ReLu(Rectified Linear Units)激活函数

时间：2015-06-19 20:13:16　　阅读：60200　　评论：1　　收藏：0　　[点我收藏+]

标签：

ReLu(Rectified Linear Units)激活函数

论文参考：Deep Sparse Rectifier Neural Networks (很有趣的一篇paper）

起源：传统激活函数、脑神经元激活频率研究、稀疏激活性

传统Sigmoid系激活函数

传统神经网络中最常用的两个激活函数，Sigmoid系（Logistic-Sigmoid、Tanh-Sigmoid）被视为神经网络的核心所在。

从数学上来看，非线性的Sigmoid函数对中央区的信号增益较大，对两侧区的信号增益小，在信号的特征空间映

分享档案　更多>
2017年06月17日 (129)
2017年06月16日 (1496)
2017年06月15日 (1639)
2017年06月14日 (1677)
2017年06月13日 (1702)
2017年06月12日 (1513)
2017年06月11日 (1535)
2017年06月10日 (1690)
2017年06月09日 (1562)
2017年06月08日 (1638)

周排行　更多
1. Visual Studio 2015 的安装与使用 2015-10-24
2. 下了个蓝屏代码查看工具，就中病毒了。。。什

射上，有很好的效果。

从神经科学上来看，中央区酷似神经元的兴奋态，两侧区酷似神经元的抑制态，因而在神经网络学习方面，可以将重点特征推向中央区，将非重点特征推向两侧区。

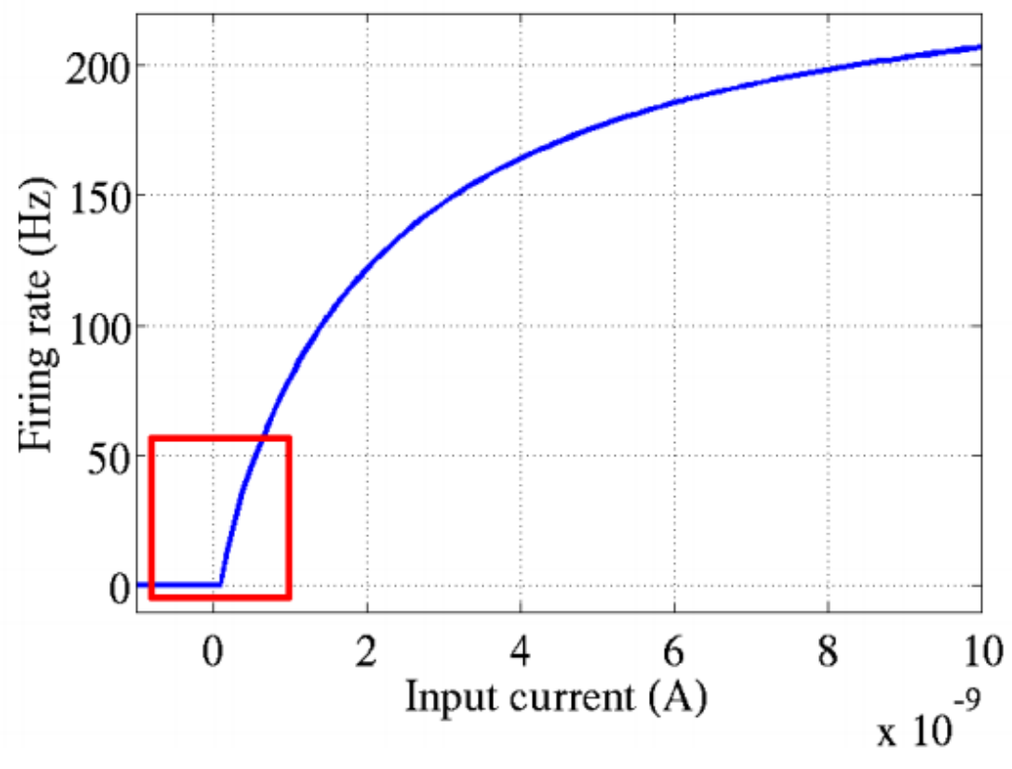
无论是哪种解释，看起来都比早期的线性激活函数(y=x),阶跃激活函数(-1/1,0/1)高明了不少。

