深度学习中的激活函数导引

原创 2016-08-01 李扬 深度学习大讲堂

摘要

近年来，深度学习在计算机视觉领域取得了引人注目的成果，其中一个重要因素是激活函数的发展。新型激活函数ReLU克服了梯度消失，使得深度网络的直接监督式训练成为可能。本文将对激活函数的历史和近期进展进行总结和概括。

激活函数的定义与作用

在人工神经网络中，神经元节点的激活函数定义了对神经元输出的映射，简单来说，神经元的输出（例如，全连接网络中就是输入向量与权重向量的内积再加上偏置项）经过激活函数处理后再作为输出。加拿大蒙特利尔大学的Bengio教授在 ICML 2016 的文章[1]中给出了激活函数的定义：激活函数是映射 h:R→R，且几乎处处可导。

神经网络中激活函数的主要作用是提供网络的非线性建模能力，如不特别说明，激活函数一般而言是非线性函数。假设一个示例神经网络中仅包含线性卷积和全连接运算，那么该网络仅能够表达线性映射，即便增加网络的深度也依旧还是线性映射，难以有效建模实际环境中非线性分布的数据。加入（非线性）激活函数之后，深度神经网络才具备了分层的非线性映射学习能力。因此，激活函数是深度神经网络中不可或缺的部分。

