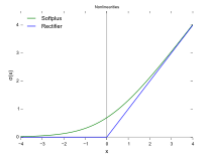
1. What does the analogy "AI is the new electricity" refer to?
Similar to electricity starting about 100 years ago, AI is transforming multiple industries.

2. Which of these are reasons for Deep Learning recently taking off?
- We have access to a lot more computational power.
- We have access to a lot more data.
- Deep learning has resulted in significant improvements in important applications such as online advertising, speech recognition, and image recognition.

3. Recall this diagram of iterating over different ML ideas. Which of the statements below are true?
- Being able to try out ideas quickly allows deep learning engineers to iterate more quickly.
- Faster computation can help speed up how long a team takes to iterate to a good idea.
- Recent progress in deep learning algorithms has allowed us to train good models faster (even without changing the CPU/GPU hardware).

4. 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?
- False

5. Which one of these plots represents a ReLU activation function?


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

7. 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?
- False

8. Why is an RNN (Recurrent Neural Network) used for machine translation, say translating English to French?
- It can be trained as a supervised learning problem.
- It is applicable when the input/output is a sequence (e.g., a sequence of words).

9. In this diagram which we hand-drew in lecture, what do the horizontal axis (x-axis) and vertical axis (y-axis) represent?
- x-axis is the amount of data
- y-axis (vertical axis) is the performance of the algorithm.

10. 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?
- Increasing the training set size generally does not hurt an algorithm's performance, and it may help significantly.
- Increasing the size of a neural network generally does not hurt an algorithm's performance, and it may help significantly.

1. What does a neuron compute?

A neuron computes a linear function (z = Wx + b) followed by an activation function

Note: The output of a neuron is a = g(Wx + b) where g is the activation function (sigmoid, tanh, ReLU)

2. Which of these is the "Logistic Loss"?

Note: We are using a cross-entropy loss function.

3. 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?

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

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

```
a = np.random.randn(2, 3) # a.shape = (2, 3)
b = np.random.randn(2, 1) # b.shape = (2, 1)
c = a + b
```

What will be the shape of "c"?

b (column vector) is copied 3 times so that it can be summed to each column of a.
Therefore, `c.shape = (2, 3)`.

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

```
a = np.random.randn(4, 3) # a.shape = (4, 3)
b = np.random.randn(3, 2) # b.shape = (3, 2)
c = a * b
```

What will be the shape of "c"?

"*" operator indicates element-wise multiplication. Element-wise multiplication requires same dimension between two matrices. It's going to be an error.

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

`(n_x, m)`

Note: A stupid way to validate this is use the formula Z^(l) = W^(l)A^(l) when l = 1, then we have

- A^(1) = X
- X.shape = (n_x, m)
- Z^(1).shape = (n^(1), m)
- W^(1).shape = (n^(1), n_x)

7. 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":

```
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)
```

What is the shape of c?

`c.shape = (12288, 45)`, this is a simple matrix multiplication example.

8. Consider the following code snippet:

```
# a.shape = (3,4)
# b.shape = (4,1)
for i in range(3):

  for j in range(4):

    c[i][j] = a[i][j] + b[j]
```

How do you vectorize this?

```
c = a + b.T
```

9. Consider the following code:

```
a = np.random.randn(3, 3)
b = np.random.randn(3, 1)
c = a * b
```

What will be c?

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

10. Consider the following computation graph.

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

Answer: `(a - 1) * (b + c)`

1. Which of the following are true?

- X is a matrix in which each column is one training example.
- a^[2]_4 is the activation output by the 4th neuron of the 2nd layer
- a^[2](12) denotes the activation vector of the 2nd layer for the 12th training example.
- a^[2] denotes the activation vector of the 2nd layer.

Note: If you are not familiar with the notation used in this course, check here.

2. 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
Note: You can check this post and (this paper)[http://yann.lecun.com/exdb/publis/pdf/lecun-98b.pdf]. 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 is a correct vectorized implementation of forward propagation for layer l, where $1 \leq l \leq L$?

- Z^[l]=W^[l]A^[l−1]+b^[l]
- A^[l]=g^[l](Z^[l])

4. 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?

- Sigmoid

Note: The output value from a sigmoid function can be easily understood as a probability.
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.

5. Consider the following code:

A = np.random.randn(4,3)
B = np.sum(A, axis = 1, keepdims = True)

What will be B.shape?

