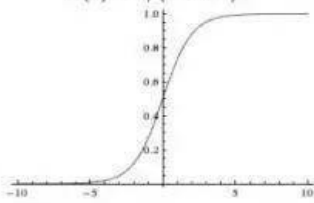
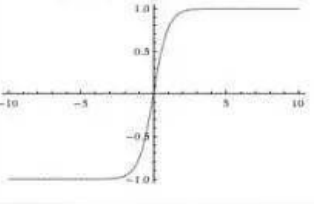
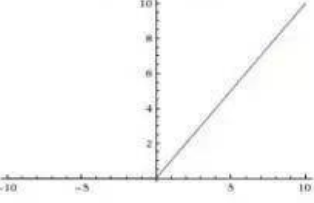
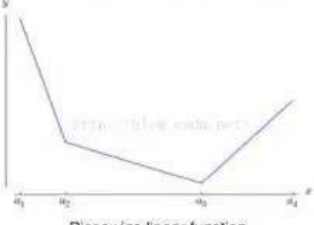


## 2. 激活函数有什么用？常见的激活函数的区别是什么？

激活函数作用：

如果不用激励函数（其实相当于激励函数是 $f(x) = x$ ），在这种情况下你每一层节点的输入都是上层输出的线性函数，很容易验证，无论你神经网络有多少层，输出都是输入的线性组合，与没有隐藏层效果相当，这种情况就是最原始的感知机（Perceptron）了，那么网络的逼近能力就相当有限。正因为上面的原因，我们决定引入非线性函数作为激励函数，这样深层神经网络表达能力就更加强大（不再是输入的线性组合，而是几乎可以逼近任意函数）。

| 激活函数       | 公式   | 缺点                                    | 优点                          |
|------------|--|---------------------------------------|-----------------------------|
| Sigmoid    | $\sigma(x) = 1/(1 + e^{-x})$<br>  | 1、会有梯度弥散<br>2、不是关于原点对称<br>3、计算exp比较耗时 | -                           |
| Tanh       | $\tanh(x) = 2\sigma(2x) - 1$<br>                                       | 梯度弥散没解决                               | 1、解决了原点对称问题<br>2、比sigmoid更快 |
| ReLU       | $f(x) = \max(0, x)$<br>   | 梯度弥散没完全解决，在（-）部分相当于神经元死亡而且不会复活        | 1、解决了部分梯度弥散问题<br>2、收敛速度更快   |
| Leaky ReLU | $f(x) = 1(x < 0)(\alpha x) + 1(x \geq 0)(x)$ -   | -                                     | 解决了神经死亡问题                   |
| Maxout     | $\max(w_1^T x + b_1, w_2^T x + b_2)$<br><br>Piecewise linear function | 参数比较多,本质上是在输出结果上又增加了一层                | 克服了ReLU的缺点,比较提倡使用           |

链接：<https://zhuanlan.zhihu.com/p/32610035>

怎么解决梯度消失问题

参考：

<https://blog.csdn.net/huangfei711/article/details/79865054>

什么是端到端学习:

参考:

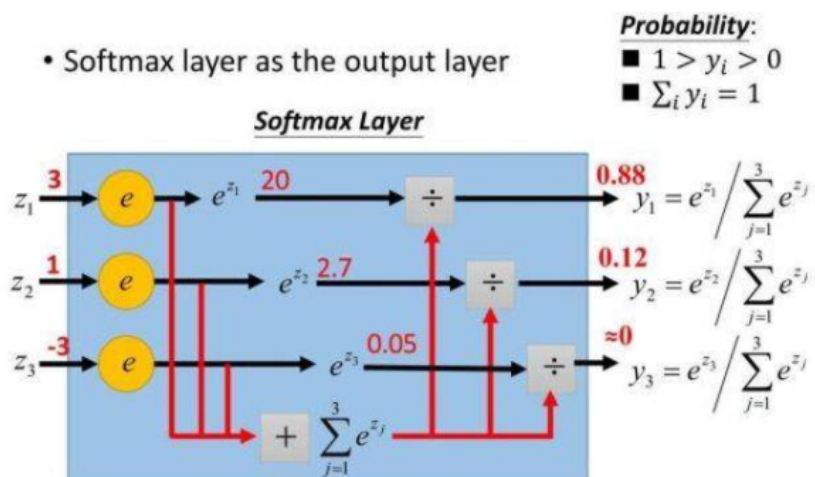
<https://blog.csdn.net/rechardchen123/article/details/86447621>

### 3.Softmax的原理是什么? 有什么作用?

Softmax用于多分类神经网络输出,目的是让大的更大。函数公式是

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}}$$

示意图如下:



Softmax是Sigmoid的扩展,当类别数 $k=2$ 时, Softmax回归退化为Logistic回归。

## 4.CNN的平移不变性是什么？如何实现的？

平移不变性（translation invariant）指的是CNN对于同一张图及其平移后的版本，都能输出同样的结果。这对于图像分类（image classification）问题来说肯定是最理想的，因为对于一个物体的平移并不应该改变它的类别。