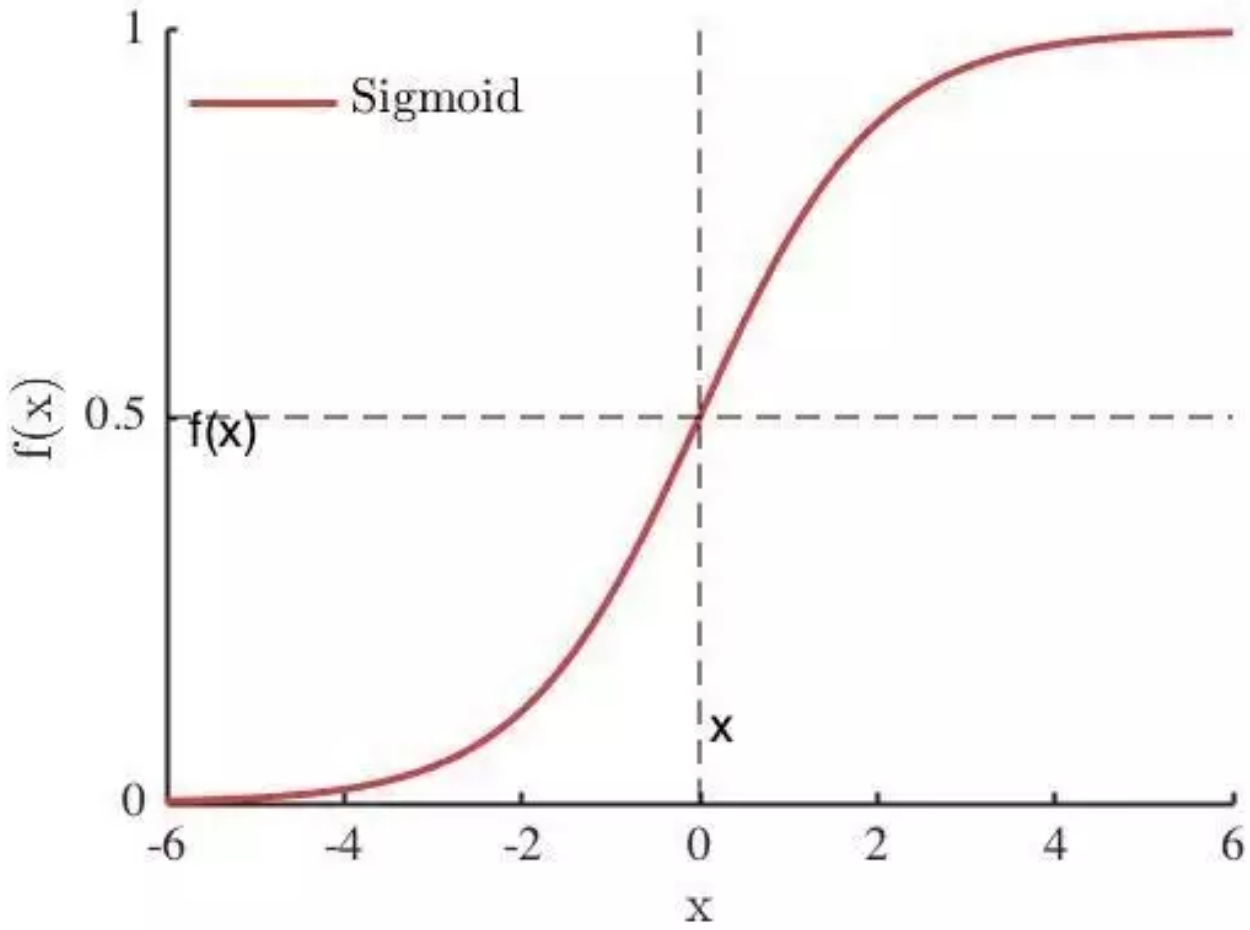
激活函数的历史发展与近期进展

从定义来看，几乎所有的连续可导函数都可以用作激活函数。但目前常见的多是分段线性和具有指数形状的非线性函数。下文将依次对它们进行总结。

Sigmoid

Sigmoid 是使用范围最广的一类激活函数，具有指数函数形状 。正式定义为：

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



可见，sigmoid 在定义域内处处可导，且两侧导数逐渐趋近于0，即：

$$\lim_{x\to\infty} f'(x)=0$$

Bengio 教授等[1]将具有这类性质的激活函数定义为软饱和激活函数。与极限的定义类似，饱和也分为左饱和与右饱和：
左饱和：

$$\lim_{x\to-\infty} f'(x)=0$$

右饱和：

$$\lim_{x \to +\infty} f'(x) = 0$$

与软饱和相对的是硬饱和激活函数，即：
f'(x)=0，当 |x| > c，其中 c 为常数。

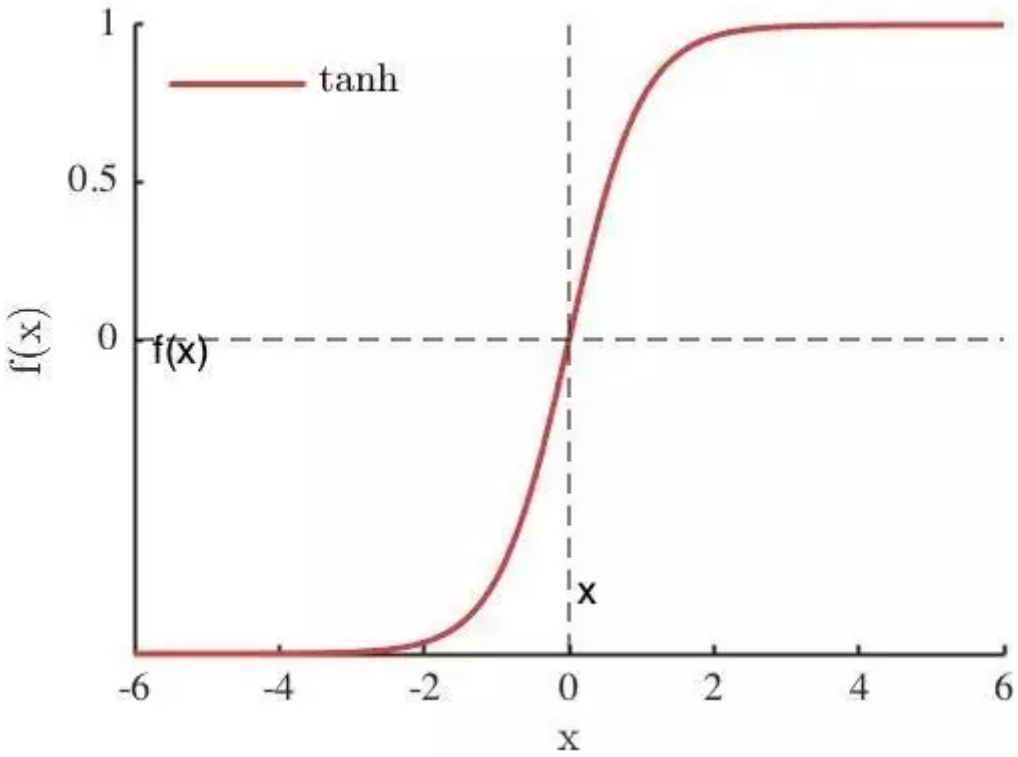
同理，硬饱和也分为左饱和和右饱和。常见的 ReLU 就是一类左侧硬饱和激活函数。

Sigmoid 的软饱和性，使得深度神经网络在二三十年里一直难以有效的训练，是阻碍神经网络发展的重要原因。具体来说，由于在后向传递过程中，sigmoid向下传导的梯度包含了一个f'(x) 因子（ sigmoid关于输入的导数），因此一旦输入落入饱和区，f'(x) 就会变得接近于0，导致了向底层传递的梯度也变得非常小。此时，网络参数很难得到有效训练。这种现象被称为梯度消失。一般来说， sigmoid 网络在 5 层之内就会产生梯度消失现象[2]。梯度消失问题至今仍然存在，但被新的优化方法有效缓解了，例如DBN中的分层预训练，Batch Normalization的逐层归一化，Xavier和MSRA权重初始化等代表性技术。

Sigmoid 的饱和性虽然会导致梯度消失，但也有其有利的一面。例如它在物理意义上最为接近生物神经元。 (0, 1) 的输出还可以被表示作概率，或用于输入的归一化，代表性的如Sigmoid交叉熵损失函数