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

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

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.

7. 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?

- 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.

8. 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?

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.

9. Consider the following 1 hidden layer neural network:

b[1] will have shape (4, 1)
W[1] will have shape (4, 2)
W[2] will have shape (1, 4)
b[2] will have shape (1, 1)
Note: Check here for general formulas to do this.

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

- Z[1] and A[1] are (4,m)

Note: Check here for general formulas to do this.

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

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.

2. Among the following, which ones are "hyperparameters"?

- size of the hidden layers n[l]
- learning rate α
- number of iterations
- number of layers L in the neural network

Note: You can check this Quora post or this blog post.

3. Which of the following statements is true?

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

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

Note: You can check the lecture videos. I think Andrew used a CNN example to explain this.

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?

☐ True

☑ False

Note: We cannot avoid the for-loop iteration over the computations among layers.

5. 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
```

6. Consider the following neural network.

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

Note: The input layer (L^[0]) does not count.

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
During backpropagation you need to know which activation was used in the forward propagation to be able to compute the correct derivative.

8. There are certain functions with the following properties:

(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

Note: See lectures, exactly same idea was explained.

9. Consider the following 2 hidden layer neural network: Which statements are True?

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

Note: See this image for general formulas.

10. 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?

W^[l] has shape (n^[l],n^[l−1])
Note: See this image for general formulas.

1. If you have 10,000,000 examples, how would you split the train/dev/test set?
98% train . 1% dev . 1% test

2. The dev and test set should:
Come from the same distribution

3. If your Neural Network model seems to have high variance, what of the following would be promising things to try?
- Add regularization
- Get more training data
Note: Check here.

4. You are working on an automated check-out kiosk for a supermarket, and are building a classifier for apples, bananas and oranges. Suppose your classifier obtains a training set error of 0.5%, and a dev set error of 7%. Which of the following are promising things to try to improve your classifier?
- Increase the regularization parameter lambda
- Get more training data
Note: Check here.

5. What is weight decay?

A regularization technique (such as L2 regularization) that results in gradient descent shrinking the weights on every iteration.

6. What happens when you increase the regularization hyperparameter lambda?

Weights are pushed toward becoming smaller (closer to 0)

7. With the inverted dropout technique, at test time:

You do not apply dropout (do not randomly eliminate units) and do not keep the 1/keep_prob factor in the calculations used in training

8. Increasing the parameter keep_prob from (say) 0.5 to 0.6 will likely cause the following:

- Reducing the regularization effect
- Causing the neural network to end up with a lower training set error

9. Which of these techniques are useful for reducing variance (reducing overfitting)?

- Dropout
- L2 regularization
- Data augmentation

10. Why do we normalize the inputs x?

It makes the cost function faster to optimize

1. Which notation would you use to denote the 3rd layer's activations when the input is the 7th example from the 8th minibatch?

a^[3]{8}(7)

Note: [i]{j}(k) superscript means i-th layer, j-th minibatch, k-th example

2. Which of these statements about mini-batch gradient descent do you agree with?

☐ You should implement mini-batch gradient descent without an explicit for-loop over different mini-batches, so that the algorithm processes all mini-batches at the same time (vectorization).

☐ Training one epoch (one pass through the training set) using mini-batch gradient descent is faster than training one epoch using batch gradient descent.

☑ One iteration of mini-batch gradient descent (computing on a single mini-batch) is faster than one iteration of batch gradient descent.

Note: Vectorization is not for computing several mini-batches in the same time.

3. Why is the best mini-batch size usually not 1 and not m, but instead something in-between?

- If the mini-batch size is 1, you lose the benefits of vectorization across examples in the mini-batch.
- If the mini-batch size is m, you end up with batch gradient descent, which has to process the whole training set before making progress.

4. Suppose your learning algorithm's cost *J*, plotted as a function of the number of iterations, looks like this:

If you're using mini-batch gradient descent, this looks acceptable. But if you're using batch gradient descent, something is wrong.

Note: There will be some oscillations when you're using mini-batch gradient descent since there could be some noisy data example in batches. However batch gradient descent always guarantees a lower ***J*** before reaching the optimal.

5. Suppose the temperature in Casablanca over the first three days of January are the same:

Jan 1st: $\theta_1 = 10$
Jan 2nd: $\theta_2 * 10$
Say you use an exponentially weighted average with $\beta = 0.5$ to track the temperature: $v_0 = 0$, $v_t = \beta v_{t-1} + (1 - \beta)\theta_t$. If v_2 is the value computed after day 2 without bias correction, and $v^{corrected}_2$ is the value you compute with bias correction. **What are these values?**

$v_2 = 7.5$, $v^{corrected}_2 = 10$

6. Which of these is NOT a good learning rate decay scheme? Here, t is the epoch number.

$\alpha = e^t * \alpha_0$

Note: This will explode the learning rate rather than decay it.

7. You use an exponentially weighted average on the London temperature dataset. You use the following to track the temperature: $v_t = \beta v_{t-1} + (1 - \beta)\theta_t$. The red line below was computed using $\beta = 0.9$. What would happen to your red curve as you vary β?

- Increasing β will shift the red line slightly to the right.
- Decreasing β will create more oscillation within the red line.

8. Consider this figure:

These plots were generated with gradient descent; with gradient descent with momentum ($\beta = 0.5$) and gradient descent with momentum ($\beta = 0.9$). **Which curve corresponds to which algorithm?**

(1) is gradient descent. (2) is gradient descent with momentum (small β). (3) is gradient descent with momentum (large β)

9. Suppose batch gradient descent in a deep network is taking excessively long to find a value of the parameters that achieves a small value for the cost function J(W[1],b[1],...,W[L],b[L]). Which of the following techniques could help find parameter values that attain a small value for J?

- [x] Try using Adam
- [x] Try better random initialization for the weights
- [x] Try tuning the learning rate α
- [x] Try mini-batch gradient descent
- [ ] Try initializing all the weights to zero

10. Which of the following statements about Adam is False?
Adam should be used with batch gradient computations, not with mini-batches.
Note: Adam could be used with both.

1. If searching among a large number of hyperparameters, you should try values in a grid rather than random values, so that you can carry out the search more systematically and not rely on chance.

- [x] False
- [ ] True

Note: Try random values, don't do grid search. Because you don't know which hyperparamerters are more important than others.
And to take an extreme example, let's say that hyperparameter two was that value epsilon that you have in the denominator of the Adam algorithm. So your choice of alpha matters a lot and your choice of epsilon hardly matters.

2. Every hyperparameter, if set poorly, can have a huge negative impact on training, and so all hyperparameters are about equally important to tune well. True or False?

- [x] False
- [ ] True

We've seen in lecture that some hyperparameters, such as the learning rate, are more critical than others.

3. During hyperparameter search, whether you try to babysit one model ("Panda" strategy) or train a lot of models in parallel ("Caviar") is largely determined by:

- [ ] Whether you use batch or mini-batch optimization
- [ ] The presence of local minima (and saddle points) in your neural network
- [x] The amount of computational power you can access
- [ ] The number of hyperparameters you have to tune

4. If you think β (hyperparameter for momentum) is between on 0.9 and 0.99, which of the following is the recommended way to sample a value for beta?

