



AI and Deep Learning

2-4 Neural Network with Multiple Hidden Layers

Zhonglei Wang

WISE, SOE, CHOW

XMU

(Version v1)

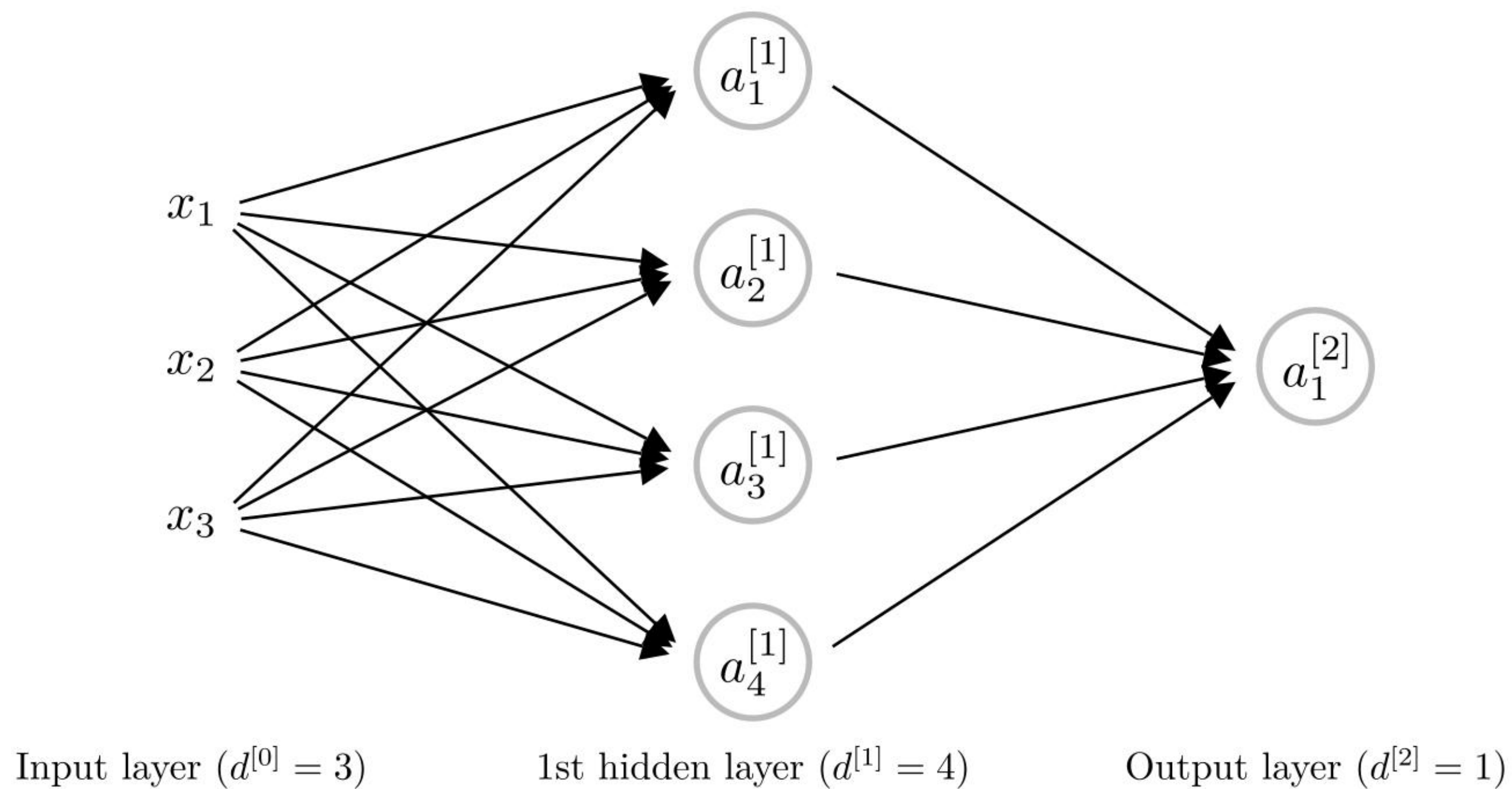
Contents

1. Forward propagation

2. Backpropagation

3. Parameter initialization

Review



Review

1. Denote $\mathbf{A}^{[0]} = \mathbf{X}$

2. Forward propagation:

$$\mathbf{Z}^{[1]} = (\mathbf{b}^{[1]})^T + \mathbf{A}^{[0]} \cdot (\mathbf{W}^{[1]})^T, \quad \mathbf{A}^{[1]} = \sigma(\mathbf{Z}^{[1]}),$$

