·计算机与信息技术·

激活函数的发展综述及其性质分析

张　焕,张　庆*,于纪言
(南京理工大学机械工程学院,智能弹药技术国防重点学科实验室,江苏 南京 210094)

摘　要: 为深入研究激活函数的作用机制,探讨优良激活函数应具备的性质,以提高卷积神经网络模型的泛化能力,文章综述了激活函数的发展,分析得到优良激活函数应具备的性质。激活函数大体可分为“S 型”激活函数、“ReLU 型”激活函数、组合型激活函数、其他类型激活函数。在深度学习发展初期,“S 型”激活函数得到了广泛应用。随着网络模型的加深,“S 型”激活函数出现了“梯度消失”问题。ReLU 激活函数的出现缓解了这一问题,但 ReLU 负半轴“置 0”则引入了“神经元坏死”的问题。随后出现的改进激活函数大多基于 ReLU 负半轴进行改动,以缓减“神经元坏死”。文章最后以多层感知机为例,推导了优良激活函数在前向、反向传播中的作用,并得出其应该具备的性质。

关键词: 深度学习;卷积神经网络;激活函数;反向传播;ReLU

中图分类号: TP183; TP391.4　　**文献标志码:** A　　**文章编号:** 1673–159X(2021)04 − 0001 − 10

doi:10.12198/j.issn.1673 − 159X.3761

Overview of the Development of Activation Function and Its Nature Analysis

ZHANG Huan, ZHANG Qing*, YU Jiyan
(*National Defense Key Discipline Laboratory of Intelligent Ammunition Technology, School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing* 210094 *China*)

Abstract: In order to study the mechanism of the activation function in depth and discuss the properties of a good activation function to improve the generalization ability of the convolutional neural network model, the article reviews the development of the activation function and analyzes the properties that a good activation function should have. Activation functions can be roughly divided into "S-type" activation functions, "ReLU-type" activation functions, combined activation functions, and other types of activation functions. In the early stage of the development of deep learning, the "S-type" activation function has been widely used. With the deepening of the network model, it's problem of "gradient disappearance" was found grandually. The emergence of the ReLU activation function alleviates this problem, but the negative half-axis of ReLU "set to 0" introduces the problem of "neuronal necrosis". Most of the subsequent improved ac-

收稿日期:2020 − 10 − 09
基金项目:国防科学技术预先研究基金项目(KO01071)。
第一作者:张焕(1996—),男,硕士研究生,主要研究方向为深度学习、图像处理。
ORCID: 0000 − 0002 − 9631 − 7883　　E-mail: 1084652540@qq.com
*** 通信作者:**张庆(1966—),男,副教授,主要研究方向为弹药战斗部设计与爆炸力学。
ORCID: 0000 − 0003 − 1312 − 0338　　E-mail: 343203908@qq.com

引用格式:张焕,张庆,于纪言. 激活函数的发展综述及其性质分析[J]. 西华大学学报(自然科学版),2021,40(4):1 − 10.
ZHANG Huan, ZHANG Qing, YU Jiyan. Overview of the Development of Activation Function and Its Nature Analysis[J]. Journal of Xihua University(Natural Science Edition), 2021, 40(4): 1 − 10.

tivation functions were modified based on the negative semi-axis of ReLU to slow down "neuronal necrosis". At the end of the article, taking the multilayer perceptron as an example, the role of a good activation function in forward and backward propagation is deduced, and the properties that it should possess are derived.

Keywords: deep learning; convolutional neural network; activation function; back propagation; ReLU

近年来,深度学习[1](deep learning, DL)成为人工智能(artificial intelligence, AI)相关领域中发展最快、最有活力的研究方向之一。卷积神经网络(convolutional neural networks, CNN)作为深度学习的最重要组成部分,其应用范围越来越广,在语音识别、自然语言处理、图像识别等领域表现优异[2-6]。卷积神经网络是由传统的人工神经网络[7-9](artificial neural network, ANN)发展而来。激活函数(activation functions)是卷积神经网络的一个必不可少的部分,它增加了网络的非线性表达能力。

激活函数可以看作卷积神经网络模型中一个特殊的层,即非线性映射层。卷积神经网络在进行完线性变换后,都会在后边叠加一个非线性的激活函数,在非线性激活函数的作用下数据分布进行再映射,以增加卷积神经网络的非线性表达能力。从模仿人类神经科学的角度来看,激活函数在模型中对数据的作用过程是模拟了生物神经元对电信号的处理过程。生物神经元的作用过程是设定一定的阈值激活或抑制接收到的电信号而进行生物信息和信号的传播。模拟生物神经元的作用过程,理想的激活函数应该是将输入数据通过一定的阈值直接输出为"0""1"这 2 种结果。但是,卷积神经网络模型在前向传播、误差反向传播的过程中要求激活函数具备连续性、可微性等性质,显然目前理想生物神经元激活函数不符合该要求。激活函数的自身函数性质决定了在作用过程的优势和缺陷。研究激活函数的性质,分析激活函数性质与优缺点的关联性,寻找时间、空间及特征采集度高效的激活函数成了一项比较重要的研究内容。

1 卷积神经网络中常见的激活函数

在深度学习发展初期,传统 S 型非线性饱和激活函数 sigmoid 和 tanh 函数得到了广泛的应用[10]。然而,随着模型深度的提高,S 型激活函数出现了梯度弥散的问题,这也是早期神经网络不能深度化发展的原因之一[11-15]。2010 年,Hinton 首次提出了修正线性单元[16](rectified linear units, ReLU)作为激活函数。Krizhevsky 等[1] 在 2012 年 ImageNet ILSVRC 比赛中使用了激活函数 ReLU。ReLU 表达式简单易于求导,使得模型训练速度大大加快,且其正半轴导数恒定为 1,很好地解决了 S 型激活函数存在的梯度弥散问题。但是 ReLU 激活函数在负半轴的梯度始终为 0,在模型学习率设置较大情况下,会发生神经元"坏死"的情况[17-18]。

为了解决 ReLU 激活函数的负半轴"神经元坏死"的情况,研究者们提出 Leaky ReLU[19]、PReLU[20]、Noisy ReLU[21]、ELUs[17]、ReLU-softplus[22]、ReLU-softsign[23]、TReLU[24] 等激活函数。这些激活函数有效缓减了"神经元坏死"的问题。下面将详细介绍各类激活函数的性质、优缺点,并总结得到优秀激活函数应该具备的特性。

