

week1-Q1.

What does the analogy “AI is the new electricity” refer to?

1. Through the “smart grid”, AI is delivering a new wave of electricity.
2. AI runs on computers and is thus powered by electricity, but it is letting computers do things not possible before.
3. Similar to electricity starting about 100 years ago, AI is transforming multiple industries.
4. AI is powering personal devices in our homes and offices, similar to electricity.

Answer: 3. AI is transforming many fields from the car industry to agriculture to supply-chain.

week1-Q2.

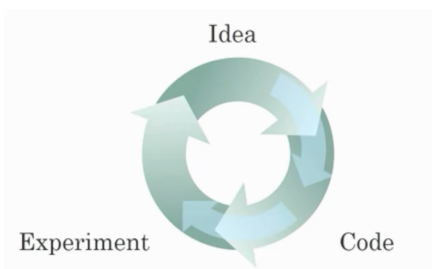
Which of these are reasons for Deep Learning recently taking off? (Check the three options that apply.)

1. Deep learning has resulted in significant improvements in important applications such as online advertising, speech recognition, and image recognition.
2. Neural Networks are a brand new field.
3. We have access to a lot more data.
4. We have access to a lot more computational power.

Answer: 1,3,4. The digitalization of our society has played a huge role in this. The development of hardware, perhaps especially GPU computing, has significantly improved deep learning algorithms' performance.

week1-Q3.

Recall this diagram of iterating over different ML ideas. Which of the statements below are true? (Check all that apply.)



1. Being able to try out ideas quickly allows deep learning engineers to iterate more quickly.
2. Faster computation can help speed up how long a team takes to iterate to a good idea.
3. It is faster to train on a big dataset than a small dataset.
4. Recent progress in deep learning algorithms has allowed us to train good models faster (even without changing the CPU/GPU hardware).

Answer: 1,2,4. For example, we discussed how switching from sigmoid to ReLU activation functions allows faster training.

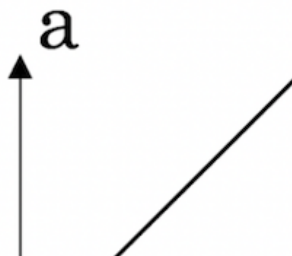
week1-Q4.

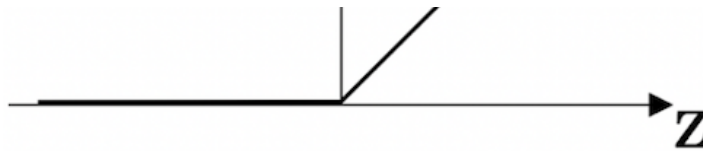
When an experienced deep learning engineer works on a new problem, they can usually use insight from previous problems to train a good model on the first try, without needing to iterate multiple times through different models. True/False?

Answer: False. Finding the characteristics of a model is key to have good performance. Although experience can help, it requires multiple iterations to build a good model.

week1-Q5.

ReLU activation function?





week1-Q6.

Images for cat recognition is an example of “structured” data, because it is represented as a structured array in a computer. True/False?

Answer: False. Images for cat recognition is an example of “unstructured” data.

week1-Q7.

A demographic dataset with statistics on different cities' population, GDP per capita, economic growth is an example of “unstructured” data because it contains data coming from different sources. True/False?

Answer: False. A demographic dataset with statistics on different cities' population, GDP per capita, economic growth is an example of “structured” data by opposition to image, audio or text datasets.

week1-Q8.

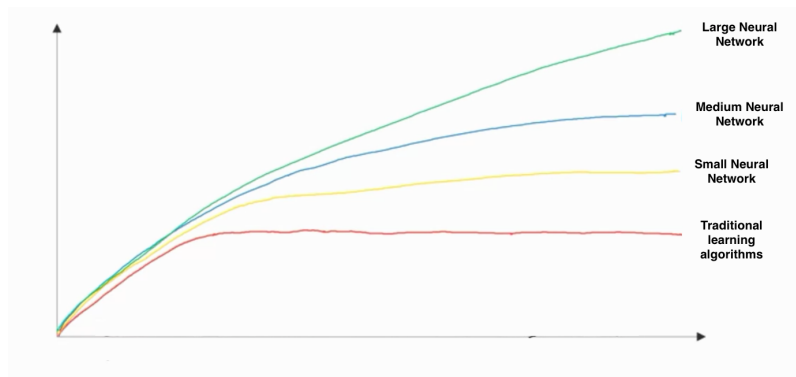
Why is an RNN (Recurrent Neural Network) used for machine translation, say translating English to French? (Check all that apply.)

