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

- True
- False

Expand

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

2. During the backpropagation process, we use gradient descent to change the hyperparameters. True/False?

1 / 1 point

- True
- False

Expand

Correct
Correct. During backpropagation, we use gradient descent to compute new values of $$W^{[l]}$$ and $$b^{[l]}$$. These are the parameters of the network.

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

1 / 1 point

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

Expand

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

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
- [x] 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 = [$n_x n$,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?

0 / 1 point

- [x] 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
- [ ] 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

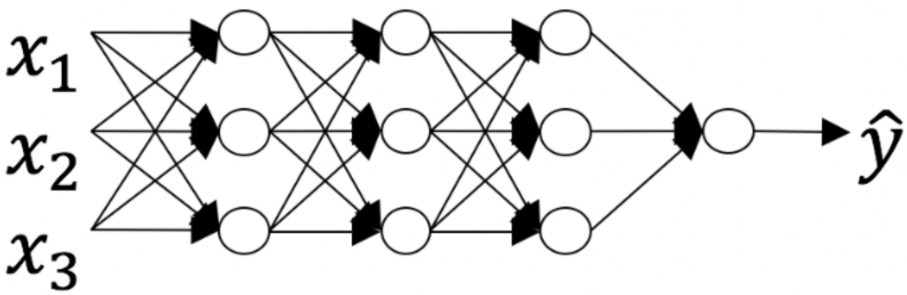
Expand

Incorrect

No. This exceeds the number of layers on the neural network.

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 3.
- 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.

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

1 / 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.
- 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

Correct

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?

- True

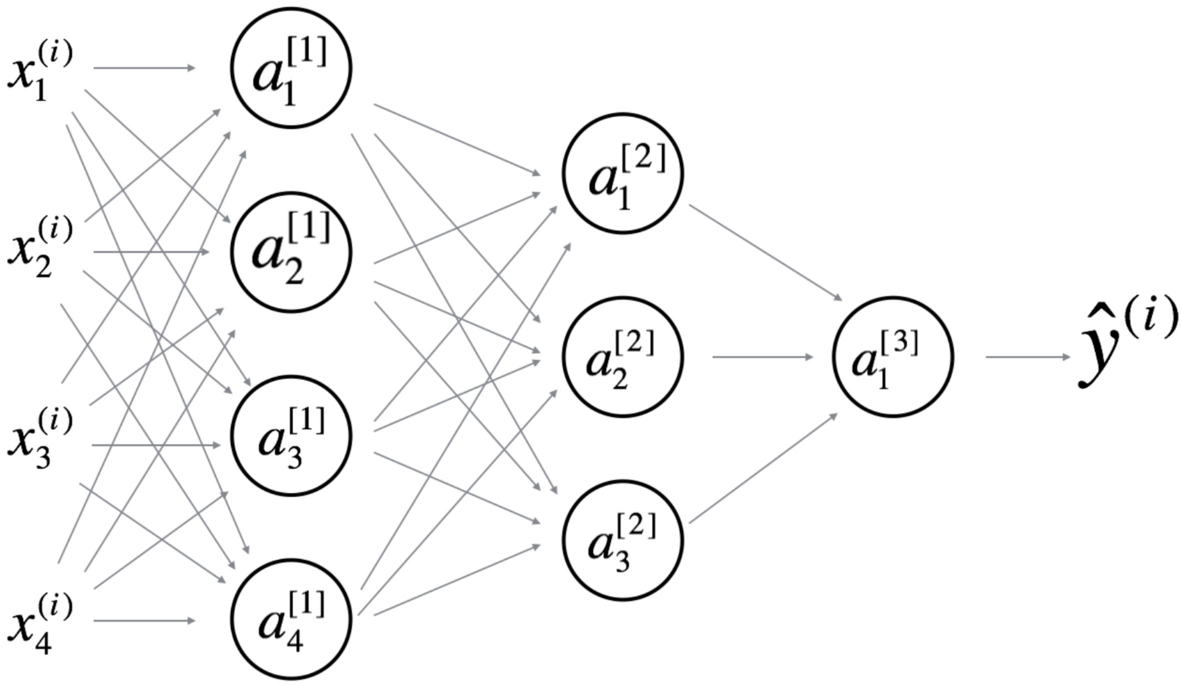
- [ ] False

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).

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

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

- [ ] $W^{[1]}$

  $W^{[1]}$ will have shape ([illegible], 4)

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

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

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

  Correct

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

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

✓ **Correct**

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

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

- [ ] $$b^{[3]}$$ 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] $$b^{[3]}$$ will have shape (1, 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 $b^{[l]}$, the bias vector associated with layer l?

0 / 1 point

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

Expand

Incorrect

No. $$b^{[l]}$$ is a column vector.