1.1 sigmoid 和 tanh 激活函数

sigmoid 和 tanh 激活函数是深度学习初期常用的 S 型激活函数,其函数、导数数学表达式为式(1)(2)(3)(4);其函数、导数图像如图 1、图 2 所示。

$$\mathrm{sigmoid}(x)=\frac{1}{1+\mathrm{e}^{-x}} \tag{1}$$

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

$$\frac{\mathrm{d}\,\mathrm{sigmoid}}{\mathrm{d}x}=\mathrm{sigmoid}(x)(1-\mathrm{sigmoid}(x)) \tag{3}$$

$$\frac{\mathrm{d}\tanh}{\mathrm{d}x}=1-\tanh^{2} \tag{4}$$

由图 1 知:sigmoid 激活函数值的范围为(0, 1),经过它激活得到的数据为非 0 均值;sigmoid 激活函数具有双向饱和性,即在一定数据范围内,其导数趋于 0 收敛。由图 2 可知:sigmoid 激活函数

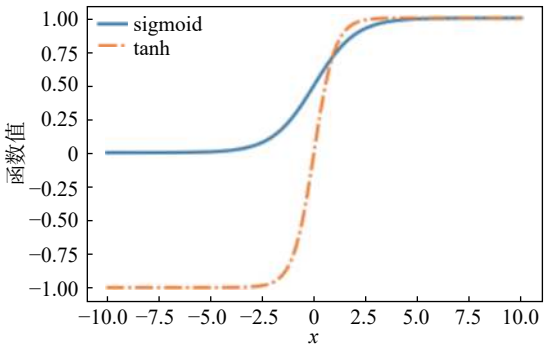

图 1　sigmoid 和 tanh 的函数图

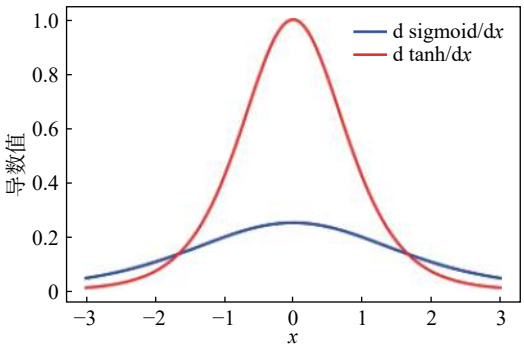

图 2　sigmoid 和 tanh 的导数图

导数范围为(0, 0.25)，且不在(−3, 3)的数据导数值很小，在反向传播过程时，导数相乘很容易造成梯度弥散；sigmoid 激活函数求导过程计算量较大，模型训练的时间复杂度较高。由图 1、图 2 对比知：tanh 激活函数解决了 sigmoid 激活函数非 0 均值的问题，且其导数范围为(0, 1)，从而略微缓减了 sigmoid 激活函数梯度弥散的问题；但 tanh 激活函数存在的双向饱和性仍然使得梯度弥散问题存在，且模型训练的时间复杂度较高。

1.2　ReLU、Nosiy ReLU、Leaky ReLU、PReLU、RReLU 激活函数

激活函数 ReLU 的提出和应用很好地解决了 sigmoid 和 tanh 函数存在的“梯度消失”问题。ReLU 可以扩展为包括高斯噪声的 Noisy ReLU (noisy rectified linear unit)，其在受限玻尔兹曼机解决计算机视觉任务中得到应用[21]。

虽然 ReLU 函数的稀疏性很好地解决了“S 型”软饱和激活函数带来的梯度消失的问题，但是 ReLU 负半轴存在的硬饱和置 0，这可能会导致“神经元坏死”，也使得它的数据分布不为 0 均值，模型在训练过程可能会发生神经元“坏死”的状况。为了解决 Relu 负半轴“神经元坏死”的情况，研究者们对 ReLU 的负半轴下功夫改造，提出了 Leaky ReLU(leaky rectified linear unit)、PReLU(parametric rectified linear unit)、RReLU(randomized leaky rectified linear unit)等激活函数。其中，RReLU 最初是在 Kaggle NDSB 竞赛中得到使用。以上所提到的函数数学表达式为式(5)(6)(7)(8)(9)，表达式中 a 为小于 1 的正数，它们的函数图像如图 3 所示。

$$\mathrm{ReLU}(x)=\begin{cases}x, & x\geqslant 0\\ 0, & x<0\end{cases} \tag{5}$$

$$\text{Noisy ReLU}(x)=\begin{cases}x+Y, & x\geqslant 0\\ 0, & x<0\end{cases},\ Y\sim N(0,\sigma(x)) \tag{6}$$

$$\text{Leaky ReLU}(x)=\begin{cases}x, & x\geqslant 0\\ ax, & x<0\end{cases},\ a\text{为常数} \tag{7}$$

$$\mathrm{PReLU}(x)=\begin{cases}x, & x\geqslant 0\\ ax, & x<0\end{cases},\ a\text{为训练参数} \tag{8}$$

$$\mathrm{RReLU}(x)=\begin{cases}x, & x\geqslant 0\\ ax, & x<0\end{cases},\ a\text{为随机数} \tag{9}$$

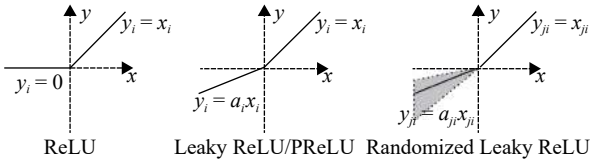

图 3　几种“变种”ReLU 激活函数的函数图

由式(5)(7)(8)(9)以及图 3 知，Leaky ReLU、RReLU 和 PReLU 分别通过手动、随机以及待训练的方式在负半轴添加一个很小的线性参数，其目的是在一定程度上缓减 ReLU 负半轴硬饱和的问题，但是引入的参数给模型训练带来了一定的麻烦。由式(6)知，Noisy ReLU(noisy rectified linear unit)在正半轴添加了高斯噪声，但是和 ReLU 存在一样的问题。

1.3　ReLU6 与神经元的稀疏性

ReLU 的稀疏性给卷积神经网络的训练带来了巨大的成功。无独有偶，2003 年 Lennie 等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。神经元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。类似神经元信号传播，在一定模型下，ReLU 的稀疏性可以提高学习的精度。然而传统的 sigmoid 激活函数几乎同时有一半的神经元被激活，这和神经科学

