Neural Networks and Deep Learning

dzlabs

September 2018

Week 1

What is a neural network?

QUIZ True or false? As explained in this lecture, every input layer feature is interconnected with every hidden layer feature.

- False
- True (X)

QUIZ Would structured or unstructured data have features such as pixel values or individual words?

- Structured data.
- Unstructured data (X)

Why is Deep Learning taking off?

The scales (large neural network and large data) drives deep learning progress. But also, computation (e.g. GPU) and algorithms. Most recent innovations on algorithms are focused on the making NN learn faster. An example is moving the activation function away from the **Sigmoid** function (which has very slow learning rate at the corners) into a **ReLU** function that has a positive learning rate on the right side of 0. Another important facts related to faster is that building NN is iterative: idea, code, experiment then over again. **QUIZ** What will the variable m denote in this course?

- Number of hidden layers
- Number of training examples (X)
- The expected output
- Slope

QUIZ - Introduction to deep learning

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

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

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

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

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

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

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?

- True (X)
- False

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

- Figure 1:
- Figure 2:
- Figure 3: (X)
- Figure 4:

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

- True
- False (X)

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?

- True
- False (X)

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

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

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. (X)
- x-axis is the performance of the algorithm, y-axis (vertical axis) is the amount of data.
- x-axis is the amount of data, y-axis is the size of the model you train.
- x-axis is the input to the algorithm, y-axis is outputs.

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? (Check all that apply.)

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

Week 2

Neural Networks Basics

We want a cost function to be convex so that Gradient will find/converge the only minimum.

QUIZ What are the parameters of logistic regression?

- W, an identity vector, and b, a real number.
- W and b, both n_x dimensional vectors.
- W, an n_x dimensional vector, and b, a real number. (X)
- W and b, both real numbers.

QUIZ What is the difference between the cost function and the loss function for logistic regression?

- The loss function computes the error for a single training example; the cost function is the average of the loss functions of the entire training set. (X)
- The cost function computes the error for a single training example; the loss function is the average of the cost functions of the entire training set.
- They are different names for the same function.

QUIZ True or false. A convex function always has multiple local optima.

- True
- False (X)

QUIZ On a straight line, the function's derivative...

- changes as values on axis increase/decrease.
- doesn't change. (X)

QUIZ One step of ________ propagation on a computation graph yields derivative of final output variable.

- Forward
- Backward (X)

QUIZ In this class, what does the coding convention dvar represent?

- The derivative of input variables with respect to various intermediate quantities.
- The derivative of any variable used in the code.
- The derivative of a final output variable with respect to various intermediate quantities. (X)

QUIZ In this video, what is the simplified formula for the derivative of the losswith respect to z?

- a / (1-a)
- a (1 - y)
- a - y (X)

QUIZ In the for loop depicted in the video, why is there only one dw variable (i.e. no i superscripts in the for loop)?

- Only one derivative is being computed.
- The value of dw in the code is cumulative. (X)
- Only the derivative of one value is relevant.

```
J = 0, $dw_1 = 0$, $dw_2 = 0$, db = 0
    for i = 1 to m:
    $z^{(i)} = w^T x^{(i)} + b$
    $a^{(i)} = \sigma(z^{(i)})$
    $J+ = -[y^{(i)} \log a^{(i)} + (1 - y^{(i)}) \log(1 - a^{(i)})]$
    $dz^{(i)} = a^{(i)} + y^{(i)}$
    $dw_1 + = x_1^{(i)} dz^{(i)}$
    $dw_2 + = x_2^{(i)} dz^{(i)}$
    $db + = dz^{(i)}$
    J = J/m, $dw_1 = dw_1/m$, $dw_2 = dw_2/m$, db = db / m
```

```
import numpy as np

for iter in range(1000):
  Z = w^T X + b = np.dot(w.T, X) + b
  A = \sigma(Z)
  dZ = A - Y
  dw = 1 /m X dZ^T
  db = 1/m np.sum(dZ)
  w = w - alpha dw
  b = b - alpha db
```

QUIZ How do you compute the derivative of b in one line of code in Python numpy?

- m(np.sum(dz))
- 1 - m(np.sum(dz))
- 1 * m(np.sum(dz))
- 1 / m*(np.sum(dz)) (X)

Broadcasting in Python

In numpy documentation look for broadcasing. (m, n) metrix +-*/ (1, n) or (m, 1) -¿ (m, n)
QUIZ Which of the following numpy line of code would sum the values in a matrix A vertically?

- A.sum(axis)
- A.sum(axis = 1)
- A.sum(axis = 0) (X)

QUIZ What kind of array has dimensions in this format: (10,) ?

- A rank 0 array
- A rank 1 array (X)
- An identity array

QUIZ True or False: Minimizing the loss corresponds with maximizing $\log p(y|x)$.

