蒋豆芽

深度学习——激活函数 已购

来自 【机器学习面试题汇总与解析（蒋豆芽面试题总结）】 | 98 浏览 | 0 回复 | 2021-04-18

蒋豆芽

+关注

机器学习面试题汇总与解析——激活函数

1. **说一下你了解的激活函数？分别应用于什么场景？** ☆☆☆☆☆
2. **说说你平时都用过什么激活函数，各自什么特点？** ☆☆☆☆☆
3. **写一下leaky ReLU的公式，跟ReLU比有什么优势？** ☆☆☆☆☆
4. **了解ReLU6吗？** ☆☆☆☆☆
5. **sigmoid有什么缺点，有哪些解决办法？** ☆☆☆☆☆
6. **relu在零点可导吗，不可导如何进行反向传播？** ☆☆☆☆☆
7. **推导sigmoid求导公式**☆☆☆☆☆
8. **Softmax公式，溢出怎么处理**☆☆☆☆☆
9. **Softmax公式求导**☆☆☆☆☆

==========

- 本专栏适合于**Python已经入门**的学生或人士，有一定的编程基础。
- 本专栏适合于**算法工程师、机器学习、图像处理求职**的学生或人士。
- 本专栏针对面试题答案进行了**优化，尽量做到好记、言简意赅。这才是一份面试题总结的正确打开方式。这样才方便背诵**
- 如专栏内容有错漏，欢迎在评论区指出或私聊我更改，一起学习，共同进步。
- **相信大家都有着高尚的灵魂，请尊重我的知识产权，未经允许严禁各类机构和个人转载、传阅本专栏的内容。**

==========

1. **说一下你了解的激活函数？分别应用于什么场景？** ☆☆☆☆☆

   参考回答

   1. sigmoid。sigmoid(x)型函数也称Logistic函数，公式如下：

$$\sigma(x) = \frac{1}{1 + exp(-x)} \quad (.)$$

蒋豆芽

问题，在深层网络中被其他激活函数替代。在**逻辑回归**中使用的该激活函数用于输出**分类**。

2. tanh。tanh(x)型函数可以解决sigmoid型函数的**期望（均值）不为0**的情况。函数输出范围为(-1,+1)。但tanh(x)型函数依然存在**梯度消失的问题**。公式如下：

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} = \frac{e^{2x} - 1}{e^{2x} + 1} = 2 \cdot sigmoid(2x) - 1 \tag{.}$$

在LSTM中使用了tanh(x)型函数。

3. ReLU。ReLU(x)型函数可以有效避免**梯度消失的问题**，公式如下：

$$ReLU(x) = \begin{cases} x, & if \quad x \geq 0 \\ 0, & if \quad x < 0 \end{cases} \tag{.}$$

ReLU(x)型函数的缺点是**负值成为“死区”**，神经网络无法再对其进行响应。Alex-Net使用了ReLU(x)型函数。当我们训练深层神经网络时，最好使用ReLU(x)型函数而不是sigmoid(x)型函数。

4. Leaky ReLU。Leaky ReLU(x)型函数为**负值增加了一个斜率**，缓解了“死区”现象，公式如下：

$$Leaky \quad ReLU(x) = \begin{cases} x, & if \quad x \geq 0 \\ \alpha \cdot x, & if \quad x < 0 \end{cases} \tag{.}$$

Leaky ReLU(x)型函数缺点是，**超参数a（阿尔法）合适的值不好设定**。当我们想让神经网络能够学到负值信息，那么使用该激活函数。

5. Mish激活函数。Mish激活函数同样允许负值有一定的梯度流入。公式如下：

$$Mish(x) = x \cdot tanh(log(1 + e^{x})) \tag{.}$$

应用场景同Leaky ReLU(x)型函数。

6. 参数化ReLU（P-ReLU）。参数化ReLU为了解决超参数a（阿尔法）合适的值不好设定的问题，干脆将这个参数也融入模型的整体训练过程中。也使用误差反向传播和随机梯度下降的方法更新参数。

7. 随机化ReLU（R-ReLU）。顾名思义，就是超参数a（阿尔法）随机化，**让不同的层自己学习不同的超参数**，但随机化的超参数的分布符合均值分布或高斯分布。