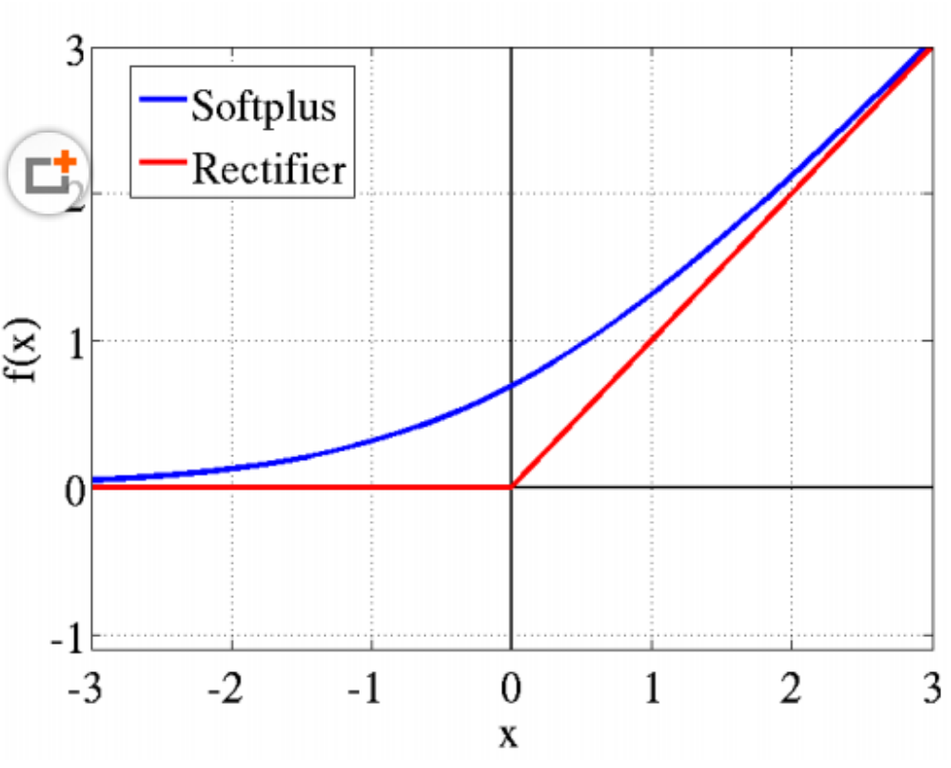
近似生物神经激活函数：Softplus&ReLu

2001年，神经科学家Dayan、Abott从生物学角度，模拟出了脑神经元接受信号更精确的激活模型，该模型如左图所示：

技术分享

技术分享





这个模型对比Sigmoid系主要变化有三点：**①单侧抑制 ②相对宽阔的兴奋边界 ③稀疏激活性**（重点，可以看到红框里前端状态完全没有激活）

同年，Charles Dugas等人在做**正数**回归预测论文中偶然使用了Softplus函数，Softplus函数是Logistic-Sigmoid函数原函数。

Softplus(x)=log(1+ex)

按照论文的说法，一开始想要使用一个指数函数（天然正数）作为激活函数来回归，但是到后期梯度实在太大，难以训练，于是加了一个log来减缓上升趋势。

偶然的是，同是2001年，ML领域的Softplus/Rectifier激活函数与神经科学领域的提出脑神经元激活频率函数有神似的地方，这促成了新的激活函数的研究。

生物神经的稀疏激活性

么鬼病毒，竟然还是用的VBS 2015-06-03
3. MUI事件管理 2016-07-13
4. 机器学习 — 监督学习和无监督学习的区别 2015-06-15
5. 优酷路由宝L1刷Breed和Padavan固件方法 2016-08-11
6. 爱奇艺、优酷、腾讯视频竞品分析报告2016（一） 2016-04-04
7. ASCLL表 2016-03-26
8. 【转】console.log 用法 2015-05-05
9. numpy的random模块 2015-03-10
10. 机器学习—逻辑回归理论简介 2015-03-06

