



人工智能与深度学习

2-4 具有多个隐藏层的神经网络

王中雷

经济学院、王亚南经济研究院、邹至庄经济研究院

厦门大学

(第一版)

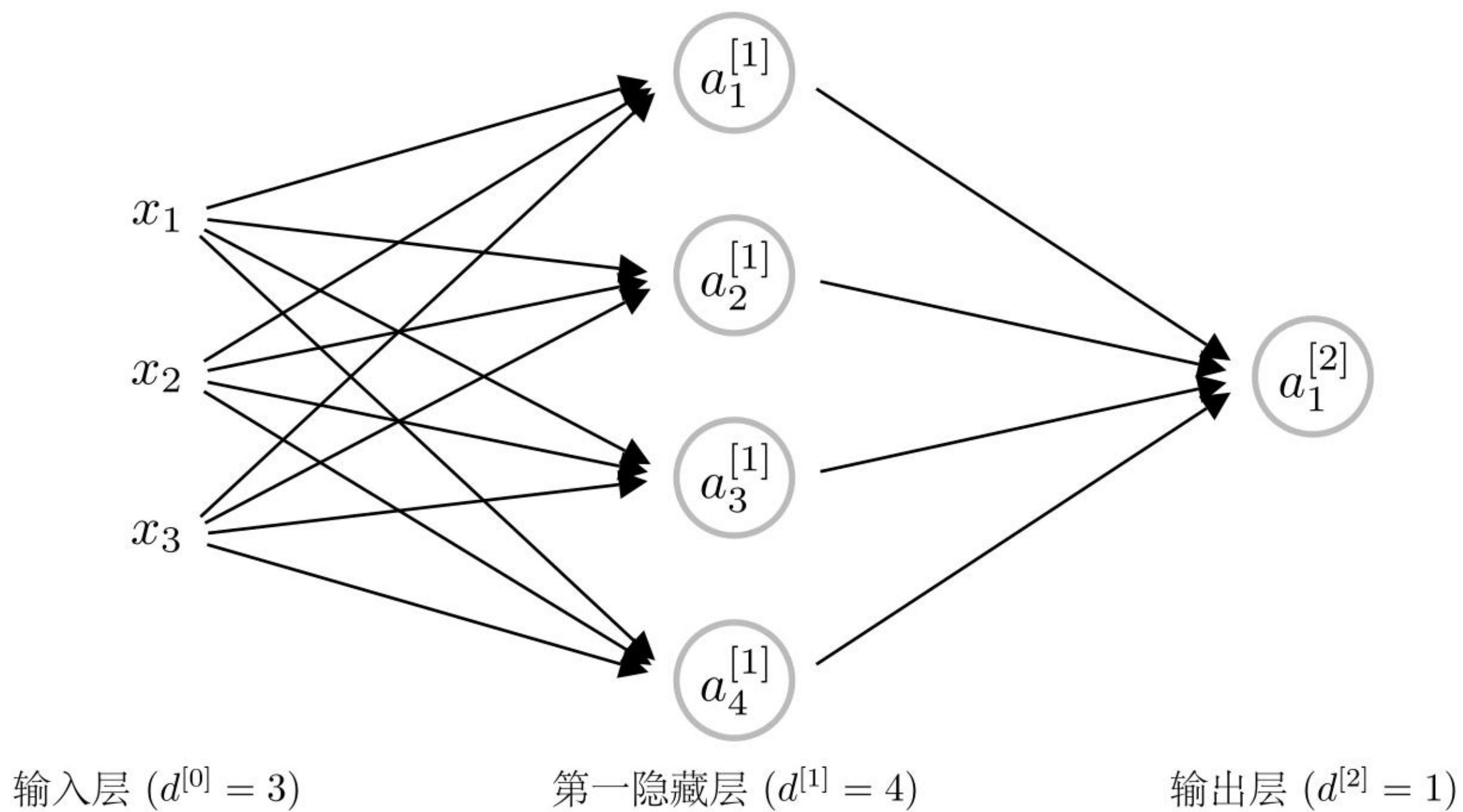
内容摘要

1. 前向传播

2. 后向传播

3. 模型参数初始化

回顾



回顾

1. 记 $\mathbf{A}^{[0]} = \mathbf{X}$

2. 前向传播:

$$\begin{aligned}\mathbf{Z}^{[1]} &= (\mathbf{b}^{[1]})^T + \mathbf{A}^{[0]} \cdot (\mathbf{W}^{[1]})^T, & \mathbf{A}^{[1]} &= \sigma(\mathbf{Z}^{[1]}), \\ \mathbf{Z}^{[2]} &= b^{[2]} + \mathbf{A}^{[1]} \cdot (\mathbf{W}^{[2]})^T, & \mathbf{A}^{[2]} &= \sigma(\mathbf{Z}^{[2]})\end{aligned}$$