$$\mathbf{Z}^{[2]} = b^{[2]} + \mathbf{A}^{[1]} \cdot (\mathbf{W}^{[2]})^T, \quad \mathbf{A}^{[2]} = \sigma(\mathbf{Z}^{[2]})$$

3. Backpropagation:

$$d\mathbf{Z}^{[2]} = n^{-1}(\mathbf{A}^{[2]} - \mathbf{Y}), \quad db^{[2]} = (d\mathbf{Z}^{[2]})^T \cdot \mathbf{1}, \quad d\mathbf{W}^{[2]} = (d\mathbf{Z}^{[2]})^T \cdot \mathbf{A}^{[1]},$$

$$d\mathbf{Z}^{[1]} = (d\mathbf{Z}^{[2]} \cdot \mathbf{W}^{[2]}) \circ \sigma'(\mathbf{Z}^{[1]}), \quad d\mathbf{b}^{[1]} = (d\mathbf{Z}^{[1]})^T \cdot \mathbf{1}, \quad d\mathbf{W}^{[1]} = (d\mathbf{Z}^{[1]})^T \cdot \mathbf{A}^{[0]}$$

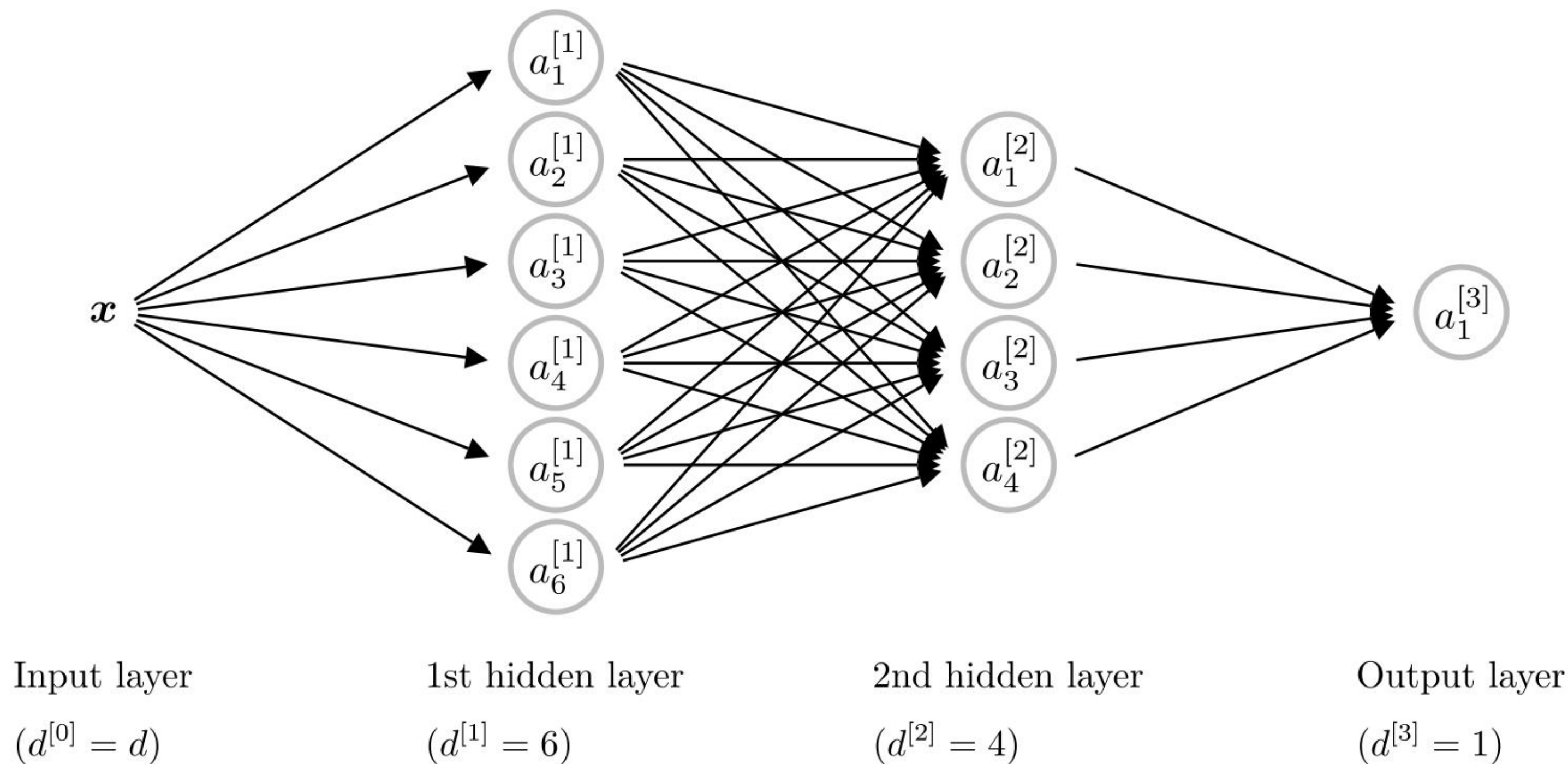
Review

1. Forward propagation and backpropagation can be coded up using for-loops
 - **Forward propagation:** Based on the current model parameters, $\mathbf{A}^{[l-1]}$ is the “input” for $\mathbf{Z}^{[l]}$ and $\mathbf{A}^{[l]}$
 - **Backpropagation:** $d\mathbf{A}^{[l]}$ is the “input” to obtain $dbZ^{[l]}$, $d\mathbf{b}^{[l]}$, $d\mathbf{W}^{[l]}$, $d\mathbf{Z}^{[l-1]}$ and $d\mathbf{A}^{[l-1]}$

Drawbacks

1. By the universal approximation theorem, a neural network with only one hidden layer can approximate any continuous function