8. 指数化线性单元（ELU）。也是为了解决死区问题，公式如下：

$$ELU(x) = \begin{cases} x, & if \quad x \geq 0 \\ \lambda \cdot (e^{x} - 1), & if \quad x < 0 \end{cases} \tag{.}$$

缺点是**指数计算量大**。

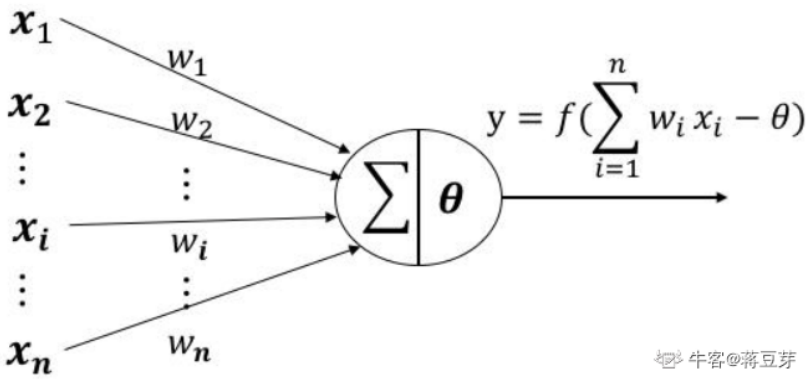
9. Maxout。与常规的激活函数不同，**Maxout**是一个可以学习的**分段线性函数**。其原理是，任何ReLU及其变体等激活函数都可以看成分段的线性函数，而Maxout加入的一层神经元正是一个可以学习参数的分段线性函数。

蒋豆芽

答案解析

为什么需要激活函数?

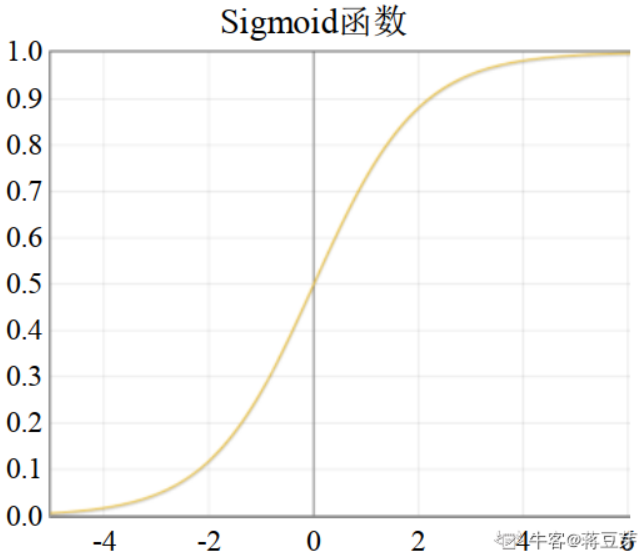
下图是神经元模型:



其中x_i为第i个输入，w_i为第i个神经元的权重，θ为门限。可以看到，y为线性函数，线性函数就只能处理线性问题，非线性的问题没办法处理了。所以为了让神经网络也能处理非线性的问题，才需要引入非线性激活函数，把线性函数给非线性化。**这就是使用激活函数的意义。**

好，我们一个个分析:

1. 首先是**sigmoid函数**，



从函数和公式可以很清楚地看到，sigmoid函数将输入映射到了[0,1]之间，因为输出都大于0，所以**sigmoid函数期望（均值）大于0的**。什么意思呢？简单理解就是(0+1)/2=0.5，均值大于0。如果输出区间在[-1,1]之间，那么均值就是(-1+1)/2=0了。

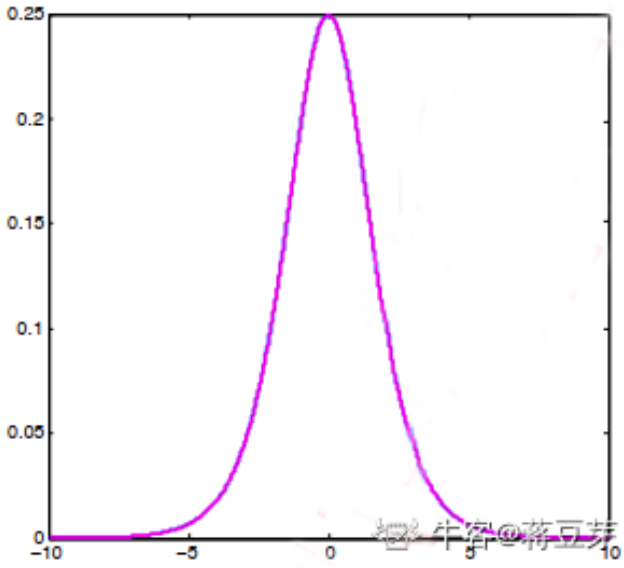
当然这只是简单理解，期望计算公式如下:

$$E(x) = \sum_{k=1}^{\infty} x_k p_k$$

蒋豆芽

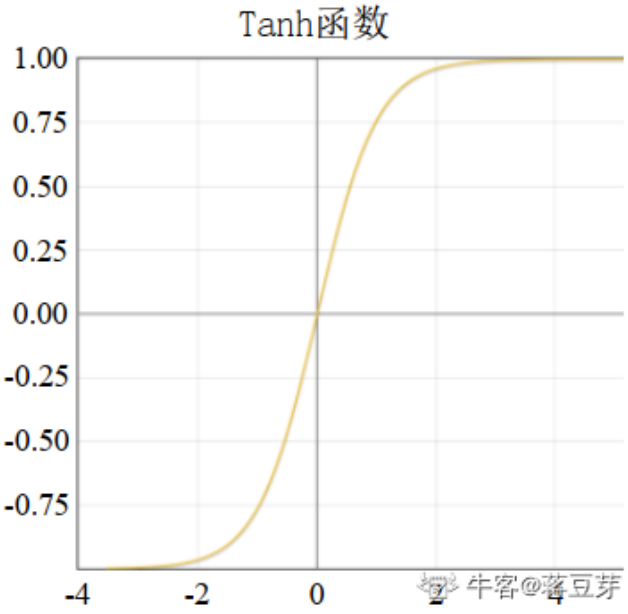
那为什么要均值为0最好呢？哈哈，以后豆芽写文章详细讲解。

那怎么又存在**梯度消失的问题**呢？梯度是啥啊？梯度就是曲线斜率吧，曲线斜率就是切线吧，大家可以回忆一下高中的知识哈，那么可以看到sigmoid函数两边的斜率已经越来越趋向0了，如图：