3. 后向传播:

$$\begin{aligned}d\mathbf{Z}^{[2]} &= n^{-1}(\mathbf{A}^{[2]} - \mathbf{Y}), & db^{[2]} &= (d\mathbf{Z}^{[2]})^T \cdot \mathbf{1}, & d\mathbf{W}^{[2]} &= (d\mathbf{Z}^{[2]})^T \cdot \mathbf{A}^{[1]}, \\ d\mathbf{Z}^{[1]} &= (d\mathbf{Z}^{[2]} \cdot \mathbf{W}^{[2]}) \circ \sigma'(\mathbf{Z}^{[1]}), & db^{[1]} &= (d\mathbf{Z}^{[1]})^T \cdot \mathbf{1}, & d\mathbf{W}^{[1]} &= (d\mathbf{Z}^{[1]})^T \cdot \mathbf{A}^{[0]}\end{aligned}$$

回顾

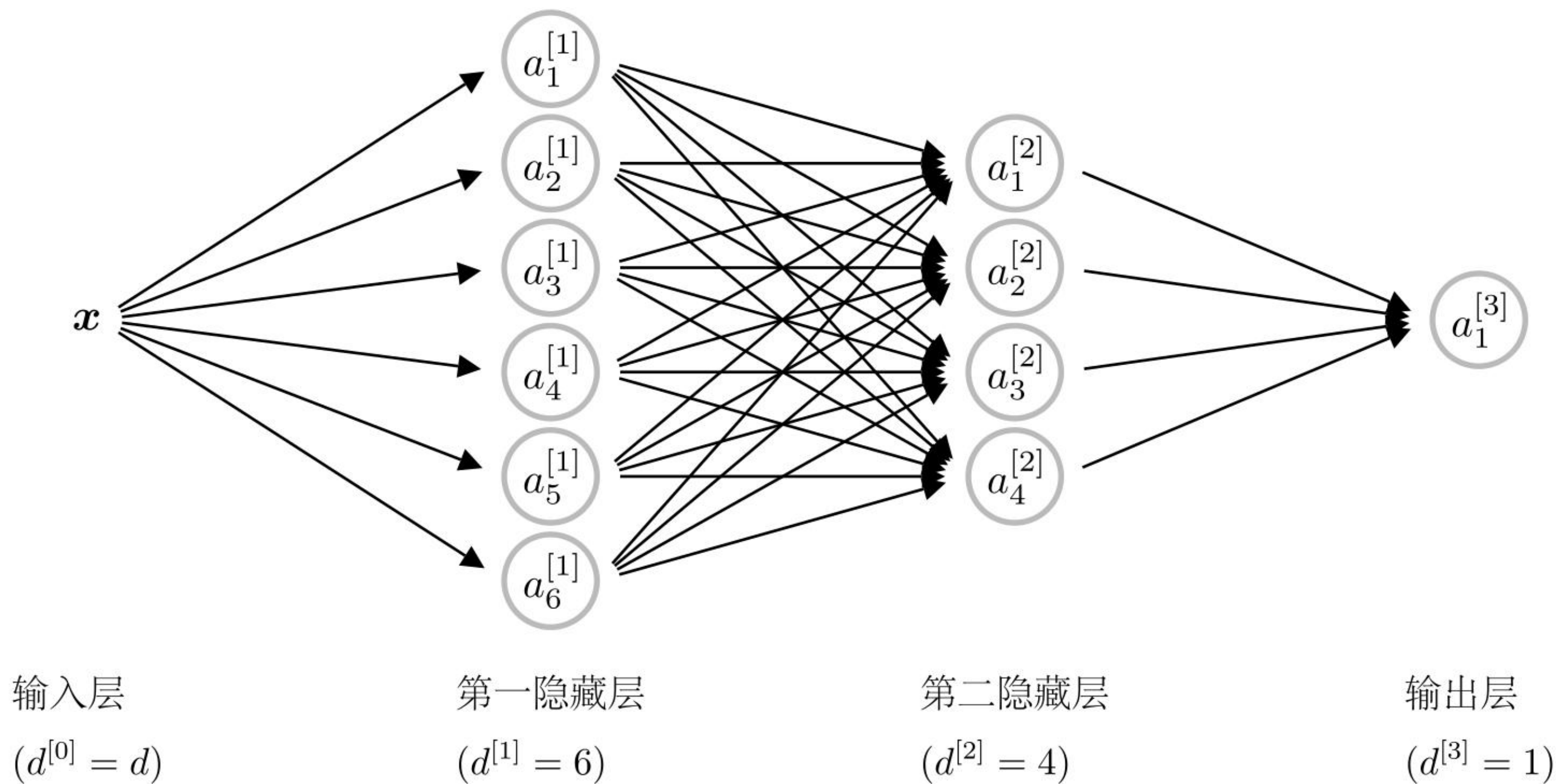
1. 前向传播和后向传播均可使用 for 循环实现

- **前向传播**: 基于当前模型参数, $\mathbf{A}^{[l-1]}$ 可被视为计算 $\mathbf{Z}^{[l]}$ 和 $\mathbf{A}^{[l]}$ 的“输入”
- **后向传播**: $d\mathbf{A}^{[l]}$ 是用来计算 $d\mathbf{Z}^{[l]}$ 、 $d\mathbf{b}^{[l]}$ 、 $d\mathbf{W}^{[l]}$ 以及 $d\mathbf{A}^{[l-1]}$ 的“输入”

缺点

1. 根据通用逼近定理，单隐藏层神经网络在一定条件下可以逼近任意连续函数
 - 然而，对于某些具有层次化结构的函数，浅层网络可能需要远多于深层网络的神经元
2. 难以高效学习层次化表示
 - 请课下自行探索 XOR（异或）问题

例子



前向传播

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

2. 对于 $l = 1, \dots, L$,

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