的研究不太相符,可能会给深度网络训练带来潜在的问题。

在深度学习中,有研究者尝试使用 ReLU6 激活函数。ReLU6 是在 ReLU 激活函数的基础上将大于 6 的数据部分置为 0,以进一步提高连接的稀疏性。式(10)为 ReLU6 的函数表达式,图 4、图 5 为其函数、导数的图像,图 6 为稀疏性连接的示意图。

$$\mathrm{ReLU6}(x)=\min(\max(0,x),6) \tag{10}$$

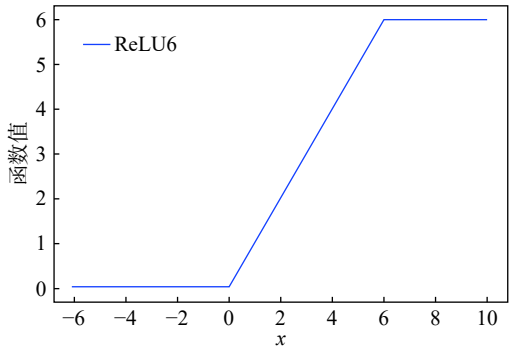

图 4 ReLU6 函数图

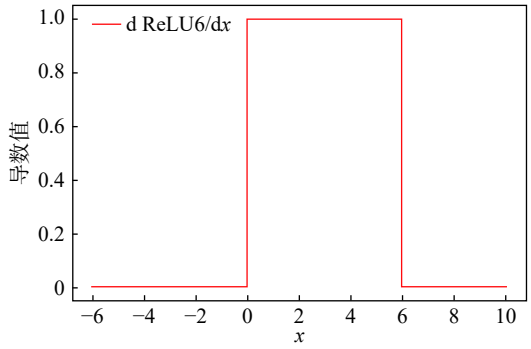

图 5 ReLU6 导数图

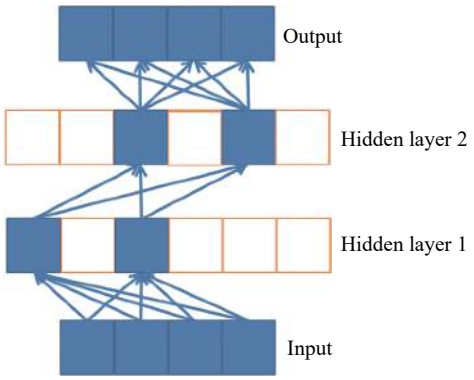

图 6 稀疏性连接示意图

1.4 Swish 和 Xwish 激活函数

Swish 激活函数[25]是谷歌提出的一个效果更优于 ReLU 的激活函数。经过测试,在保持模型其他参数不改变的条件下,只把原模型中的 ReLU 激活函数修改为 Swish 激活函数,模型的准确率均有提升。Swish 激活函数的数学表达式为式(11)所示,函数图像为图 7 所示,导数图像为图 8 所示。式(11)中 β 是常数或可训练的参数。Swish 激活函数没有上界有下界,具有可微、非单调的性质。当 $\beta=0$ 时,Swish 变为线性函数;当 $\beta\to\infty$,Swish 成为了 ReLU 函数:因此,可以将 Swish 函数看成线性函数和 ReLU 函数之间的线性插值的平滑激活函数。

$$\mathrm{Swish}(x)=x\cdot\mathrm{sigmoid}(\beta x) \tag{11}$$

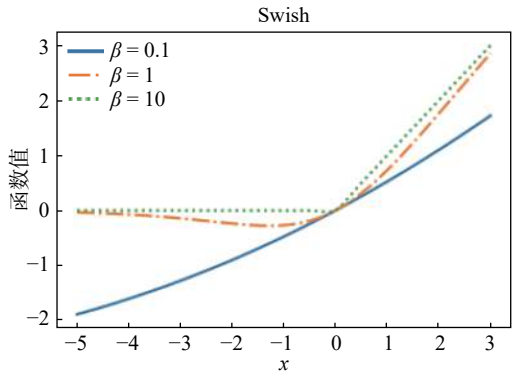

图 7 Swish 函数图

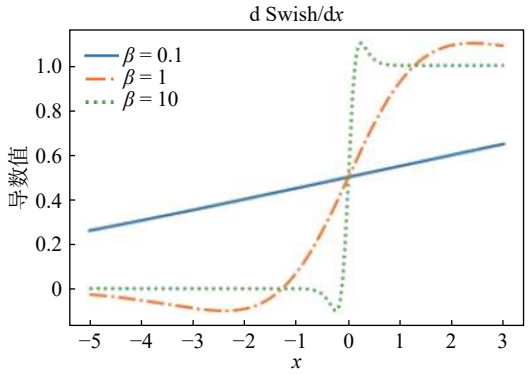

图 8 Swish 导数图

刘宇晴等提出的 Xwish 激活函数[26]与 Swish 激活函数有相似的函数曲线及性质。其函数、导数数学表达式为式(12)(13)所示,函数、导数图像为图 9、图 10 所示。

$$\mathrm{Xwish}(x)=x\cdot[\arctan(\beta x)+0.5\pi]/\pi \tag{12}$$

$$\frac{\mathrm{d\,Xwish}}{\mathrm{d}x}=\frac{1}{\pi}[\arctan(\beta x)+0.5\pi]+\frac{1}{\pi}\frac{\beta x}{1+(\beta x)^2} \tag{13}$$

1.5 Maxout 激活函数

Maxout 激活函数的原理是通过线性分段函数来拟合可能的目标凸函数,并将其作为激活函数。它可以看作在卷积神经网络中添加的一层激活层。它包含 1 个参数 k。相比其他激活函数,它的