tanh

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



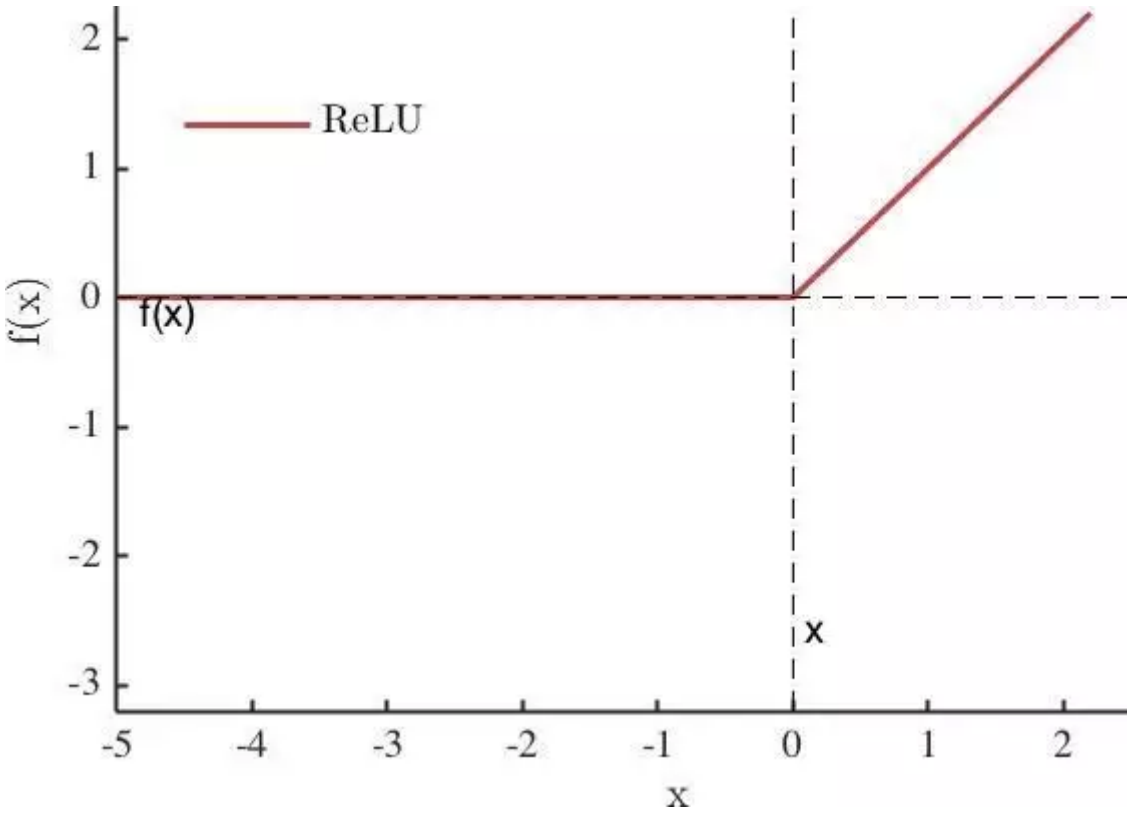
可见，tanh(x)=2sigmoid(2x)-1，也具有软饱和性。Xavier在文献[2]中分析了sigmoid与tanh的饱和现象及特点，具体见原论文。此外，文献 [3] 中提到tanh 网络的收敛速度要比sigmoid快。因为 tanh 的输出均值比 sigmoid 更接近 0，SGD会更接近 natural gradient[4]（一种二次优化技术），从而降低所需的迭代次数。

ReLU

虽然2006年Hinton教授提出通过分层无监督预训练解决深层网络训练困难的问题，但是深度网络的直接监督式训练的最终突破，最主要的原因是采用了新型激活函数ReLU[5, 6]。与传统的sigmoid激活函数相比，ReLU能够有效缓解梯度消失问题，从而直接以监督的方式训练深度神经网络，无需依赖无监督的逐层预训练，这也是2012年深度卷积神经网络在ILSVRC竞赛中取得里程碑式突破的重要原因之一。

ReLU的 正式定义为：

$$y = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$



可见，ReLU 在x<0 时硬饱和。由于 x>0时导数为 1，所以，ReLU 能够在x>0时保持梯度不衰减，从而缓解梯度消失问题。但随着训练的推进，部分输入会落入硬饱和区，导致对应权重无法更新。这种现象被称为“神经元死亡”。

ReLU还经常被“诟病”的一个问题是输出具有偏移现象[7]，即输出均值恒大于零。偏移现象和神经元死亡会共同影响网络的收敛性。本文作者公开在arxiv的文章[8]中的实验表明，如果不采用 Batch Normalization，即使用 MSRA 初始化30层以上的ReLU网络，最终也难以收敛。相对的，PReLU和ELU网络都能顺利收敛，这两种改进的激活函数将在后面介绍。实验所用代码见 https://github.com/Coldmooon/Code-for-MPELU/ 。

ReLU另外一个性质是提供神经网络的稀疏表达能力，在Bengio教授的Deep Sparse Rectifier Neural Network[6]一文中被认为是ReLU带来网络性能提升的原因之一。但后来的研究发现稀疏性并非性能提升的必要条件，文献 RReLU [9]也指明了这一点。