- $\sigma^{[l]}(z)$: 第 l 层的激活函数
- Python 中的广播机制可用于计算线性变换和非线性激活

后向传播

1. $\mathbf{dA}^{[L]}$ 可以通过代价函数计算得到（假设已经得到）

2. 假设 $\mathbf{dA}^{[l]}$ 已知 $(l = L, \dots, 1)$

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

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

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

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

3. 在某些应用领域, $\mathbf{dX} = \mathbf{dA}^{[0]}$ 通常用于做敏感性分析

4. 请自行检验以上结果

5. 第 2-2 节中的部分内容可能有所帮助

模型参数初始化

1. 到目前为止，我们已经知道如何基于当前模型参数计算梯度
2. 对于梯度下降法而言，我们尚未讨论如何对模型参数进行初始化
 - 也就是说，我们该如何完成梯度下降法的第一步呢
3. 模型参数包括 $\{(\mathbf{b}^{[l]}, \mathbf{W}^{[l]}) : l = 1, \dots, L\}$
4. 我们通常将偏置项 $\{\mathbf{b}^{[l]} : l = 1, \dots, L\}$ 初始化为 0
5. 我们只需讨论，如何初始化权重项 $\{\mathbf{W}^{[l]} : l = 1, \dots, L\}$ 即可
6. 简单起见，在后续的讨论中，我们假设激活函数为 ReLU

对称性问题

1. 为什么不能用同一个数值，初始化一个权重项的每一个元素？
2. 如果这样做，将导致对称性问题的出现
 - 对于同一层的不同神经元，它们的前向传播、后向传播以及梯度结果将完全一致
 - 本质上讲，这一层只有一个神经元，不能学习到不同的特征信息
3. 如果将权重项都初始化为 0，情况将更加糟糕
 - 对称性问题
 - 死亡 ReLU
4. 为了避免对称性问题，我们必须随机初始化权重矩阵