都趋向0了，我们神经网络反向传播参数的时候参数就不更新了呀，不更新就代表网络没有继续学习了，这就是**梯度消失。**

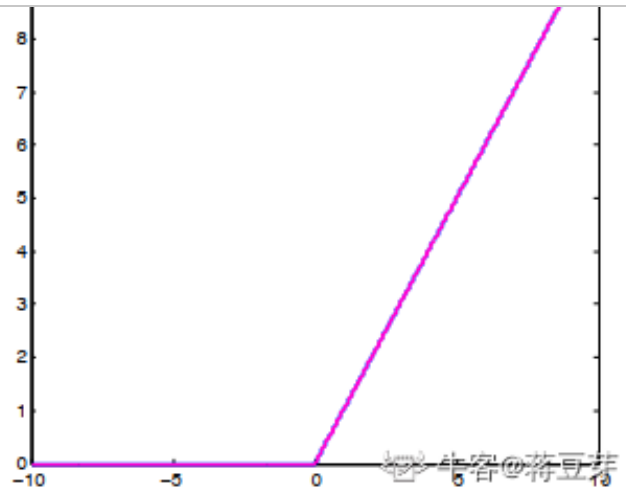
2. 继续tanh函数，可以看到函数曲线：



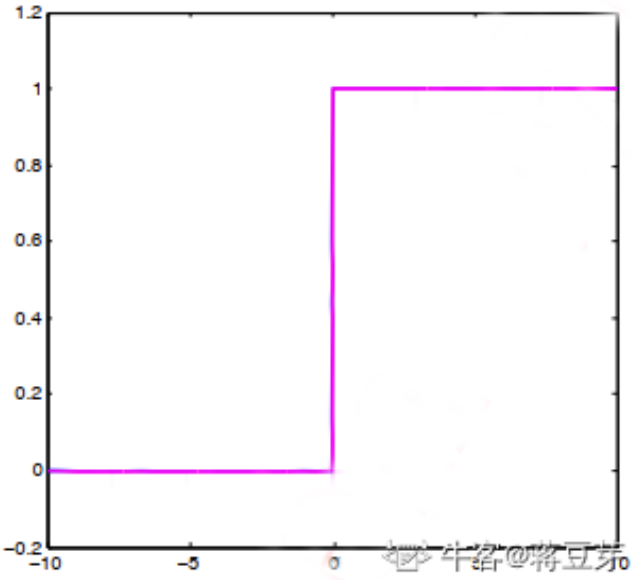
输出范围成了[-1,1]，期望（均值）为0。梯度消失的问题同sigmoid。

3. 后来发展出的ReLU函数改进了梯度消失的问题，函数如图：

蒋豆芽



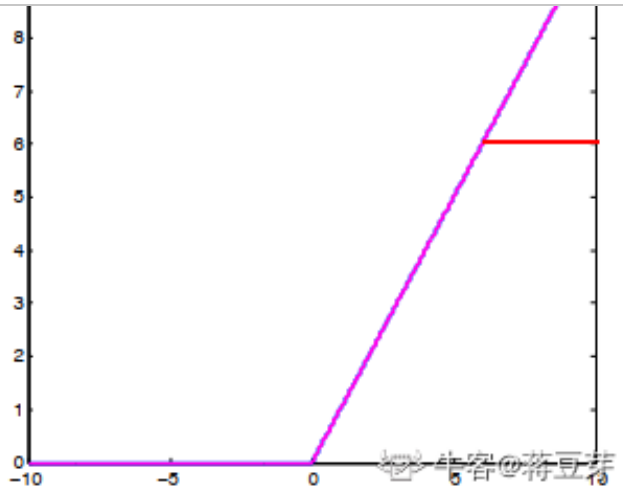
可以看到，ReLU在大于0时，斜率固定，这有助于网络训练。再看看梯度函数：



ReLU梯度稳定，值还比sigmoid大，所以**可以加快网络训练**。不过ReLU的缺点就是小于0的值不再得到响应，这又不利于网络训练。我们在输入图像时就要注意，应该使用Min-Max归一化，而不能使用Z-score归一化。

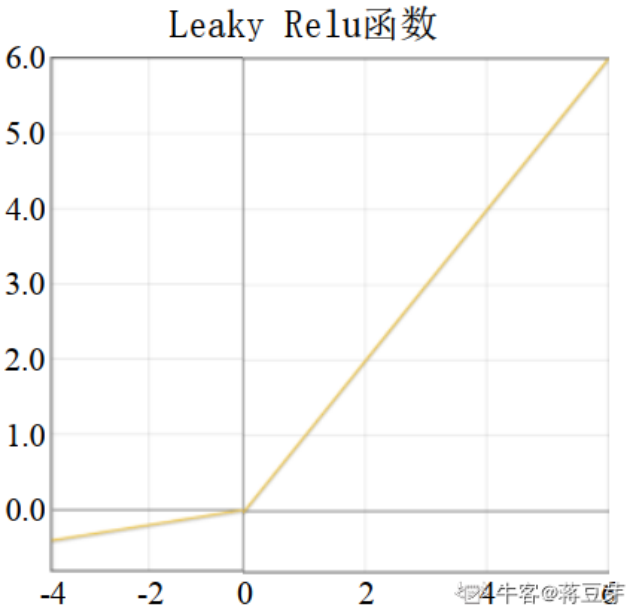
4. 大家可以注意到ReLU的正值输出为[0,无穷大]，关键是我们计算机内存有限，能存储无穷大的数吗？当然不能，所以将ReLU应用到实际中时需要限定输出的最大值，所以就成了ReLU6了，如图：