在神经科学方面，除了新的激活频率函数之外，神经科学家还发现了神经元的稀疏激活性。

还是2001年，Attwell等人基于大脑能量消耗的观察学习上，推测神经元编码工作方式具有稀疏性和分布性。

2003年Lennie等人估测大脑同时被激活的神经元只有1~4%，进一步表明神经元工作的稀疏性。

从信号方面来看，即神经元同时只对输入信号的少部分选择性响应，大量信号被刻意的屏蔽了，这样可以提高学习的精度，更好更快地提取稀疏特征。

从这个角度来看，在经验规则的初始化W之后，传统的Sigmoid系函数同时近乎有一半的神经元被激活，这不符合神经科学的研究，而且会给深度网络训练带来巨大问题。

Softplus照顾到了新模型的前两点，却没有稀疏激活性。因而，校正函数max(0,x)成了近似符合该模型的最大赢家。

Part I：关于稀疏性的观点

Machine Learning中的颠覆性研究是稀疏特征，基于数据的稀疏特征研究上，派生了Deep Learning这一分支。

稀疏性概念最早由Olshausen、Field在1997年对信号数据稀疏编码的研究中引入，并最早在卷积神经网络中得以大施拳脚。

近年来，稀疏性研究不仅在计算神经科学、机器学习领域活跃，甚至信号处理、统计学也在借鉴。

总结起来稀疏性大概有以下三方面的贡献：

1.1 信息解离

当前，深度学习一个明确的目标是从数据变量中解离出关键因子。原始数据（以自然数据为主）中通常缠绕着高度密集的特征。原因

是这些特征向量是相互关联的，一个小小的关键因子可能牵扰着一堆特征，有点像蝴蝶效应，牵一发而动全身。

基于数学原理的传统机器学习手段在解离这些关联特征方面具有致命弱点。

然而，如果能够解开特征间缠绕的复杂关系，转换为稀疏特征，那么特征就有了鲁棒性（去掉了无关的噪声）。

1.2 线性可分性

稀疏特征有更大可能线性可分，或者对非线性映射机制有更小的依赖。因为稀疏特征处于高维的特征空间上（被自动映射了）

从流形学习观点来看（参见降噪自动编码器），稀疏特征被移到了一个较为纯净的低维流形面上。

线性可分性亦可参照天然稀疏的文本型数据，即便没有隐层结构，仍然可以被分离的很好。

1.3 稠密分布但是稀疏

稠密缠绕分布着的特征是信息最富集的特征，从潜在性角度，往往比局部少数点携带的特征成倍的有效。

而稀疏特征，正是从稠密缠绕区解离出来的，潜在价值巨大。

1.4 稀疏性激活函数的贡献的作用：

不同的输入可能包含着大小不同关键特征，使用大小可变的数据结构去做容器，则更加灵活。

假如神经元激活具有稀疏性，那么不同激活路径上：不同数量（选择性不激活）、不同功能（分布式激活），

两种可优化的结构生成的激活路径，可以更好地从有效的数据的维度上，学习到相对稀疏的特征，起到自动化解

离效果。

Part II：基于稀疏性的校正激活函数

2.1 非饱和线性端

撇开稀疏激活不谈，校正激活函数max(0,x)，与Softplus函数在兴奋端的差异较大(线性和非线性)。

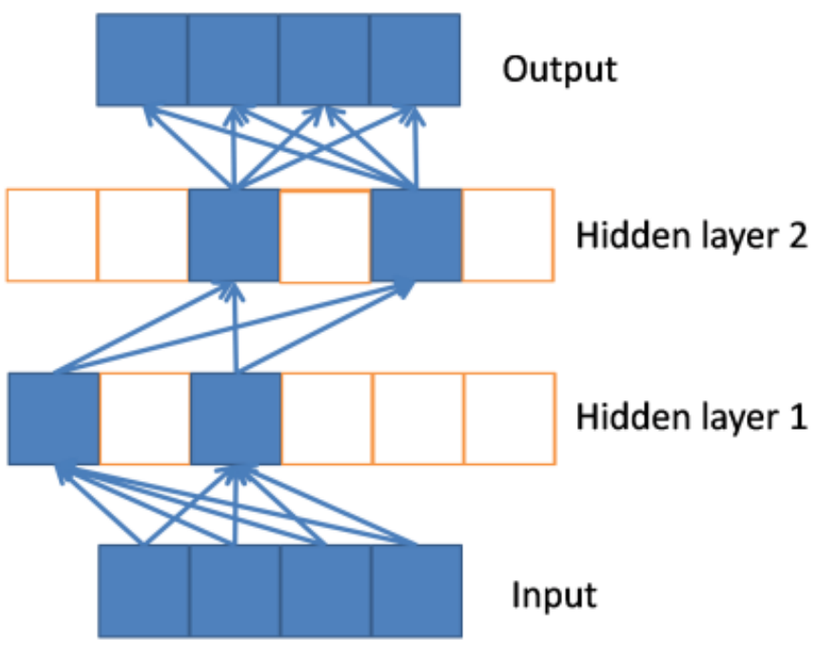
几十年的机器学习发展中，我们形成了这样一个概念：非线性激活函数要比线性激活函数更加先进。