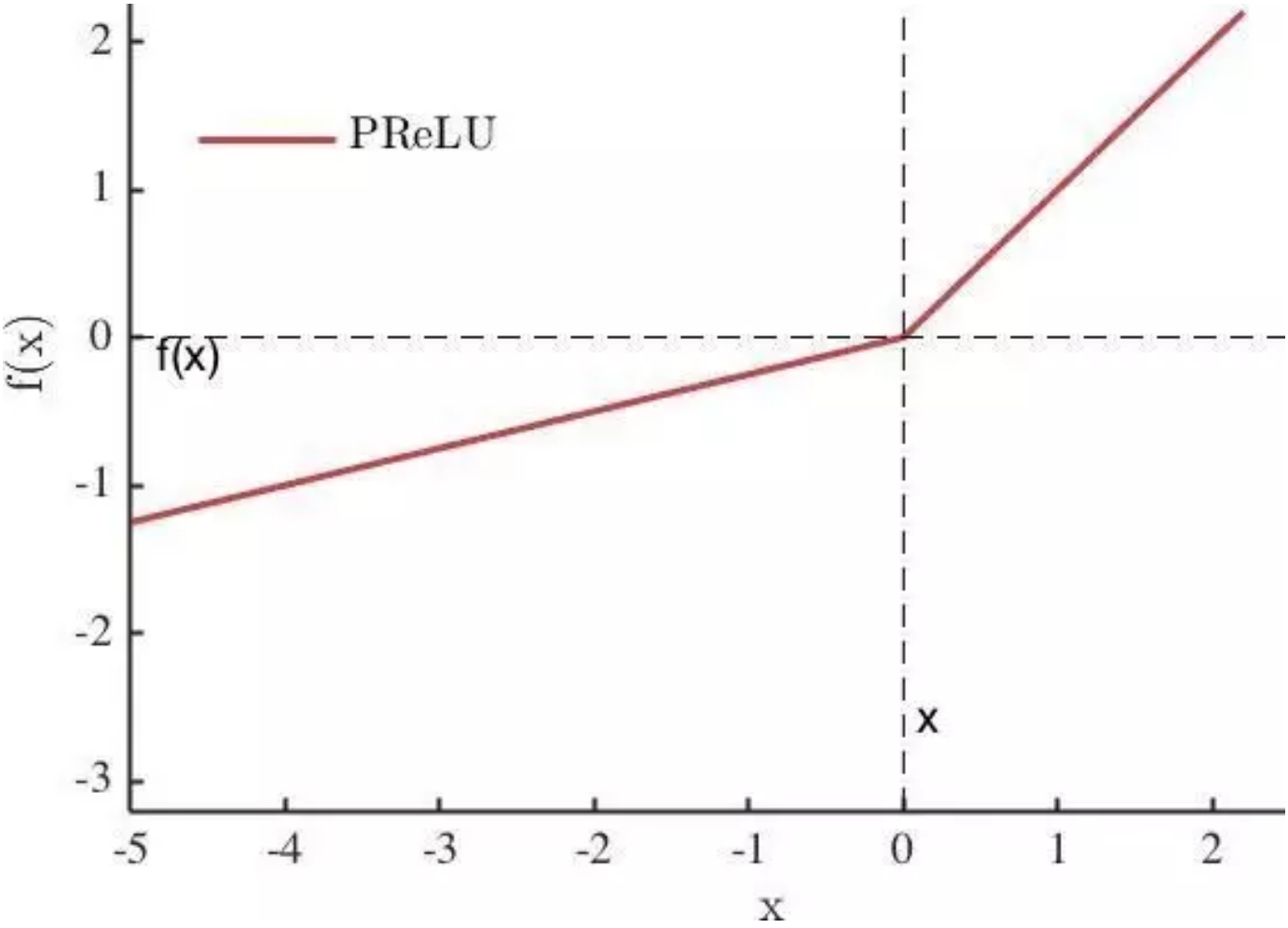
PReLU[10]、ELU[7]等激活函数不具备这种稀疏性，但都能够提升网络性能。本文作者在文章[8]中给出了一些实验比较结果。首先，在cifar10上采用NIN网络，实验结果为 PReLU > ELU > ReLU，稀疏性并没有带来性能提升。其次，在 ImageNet上采用类似于[11] 中model E的15 层网络，实验结果则是ReLU最好。为了验证是否是稀疏性的影响，以 LReLU [12]为例进一步做了四次实验，负半轴的斜率分别为1，0.5，0.25，0.1，需要特别说明的是，当负半轴斜率为1时，LReLU退化为线性函数，因此性能损失最大。实验结果展现了斜率大小与网络性能的一致性。综合上述实验可知，ReLU的稀疏性与网络性能之间并不存在绝对正负比关系。

LReLU 斜率 a	ImageNet top-1 分类精度(%)
0	62.34
0.1	62.08
0.25	61.46
0.5	57.24
1	39.73

PReLU

PReLU [10]是ReLU 和 LReLU的改进版本，具有非饱和性：

$$f(y_i) = \begin{cases} y_i, & \text{if } y_i > 0 \\ a_i y_i, & \text{if } y_i \leq 0 \end{cases}.$$



与LReLU相比，PReLU中的负半轴斜率a可学习而非固定。原文献建议初始化a为0.25，不采用正则。个人认为，是否采用正则应当视具体的数据库和网络，通常情况下使用正则能够带来性能提升。

虽然PReLU 引入了额外的参数，但基本不需要担心过拟合。例如，在上述cifar10+NIN实验中，PReLU比ReLU和ELU多引入了参数，但也展现了更优秀的性能。所以实验中若发现网络性能不好，建议从其他角度寻找原因。

与ReLU相比，PReLU收敛速度更快。因为PReLU的输出更接近0均值，使得SGD更接近natural gradient。证明过程参见原文[10]。

此外，作者在ResNet 中采用ReLU，而没有采用新的PReLU。这里给出个人浅见，不一定正确，仅供参考。首先，在上述LReLU实验中，负半轴斜率对性能的影响表现出一致性。对PReLU采用正则将激活值推向0也能够带来性能提升。这或许表明，小尺度或稀疏激活值对深度网络的影响更大。其次，ResNet中包含单位变换和残差两个分支。残差分支用于学习对单位变换的扰动。如果单位变换是最优解，那么残差分支的扰动应该越小越好。这种假设下，小尺度或稀疏激活值对深度网络的影响更大。此时，ReLU或许是比PReLU更好的选择。

RReLU

数学形式与PReLU类似，但RReLU[9]是一种非确定性激活函数，其参数是随机的。这种随机性类似于一种噪声，能够在一定程度上起到正则效果。作者在cifar10/100上观察到了性能提升。