1. It can be trained as a supervised learning problem.
2. It is strictly more powerful than a Convolutional Neural Network (CNN).
3. It is applicable when the input/output is a sequence (e.g., a sequence of words).
4. RNNs represent the recurrent process of Idea->Code->Experiment->Idea->....

Answer: 1,3. An RNN can map from a sequence of english words to a sequence of french words.

week1-Q9.

In this diagram which we hand-drew in lecture, what do the horizontal axis (x-axis) and vertical axis (y-axis) represent?



Answer: x-axis is the amount of data. y-axis (vertical axis) is the performance of the algorithm.

week1-Q10.

Assuming the trends described in the previous question's figure are accurate (and hoping you got the axis labels right), which of the following are true? (Check all that apply.)

1. Decreasing the training set size generally does not hurt an algorithm's performance, and it may help significantly.
2. Increasing the size of a neural network generally does not hurt an algorithm's performance, and it may help significantly.
3. Increasing the training set size generally does not hurt an algorithm's performance, and it may help significantly.
4. Decreasing the size of a neural network generally does not hurt an algorithm's performance, and it may help significantly.

Answer: 2,3. According to the trends in the figure above, big networks usually perform better than small networks. Bringing more data to a model is almost always beneficial.

week2-Q3.

Suppose img is a (100, 100, 3) array representing a 100x100 image with 3 color channels: red, green and blue. How do you reshape this into a

Suppose `img` is a `(32,32,3)` array, representing a `32x32` image with 3 color channels red, green and blue. How do you reshape this into a column vector?

Answer: `x = img.reshape((32*32*3,1))`.

week2-Q4.

Consider the two following random arrays "a" and "b". What will be the shape of "c"?

```
In [1]: import numpy as np
a = np.random.randn(2, 3) # a.shape = (2, 3)
b = np.random.randn(2, 1) # b.shape = (2, 1)
c = a + b
print(c.shape)

(2, 3)
```

Answer: This is broadcasting. `b` (column vector) is copied 3 times so that it can be summed to each column of `a`.

week2-Q5.

Consider the two following random arrays "a" and "b". What will be the shape of "c"?

```
In [2]: a = np.random.randn(4, 3) # a.shape = (4, 3)
b = np.random.randn(3, 2) # b.shape = (3, 2)

# operands could not be broadcast together with shapes (4,3) (3,2)
# print((a*b).shape)

print((np.dot(a,b)).shape)

(4, 2)
```

Answer: The computation cannot happen because the sizes don't match. It's going to be "Error"! In numpy the `"**"` operator indicates element-wise multiplication. It is different from `"np.dot()"`. If you would try `"c = np.dot(a,b)"` you would get `c.shape = (4, 2)`.

week2-Q7.

Recall that `"np.dot(a,b)"` performs a matrix multiplication on `a` and `b`, whereas `"a*b"` performs an element-wise multiplication.

Consider the two following random arrays "a" and "b":

```
In [3]: a = np.random.randn(12288, 150) # a.shape = (12288, 150)
b = np.random.randn(150, 45) # b.shape = (150, 45)
c = np.dot(a,b)
print(c.shape)

(12288, 45)
```

Answer: Remember that a `np.dot(a, b)` has shape (number of rows of `a`, number of columns of `b`). The sizes match because : "number of columns of `a` = 150 = number of rows of `b`"

week2-Q8.

Consider the following code snippet. How do you vectorize this?

```
In [6]: a = np.random.randn(3,4)
b = np.random.randn(4,1)

for i in range(3):
    for j in range(4):
        c[i][j] = a[i][j] + b[j]

print(c.shape)

(3, 4)
```

```
In [7]: c = a + b.T
print(c.shape)

(3, 4)
```

week2-Q9.

Consider the following code:

```
In [8]: a = np.random.randn(3, 3)
b = np.random.randn(3, 1)
c = a*b
print(c.shape)

(3, 3)
```

What will be c? (If you're not sure, feel free to run this in python to find out).