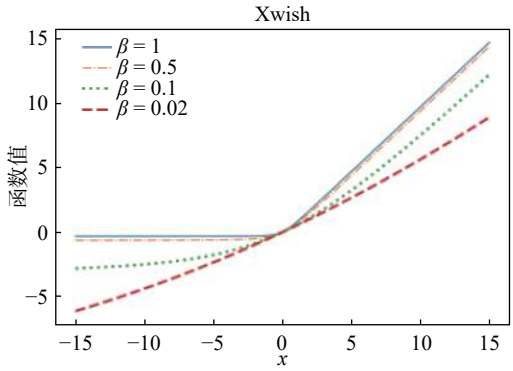

图 9　Xwish 函数图

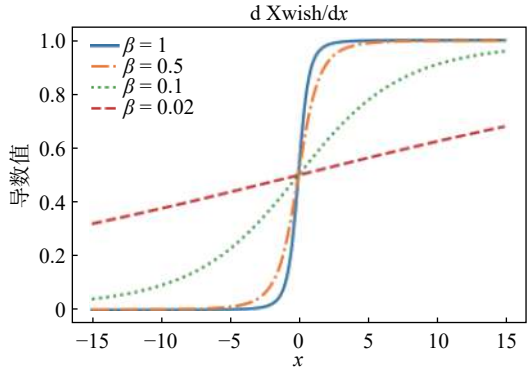

图 10　Xwish 在不同参数下的导数图

特殊之处在于,增加了 k 个神经元,经过神经元输出最大的激活值。Maxout 激活函数可以看作 ReLU 激活函数的推广。Maxout 激活函数能增强 Dropout 函数的功能,实验证明二者一起使用时能发挥比较好的效果[27]。

任意的凸函数都可以由分段线性函数拟合,而 Maxout 取 k 个线性隐藏层节点的最大值。图 11 依次示出 Maxout 激活函数拟合线性激活函数、绝对值激活函数、二次激活函数的过程。图中展示了在一维输入下 Maxout 激活函数拟合二维平面函数的过程。实际上,Maxout 激活函数可以逼近拟合更高维度的凸函数。Maxout 具有 ReLU 的优点,即线性和不饱和性,同时它也解决了 ReLU 存在的“神经元坏死”的问题。但是,Maxout 引入了训练参数而导致了模型整体参数数量的激增,导致模型的复杂度增高。

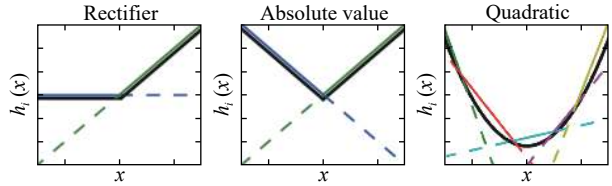

图 11　Maxout 示意图

1.6　ELU 和 TReLU 激活函数

ELU 和 TReLU 激活函数的正半轴与 ReLU 激活函数保持一致,通过对负半轴引入软饱和以代替置“0”。式(14)(15)为 ELU 激活函数的函数、导数数学表达式,图 12、13 为其函数、导数图像。从数学表达式(14)(15)和图 12、13 可以看出,ELU 激活函数[13] 在正半轴具有与 ReLU 激活函数一样的优势,同时引入了负半轴的定义使得整体输出均值接近 0。与 LeakyReLU 和 PReLU 相比,虽同样都是激活了负半轴,但 ELU 的负半轴为软饱和区,斜率具有衰减性,这使得其对噪声有一些鲁棒性。同时,参数a控制着函数的斜率变化。

$$\mathrm{ELU}(x)=\begin{cases} x, & x\geqslant 0 \\ a(\mathrm{e}^{x}-1), & x<0 \end{cases}\quad (a\text{为参数},a\geqslant 0) \tag{14}$$

$$\frac{\mathrm{d}\,\mathrm{ELU}}{\mathrm{d}x}=\begin{cases} 1, & x\geqslant 0 \\ a\mathrm{e}^{x}, & x<0 \end{cases}\quad (a\text{参数},a\geqslant 0) \tag{15}$$

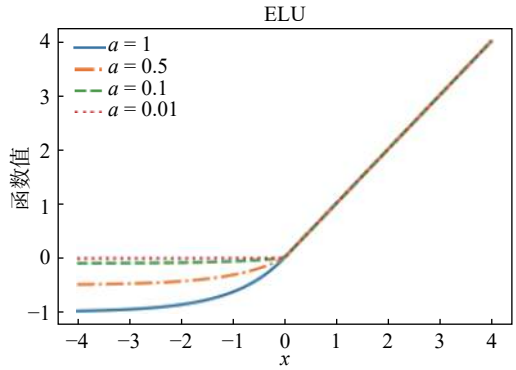

图 12　ELU 在不同参数下的函数图

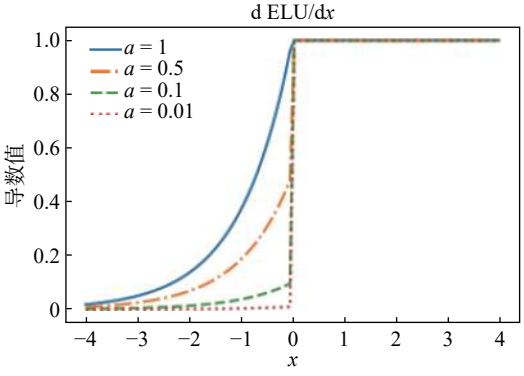

图 13　ELU 在不同参数下的导数图

张涛等同样提出了负半轴为饱和区的 TReLU 激活函数[24]。式(16)(17)为其函数、导数数学表达式。式中a为可变参数,用来控制非饱和区域的斜率变化。TReLU 拥有和 ELU 相似的优势:缓减了

梯度弥散的问题;激活了负半轴,从而缓减了“神经元坏死”的问题;近似于 0 均值分布;负半轴的软饱和性使得其对噪声具有鲁棒性。图 14、15 为其函数、导数图像。

$$\mathrm{TReLU}(x)=\begin{cases} x, & x\geqslant 0 \\ \tanh(ax), & x<0 \end{cases} \quad (a\text{参数}, a\geqslant 0) \tag{16}$$

$$\frac{\mathrm{d}\,\mathrm{TReLU}}{\mathrm{d}x}=\begin{cases} 1, & x\geqslant 0 \\ a^2\left(1-(\tanh(ax))^2\right), & x<0 \end{cases} \tag{17}$$

