← Back **Shallow Neural Networks**
Graded Quiz • 50 min

**Due** Aug 21, 11:59 PM +03

## Congratulations! You passed!

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

Go to next item

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

**1 / 1 point**

- [x] $w_3^{[4]}$ is the column vector of parameters of the fourth layer and third neuron.

✓ **Correct**

Yes. The vector $w_j^{[i]}$ is the column vector of parameters of the i-th layer and j-th neuron of that layer.

- [ ] $W_1$ is a matrix with rows equal to the parameter vectors of the first layer.
- [ ] $w_3^{[4]}$ is the column vector of parameters of the third layer and fourth neuron.
- [x] $W^{[1]}$ is a matrix with rows equal to the transpose of the parameter vectors of the first layer.

✓ **Correct**

Yes. We construct $W^{[1]}$ stacking the parameter vectors $w_j^{[1]}$ of all the neurons of the first layer.

- [ ] $w_3^{[4]}$ is the row vector of parameters of the fourth layer and third neuron.
- [ ] $W^{[1]}$ is a matrix with rows equal to the parameter vectors of the first layer.

Expand

**Correct**
Great, you got all the right answers.

**2.** The tanh activation is not always better than sigmoid activation function for hidden units because the mean of its output is closer to zero, and so it centers the data, making learning complex for the next layer. True/False?

**1 / 1 point**

- False
- True

Expand

Correct

Yes. As seen in lecture the output of the tanh is between -1 and 1, it thus centers the data which makes the learning simpler for the next layer.

**3.** Which of these is a correct vectorized implementation of forward propagation for layer $l$, where $1 \leq l \leq L$?

1 / 1 point

- $Z^{[l]} = W^{[l]}A^{[l]} + b^{[l]}$
  $A^{[l+1]} = g^{[l]}(Z^{[l]})$
- $Z^{[l]} = W^{[l]}A^{[l-1]} + b^{[l]}$
  $A^{[l]} = g^{[l]}(Z^{[l]})$
- $Z^{[l]} = W^{[l-1]}A^{[l]} + b^{[l-1]}$
  $A^{[l]} = g^{[l]}(Z^{[l]})$
- $Z^{[l]} = W^{[l]}A^{[l]} + b^{[l]}$
  $A^{[l+1]} = g^{[l+1]}(Z^{[l]})$

Expand

Correct

**4.** The use of the ReLU activation function is becoming more rare because the ReLU function has no derivative for $c = 0$. True/False?

1 / 1 point

- False
- True

Expand

Correct
Yes. Although the ReLU function has no derivative at $c = 0$ this rarely causes any problems in practice. Moreover it has become the default activation function in many cases, as explained in the lectures.

**5.** Consider the following code:

1 / 1 point

#+begin_src python

x = np.random.rand(4, 5)

y = np.sum(x, axis=1)

#+end_src

What will be y.shape?

- (4, 1)
- (1, 5)
- (4, )
- (5, )

Expand

Correct
Yes. By using axis=1 the sum is computed over each row of the array, thus the resulting array is a column vector with 4 entries. Since the option keepdims was not used the array doesn't keep the second dimension.

**6.** Suppose you have built a neural network with one hidden layer and tanh as activation function for the hidden layers. Which of the following is a best option to initialize the weights?

1 / 1 point

- Initialize the weights to large random numbers.
- Initialize the weights to small random numbers.
- Initialize all weights to a single number chosen randomly.

- [ ] Initialize all weights to 0.

Expand

**Correct**
The use of random numbers helps to "break the symmetry" between all the neurons allowing them to compute different functions. When using small random numbers the values $z^{[k]}$ will be close to zero thus the activation values will have a larger gradient speeding up the training process.

**7.** Using linear activation functions in the hidden layers of a multilayer neural network is equivalent to using a single layer. True/False?

**1 / 1 point**

- [x] True
- [ ] False

Expand

**Correct**
Yes. When the identity or linear activation function $g(c) = c$ is used the output of composition of layers is equivalent to the computations made by a single layer.

**8.** You have built a network using the tanh activation for all the hidden units. You initialize the weights to relatively large values, using np.random.randn(..,..)*1000. What will happen?

**1 / 1 point**

- [ ] This will cause the inputs of the tanh to also be very large, thus causing gradients to also become large. You therefore have to set $\alpha$ to a very small value to prevent divergence; this will slow down learning.
- [ ] This will cause the inputs of the tanh to also be very large, causing the units to be “highly activated” and thus speed up learning compared to if the weights had to start from small values.
- [ ] So long as you initialize the weights randomly gradient descent is not affected by whether the weights are large or small.
- [x] This will cause the inputs of the tanh to also be very large, thus causing gradients to be close to zero. The optimization algorithm will thus become slow.

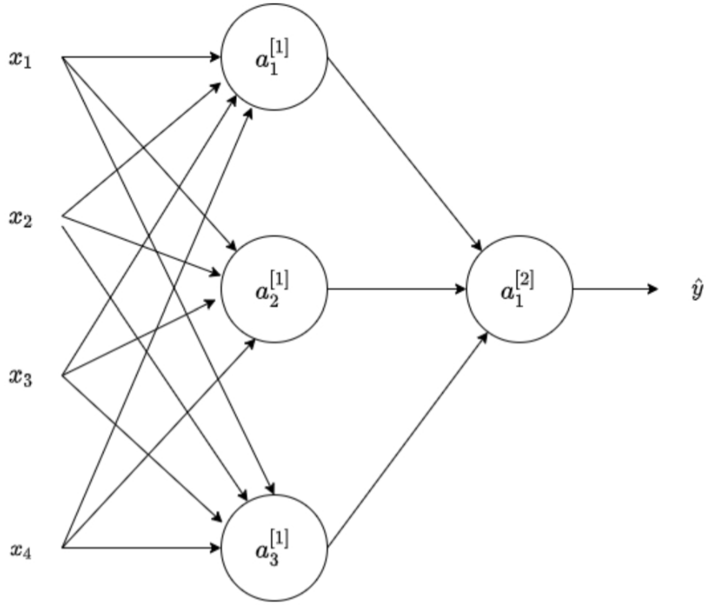
Expand

Correct

Yes. tanh becomes flat for large values; this leads its gradient to be close to zero. This slows down the optimization algorithm.

**9.** Consider the following 1 hidden layer neural network:

1 / 1 point




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

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

Correct

Yes. $b^{[k]}$ is a column vector and has the same number of rows as neurons in the k-th layer.

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

Correct

Yes. $b^{[k]}$ is a column vector and has the same number of rows as neurons in the k-th layer.

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

✓ **Correct**

Yes. The number of rows in $W^{[k]}$ is the number of neurons in the k-th layer and the number of columns is the number of inputs of the layer.

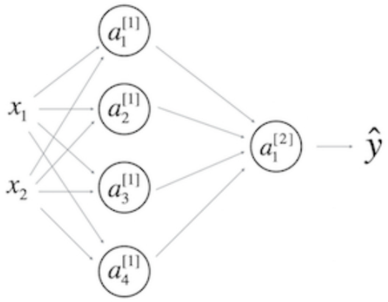
Expand

**Correct**

Great, you got all the right answers.

**10.** What are the dimensions of $Z^{[1]}$ and $A^{[1]}$?

**1 / 1 point**




- ○ $Z^{[1]}$ and $A^{[1]}$ are (4,2)
- ○ $Z^{[1]}$ and $A^{[1]}$ are (1,4)
- ◉ $Z^{[1]}$ and $A^{[1]}$ are (4,m)
- ○ $Z^{[1]}$ and $A^{[1]}$ are (4,1)

Expand

Correct