 - However, number of neurons should be **exponentially** large
2. Inability to learn hierarchical representation
 - Check the XOR problem by yourself

Example



Forward propagation

1. Let $\mathbf{A}^{[0]} = \mathbf{X} \in \mathbb{R}^{n \times d}$

2. For $l = 1, \dots, L$,

$$\mathbf{Z}^{[l]} = (\mathbf{b}^{[l]})^T + \mathbf{A}^{[l-1]} \cdot (\mathbf{W}^{[l]})^T,$$
$$\mathbf{A}^{[l]} = \sigma^{[l]}(\mathbf{Z}^{[l]})$$

- $\sigma^{[l]}(z)$: Activation function for the l th layer
- Broadcasting is used for the linear transformation as well as the nonlinear activation

Backpropagation

1. $\mathbf{dA}^{[L]}$ can be obtained from the cost function
2. Assume $\mathbf{dA}^{[l]}$ is available for $l = L, \dots, 2$, then we have

$$\mathbf{dZ}^{[l]} = \mathbf{dA}^{[l]} \circ \sigma^{[l]'}(\mathbf{Z}^{[l]}),$$

$$\mathbf{db}^{[l]} = (\mathbf{dZ}^{[l]})^T \cdot \mathbf{1},$$

$$\mathbf{dW}^{[l]} = (\mathbf{dZ}^{[l]})^T \cdot \mathbf{A}^{[l-1]},$$

$$\mathbf{dA}^{[l-1]} = \mathbf{dZ}^{[l]} \cdot \mathbf{W}^{[l]}$$

3. $\mathbf{dX} = \mathbf{dA}^{[0]}$ is often used for sensitivity analysis in applied fields
4. Check the results above by yourself
5. Materials in Section 2-2 may be helpful