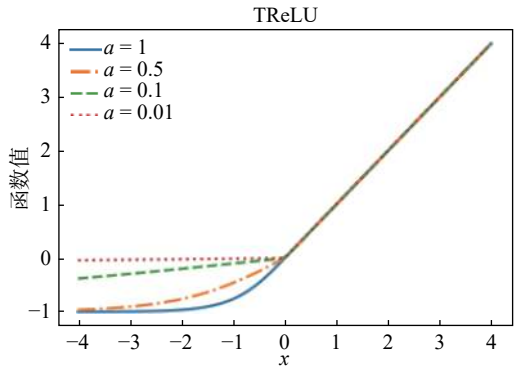

图 14 TReLU 在不同参数下的函数图

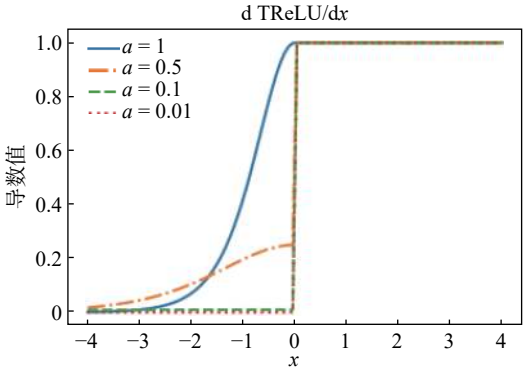

图 15 TReLU 在不同参数下的导数图

1.7 softplus、softsign 和relu-softplus、relu-softsign 激活函数

softplus 是对所有输入数据进行非线性映射的一种激活函数。式(18)(19)为其函数、导数表达式,图 16、图 17 为其函数、导数图像。从数学表达式和函数图像可以看出:softplus 无上界,具有负半轴单向软饱和性,函数值始终大于 0;同时引入了对数和指数运算,计算量较大。

$$\mathrm{softplus}(x)=\ln(1+\mathrm{e}^x) \tag{18}$$

$$\frac{\mathrm{d}\,\mathrm{softplus}}{\mathrm{d}x}=\frac{\mathrm{e}^x}{1+\mathrm{e}^x} \tag{19}$$

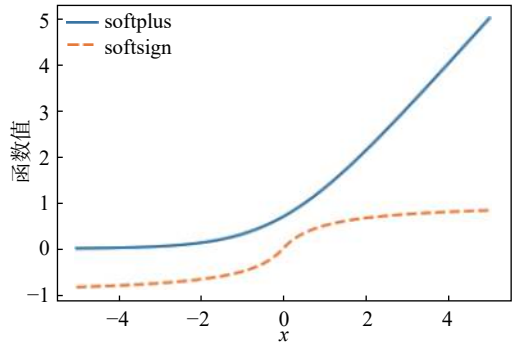

图 16 softplus 和 softsign 函数图

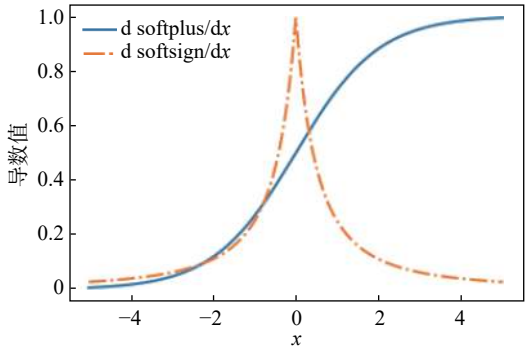

图 17 softplus 和 softsign 导数图

softsign 激活函数是一种双向软饱和“S 型”激活函数,可以看作 tanh 激活函数的改进版。式(20)(21)为其函数、导数表达式,图 16、图 17 为其函数、导数图像。图 18 示出 softplus、softsign、tanh 的函数图像比较结果。图 19 示出 softplus、softsign、tanh 的导数图像比较结果。从式(20)(21)和图 18、图 19 可以看出,softsign 激活函数是 0 均值分布的,且相比于 tanh 激活函数,softsign 激活函数的曲线变化更加平缓,其导数下降的速率较慢。从理论上看,相较于 tanh 激活函数,其能够进一步缓减双向软饱和“S 型”激活函数存在的梯度弥散问题。

$$\mathrm{softsign}(x)=\frac{x}{1+|x|} \tag{20}$$

$$\frac{\mathrm{d}\,\mathrm{softsign}}{\mathrm{d}x}=\begin{cases} \dfrac{1}{(1+x)^2}, & x\geqslant 0 \\ \dfrac{1}{(1-x)^2}, & x<0 \end{cases} \tag{21}$$

曲之琳等[22]将 ReLU 激活函数和 softplus 激活函数进行了结合,并对 softplus 的负半轴做减常数 ln2 处理,提出了 relu-softplus 组合激活函数。式(22)(23)为其函数、导数表达式,图 20、图 21 为函数、导数图像。和 ELU、TReLU 等激活函数一

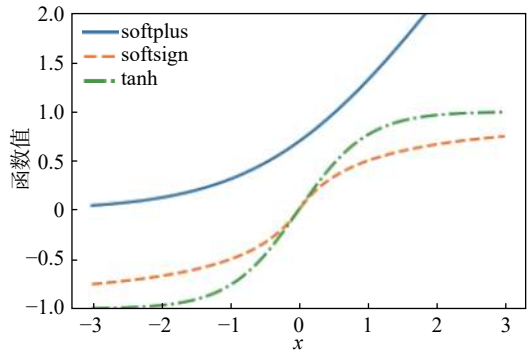

图 18　softplus/softsign/tanh 函数对比图

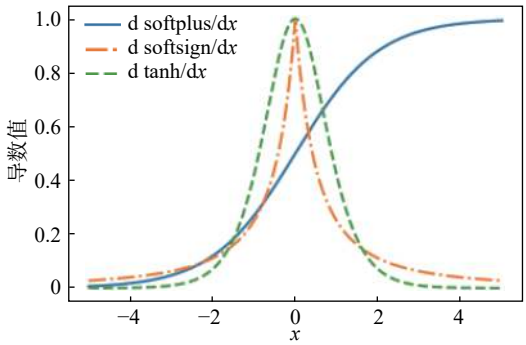

图 19　softplus/softsign/tanh 导数对比图

样, relu-softplus 激活函数的负半轴软饱和缓减了“神经元坏死“问题, 且负半轴做减常数处理, 巧妙地完成了与正半轴的连接。但是 relu-softplus 激活函数存在指数, 在负半轴零点附近的导数存在突变, 持续减小直至 0。其存在的问题是对学习率要求较高: 若学习率设置过大, 容易出现模型不收敛的问题; 学习率设计较小, 模型收敛得慢。

$$\mathrm{relu-softplus}(x)=\begin{cases} x, & x\geqslant 0 \\ \ln(1+\mathrm{e}^{x})-\ln 2, & x<0 \end{cases} \tag{22}$$

$$\frac{\mathrm{d\ relu-softplus}}{\mathrm{d}x}=\begin{cases} 1, & x\geqslant 0 \\ \dfrac{\mathrm{e}^{x}}{1+\mathrm{e}^{x}}, & x<0 \end{cases} \tag{23}$$