尤其是在布满Sigmoid函数的BP神经网络，布满径向基函数的SVM神经网络中，往往有这样的幻觉，非线性函数对非线性网络贡献巨大。

该幻觉在SVM中更加严重。核函数的形式并非完全是SVM能够处理非线性数据的主力功臣（支持向量充当着隐层角色）。

那么在深度网络中，对非线性的依赖程度就可以缩一缩。另外，在上一部分提到，稀疏特征并不需要网络具有很强的处理线性不可分机制。

综合以上两点，在深度学习模型中，使用简单、速度快的线性激活函数可能更为合适。



如图，一旦神经元与神经元之间改为线性激活，网络的非线性部分仅仅来自于神经元部分选择性激活。

2.2 Vanishing Gradient Problem

更倾向于使用线性神经激活函数的另外一个原因是，减轻梯度法训练深度网络时的Vanishing Gradient Problem。

看过BP推导的人都知道，误差从输出层反向传播算梯度时，在各层都要乘当前层的输入神经元值，激活函数的一阶导数。

即Grad=Error·Sigmoid′(x)·x。使用双端饱和(即值域被限制)Sigmoid系函数会有两个问题：

①Sigmoid‘(x)∈(0,1) 导数缩放

②x∈(0,1)或x∈(-1,1) 饱和值缩放

这样，经过每一层时，Error都是成倍的衰减，一旦进行递推式的多层的反向传播，梯度就会不停的衰减，消失，使得网络学习变慢。

而校正激活函数的梯度是1，且只有一端饱和，梯度很好的在反向传播中流动，训练速度得到了很大的提高。

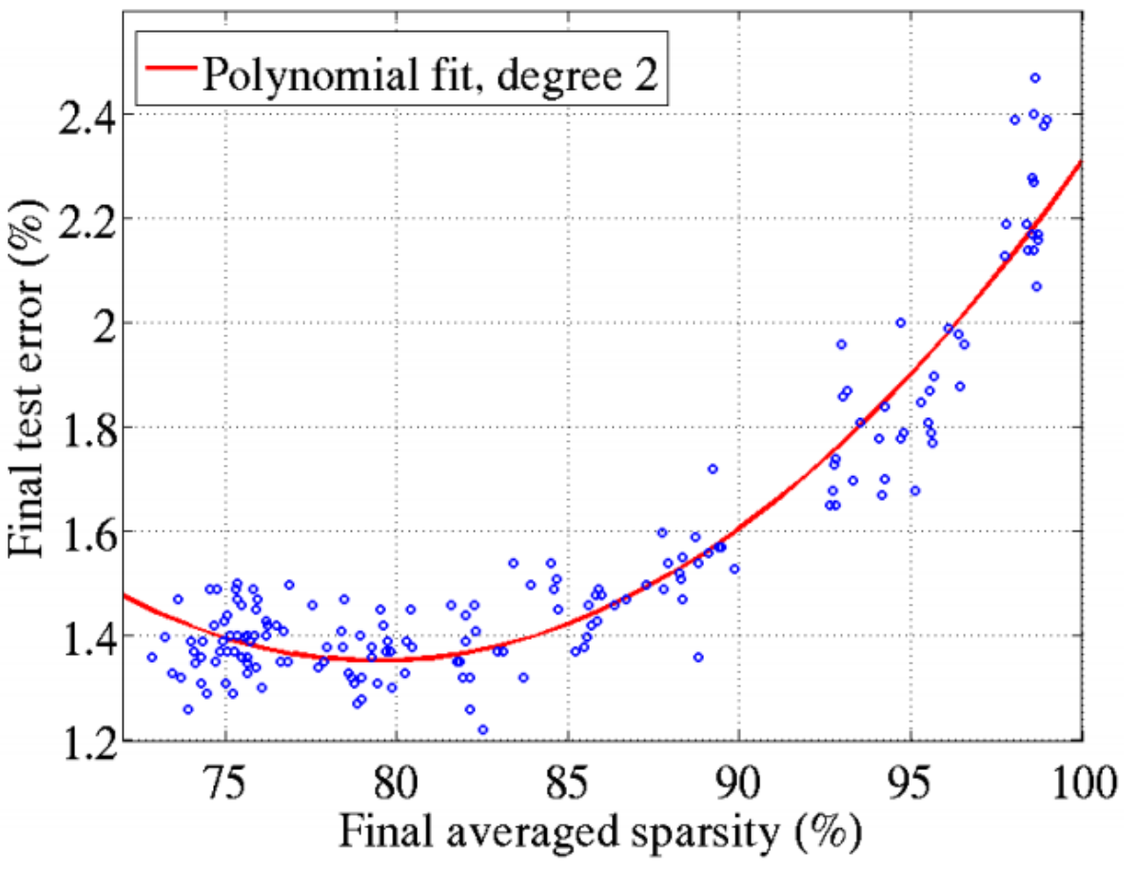
Softplus函数则稍微慢点，Softplus‘(x)=Sigmoid(x)∈(0,1) ，但是也是单端饱和，因而速度仍然会比Sigmoid系函数快。