蒋豆芽



就是因为最大输出限定在6，所以称为ReLU6了。要是输出限定为豆芽，就应该叫ReLU-豆芽，就是这样，自己创造的东西，想怎么皮就怎么皮。哈哈

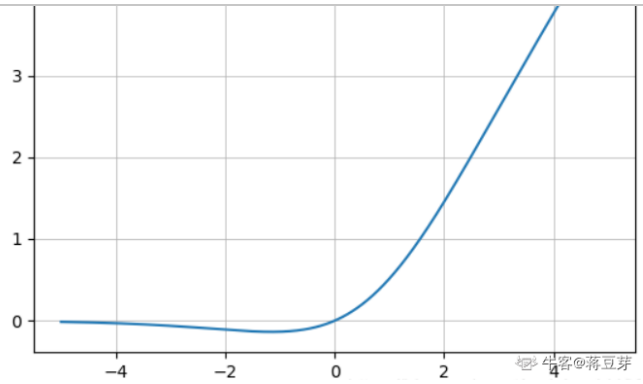
5. 然后解决ReLU的“死区”问题，Leaky ReLU函数如图：



可以看到，0的左边存在斜率，那么负值也可以得到响应，有利于网络学习到更多的信息。但是呢这个超参数不好设定，要根据不同的任务手动设置不同的参数。

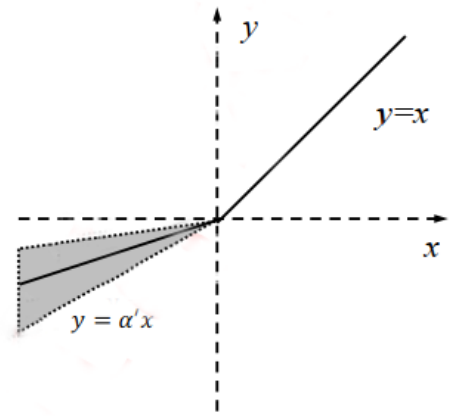
6. Mish函数，如图：

蒋豆芽



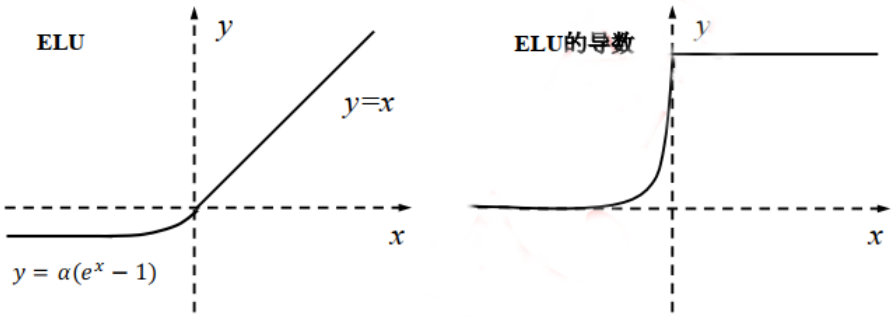
可以看出，在负值中，允许有一定的梯度流入。

7. 然后就有了参数化ReLU，让网络自己学习到超参数。
8. 然后是随机化ReLU，每个层都有自己的阿尔法参数，它们具有一定的分布，我们可以将其限定为均值分布或高斯分布，如图：