Answer: This will invoke broadcasting, so b is copied three times to become (3,3), and * is an element-wise product so c.shape will be (3, 3).

week3-Q5.

Consider the following code:

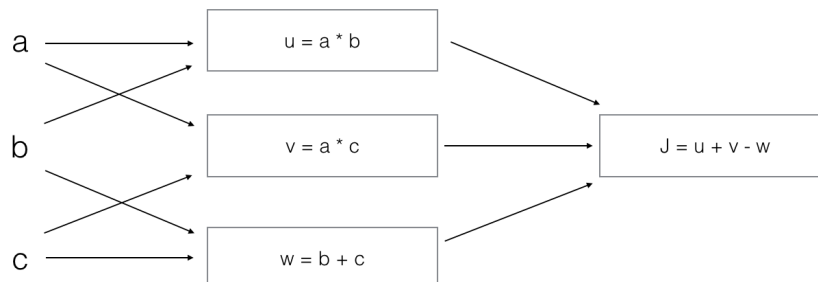
```
In [9]: import numpy as np
A = np.random.randn(4,3)
B = np.sum(A, axis=1, keepdims=True)
print(B.shape)

(4, 1)
```

Answer: We use (keepdims = True) to make sure that A.shape is (4,1) and not (4,). It makes our code more rigorous.

week2-Q10.

Consider the following computation graph.



What is the output J?

Answer: $J = (a - 1) * (b + c)$

$J = u + v - w = a*b + a*c - (b + c) = a * (b + c) - (b + c) = (a - 1) * (b + c).$

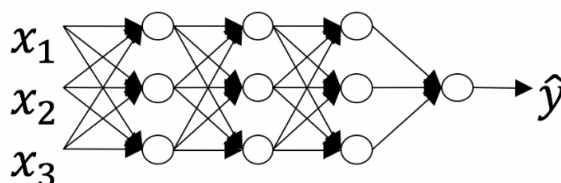
week2-Q2.

Which of these is the "Logistic Loss"?

Answer: $\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) = y^{(i)}\log(\hat{y}^{(i)}) + (1 - y^{(i)})\log(1 - \hat{y}^{(i)})$

week-Q6.

How many layers does this network have?



Answer: The number of layers L is 4. The number of hidden layers is 3. 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.

week3-Q1.

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

- $a^{[2]}$ denotes the activation vector of the 2nd layer. (True)
- $a_4^{[2]}$ is the activation output by the 4th neuron of the 2nd layer. (True)
- $a^{[2](12)}$ denotes the activation vector of the 2nd layer for the 12th training example. (True)
- X is a matrix in which each column is one training example. (True)

week2-Q1.

What does a neuron compute?

Answer: A neuron computes a linear function $z = Wx + b$ followed by an activation function. The output of a neuron is $a = g(Wx + b)$ where g is the activation function (sigmoid, tanh, ReLU, ...).

week3-Q3.

Vectorized implementation of forward propagation for layer l , where $1 \leq l \leq L$?

Answer:

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

$$A^{[l]} = g^{[l]}(Z^{[l]})$$

week-Q4.

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 $l=1, 2, \dots, L$. True/False?

Answer: False. 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: ($a^{[l]} = g^{[l]}(z^{[l]})$, $z^{[l]} = W^{[l]}a^{[l-1]} + b^{[l]}$, ...).

week-Q5.

Assume we store the values for $n^{[l]}$ in an array called layers, 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?

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

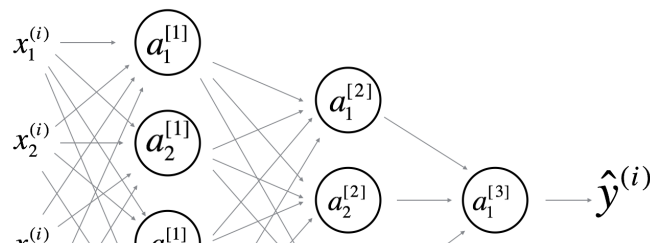
week4-Q10.

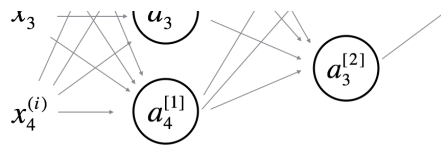
Whereas the previous question used a specific network, in the general case what is the dimension of $W^{[l]}$, the weight matrix associated with layer l ?

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

week4-Q9.

