神经网络基础———激活函数

一.为什么要用激活函数

1.卷积神经网络的卷积层相当于一个线性函数，而单纯将多个卷基层组合到一起所起到的作用完全可以通过少数几个卷基层实现，不能达到效果；同时单纯用卷积层提取的特征是线性特征，而特征往往是非线性的（特征线性度的定义还不知道，但是从直观上看，特征应该是复杂的），故需要引入非线性来逐步提取非线性特征。

2.激活函数可以提取具有重要作用的神经元，忽视那些冗余的神经元，即引入了稀疏性。据研究表明大脑工作的时候，只有1%-4%的神经元在工作，稀疏性是具有很大意义的。而如何提取网络中具有代表作用的神经元便是关键。

3.由于神经网络反向传播的需要，激活函数一定要可导（只在某些点不可导是可以的，之后会继续研究反向传播算法）。

4.激活函数要保证是单调的，这样可以使得每一层都是凸函数，进而使得Loss尽可能近似与凸函数，免得陷入局部最优。

5.输出有限的激活函数训练起来更加稳定；输出无限的激活函数训练效率高，但是要选择较小的初始学习率。

二.激活函数的种类

1.sigmod函数.

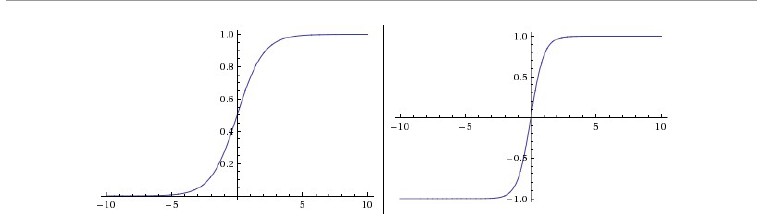


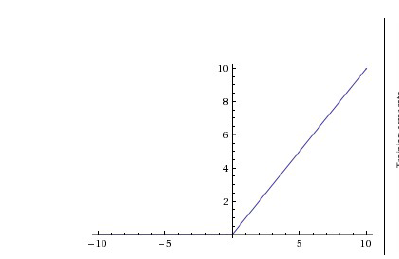
图1

如左图，sigmod函数将位于0附近的神经元激活，而将绝对值较大的神经元抑制，起到了筛选关键神经元的作用。缺点是当神经元的值较大时是被抑制的，但是正常来讲不应该是抑制的。而且输出的均值不是在0附近的。sigmod函数将神经元归一化到0-1之间，起到了数据归一化的作用（数据归一化为什么有好处将会在BN中研究）。

2.tanh函数.

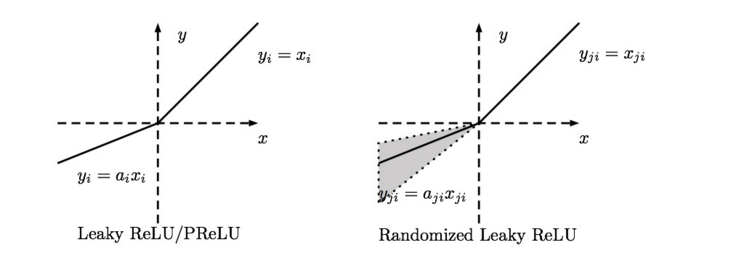
如图1右，相比于sigmod函数，其不同在于数值是正负都有，均值是0.其他的应该相似。

3.RELU函数.

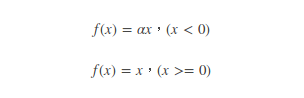


Relu函数如图，他将为负值的神经元彻底抑制，正的神经元保留原值。相当于它将数值是正的神经元提取为关键神经元，相比与sigmog更加合理。增加了网络的稀疏性。斜率固定，在反向传播的过程中具有优势。思考便是：将值是负的神经元抑制是否合理。若某一个神经元的梯度很大，但是恰好值是负的，那么通过Relu后该神经元便不具有响应的效果了。该方法相当于在数值的正负与梯度的大小之间选择了前者而完全忽视了后者。在使用的过程中一定要保证尽量不产生较大的梯度，以免造成梯度被错误的抑制。负值神经元肯定是包含一定有用信息的，之后对负值的添加形成了新的激活函数。

4.Leaky-Relu函数



如左图Leaky Relu



a是一个很小的数，是事先根据一定的条件定好的。相比与Relu其优点便是将负值并没有完全抑制而是尽量抑制。将负值缩小到很小的范围内。我们可以想既然是很小的负值，在以后的计算中的很可能一直保持这么小，对于数值的贡献很可能不是很大，除非是遇到了相当大的参数。而它的意义很可能便是在反向传播的过程中对于数值是负数但是梯度很大的神经元一个“重出江湖”的机会，故可以解决Relu的相应缺点，在数值和梯度之间之间起到平衡的作用。但是相比与Relu，明显的一点便是网络不够稀疏（对于稀疏的定义？），计算的复杂性会相对增加，但是可能由于负数的数值小，复杂性也不会提升得太多。

5.RRelu函数

如右图，在PRelu中参数a不再是固定的而是随机的从高斯分布中采样而获得的。



在测试阶段是将将所有的a取平均值。相当与在负数和梯度之间一取平均的方式找平衡。

6.PRelu函数

将a作为网络的一个参数，进行学习。如果神经网络实现a的共享，则相当于每个输出的feature map增加了一个元素，对于计算量的增加较小。以反向传播算法对其进行更新。相当于找到在负数和梯度之间之间更好的平衡。

7.maxout

maxout与卷积层、池化层一样是神经网络的层。对于输入它学习多个输出，之后取所有输出中最大的那个，显然其计算量会有很大程度的提升。

三.总结

激活函数的实质有两个：引入非线性和增加网络的稀疏性。

非线性可以有很多的方法，同时可以定义一种新的卷积内部的计算方式，实现非线性映射，不一定非要用激活函数，如maxout。但是想要增加网络的稀疏性要选择一个找出关键神经元的方法，目前的主流方式是以正负值和梯度的组合来实现评估。同时PRelu已经以学习的方式找到两者最好的平衡方式，但是网络的稀疏性有所降低。故如何找到一个性的评价关键神经元的方法、在稀疏性和因素的选择之间找好平衡是关键。

四.创新

我认为可以将Relu的思想和PRelu进行结合，在许多个PRelu后增加Relu来增加网络的稀疏性。