Parameter initialization

1. Up to now, we have discussed how to obtain gradients based on the **CURRENT** model parameters
2. One question remains for the gradient descent algorithm: How to initialize model parameters?
 - Specifically, how to finish the **first step** of the gradient descent algorithm?
3. Model parameters involve $\{(\mathbf{b}^{[l]}, \mathbf{W}^{[l]}) : l = 1, \dots, L\}$
4. We usually initialize bias terms $\{\mathbf{b}^{[l]} : l = 1, \dots, L\}$ to be zero
5. It remains to discuss how to initialize those weight terms $\{\mathbf{W}^{[l]} : l = 1, \dots, L\}$
6. For simplicity, assume activation functions to be ReLU in the following discussion

Symmetry issue

1. Why not use a single value to initialize all elements of a weight term?
2. Then, it may cause the symmetry issue
 - For neurons in the same layer, they may be identically the same for both forward and backpropagation as well as gradients
 - Essentially, they play the same role and cannot learn different information
3. The case is even worse if we initialize weight terms by 0
 - Symmetry issue
 - Dying ReLU
4. To avoid symmetry issue, we should initialize weight terms randomly

Intuition for model parameter initialization

1. Forward and backpropagation are essentially conducted for each training example
2. Concentrate on a **SINGLE** training example $(\mathbf{x}, y)$
3. We mainly focus on He initialization; see He et al. (2015) for more discussion
 - ReLU is used as the activation function for hidden layers
 - Initialize **all bias terms** to be zero
 - For $l = 1, \dots, L$, elements of the weight matrix $W^{[l]} = (w_{ij}^{[l]})$ are independently initialized
 - ▷ $w_{ij}^{[l]} \sim N(0, \sigma^{[l]2})$ ($i = 1, \dots, d^{[l]}; j = 1, \dots, d^{[l-1]}$)
 - ▷ Denote $\sigma^{[l]}$ to be the variance associated with the l th layer
 - ▷ Different layers correspond to different variances

Intuition for model parameter initialization

1. Forward propagation:

$$\mathbf{a}^{[l-1]} = \text{ReLU}(\mathbf{z}^{[l-1]}), \quad \mathbf{z}^{[l]} = \mathbf{W}^{[l]} \cdot \mathbf{a}^{[l-1]}$$

2. Backpropagation:

$$d\mathbf{z}^{[l]} = d\mathbf{a}^{[l]} \circ \text{ReLU}'(\mathbf{z}^{[l]}), \quad d\mathbf{a}^{[l-1]} = (\mathbf{W}^{[l]})^T \cdot (d\mathbf{z}^{[l]})$$

- $\text{ReLU}'(z)$ is the derivative of ReLU evaluated at z , and it equals to 0 or 1

3. ReLU as well as its derivative only clip part of “values” to zero, so both are nonlinear

4. Forward propagation and backpropagation are essentially “the same”. Both include:

- Linear transformation, involving a weight matrix $\mathbf{W}^{[l]}$, which affects the variance between its “input” and “output”
- Activation, involving ReLU or its derivative, which may clip variance to half after activation

Intuition for model parameter initialization

1. Thus, it is the initialization of $\mathbf{W}^{[l]}$ that may **dramatically** change the variability before and after calculation
2. Elements of $\mathbf{W}^{[l]}$ are randomly generated from a normal distribution $N(0, \sigma^{[l]2})$
3. It is $\sigma^{[l]2}$ that controls the change in variance before and after calculation
 - Large $\sigma^{[l]2}$ may lead to **gradient exploding**
 - Small $\sigma^{[l]2}$ may result in **gradient vanishing**
4. **Goal:** Choose $\{\sigma^{[l]2} : l = 1, \dots, L\}$ properly to stabilize variance
 - To avoid gradient exploding or vanishing problems