Maxout

Maxout[13]是ReLU的推广，其发生饱和是一个零测集事件（measure zero event）。正式定义为：

$$\max(w_1^T x + b_1, w_2^T x + b_2, \ldots, w_n^T x + b_n)$$

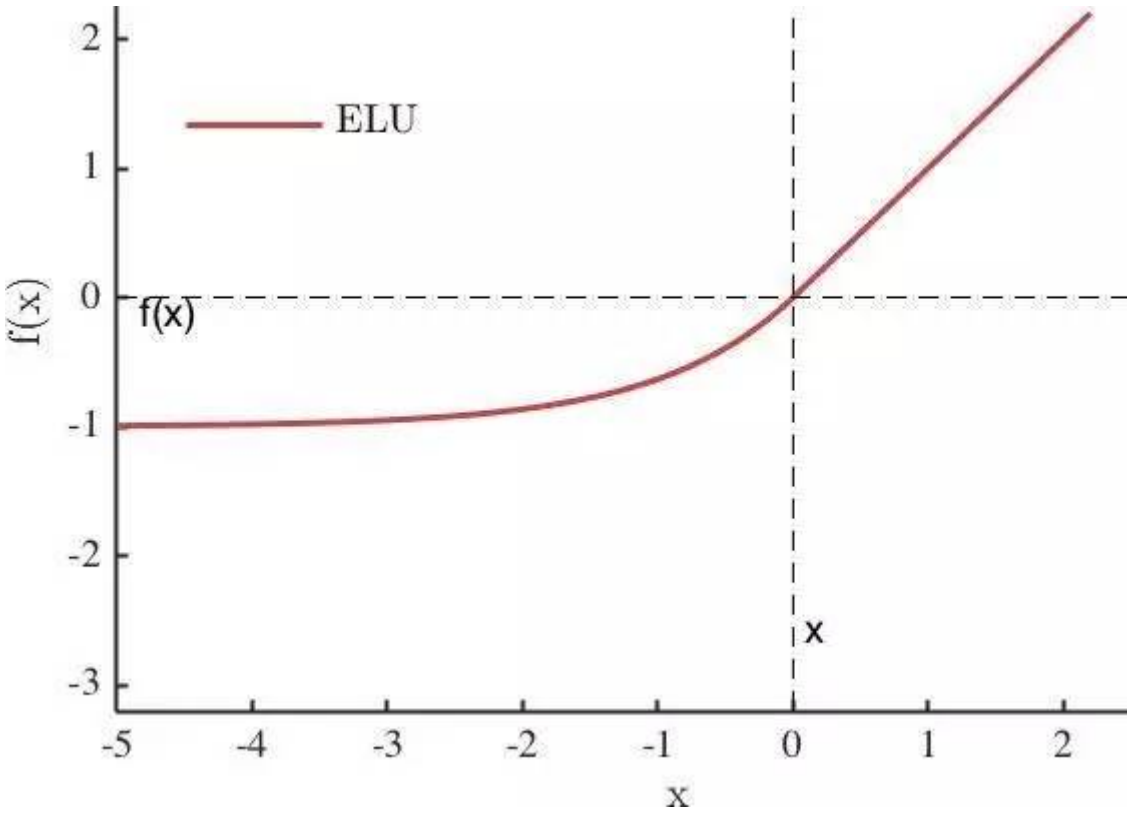
Maxout网络能够近似任意连续函数，且当w2,b2,...,wn,bn为0时，退化为ReLU。 其实，Maxout的思想在视觉领域存在已久。例如，在HOG特征里有这么一个过程：计算三个通道的梯度强度，然后在每一个像素位置上，仅取三个通道中梯度强度最大的数值，最终形成一个通道。这其实就是Maxout的一种特例。

Maxout能够缓解梯度消失，同时又规避了ReLU神经元死亡的缺点，但增加了参数和计算量。

ELU

ELU[7]融合了sigmoid和ReLU，具有左侧软饱性。其正式定义为：

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha\left(\exp(x) - 1\right) & \text{if } x \leq 0 \end{cases}$$