模型参数初始化的直观理解

1. 本质上讲，前向和后向传播是对每个训练样本单独进行计算的
2. 考虑一个训练样本 $(\boldsymbol{x}, y)$
3. 我们主要讨论 He 初始化方法，更多讨论参见 He et al. (2015)
 - 隐藏层中的激活函数是 ReLU
 - 将模型所有的偏置项初始化为 0
 - 对于 $l = 1, \dots, L$ ，权重矩阵 $W^{[l]} = (w_{ij}^{[l]})$ 中的各元素独立同分布地初始化
 - ▷ $w_{ij}^{[l]} \sim N(0, \sigma^{[l]2})$ ($i = 1, \dots, d^{[l]}; j = 1, \dots, d^{[l-1]}$)
 - ▷ 记 $\sigma^{[l]2}$ 为第 l 层对应的方差
 - ▷ 不同层的方差可能不尽相同

模型参数初始化的直观理解

1. 前向传播:

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

2. 后向传播:

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

- $\text{ReLU}'(z)$ 表示 ReLU 激活函数在 z 处的导数，其取值是 0 或者 1（按约定处理不可导点）

3. ReLU 及其导数，只是将“某些值”压缩至 0；因此，两个都是非线性变换

4. 本质上讲，前向和后向传播是“一样的”。两个过程均包含：

- 线性变换，涉及权重矩阵 $\mathbf{W}^{[l]}$ ，其显著影响变换前后的方差
- 关于零均值的输入，ReLU 激活会将输出的二阶矩缩小为输入二阶矩的一半（后面讨论）

模型参数初始化的直观理解

1. 因此，直接导致计算前后方差可能产生剧烈变化的是，权重项 $\mathbf{W}^{[l]}$ 的初始化方式
2. 权重矩阵 $\mathbf{W}^{[l]}$ 中的元素，是从正态分布 $N(0, \sigma^{[l]2})$ 中随机产生的
3. 换句话说，控制计算前后方差变化的是 $\sigma^{[l]2}$
 - 较大的 $\sigma^{[l]2}$ 容易造成梯度爆炸
 - 较小的 $\sigma^{[l]2}$ 容易造成梯度消失
4. 目标：分析前向传播和后向传播，确定 $\{\sigma^{[l]2} : l = 1, \dots, L\}$ 以稳定方差
 - 尽量避免梯度爆炸或者梯度消失问题

前向传播中权重矩阵的初始化

1. 由于 $\mathbf{b}^{[l]} = 0$ ，对应于初始化模型参数的前向传播的计算为

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

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

2. 对于 $l = 1, \dots, L$ ，由于权重矩阵 $\mathbf{W}^{[l]}$ 中每个元素，以相同的方式进行随机初始化，

- $\mathbf{z}^{[l]}$ 中每个元素，具有相同的（关于 0）对称分布，且满足 $E(z_j^{[l]}) = 0$
- 记 $\mathbf{z}^{[l]}$ 中每个元素的方差为 $\tau^{[l]2}$

前向传播中权重矩阵的初始化

1. 对于 $l = 2, \dots, L$, 考虑

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

- $j = 1, \dots, d^{[l-1]}$
- $p^{[l-1]}(z)$: $z_j^{[l-1]}$ 的概率密度函数
- 最后两个等式成立, 是由于 $z_j^{[l-1]}$ 的分布是关于 0 对称的, 且其均值为 0

前向传播中权重矩阵的初始化

1. 考虑

$\text{var}(X)$

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

$$\begin{aligned}\tau^{[l]2} &= \text{var}(z_j^{[l]}) = \text{var}\left(\sum_{k=1}^{d^{[l-1]}} w_{jk}^{[l]} a_k^{[l-1]}\right) \\&= E\left\{\text{var}\left(\sum_{k=1}^{d^{[l-1]}} w_{jk}^{[l]} a_k^{[l-1]} \mid \mathbf{a}^{[l-1]}\right)\right\} + \text{var}\left\{E\left(\sum_{k=1}^{d^{[l-1]}} w_{jk}^{[l]} a_k^{[l-1]} \mid \mathbf{a}^{[l-1]}\right)\right\} \\&= \sum_{k=1}^{d^{[l-1]}} \text{var}(w_{jk}^{[l]}) \cdot E\{(a_k^{[l-1]})^2\} + 0 \\&= \sigma^{[l]2} \sum_{k=1}^{d^{[l-1]}} E\{(a_k^{[l-1]})^2\} = \frac{d^{[l-1]} \cdot \sigma^{[l]2} \cdot \tau^{[l-1]2}}{2}\end{aligned}$$

