- True
- False

Expand

Correct
Forward propagation propagates the input through the layers, although for shallow networks we may just write all the lines ($a^{[2]} = g^{[2]}(z^{[2]}), z^{[2]} = W^{[2]}a^{[1]} + b^{[2]}$, ...) in a deeper network, we cannot avoid a for loop iterating over the layers: ($a^{[l]} = g^{[l]}(z^{[l]}), z^{[l]} = W^{[l]}a^{[l-1]} + b^{[l]}$, ...).

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

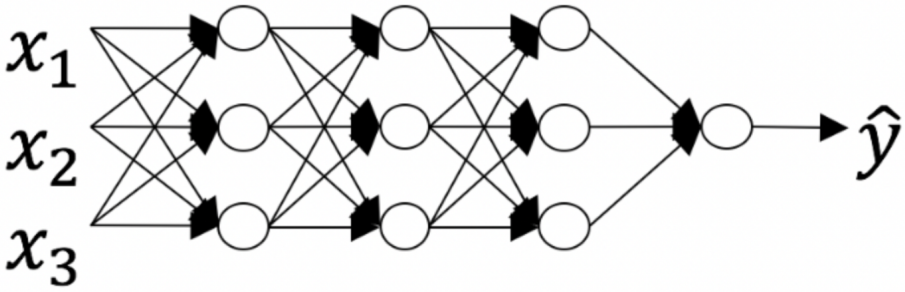
- for i in range(1, len(layer_dims)/2):
  parameter['W' + str(i)] = np.random.randn(layer_dims[i], layer_dims[i-1]) * 0.01
  parameter['b' + str(i)] = np.random.randn(layer_dims[i], 1) * 0.01
- **(selected)** for i in range(1, len(layer_dims)):
  parameter['W' + str(i)] = np.random.randn(layer_dims[i], 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], 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 4. 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 5. The number of hidden layers is 4.
- **(selected)** The number of layers L is 4. The number of hidden layers is 3.

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.

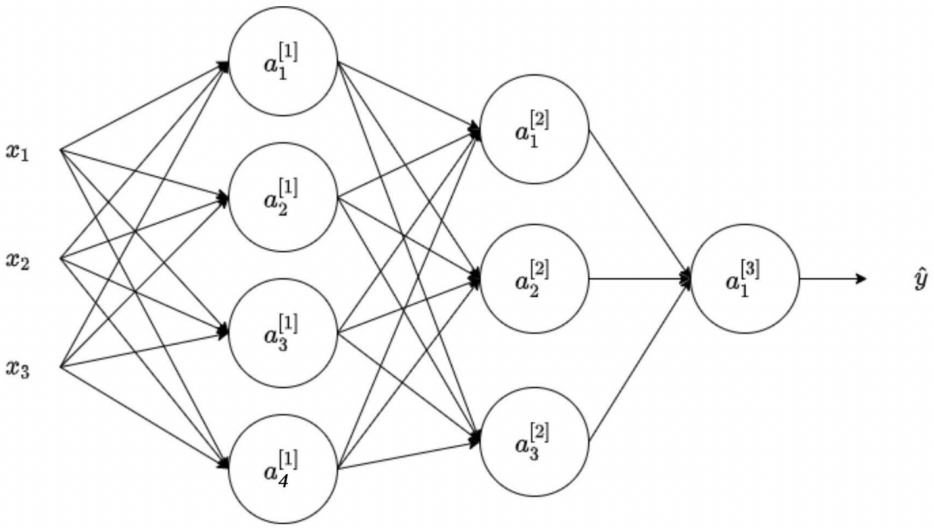
8. A shallow neural network with a single hidden layer and 6 hidden units can compute any function that a neural network with 2 hidden layers and 6 hidden units can compute. True/False? **1 / 1 point**

- ◉ False
- ○ True

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 is true? (Check all that apply).

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

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

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

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 $b^{[l]}$, the bias vector associated with layer l? **1 / 1 point**

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

Expand

Correct
True. $b^{[l]}$ is a column vector with the same number of rows as units in the respective layer.