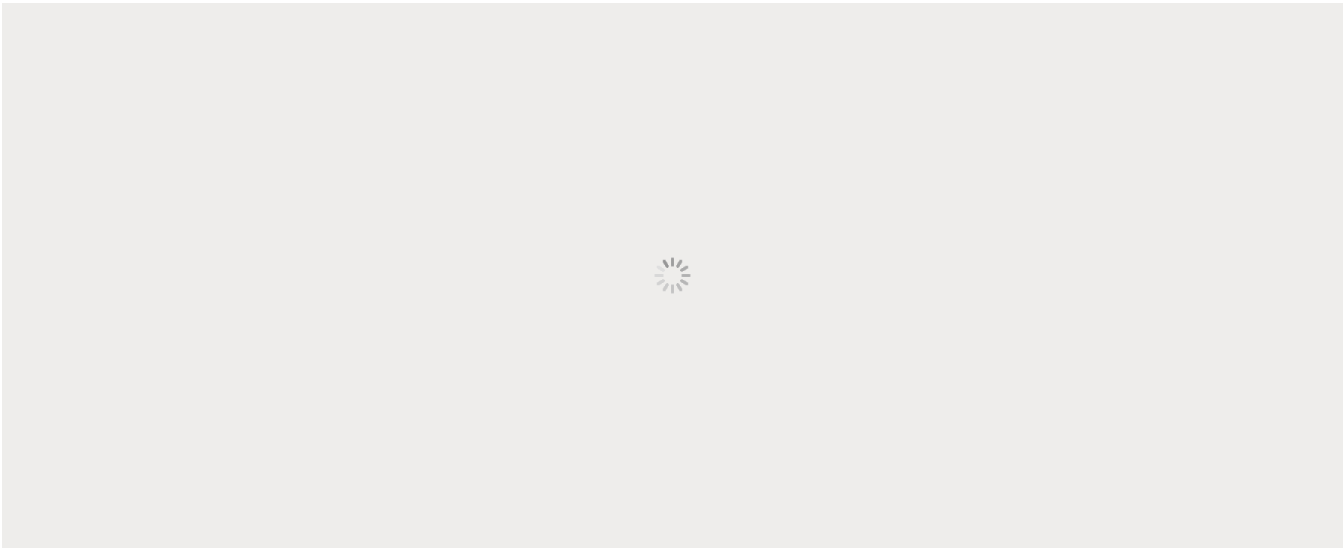


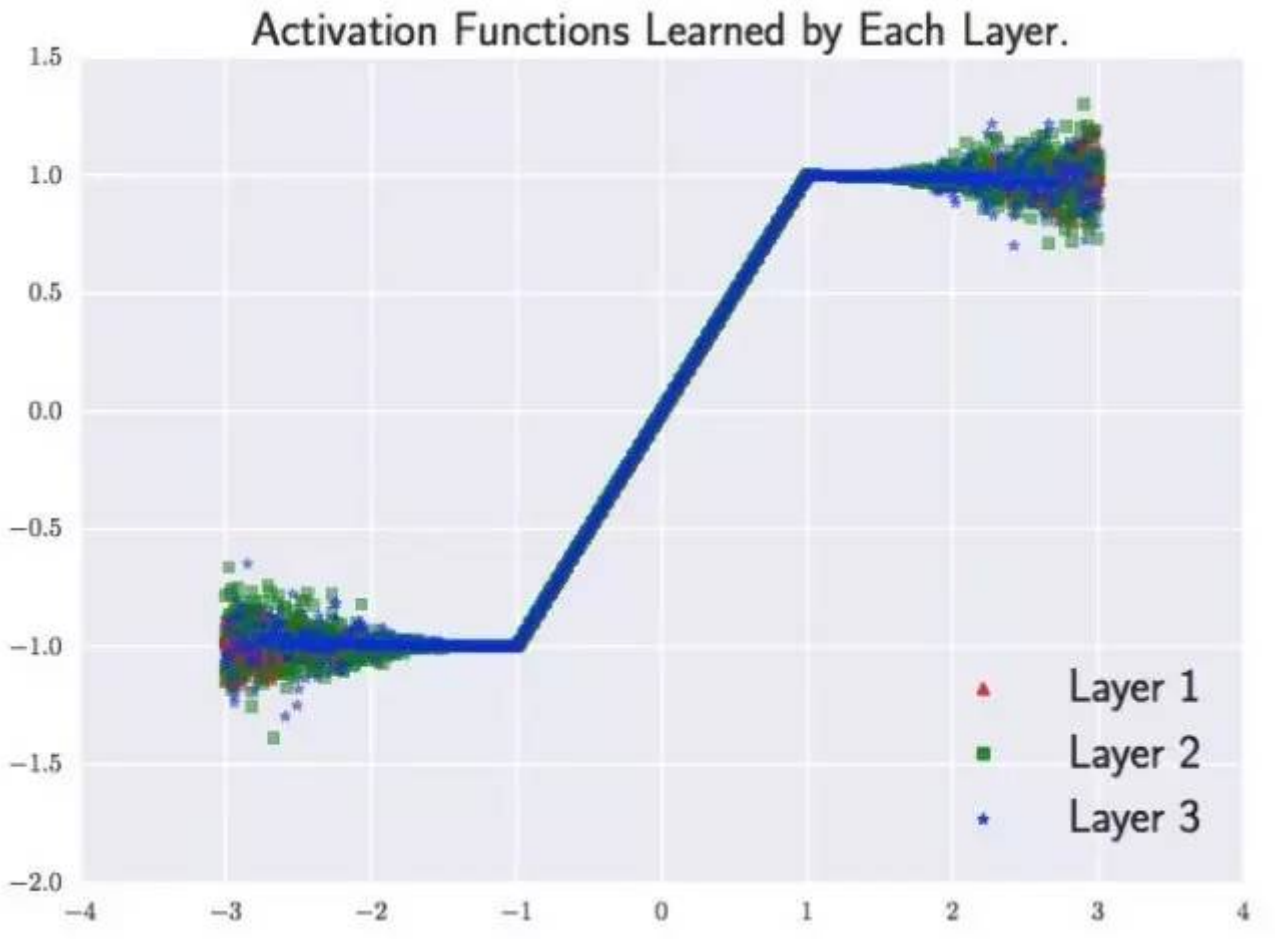
右侧线性部分使得ELU能够缓解梯度消失，而左侧软饱能够让ELU对输入变化或噪声更鲁棒。ELU的输出均值接近于零，所以收敛速度更快。经本文作者实验，ELU的收敛性质的确优于ReLU和PReLU。在cifar10上，ELU 网络的loss 降低速度更快；在 ImageNet上，不加 Batch Normalization 30 层以上的 ReLU 网络会无法收敛，PReLU网络在MSRA的Fan-in （ caffe ）初始化下会发散，而 ELU 网络在Fan-in/Fan-out下都能收敛 。实验代码见 https://github.com/Coldmooon/Code-for-MPELU/。