Weight initialization for forward propagation

1. Since $\mathbf{b}^{[l]} = 0$, the forward propagation for the initial step is

$$\mathbf{a}^{[l-1]} = \text{ReLU}(\mathbf{z}^{[l-1]}), \quad \mathbf{z}^{[l]} = \mathbf{W}^{[l]} \cdot \mathbf{a}^{[l-1]}$$

- $\mathbf{a}^{[l-1]} = (a_1^{[l-1]}, \dots, a_{d^{[l-1]}}^{[l-1]})^T \in \mathbb{R}^{d^{[l-1]} \times 1}$, $a_j^{[l-1]} = \text{ReLU}(z_j^{[l-1]}) \quad (j = 1, \dots, d^{[l-1]})$
- $\mathbf{a}^{[0]} = \mathbf{x}$
- $\mathbf{z}^{[l]} = (z_1^{[l]}, \dots, z_{d^{[l]}}^{[l]})^T \in \mathbb{R}^{d^{[l]} \times 1}$, $z_j^{[l]} = \sum_{k=1}^{d^{[l-1]}} w_{jk}^{[l]} a_k^{[l-1]} \quad (j = 1, \dots, d^{[l]})$

2. For $l = 1, \dots, L$, each element of $\mathbf{W}^{[l]}$ is randomly initialized in the **SAME** way,

- Each element in $\mathbf{z}^{[l]}$ has the same **symmetric** distribution and satisfies $E(z_j^{[l]}) = 0$
- Denote the variance of each element in $\mathbf{z}^{[l]}$ to be $\tau^{[l]2}$

Weight initialization for forward propagation

1. For $l = 2, \dots, L$, consider

$$\begin{aligned} E \left\{ (a_j^{[l-1]})^2 \right\} &= E \left[\{\text{ReLU}(z_j^{[l-1]})\}^2 \right] = \int_0^\infty z^2 p^{[l-1]}(z) dz \\ &= \frac{1}{2} E \{ (z_j^{[l-1]})^2 \} = \frac{1}{2} \text{var}(z_j^{[l-1]}) = \frac{1}{2} \tau^{[l-1]2} \end{aligned}$$

- $j = 1, \dots, d^{[l-1]}$
- $p^{[l-1]}(z)$: Density function of $z_j^{[l-1]}$
- The last two equalities hold, since $z_j^{[l-1]}$ is symmetrically distributed with mean 0

Weight initialization for forward propagation

1. Consider

$\text{var}(X)$

$$= E\{\text{var}(X | Y)\} + \text{var}\{E(X | Y)\}$$

$$\tau^{[l]2} = \text{var}(z_j^{[l]}) = \text{var}\left(\sum_{k=1}^{d^{[l-1]}} w_{jk}^{[l]} a_k^{[l-1]}\right)$$

$$= E\left\{\text{var}\left(\sum_{k=1}^{d^{[l-1]}} w_{jk}^{[l]} a_k^{[l-1]} \mid \mathbf{a}^{[l-1]}\right)\right\} + \text{var}\left\{E\left(\sum_{k=1}^{d^{[l-1]}} w_{jk}^{[l]} a_k^{[l-1]} \mid \mathbf{a}^{[l-1]}\right)\right\}$$

$$= \sum_{k=1}^{d^{[l-1]}} \text{var}(w_{jk}^{[l]}) \cdot E\{(a_k^{[l-1]})^2\} + 0$$

$$= \sigma^{[l]2} \sum_{k=1}^{d^{[l-1]}} E\{(a_k^{[l-1]})^2\} = \frac{d^{[l-1]} \cdot \sigma^{[l]2} \cdot \tau^{[l-1]2}}{2}$$

Weight initialization for forward propagation

1. To conclude, we have

$$\tau^{[l]2} = \frac{d^{[l-1]} \cdot \sigma^{[l]2} \cdot \tau^{[l-1]2}}{2}$$

2. Thus, to stabilize the variance for the forward propagation, we set

$$\sigma^{[l]2} = \frac{2}{d^{[l-1]}}$$

- We reach the conclusion to initialize $\mathbf{W}^{[l]}$ for $l = 2, \dots, L$
- Actually, we can apply the same idea to initialize $\mathbf{W}^{[1]}$