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

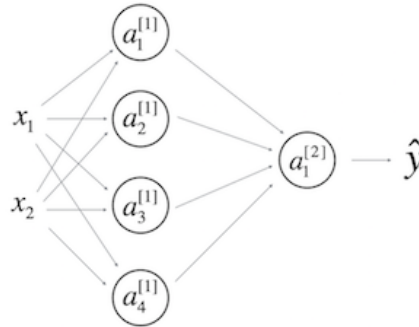




1. $W^{[1]}$ will have shape (4, 4). True, shape of $W^{[l]}$ is $(n^{[l]}, n^{[l-1]})$
2. $W^{[2]}$ will have shape (3, 4). True, shape of $W^{[l]}$ is $(n^{[l]}, n^{[l-1]})$
3. $W^{[3]}$ will have shape (1, 3). True, shape of $W^{[l]}$ is $(n^{[l]}, n^{[l-1]})$
4. $b^{[1]}$ will have shape (4, 1). True, shape of $b^{[l]}$ is $(n^{[l]}, 1)$
5. $b^{[2]}$ will have shape (3, 1). True, shape of $b^{[l]}$ is $(n^{[l]}, 1)$
6. $b^{[3]}$ will have shape (1, 1). True, shape of $b^{[l]}$ is $(n^{[l]}, 1)$

week3-Q9.

Consider the following 1 hidden layer neural network:



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

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

week2-Q6.

Suppose you have n_x input features per example. Recall that $X = [x^{(1)} x^{(2)} \dots x^{(m)}]$. What is the dimension of X ?

Answer: (n_x, m)

week3-Q10.

In the same network as the previous question, what are the dimensions of $Z^{[1]}$ and $A^{[1]}$?

Answer: $Z^{[1]}$ and $A^{[1]}$ are (4, m).

week3-Q4.

You are building a binary classifier for recognizing cucumbers ($y=1$) vs. watermelons ($y=0$). Which one of these activation functions would you recommend using for the output layer?

Answer: sigmoid. Sigmoid outputs a value between 0 and 1 which makes it a very good choice for binary classification. You can classify as 0 if the output is less than 0.5 and classify as 1 if the output is more than 0.5. It can be done with tanh as well but it is less convenient as the output is between -1 and 1.

week3-Q2.

The tanh activation usually works better than sigmoid activation function for hidden units because the mean of its output is closer to zero, and so it centers the data better for the next layer. True/False?

Answer: True, 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.

week3-Q8.

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

Answer: 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. `tanh` becomes flat for large values, this leads its gradient to be close to zero. This slows down the optimization algorithm.

week3-Q6.

Suppose you have built a neural network. You decide to initialize the weights and biases to be zero. Which of the following statements is true?

Answer: Each neuron in the first hidden layer will perform the same computation. So even after multiple iterations of gradient descent each neuron in the layer will be computing the same thing as other neurons.

week3-Q7.

Logistic regression's weights w should be initialized randomly rather than to all zeros, because if you initialize to all zeros, then logistic regression will fail to learn a useful decision boundary because it will fail to "break symmetry", True/False?

Answer: False. Logistic Regression doesn't have a hidden layer. If you initialize the weights to zeros, the first example x fed in the logistic regression will output zero but the derivatives of the Logistic Regression depend on the input x (because there's no hidden layer) which is not zero. So at the second iteration, the weights values follow x 's distribution and are different from each other if x is not a constant vector.

week4-Q1.

What is the "cache" used for in our implementation of forward propagation and backward propagation?

Answer: We use it to pass variables computed during forward propagation to the corresponding backward propagation step. It contains useful values for backward propagation to compute derivatives. The "cache" records values from the forward propagation units and sends it to the backward propagation units because it is needed to compute the chain rule derivatives.

week4-Q2.

Among the following, which ones are "hyperparameters"? (Check all that apply.)

Answer: learning rate α , number of layers L in the neural network, number of iterations, size of the hidden layers $n^{[l]}$.

Not hyperparameters: bias vectors $b^{[l]}$, weight matrices $W^{[l]}$, activation values $a^{[l]}$.

week4-Q3.

Which of the following statements is true?

1. The deeper layers of a neural network are typically computing more complex features of the input than the earlier layers.
2. The earlier layers of a neural network are typically computing more complex features of the input than the deeper layers.