论文的另一个重要贡献是分析了Bias shift 现象与激活值的关系，证明了降低Bias shift 等价于把激活值的均值推向0。

Noisy Activation Functions

engio教授在ICML 2016 提出了一种激活策略[1]，可用于多种软饱和激活函数，例如 sigmoid和tanh。





当激活函数发生饱和时， 网络参数还能够在两种动力下继续更新：正则项梯度和噪声梯度。引入适当的噪声能够扩大SGD的参数搜索范围，从而有机会跳出饱和区。在激活函数中引入噪声的更早工作可追溯到[5]，但文献[5]的工作并不考虑噪声引入的时间和大小。本篇的特点在于，只在饱和区才引入噪声，且噪声量与饱和程度相关——原式与泰勒展开式一次项之差 δ。算法1中g表示sigmoid，用于归一化 δ。注意，ReLU的 δ恒为0，无法直接加噪声，所以作者把噪声加在了输入上。

CReLU

CReLU [14]是Wenling Shang 发表在 ICML 2016的工作，本篇同样提出了一种激活策略:

$$\forall x \in \mathbb{R}, \rho_c(x) \triangleq ([x]_+, [-x]_+).$$

其中，[] 表示 ReLU（其他亦可）。

作者在观察第一层滤波器（filter）时发现，滤波器相位具有成对现象（pair-grouping phenomenon）。这一发现揭示了网络的底层学到了一些冗余滤波器来提取输入的正负相位信息的可能性。因此可以考虑采用适当的操作移除这些冗余滤波器。对此，作者提出了CReLU，将激活函数的输入额外做一次取反，等价于将输入相位旋转180°。这种策略可以看作在网络中加入相位的先验。实验在cifar10上观察到能以更少的参数获得性能提升。

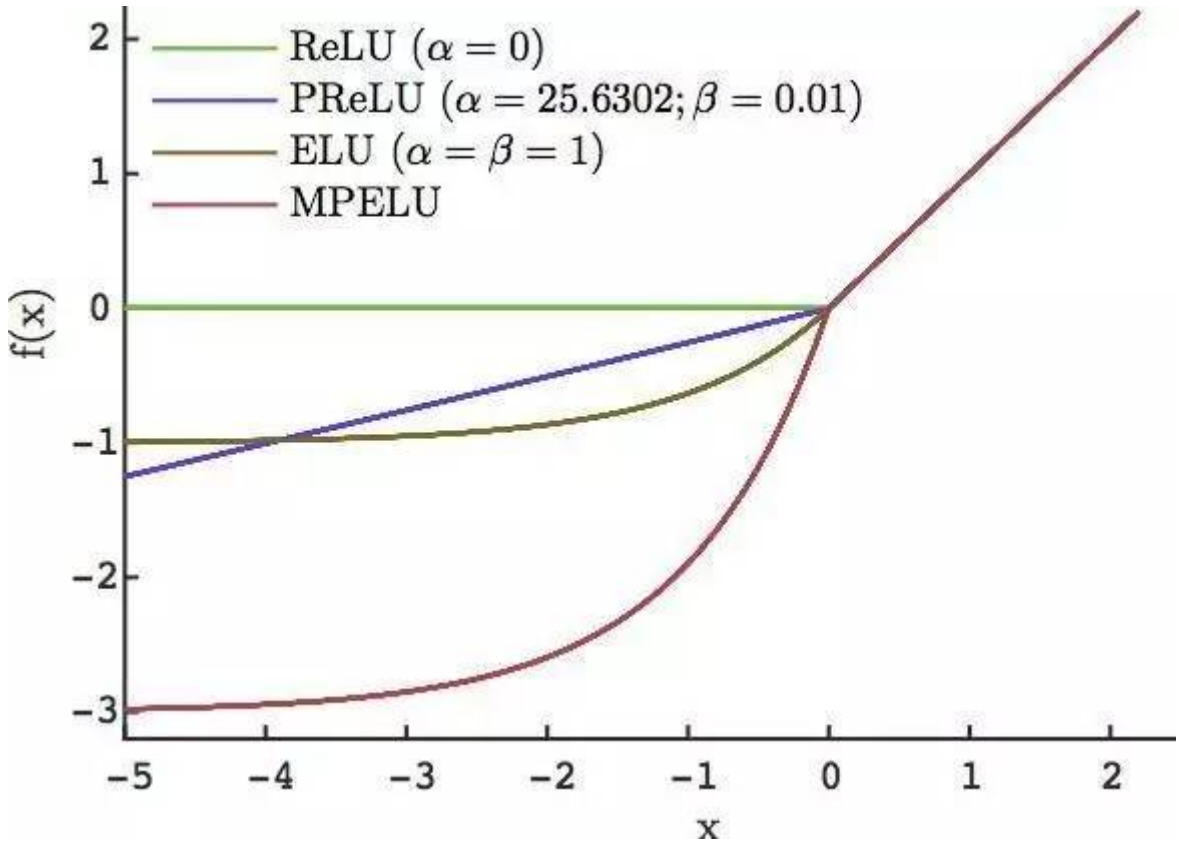
Model	CIFAR-10				CIFAR-100				params.
	Single		Average	Vote	Single		Average	Vote	
	train	test			train	test			
Baseline	1.09	9.17	10.20±0.09	7.55	13.68	36.30	38.52±0.12	31.26	1.4M
+ (double)	0.47	8.65	9.87±0.09	7.28	6.03	34.77	36.73±0.15	28.34	5.6M
AVR	4.10	**8.32**	10.26±0.10	7.76	19.35	35.00	37.24±0.20	29.77	1.4M
CReLU	4.23	8.43	**9.39**±0.11	**7.09**	14.25	**31.48**	**33.76**±0.12	**27.60**	2.8M
+ (half)	4.73	**8.37**	**9.44**±0.09	**7.09**	21.01	33.68	36.20±0.18	29.93	0.7M