Part III 潜在问题

强制引入稀疏零的合理性?

诚然，稀疏性有很多优势。但是，过分的强制稀疏处理，会减少模型的有效容量。即特征屏蔽太多，导致模型无法学习到有效特征。

论文中对稀疏性的引入度做了实验，理想稀疏性（强制置0）比率是70%~85%。超过85%，网络就容量就成了问题，导致错误率极高。



对比大脑工作的95%稀疏性来看，现有的计算神经网络和生物神经网络还是有很大差距的。

庆幸的是，ReLu只有负值才会被稀疏掉，即引入的稀疏性是可以训练调节的，是动态变化的。

只要进行梯度训练，网络可以向误差减少的方向，自动调控稀疏比率，保证激活链上存在着合理数量的非零值。

Part IV ReLu的贡献

4.1 缩小做和不做非监督预训练的代沟

ReLu的使用，使得网络可以自行引入稀疏性。这一做法，等效于无监督学习的预训练。

Neuron	MNIST	CIFAR10	NISTP	NORB
With **unsupervised pre-training**				
Rectifier	**1.20%**	**49.96%**	**32.86%**	**16.46%**
Tanh	**1.16%**	**50.79%**	35.89%	17.66%
Softplus	**1.17%**	**49.52%**	**33.27%**	19.19%
Without **unsupervised pre-training**				
Rectifier	**1.43%**	**50.86%**	**32.64%**	**16.40%**
Tanh	1.57%	52.62%	36.46%	19.29%
Softplus	1.77%	53.20%	35.48%	17.68%