前向传播中权重矩阵的初始化

1. 总结以上推导过程，我们有

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

2. 因此，为了稳定前向传播的方差，我们令

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

- 我们主要基于 $l = 2, \dots, L$ ，讨论了 $\mathbf{W}^{[l]}$ 的初始化
- 实际上，我们可将上述过程应用于初始化 $\mathbf{W}^{[1]}$

后向传播中权重矩阵的初始化

1. $d\mathbf{a}^{[L]}$ 的形式，由代价函数 $\mathcal{J}$ 决定

- 因此，我们通常不能保证 $E(d\mathbf{a}^{[L]}) = 0$

2. 对于 $l = L - 1, \dots, 1$ ，考虑

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

- 基于初始化 $\mathbf{W}^{[l]}$ 的方式及相关独立性近似， $d\mathbf{a}^{[l-1]}$ 中的每个元素是独立同分布的，且均值为零
- 基于上面的假设，我们还能证明： $\text{ReLU}'(\mathbf{z}^{[l]})$ 中的每个元素，以相同概率等于 0 或者 1
- 如果假设 $\text{ReLU}'(\mathbf{z}^{[l]})$ 和 $d\mathbf{a}^{[l]}$ 是独立的， $d\mathbf{z}^{[l]}$ 中每个元素的方差，是 $d\mathbf{a}^{[l]}$ 中元素方差的一半（自行验证）

后向传播中权重矩阵的初始化

1. 因此，类似于前面的讨论，为了稳定后向传播的方差，我们令

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

- 我们主要基于 $l = L - 1, \dots, 1$ ，讨论了 $\mathbf{W}^{[l]}$ 的初始化
- 实际上，我们可将上述过程应用于初始化 $\mathbf{W}^{[L]}$

权重初始化的总结

1. 为了稳定方差，我们需要

- 在前向传播中，令 $\sigma^{[l]2} = 2/d^{[l-1]}$
- 在后向传播中，令 $\sigma^{[l]2} = 2/d^{[l]}$

2. 在 PyTorch 中，我们可以用任何一种初始化方式（fan_in 或 fan_out 模式）

3. 当然，我们也可以折中地令

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

4. 我们也可以选择其他的模型参数初始化方案（请大家课下自行学习）

权重初始化的总结

1. 对于前向传播 ($\mathbf{a}^{[l-1]} = \text{ReLU}(\mathbf{z}^{[l-1]}); \quad \mathbf{z}^{[l]} = \mathbf{W}^{[l]} \cdot \mathbf{a}^{[l-1]}$), 考虑两种情况

- 如果 $\{\sigma^{[l]2} : l = 1, \dots, L\}$ 很大, 那么, 权重矩阵每个元素的方差就会很大
 - ▷ $\mathbf{z}^{[l]}$ 每个元素的方差, 将显著大于 $\mathbf{z}^{[l-1]}$ 中每个元素的方差; 即 $\tau^{[l]2}$ 将显著大于 $\tau^{[l-1]2}$
 - ▷ 随着层数的加深, $\mathbf{z}^{[l]}$ 以及 $\mathbf{a}^{[l]}$ 相应元素的方差会急剧增加
 - ▷ 前向传播中的激活值可能急剧增大, 并可能伴随后向传播中的梯度爆炸问题
- 如果 $\{\sigma^{[l]2} : l = 1, \dots, L\}$ 很小, 那么, 权重矩阵每个元素的方差就会很小
 - ▷ $\mathbf{z}^{[l]}$ 每个元素的方差, 将显著小于 $\mathbf{z}^{[l-1]}$ 每个元素的方差; 即 $\tau^{[l]2}$ 将显著小于 $\tau^{[l-1]2}$
 - ▷ 随着层数的加深, $\mathbf{z}^{[l]}$ 以及 $\mathbf{a}^{[l]}$ 相应元素的方差会急剧减小
 - ▷ 前向传播中的激活值可能逐层衰减, 并可能伴随后向传播中的梯度消失问题

课程总结

1. 针对具有多个隐藏层的神经网络模型，讨论了统一的计算框架

- 前向传播：输入是上一层的激活值，输出是当前层的激活值
- 后向传播：输入是代价函数对当前层激活的梯度，输出是对上一层激活值的梯度

2. 参数初始化

- 将偏置项初始化为 0
- 随机初始化权重项，以避免对称性问题
- 确定随机初始化权重项时，所用的方差大小的原则是：稳定前向传播和后向传播