- False
- True (X)

QUIZ - Neural Network Basics

1. What does a neuron compute?

- A neuron computes the mean of all features before applying the output to an activation function
- A neuron computes an activation function followed by a linear function (z = Wx + b)
- A neuron computes a linear function (z = Wx + b) followed by an activation function (X)
- A neuron computes a function g that scales the input x linearly (Wx + b)

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

- $\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) =\mid y^{(i)} - \hat{y}^{(i)} \mid$
- $\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) = max(0, y^{(i)} - \hat{y}^{(i)})$
- $\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) =\mid y^{(i)} - \hat{y}^{(i)} \mid^2$
- $\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) = -(y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})$ (X)

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

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"?

- The computation cannot happen because the sizes don't match. It's going to be "Error"!
- c.shape = (2, 1)
- c.shape = (2, 3) (X)
- c.shape = (3, 2)

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"?

- c.shape = (4,2)
- The computation cannot happen because the sizes don't match. It's going to be "Error"! (X)
- c.shape = (3, 3)
- c.shape = (4, 3)

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?

- (1,m)(1,m)
- (m, n_x)
- (n_x, m) (X)
- (m,1)(m,1)

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, 150)
- c.shape = (12288, 45) (X)
- c.shape = (150,150)
- The computation cannot happen because the sizes don't match. It's going to be "Error"!

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.T + b.T
- c = a.T + b
- c = a + b.T
- c = a + b (X)

9. Consider the following code:

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

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

- 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) (X)
- This will invoke broadcasting, so b is copied three times to become (3, 3), and * invokes a matrix multiplication operation of two 3x3 matrices so c.shape will be (3, 3)
- This will multiply a 3x3 matrix a with a 3x1 vector, thus resulting in a 3x1 vector. That is, c.shape = (3,1).
- It will lead to an error since you cannot use "*" to operate on these two matrices. You need to instead use np.dot(a,b)

10. Consider the following computation graph. What is the output J?

- J = (c - 1)*(b + a)
- J = (a - 1) * (b + c) (X)
- J = a*b + b*c + a*c
- J = (b - 1) * (c + a)

Week 3

Shallow neural networks

Activation functions

Never use a **Sigmoid** function as the **tanh** $tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ is also superior.

Another good activation function is the **LeRU** and the **leacky LeRU**

Why do you need non-linear activation functions?

Using a linear activation function (i.e. $g(z) = z$) is useless as it turns out the NN into a linear regression. As a composition of functions is also a function.

Having a hidden layer using linear activation and an output layer using a sigmoid function, then the NN ends up a logistic regression.

The only case where using a linear activation function may be usefull is when at the output layer in the case where $y \in R$ (e.g. when predicting the housing prices).

Random Initialization

Initial $w^{[i]}$ to a random array, otherwise all the nodes of the ith layer will be computing the same function. Then, multiply it by a a very small number to avoid an initialization that's very large which causes saturation and very slow learning. It's OK for b to be initialized to zero.

```
w^{[1]} = np.random.randn((2, 2)) * 0.01
b^{[1]} = np.zeros((2, 1))
```

QUIZ - Shallow Neural Networks

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

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

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/False?

- True (X)
- False

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

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

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?

- ReLU

- Leaky ReLU
- sigmoid (X)
- tanh

5. Consider the following code:

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

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

- (, 3)
- (4, 1) (X)
- (1, 3)
- (4,)

6. 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?

- 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. (X)
- Each neuron in the first hidden layer will perform the same computation in the first iteration. But after one iteration of gradient descent they will learn to compute different things because we have "broken symmetry".
- Each neuron in the first hidden layer will compute the same thing, but neurons in different layers will compute different things, thus we have accomplished "symmetry breaking" as described in lecture.
- The first hidden layer's neurons will perform different computations from each other even in the first iteration; their parameters will thus keep evolving in their own way.

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?

- True
- False (X)

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?

- It doesn't matter. So long as you initialize the weights randomly gradient descent is not affected by whether the weights are large or small.
- 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.
- 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. (X)
- 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 α to be very small to prevent divergence; this will slow down learning.

9. Consider the following 1 hidden layer neural network: Which of the following statements are True? (Check all that apply).