当然，效果肯定没预训练好。论文中给出的数据显示，没做预训练情况下，ReLu激活网络遥遥领先其它激活函数。

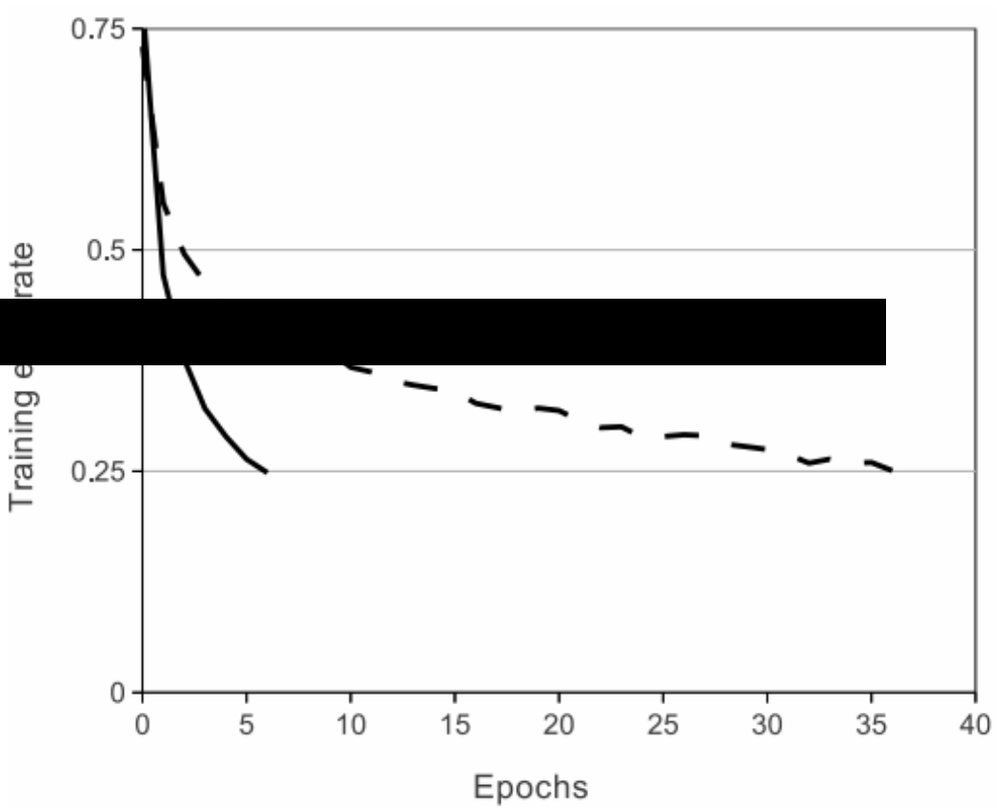
甚至出现了比普通激活函数预训练后更好的奇葩情况。当然，在预训练后，ReLu仍然有提升空间。

从这一层面来说，ReLu缩小了非监督学习和监督学习之间的代沟。当然，还有更快的训练速度。

4.2 更快的特征学习

在MNIST+LeNet4中，ReLu+Tanh的组合在epoch 50左右就能把验证集错误率降到1.05%

但是，全Tanh在epoch 150时，还是1.37%，这个结果ReLu+Tanh在epoch 17时就能达到了。



该图来自AlexNet的论文对ReLu和普通Sigmoid系函数做的对比测试，可以看到，ReLu的使用，使得学习周期

大大缩短。综合速率和效率，DL中大部分激活函数应该选择ReLu。

Part V Theano中ReLu的实现

ReLu可以直接用T.maximum(0,x)实现，用T.max(0,x)不能求导.

Part VI ReLu训练技巧

见Cifar-10训练技巧

标签：

赞	踩
(64)	(3)

举报

评论

一句话评论（**0**）

共**0**条

登录后才能评论！ 登录

友情链接

兰亭集智 国之画 百度统计 站长统计 阿里云 chrome插件