使用CReLU时，要有意识的将滤波器数量减半，否则， 网络参数变为2倍。

MPELU

MPELU[8]是我们组的工作，将分段线性与ELU统一到了一种形式下。在NIN+CIFAR10，本文作者发现ELU与LReLU性能一致，而与PReLU差距较大。经过分析，ELU泰勒展开的一次项就是LReLU。当在ELU前加入BN让输入集中在0均值附近， 则ELU与LReLU之差——泰勒展开高次项会变小，粗略估计，约55.57%的激活值误差小于0.01。因此，受PReLU启发，令α可学习能够提高性能。此外，引入参数β能够进一步控制ELU的函数形状。正式定义为：

$$f(y_i) = \begin{cases} y_i & if\ y_i > 0 \\ \alpha_c(e^{\beta_c y_i} - 1) & if\ y_i \leqslant 0 \end{cases}.$$



α 和 β可以使用正则。α, β 固定为1时，MPELU 退化为 ELU； β 固定为很小的值时，MPELU 近似为 PReLU；当α=0，MPELU 等价于 ReLU。

MPELU 的优势在于同时具备 ReLU、PReLU和 ELU的优点。首先，MPELU具备ELU的收敛性质，能够在无 Batch Normalization 的情况下让几十层网络收敛。其次，作为一般化形式， MPELU 较三者的推广能力更强。简言之，MPELU = max(ReLU, PReLU, ELU)。

参数初始值		ImageNet top-1 分类精度(%)(同参数条件下横向比较)			
(beta for MPELU)		MSRA 初始化		Taylor 初始化	
alpha	beta	ReLU	PReLU (加L2正则)	ELU	MPELU (加L2正则)
0	1	62.55			62.59
0.25	1		61.42		62.53
1	1			60.172	62.67

当前对ELU网络普遍采用的初始化方法是 MSRA。这在实际中是可行的，只是不具备理论解释性。我们的工作利用泰勒公式和MSRA的推导过程，为ELU网络初始化提供了理论解释。此外，Dmytro 提出了 LSUV[15]，理论上可以用于 ELU/MPELU 的初始化。但在30/52层ELU网络上，发现 LSUV 会导致ELU网络在几次迭代之内发散，网络文件见 https://github.com/Coldmooon/Code-for-MPELU/。

总结

深度学习的快速发展，催生了形式各异的激活函数。面对琳琅满目的成果，如何做出选择目前尚未有统一定论，仍需依靠实验指导。一般来说，在分类问题上建议首先尝试 ReLU，其次ELU，这是两类不引入额外参数的激活函数。然后可考虑使用具备学习能力的PReLU和本文作者提出的MPELU，并使用正则化技术，例如应该考虑在网络中增加Batch Normalization层。

本文围绕深度卷积神经网络结构，对十余种激活函数进行了总结，相关代码可在作者的github主页上下载：https://github.com/Coldmooon/Code-for-MPELU/。个人浅见如有疏漏之处，请诸位读者不吝斧正。

致谢

这是一次严肃又愉快的写作过程，本文作者在撰稿过程中得到了两位审稿人的建设性意见，改善了文章的可读性，并提示了若干重要的引证文献，在此特表感谢！

参考文献

1. Gulcehre, C., et al., Noisy Activation Functions, in ICML 2016. 2016.
2. Glorot, X. and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. AISTATS 2010.
3. LeCun, Y., et al., Backpropagation applied to handwritten zip code recognition. Neural computation, 1989. 1(4): p. 541-551.
4. Amari, S.-I., Natural gradient works efficiently in learning. Neural computation, 1998. 10(2): p. 251-276.
5. Nair, V. and G.E. Hinton. Rectified linear units improve Restricted Boltzmann machines. ICML 2010.
6. Glorot, X., A. Bordes, and Y. Bengio. Deep Sparse Rectifier Neural Networks.AISTATS 2011.
7. Djork-Arné Clevert, T.U., Sepp Hochreiter. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). ICLR 2016.
8. Li, Y., et al., Improving Deep Neural Network with Multiple Parametric Exponential Linear Units. arXiv preprint arXiv:1606.00305, 2016.
9. Xu, B., et al. Empirical Evaluation of Rectified Activations in Convolutional Network. ICML Deep Learning Workshop 2015.
10. He, K., et al. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. ICCV 2015.
11. He, K. and J. Sun Convolutional Neural Networks at Constrained Time Cost. CVPR 2015.
12. Maas, A.L., Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. in ICML 2013.

13. Goodfellow, I.J., et al. Maxout Networks. ICML 2013..
14. Shang, W., et al., Understanding and Improving Convolutional Neural Networks via Concatenated Rectified Linear Units.ICML 2016.
15. Mishkin, D. and J. Matas All you need is a good init. ICLR 2016.

作者简介

李扬，北京邮电大学电子工程学院通信与网络中心实验室博士生，导师范春晓教授。本科毕业于合肥工业大学光信息科学与技术专业，硕士师从北京邮电大学杨义先教授学习并从事信息安全项目研发。2015年转向深度学习领域，目前专注于深度学习及其在目标检测的应用。

往期精彩回顾

深度学习在人脸识别中的应用 ——优图祖母模型的“进化”

【CVPR2016论文快讯】面部特征点定位的最新进展

人群数量估计领域研究进展

CVPR2016 论文快讯：人脸专题

CVPR 2016论文快讯：目标检测领域的新进展

深度学习在计算机视觉领域成功的启示与开放问题讨论

【CVPR2016论文快讯】细粒度视觉分类的最新进展

阅读原文

作者本人github主页链接：

阅读原文