卷积+最大池化约等于平移不变性。卷积：简单地说，图像经过平移，相应的特征图上的表达也是平移的。在神经网络中，卷积被定义为不同位置的特征检测器，也就意味着，无论目标出现在图像中的哪个位置，它都会检测到同样的这些特征，输出同样的响应。

池化：比如最大池化，它返回感受野中的最大值，如果最大值被移动了，但是仍然在这个感受野中，那么池化层也仍然会输出相同的最大值。这两种操作共同提供了一些平移不变性，即使图像被平移，卷积保证仍然能检测到它的特征，池化则尽可能地保持一致的表达。

链接：<https://zhangting2020.github.io/2018/05/30/Transform-Invariance/>

## 5.AlexNet, VGG, GoogLeNet, ResNet等网络之间的区别是什么？

网络结构对比：

| 模型名                          | AlexNet        | VGG            | GoogLeNet | ResNet  |
|------------------------------|----------------|----------------|-----------|---------|
| 初入江湖                         | 2012           | 2014           | 2014      | 2015    |
| 层数                           | 8              | 19             | 22        | 152     |
| Top-5错误                      | 16.4%          | 7.3%           | 6.7%      | 3.57%   |
| Data Augmentation            | +              | +              | +         | +       |
| Inception(NIN)               | -              | -              | +         | -       |
| 卷积层数                         | 5              | 16             | 21        | 151     |
| 卷积核大小                        | 11,5,3         | 3              | 7,1,3,5   | 7,1,3,5 |
| 全连接层数                        | 3              | 3              | 1         | 1       |
| 全连接层大小                       | 4096,4096,1000 | 4096,4096,1000 | 1000      | 1000    |
| Dropout                      | +              | +              | +         | +       |
| Local Response Normalization | +              | -              | +         | -       |
| Batch Normalization          | -              | -              | -         | +       |

特点：

AlexNet相比传统的CNN，主要改动包括Data Augmentation（数据增强）、Dropout 方法、激活函数用ReLU代替了传统的Tanh或者Logistic、Local Response Normalization（LRN，实际就是利用临近的数据做归一化）、Overlapping Pooling（有重叠，即Pooling的步长比Pooling Kernel的对应边要小）、多GPU并行。

VGG很好地继承了AlexNet的特点，但是网络更深。

GoogLeNet, 网络更深, 但主要的创新在于他的Inception, 这是一种网中网 (Network In Network) 的结构, 即原来的结点也是一个网络。相比于前述几个网络, 用了Inception之后整个网络结构的宽度和深度都可扩大, 能够带来2-3倍的性能提升。

ResNet在网络深度上有了进一步探索。但主要的创新在残差网络, 网络的提出本质上还是要解决层次比较深的时候无法训练的问题。这种借鉴了Highway Network思想的网络相当于旁边专门开个通道使得输入可以直达输出, 而优化的目标由原来的拟合输出  $H(x)$  变成输出和输入的差  $H(x)-x$ , 其中  $H(X)$  是某一层原始的期望映射输出,  $x$  是输入。

Pooling层是做什么的?

请参考: [https://blog.csdn.net/bobo\\_jiang/article/details/79080379](https://blog.csdn.net/bobo_jiang/article/details/79080379)

ROI pooling 的不足是什么?

## 二、ROI Pooling存在的问题

- 1、候选框从原图坐标映射到的feature Map坐标时, 位置坐标可能存在浮点数, 此时进行取整操作而出现第一次量化;
- 2、在ROI Pooling求取每个小网格的位置时也同样存在浮点数取整的情况, 此时进行取整操作出现第二次量化。

ROI align的具体做法是什么?

请参考: <https://www.cnblogs.com/ranjiewen/articles/8869703.html>

inception module的优点是什么?

请参考: <https://blog.csdn.net/u012193416/article/details/82622657>

附加题:

CV:

## 7.mini-Batch SGD相对于GD有什么优点

优点:

- 每次迭代计算量小, 对硬件算力要求低
- 可训练更大得数据集

NLP:

为什么Self-attention Model在长距离序列中如此强大?

分析：建议在对其它典型的神经网络（如卷积、循环等）结构比较熟悉的情况下，通过比较与这些结构的不同，来分析Self-attention机制的长距离捕捉能力。

案例1：这可能是因为Transformer的参数量很大，模型结构比较复杂，因此编码能力很强呗。（直接Pass【拍死】）

案例2：在self-Attention机制出现之前，我们一般用CNN或者RNN来处理文本，无论卷积还是循环神经网络其实都是对序列的一种局部编码，前者是基于N-gram的局部编码，而后者由于梯度消失等问题也只能建立短距离依赖。而在Self-attention Model，序列中所有输入元素之间并行地计算相关性，能够捕捉长距离的关系。（对比了CNN与RNN，回答地不错）

BI:

什么是wide&deep模型？

请参考: <https://blog.csdn.net/zero112535/article/details/109321147>