王红霞等[23]将 ReLU 激活函数和 softsign 激活函数进行了结合, 提出了 relu-softsign 组合激活函数。式(24)(25)为其函数、导数表达式, 图 20、图 21 示出 relu-softplus 和 relu-softsign 的函数、导数的比较结果。从公式(24)(25)以及图 20、图 21 中可以看出: relu-softsign 激活函数在正半轴为线性单元, 其导数始终为 1, 这一特点保证了模型的加速收敛; relu-softsign 激活函数在负半轴为 softsign 激活函数的负半轴软饱和部分, 其导数在零点不存在突变, 导数值由 1 减小至趋于 0。relu-softsign 激活函数由于在负半轴提供的非 0 导数, 增加了对非正值处理的鲁棒性, 也加速了模型的收敛速度。

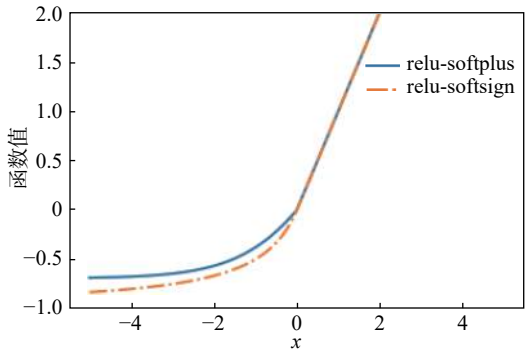

图 20　relu-softsign 和 relu-softplus 函数图

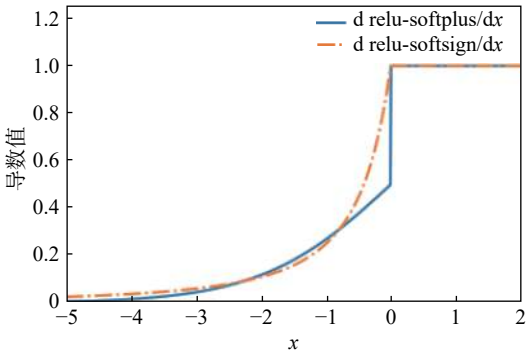

图 21　relu-softsign 和 relu-softplus 导数图

比较二者的负半轴斜率, 由图 20、图 21 可以看出, relu-softsign 激活函数在负半轴零点附近较 relu-softplus 激活函数整体有更大的导数, 前者在零点附近导数由 1 衰减较快, 但保证了模型在零点附近的数据特征下有较快的收敛性; 后者在零点附近导数值由 1 突变为 0.5, 相较于前者, 其模型在零点附近的数据特征下收敛性不足。王红霞等[23]通过仿真实验验证了此理论分析。

$$\mathrm{relu-softsign}(x)=\begin{cases} x, & x\geqslant 0 \\ \dfrac{x}{1-x}, & x<0 \end{cases} \tag{24}$$

$$\frac{\mathrm{d\ relu-softsign}}{\mathrm{d}x}=\begin{cases} 1, & x\geqslant 0 \\ \dfrac{1}{(1-x)^{2}}, & x<0 \end{cases} \tag{25}$$

2　激活函数的性质分析

依据上文的讨论及激活函数的发展规律, 可以初步得出一个良好的激活函数常具备以下一些特点: 1)非线性以及可微性; 2)解决梯度消失问题, 也

避免出现梯度爆炸问题; 3)解决“神经元坏死“问题; 4)符合或近似符合 0 均值分布; 5)计算的时间、空间复杂度小; 6)存在一定的稀疏性; 7)模型收敛速度相对较快; 8)对数据噪声具有一定的鲁棒性等。

sigmoid 和 tanh 激活函数符合 1)、4)特点, 但不符合 2)、5)、6)、7); ReLU 激活函数缓和了“S”型激活函数存在的问题 2)、5)、6)、7), 但是引入了问题 3)、4); Nosiy ReLU、Leaky ReLU、PReLU、RReLU 激活函数对 ReLU 激活函数负半轴进行改造, 缓减了 ReLU 激活函数存在的问题 3); ReLU6 激活函数引进稀疏性, 因此符合特点 6)、8); Swish 和 Xwish 激活函数可看成线性函数和 ReLU 函数之间的线性插值的平滑激活函数, 其保留了负半轴的特征, 缓和了 ReLU 存在的问题 3)、4); Maxout 激活函数缓和了 ReLU 激活函数存在的问题 3), 但是引入参数也带来了问题 5); ELU 和 TReLU 激活函数缓减了 ReLU 激活函数的问题 3)、4), 且引入的软饱和满足特点 8); relu-softplus、relu-softsign 激活函数结合了 softplus/softsign 和 ReLU 的正负半轴, 不但缓和了 ReLU 激活函数存在的问题 3)、4), 且符合特点 7)、8), 将激活函数负半轴斜率变化快慢和模型收敛速度结合了起来。

为了进一步分析激活函数的性质, 本文以多层感知机为例, 推导激活函数在前向传播和反向传播过程中的作用表达式。

2.1 损失函数

在多层感知机中, 给定样本集合, 其整体代价函数为式(26)。其中, 前一项为误差项, 常见的有平方误差项、交叉熵误差项等; 后一项为正则化项, 此处使用的是 L2 正则化。

$$J(W,b)=\left[\frac{1}{m}\sum J\left(W,b;x^{(i)},y^{(i)}\right)\right]+\frac{\lambda}{2}\sum\sum\sum W^2 \tag{26}$$

2.2 前向传播

多层感知机中, 输入信号通过各个网络层的隐节点产生输出的过程称为前向传播。在网络训练过程中, 前向传播会生成一个标量损失函数。定义第i层的输入、输出为$x^{[i]}$、$a^{[i]}$,上一层的输出作为下一层的输入。$w^{[i]}$、$b^{[i]}$为第i层的权值参数和偏置, $z^{[i]}$是第i层输入神经元未经激活的值, $g^{[i]}(x)$为第i层的激活函数。前向传播过程的表达式为

$$z^{[i]}=w^{[i]}a^{[i-1]}+b^{[i]}a^{[i]}=g^{[i]}\left(z^{[i]}\right) \tag{27}$$

分析式(27), 在神经网络的前向传播的过程中, 输入和本层的权值相乘, 加上偏置, 并将各项结果累加, 得到下一层神经元的初步输入值, 其与上一层的关系为线性关系。这个初步输入值经过激活函数的加工, 对初步输入值进行非线性映射, 增强了表达能力。因此激活函数应该具备以下性质:

1)激活函数具有较强的非线性表达能力;

2)激活函数应该符合或近似符合 0 均值分布条件, 以增加其对数据的适应性;

3)激活函数应该具有良好的计算特性。

2.3 反向传播

多层感知机中, 反向传播过程是将损失函数的梯度信息沿着网络向后传播, 以更新权值参数。其过程是将$\mathrm{d}a^{[j]}$作为输入, 得到$\mathrm{d}w^{[j]}$、$\mathrm{d}b^{[j]}$, 作为输出, a 为学习率, 其余参数表达参照 2.2 中的前向传播, 其过程表达式如式(28)—(34)所示。

$$\mathrm{d}z^{[j]}=\mathrm{d}a^{[j]}\cdot g^{[j]'}\left(z^{[j]}\right) \tag{28}$$

$$\mathrm{d}w^{[j]}=\mathrm{d}z^{[j]}\cdot a^{[j-1]^{\mathrm{T}}} \tag{29}$$

$$\mathrm{d}b^{[j]}=\mathrm{d}z^{[j]} \tag{30}$$

$$\mathrm{d}a^{[j-1]}=\left(w^{[j]}\right)^{\mathrm{T}}\cdot\mathrm{d}z^{[j]} \tag{31}$$

$$\mathrm{d}z^{[j-1]}=\left(w^{[j]}\right)^{\mathrm{T}}\cdot\mathrm{d}z^{[j]}\cdot g^{[j-1]'}\left(z^{[j-1]}\right) \tag{32}$$

$$w=w^{[j]}-a\cdot\left(w^{[j+1]}\right)^{\mathrm{T}}\cdot\mathrm{d}z^{[j+1]}\cdot g^{[j]'}\left(z^{[j]}\right)\cdot a^{[j-1]^{\mathrm{T}}} \tag{33}$$

$$b=b-a\cdot\mathrm{d}b=b^{[j]}-a\cdot\left(w^{[j+1]}\right)^{\mathrm{T}}\cdot\mathrm{d}z^{[j+1]}\cdot g^{[j]'}\left(z^{[j]}\right) \tag{34}$$

权值更新式(34)(35)中, $g^{[j]'}\left(z^{[j]}\right)$为第 i 层激活函数的导数值。以上文中各类激活函数为例, 激活函数求导展开式为式(35)—(43)所示。

对于 sigmoid 激活函数, 其求导展开式为

$$g^{[j]'}\left(z^{[j]}\right)=\frac{1}{1+\mathrm{e}^{-z^{[j]}}}\left(1-\frac{1}{1+\mathrm{e}^{-z^{[j]}}}\right) \tag{35}$$

对于 tanh 激活函数, 其求导展开式为

$$g^{[j]'}\left(z^{[j]}\right)=1-\left(\frac{\mathrm{e}^{z^{[j]}}-\mathrm{e}^{-z^{[j]}}}{\mathrm{e}^{z^{[j]}}+\mathrm{e}^{-z^{[j]}}}\cdot\frac{\mathrm{e}^{z^{[j]}}-\mathrm{e}^{-z^{[j]}}}{\mathrm{e}^{z^{[j]}}+\mathrm{e}^{-z^{[j]}}}\right) \tag{36}$$

对于 ReLU 激活函数, 其求导展开式为

$$g^{[j]'}\left(z^{[j]}\right)=\begin{cases}1, z^{[j]}\geqslant 0\\ 0, z^{[j]}<0\end{cases} \tag{37}$$

对于 ReLU6 激活函数,其求导展开式为

$$g^{[j]'}\left(z^{[j]}\right)=\begin{cases}0, & z^{[j]}\geqslant 6\\ 1, & 0<z^{[j]}<6\\ 0, & z^{[j]}\leqslant 0\end{cases} \tag{38}$$

对于 Xwish 激活函数,其求导展开式为

$$g^{[j]'}\left(z^{[j]}\right)=\frac{1}{\pi}\left[\arctan\left(\beta z^{[j]}\right)+0.5\pi\right]+\frac{1}{\pi}\frac{\beta z^{[j]}}{1+\left(\beta z^{[j]}\right)^2} \tag{39}$$

对于 ELU 激活函数,其求导展开式为

$$g^{[j]'}\left(z^{[j]}\right)=\begin{cases}1, & z^{[j]}\geqslant 0\\ a\cdot \mathrm{e}^{z^{[j]}}, & z^{[j]}<0\left(a\text{参数}, a\geqslant 0\right)\end{cases} \tag{40}$$

对于 TReLU 激活函数,其求导展开式为

$$g^{[j]'}\left(z^{[j]}\right)=\begin{cases}1, & z^{[j]}\geqslant 0\\ a^2\left(1-\left(\tanh\left(az^{[j]}\right)\right)^2\right), & z^{[j]}<0\end{cases} \tag{41}$$

对于 relu-softplus 激活函数,其求导展开式为

$$g^{[j]'}\left(z^{[j]}\right)=\begin{cases}1, & z^{[j]}\geqslant 0\\ \dfrac{\mathrm{e}^{z^{[j]}}}{1+\mathrm{e}^{z^{[j]}}}, & z^{[j]}<0\end{cases} \tag{42}$$

对于 relu-softsign 激活函数,其求导展开式为

$$g^{[j]'}\left(z^{[j]}\right)=\begin{cases}1, & z^{[j]}\geqslant 0\\ \dfrac{1}{\left(1-z^{[j]}\right)^2}, & z^{[j]}<0\end{cases} \tag{43}$$

分析式(28)—(43),可以得出,权值参数(*Wb*)的更新与激活函数导数值的大小存在线性相关关系,深层神经网络的参数更新中会出现激活函数的导数连乘。因此分析反向传播过程,激活函数应该具备以下性质:

1)在连乘情况下避免出现梯度消失问题,也避免出现梯度爆炸问题;

2)避免出现激活函数导数过于置 0 从而导致参数不更新,出现“神经元坏死“问题;

3)激活函数的导数计算的时间、空间复杂度应该较小;

4)由神经科学学科的研究论证,激活函数应该存在一定的稀疏性;

5)模型刚开始训练的时候,激活函数的导数应该较大,加速模型收敛,在模型收敛后半段,激活函数有一定的软饱和性,即导数渐渐趋于 0,使得模型收敛至最优值;