```
r = np.random.rand()
beta = 1 - 10 ** (-r - 1)
```

5. Finding good hyperparameter values is very time-consuming. So typically you should do it once at the start of the project, and try to find very good hyperparameters so that you don't ever have to revisit tuning them again. True or false?

- [x] False
- [ ] True

Note: Minor changes in your model could potentially need you to find good hyperparameters again from scratch.

6. In batch normalization as presented in the videos, if you apply it on the lth layer of your neural network, what are you normalizing?

- z^[l]

7. In the normalization formula, why do we use epsilon?

- To avoid division by zero

8. Which of the following statements about γ and β in Batch Norm are true? Only correct options

- They can be learned using Adam, Gradient descent with momentum, or RMSprop, not just with gradient descent.
- They set the mean and variance of the linear variable z^[l] of a given layer.

9. After training a neural network with Batch Norm, at test time, to evaluate the neural network on a new example you should:

- Perform the needed normalizations, use μ and σ^2 estimated using an exponentially weighted average across mini-batches seen during training.

10. Which of these statements about deep learning programming frameworks are true?

- ☑ A programming framework allows you to code up deep learning algorithms with typically fewer lines of code than a lower-level language such as Python.
- ☑ Even if a project is currently open source, good governance of the project helps ensure that the it remains open even in the long term, rather than become closed or modified to benefit only one company.
- ☐ Deep learning programming frameworks require cloud-based machines to run.