Weight initialization for backpropagation

1. The form of $d\mathbf{a}^{[L]}$ is determined by the cost function $\mathcal{J}$

- Thus, we may not guarantee $E(d\mathbf{a}^{[L]}) = 0$ in general

2. For $l = L - 1, \dots, 1$, consider

$$d\mathbf{z}^{[l]} = d\mathbf{a}^{[l]} \circ \text{ReLU}'(\mathbf{z}^{[l]}), \quad d\mathbf{a}^{[l-1]} = (\mathbf{W}^{[l]})^T \cdot (d\mathbf{z}^{[l]})$$

- Elements of $d\mathbf{a}^{[l-1]}$ are i.i.d. with mean zero (check by yourself)
- Each element of $\text{ReLU}'(\mathbf{z}^{[l]})$ has the same probability to be 0 or 1
- If we assume that $\text{ReLU}'(\mathbf{z}^{[l]})$ and $d\mathbf{a}^{[l]}$ are independent, then the variance of each element in $d\mathbf{z}^{[l]}$ is half of that in $d\mathbf{a}^{[l]}$ (check by yourself)

Weight initialization for backpropagation

1. By a similar argument, to stabilize the variance for backpropagation, we set

$$\sigma^{[l]2} = \frac{2}{d^{[l]}}$$

- We reach the conclusion to initialize $\mathbf{W}^{[l]}$ for $l = L - 1, \dots, 1$
- Actually, we can apply the same idea to initialize $\mathbf{W}^{[L]}$

Summary for weight initialization

1. To stabilize the variance, we need

- Set $\sigma^{[l]2} = 2/d^{[l-1]}$ for forward propagation
- Set $\sigma^{[l]2} = 2/d^{[l]}$ for backpropagation

2. For PyTorch, we can use either

3. Of course, we can also compromise to set

$$\sigma^{[l]2} = \frac{4}{d^{[l-1]} + d^{[l]}}$$

4. We may consider other initialization methods (check by yourself)

Summary for weight initialization

1. For forward propagation ($\mathbf{a}^{[l-1]} = \text{ReLU}(\mathbf{z}^{[l-1]})$, $\mathbf{z}^{[l]} = \mathbf{W}^{[l]} \cdot \mathbf{a}^{[l-1]}$), consider two cases:

- If $\{\sigma^{[l]2} : l = 1, \dots, L\}$ are **large**, the variance of weight elements is **large**
 - ▷ The variance ($\tau^{[l]2}$) of each element in $\mathbf{z}^{[l]}$ may be **much larger** than that ($\tau^{[l-1]2}$) of $\mathbf{z}^{[l-1]}$
 - ▷ The variance of $\mathbf{z}^{[l]}$ as well as $\mathbf{a}^{[l]}$ will **dramatically increase** as l increases
 - ▷ Lead to the **exploding gradient problem**, making training procedure unstable
- If $\{\sigma^{[l]2} : l = 1, \dots, L\}$ are **small**, the variance of weight elements is **small**
 - ▷ The variance ($\tau^{[l]2}$) of each element in $\mathbf{z}^{[l]}$ may be **smaller** than that ($\tau^{[l-1]2}$) of $\mathbf{z}^{[l-1]}$
 - ▷ The variance of $\mathbf{z}^{[l]}$ as well as $\mathbf{a}^{[l]}$ will **dramatically decrease** as l increases
 - ▷ Lead to the **gradient vanishing problem**, making training procedure inefficient

Summary

1. Unified framework for neural networks with multiple hidden layers

- Forward propagation: Input is the activated value from the previous layer, output is the one for the current layer
- Backpropagation: Input is the gradient for the activation in current layer, output is the one for the previous layer

2. Parameter initialization

- Initialize bias terms to be 0
- Randomly initialize weight terms to avoid symmetry issues
- Variance of weight initialization is used to stabilize forward and backpropagation