6)参数 w 的更新方向与该层的输入$a^{[j-1]}$有关,参数的更新方向应该自由,因此应该选择一个正负值都可以输出的激活函数;

7)模型对数据噪声应具有一定的鲁棒性,因此激活函数应该具备一定的饱和性。

3　结束语与展望

本文较详细地列举了激活函数的发展历程及当前主流激活函数所固有的特点,并推导了多层感知机的前向传播、反向传播过程,结合激活函数的发展经验提出了优良激活函数应该具备的一些性质。这为深入了解激活函数提供了便利,为研究改进激活函数提供了一种思路。随着计算机计算水平以及深度学习理论的不断发展,激活函数的角色还会发生变化。未来可能从以下几个方向突破:1)从计算机的计算能力限制中解放出来,应用更复杂、特征映射更精细的激活函数;2)深度学习理论的重大突破及神经科学的解密可能会带来激活函数发展的新思路;3)在 ReLU 激活函数的模板下,对激活函数进行负半轴改造;4)稀疏性连接理念在激活函数中的应用启发;5)针对特定任务应用而设计激活函数(精度与时间复杂度的权衡)。

激活函数作为卷积神经网络的一个必不可少的组成部分,不论是在 ReLU 的基础上进行改进,还是构造全新激活函数;不论是通用型激活函数,还是单适用性激活函数:其最终的目的是为了增强或更快速地对数据特征进行非线性映射,最终实现模型的高泛化能力或低时间复杂度。

参　考　文　献

[1] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. ImageNet c-lassification with deep convolutional neural netwo-rks[J]. Advances in Neural Information Processing System, 2012, 25(2): 1097 − 1105.

[2] 崔雍浩, 商聪, 陈锶奇, 等. 人工智能综述: AI 的发展[J]. 无线电通信技术, 2019, 45(3): 225 − 231.

[3] 黄凯奇, 任伟强, 谭铁牛. 图像物体分类与检测算法综述[J]. 计算机学报, 2014, 36(6): 1225 − 1240.

[4] 常亮, 邓小明, 周明全, 等. 图像理解中的卷积神经网络[J]. 自动化学报, 2016, 42(9): 1300 − 1312.

[5] 吴正文. 卷积神经网络在图像分类中的应用研究[D]. 成都: 电子科技大学, 2015.

[6] 魏溪含, 涂铭, 张修鹏. 深度学习与图像识别原理与实践 [M]. 北京: 机械工业出版社, 2019: 3 − 8.

[7] 叶世伟, 史忠植. 神经网络原理 [M]. 北京: 机械工业出版社, 2004.

[8] HAYKIN S. 神经网络与机器学习: 英文版 [M]. 北京: 机械工业出版社, 2009.

[9] GE S S, HANG C C, LEE T H, et al.Stable adaptive neural network control[M]. Berlin: Spring Publishing Company Incorporated, 2010.

[10] 李宏伟, 吴庆祥. 智能传感器中神经网络激活函数的实现方案[J]. 传感器与微系统, 2014, 33(1): 46 − 48.

[11] GLOROT X, BENGIO Y. Understanding the difficulty of training deep feedforward neural networks[J]. Journal of Machine Learning Research, 2010, 9: 249 − 256.

[12] BALDI P, SADOWSKI P, LU Z Q. Learning in the machine: random backpropagation and the deep learning channel[J]. Artificial Intelligence, 2018, 260: 1 − 35.

[13] LECUN Y, BENGIO Y, HINTON G. Deep learning[J]. Nature, 2015, 521: 436 − 444.

[14] GOODFELLOW I, BENGIO Y, COURVILLE A, et al.Deep learning[M]. Cambridge: MIT Press, 2016.

[15] CHATTERJEE A, GUPTA U, CHINNAKOTLA M K, et al. Understanding emotions in text using deep learning and big data[J]. Computers in Human Behavior, 2019, 93: 309 − 317.

[16] NAIR V, HINTON G E.Rectified linear units improve restricted boltzmann machines[C]// Proceedings of the 27th International Conference on Machine Learning (ICML-10). Haifa, Israel: DBLP, 2010: 807 − 814.

[17] DJORK-ARNÉ C, UNTERTHINER T, HOCHREITER S.Fast andaccurate deep network learning by exponential lin-ear units(ELUs)[J/OL]. Computer Science, 2015. https://arxiv.org/abs/1511.07289.

[18] DOLEZEL P, SKRABANEK P, GAGO L. Weight initialization possibilities for feedforward neural network with linear saturated activation functions[J]. IFACPapersOnLine, 2016, 49(25): 49 − 54.

[19] MAAS A L, HANNUN A Y, NG A Y.Rectifier nonlinea-rities improve neural network acoustic models[C]// Proceedings of the 30th International Conference on Machine Learning.Atlanta: ACM, 2013: 456 − 462.

[20] HE K, ZHANG X, REN S, et al.Delving deep into rectifiers: surpassing human-level performance on ImageNet classification[C]//Proceedings of the IEEE International Conference on Computer Vision.Santiago : IEEE, 2015: 1026 −1034.

[21] GULCEHRE C, MOCZULSKI M, DENIL M, et al.Noisy activation functions[C]// Proceedings of the 33rd International Conference on International Conference on Machine Learning.[S.l.]: JMLR, 2016: 3059 − 3068.

[22] 曲之琳, 胡晓飞. 基于改进激活函数的卷积神经网络研究[J]. 计算机技术与发展, 2017, 27(12): 77 − 80.

[23] 王红霞, 周家奇, 辜承昊, 等. 用于图像分类的卷积神经网络中激活函数的设计[J]. 浙江大学学报(工学版), 2019, 53(7): 1363 − 1373.

[24] 张涛, 杨剑, 宋文爱, 等. 关于改进的激活函数 TReLU 的研究[J]. 小型微型计算机系统, 2019(1): 58 − 63.

[25] RAMACHANDRAN P, ZOPH B, LE Q V. Searching for an activation functions[EB/OL]. (2017-10-27)[2020-09-09]. https://arxiv.org/abs/1710.05941.

[26] 刘宇晴, 王天昊, 等. 深度学习神经网络的新型自适应激活函数[J]. 吉林大学学报(理学版), 2019, 57(4): 857 − 859.

[27] 赵惠珍, 刘付显, 李龙跃, 等. 基于混合 maxout 单元的卷积神经网络性能优化[J]. 通信学报, 2017, 38(7): 105 − 114.

(编校: 饶莉)