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

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

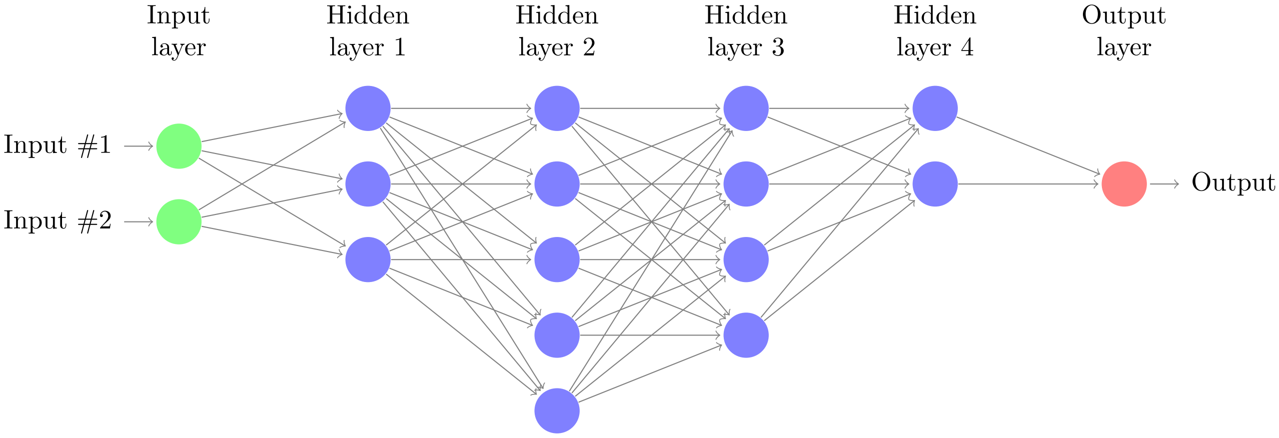
10. Question 10 In the same network as the previous question, what are the dimensions of $Z^{[1]}$ and $A^{[1]}$?

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

Week 4

Deep Neural Network

Getting your matrix dimensions right



With l denoting the layer number, $n^{[l]}$ the number of nodes at layer l:

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

The dimensions of the matrices W and b for one sample are:

$$W^{[l]} : (n^{[l]}, n^{[l-1]}) \tag{2}$$

$$b^{[l]} : (n^{[l]}, 1) \tag{3}$$

For m samples:

$$Z^{[l]} : (n^{[l]}, m) \tag{4}$$

$$A^{[l-1]} : (n^{[l-1]}, m) \tag{5}$$

$$b^{[l]} : (n^{[l]}, 1) \tag{6}$$

Backward propagation for layer l

Input $da^{[l]}$ Output $da^{[l-1]}, dW^{[l]}, db^{[l]}$ Then

$$dz^{[l]} = da^{[l]} * g^{[l]'}(z^{[l]}) \tag{7}$$

$$dw^{[l]} = dz^{[l]} * a^{[l-1]} \tag{8}$$

$$db^{[l]} = dz^{[l]} \tag{9}$$

$$da^{[l-1]} = W^{[l]T} dz^{[l]} \tag{10}$$

Then $dz^{[l]}$ becomes

$$dz^{[l]} = w^{[l+1]T} dz^{[l+1]} * g^{[l]'}(z^{[l]}) \tag{11}$$

The vectorized version will looks like:

$$dZ^{[l]} = dA^{[l]} * g^{[l]'}(Z^{[l]}) \tag{12}$$

$$dW^{[l]} = \frac{1}{m} dZ^{[l]}.A^{[l-1]T} \tag{13}$$

$$db^{[l]} = \frac{1}{m} np.sum(dZ^{[l]}, axis = 1, keepdims = True) \tag{14}$$

$$dA^{[l-1]} = W^{[l]T}.dZ^{[l]} \tag{15}$$

What are hyperparameters?

Parameters: $W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]}, W^{[3]}, b^{[3]}...$ Hyperparameters: paramters that determines the raw parameters above

- learning rate α
- number of iterations
- number of hidden layers L
- number of hidden units $n^{[1]}, n^{[2]}, ...$
- choice of activation layers

Later more hyperprameters: Momentum, mini-batch size, regulations

QUIZ - Key concepts on Deep Neural Networks

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. (X)
- We use it to pass variables computed during backward propagation to the corresponding forward propagation step. It contains useful values for forward propagation to compute activations.
- It is used to cache the intermediate values of the cost function during training.
- It is used to keep track of the hyperparameters that we are searching over, to speed up computation.

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

- number of iterations (X)
- activation values $a^{[l]}$
- number of layers LL in the neural network (X)
- weight matrices $W^{[l]}$
- size of the hidden layers $n^{[l]}$ (X)
- bias vectors $b^{[l]}$
- learning rate α (X)

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. (X)
- The earlier layers of a neural network are typically computing more complex features of the input than the deeper layers.

4. Vectorization allows you to compute forward propagation in an LL-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 (X)

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.
How many layers does this network have?

- The number of layers LL is 4. The number of hidden layers is 3. (X)
- The number of layers LL is 3. The number of hidden layers is 3.
- The number of layers LL is 4. The number of hidden layers is 4.
- The number of layers LL is 5. The number of hidden layers is 4.

7. During forward propagation, in the forward function for a layer ll 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 ll, since the gradient depends on it. True/False?

- True
- False (X)

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 (X)
- False

9. Consider the following 2 hidden layer neural network: Which of the following statements are True? (Check all that apply).

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

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 ll?

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