随机化ReLU 牛客@蒋豆芽

9. ELU函数，如图：

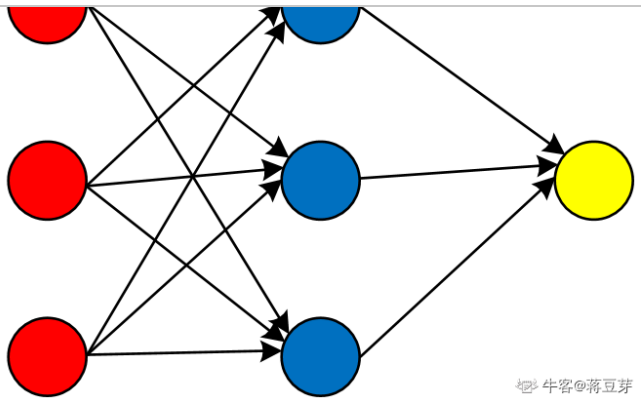


(a) 指数化线性单元 ELU。 (b) ELU 的导数 牛客@蒋豆芽

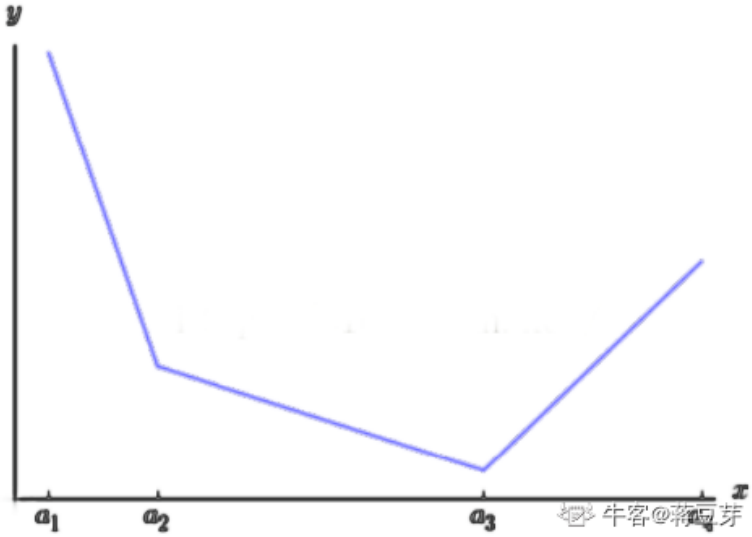
缺点计算量大。

10. Maxout。Maxout是深度学习网络中的一层网络，就像池化层、卷积层一样等，我们可以把maxout 看成是网络的激活函数层。如图：

蒋豆芽



从图中可以看出，Maxout就像一个层一样，k个节点就对应k个参数。这个层最终表现为一个分段函数。



类似的问题还有：

2. **说说你平时都用过什么激活函数，各自什么特点？** ☆☆☆☆☆

   参考回答

   回答参考上面。

   答案解析

   无。

3. **写一下leaky ReLU的公式，跟ReLU比有什么优势？** ☆☆☆☆☆

   参考回答

   回答参考上面。

蒋豆芽

4. **了解ReLU6吗？** ☆☆☆☆☆

   参考回答

   回答参考上面。

   答案解析

   无。

5. **sigmoid有什么缺点，有哪些解决办法？** ☆☆☆☆☆

   参考回答

   回答参考上面。

   答案解析

   无。

6. **relu在零点可导吗，不可导如何进行反向传播？** ☆☆☆☆☆

   参考回答

   不可导。

   人为将梯度规定为0.

   答案解析

```
caffe源码~/caffe/src/caffe/layers/relu_layer.cpp倒数第十行代码：

bottom_diff[i] = top_diff[i] * ((bottom_data[i] > 0)+ negative_slope * (bottom_data[i] <=
```

   可以清楚看到，默认情况下（negative_slope=0），间断点处（<=0）的导数认为是0.

7. **推导sigmoid求导公式** ☆☆☆☆☆

   参考回答

   sigmoid公式为：

蒋豆芽

那么求导推导如下：

$$\begin{aligned}\sigma'(x) &= \left(\frac{1}{1+e^{-z}}\right)' \\ &= \frac{1}{(1+e^{-z})^2}\cdot(e^{-z}) \\ &= \frac{1}{1+e^{-z}}\cdot\frac{e^{-z}}{1+e^{-z}} \\ &= \frac{1}{1+e^{-z}}\cdot\left(1-\frac{1}{1+e^{-z}}\right) \\ &= \sigma(z)(1-\sigma(z))\end{aligned} \tag{.}$$

答案解析

无。

8. **Softmax公式，溢出怎么处理**☆☆☆☆☆

参考回答

$$f(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}, j = 1, 2, \ldots, n$$

令 $M = max(x_i), i = 1, 2, \cdots, n$ ，即 M 为所有 x_i 中最大的值，那么我们只需要把计算 $f(x_i)$的值，改为计算 $f(x_i - M)$ 的值，就可以解决上溢出、下溢出的问题了，并且，计算结果理论上仍然和 $f(x_i)$ 保持一致。

操作类似**Min-Max归一化。**

答案解析

参考文章：https://www.cnblogs.com/guoyaohua/p/8900683.html

9. **Softmax公式求导**☆☆☆☆☆

参考回答

略。

答案解析

参考文章：https://blog.csdn.net/bqw18744018044/article/details/83120425

蒋豆芽

收藏 赞

相关专栏

机器学习面试题汇总与解析（蒋豆芽面试题总结）

27篇文章 | 90订阅

已订阅

0条评论 默认排序

没有回复

请留下你的观点吧~

发布