

# 6. 激活函数

## 1. 激活函数的作用

1. 激活函数的作用:

如果不使用激活函数, 这种情况下每层输出都是上一层输入的线性函数, 无论神经网络有多少层, 输出都是输入的线性函数, 这就像只有一个隐藏层的效果是一样的。这种情况相当于多层感知机。

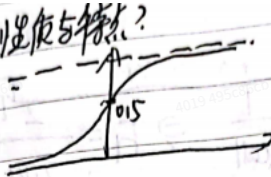
使用激活函数是为了增加模型的非线性因素, 从而使模型表达能力更强大。

## 2. Sigmoid

2) 有哪些激活函数, 都有什么性质与特点?

① sigmoid

$$f(z) = \frac{1}{1 + e^{-z}}$$



它能够将输入的连续实值变换为 0~1 之间的输出。

缺点: 在深度神经网络中, 梯度反向传递时有可能产生梯度爆炸或梯度消失。

① sigmoid  $\sigma(x) = \frac{1}{1 + e^{-x}}$  (只有在 0 附近有较好的收敛性)

优点: (1) 便于求导的平滑函数

(2) 能压缩数据, 保证数据幅度 [0, 1]。

(3) 适用于前向传播

缺点: (1) 容易出现梯度消失 (gradient vanishing) 的现象:

当激活函数接近饱和区时, 变化太缓慢, 导数接近于 0, 而根据反向传播的链式法则, 当前导数需要之前各层导数的乘积, 几个较小的数相乘, 导数结果接近 0, 从而无法完成深层网络的训练。

梯度消失

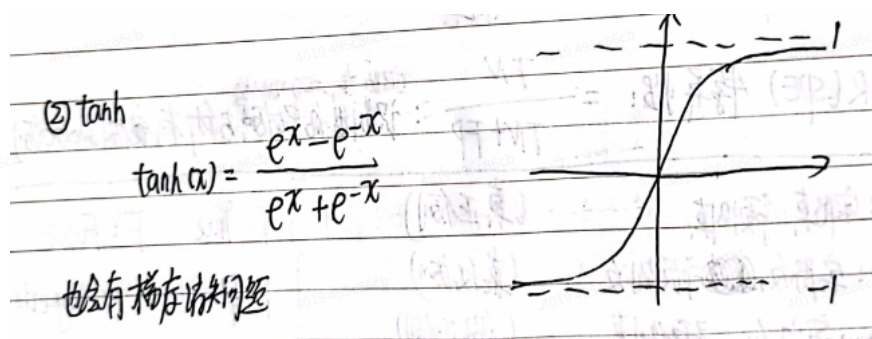
(2) Sigmoid 的输出不是 0 均值 (zero-centered)

这会导致后层的神元输入是非 0 均值信号, 这会对训练产生影响。以  $h = \text{sigmoid}(w \cdot x + b)$  为例, 假设输入为正, 对  $w$  的导数也为正, 这样在反向传播中一直向正方向更新, 使收敛很慢。

(3) 累运算相对耗时。



### 3. Tanh



Date: \_\_\_\_\_

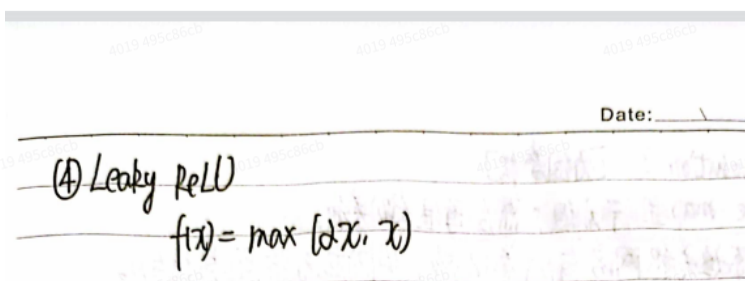
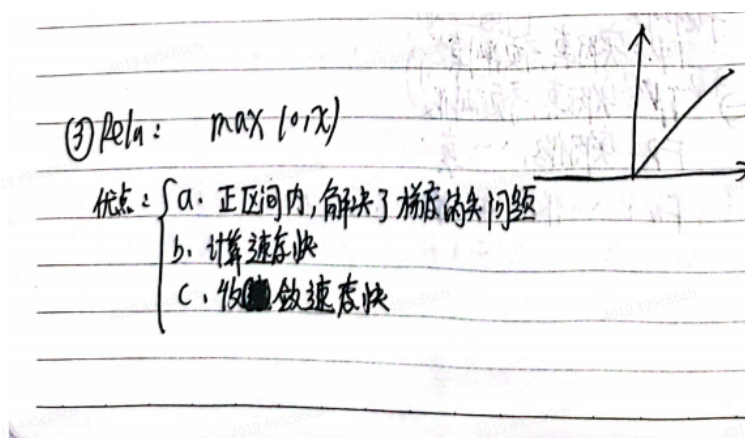
② tanh

• tanh函数将输入值压缩到  $-1$  到  $1$  的范围，因为它呈0均值分布。

缺点：存在 梯度消失 和 幂运算 的问题

实现：  $\tanh(x) = 2 \operatorname{sigmoid}(x) - 1$

### 4. ReLU



优点:

(1) 收敛速度比 sigmoid 和 tanh 快。原因是计算复杂度低, 不需要进行指数运算。

(2) 梯度不会饱和, 解决了梯度消失的问题

(3) 适用于反向传播

缺点: (1) ReLU 的输出不是 zero-centered

(2) 有 Dead ReLU Problem (神经元坏死现象):

某些神经元可能永远不能被激活, 导致相应参数永远不会被更新 (在负数部分, 梯度为 0)。

产生这种现象两个原因:

a. 参数初始化问题: 可以采用 Xavier 初始化方法

b. learning rate 太高 导致在训练过程中参数更新太大;

避免将 lr 设置过大, 或使用 adagrad 等自动调节 lr 的方法。

(3) ReLU 不会对数据做幅度压缩, 所以数据的幅度会随着模型层数的增加不断扩张

④ Leaky ReLU 函数

$$f(x) = \max(0.01x, x)$$

用来解决 ReLU 带来的 神经元坏死的问题。

但效果不如 ReLU 好。