DOI:10.11918/j.issn.0367-6234.201703117

# 一种新的深度卷积神经网络的 SLU 函数

赵慧珍,刘付显,李龙跃

(空军工程大学 防空反导学院,西安 710051)

**摘 要**:修正线性单元(rectified linear unit, ReLU)是深度卷积神经网络常用的激活函数,但当输入为负数时,ReLU 的输出为零,造成了零梯度问题;且当输入为正数时,ReLU 的输出保持输入不变,使得 ReLU 函数的平均值恒大于零,引起了偏移现象,从而限制了深度卷积神经网络的学习速率和学习效果.针对 ReLU 函数的零梯度问题和偏移现象,根据"输出均值接近零的激活函数能够提升神经网络学习性能"原理对其进行改进,提出 SLU(softplus linear unit)函数.首先,对负数输入部分进行 softplus 处理,使得负数输入时 SLU 函数的输出为负,从而输出平均值更接近于零,减缓了偏移现象;其次,为保证梯度平稳,对 SLU 的参数进行约束,并固定正数部分的参数;最后,根据 SLU 对正数部分的处理调整负数部分的参数,确保激活函数在零点处连续可导,信息得以双向传播.设计深度自编码模型在数据集 MINST 上进行无监督学习,设计网中网卷积神经网络模型在数据集 CIFAR-10 上进行监督学习.实验结果表明,与 ReLU 及其相关改进单元相比,基于 SLU 函数的神经网络模型具有更好的特征学习能力和更高的学习精度.

**关键词**:深度学习;卷积神经网络;激活函数;softplus 函数;修正线性单元

**中图分类号**:TP183 **文献标志码**:A **文章编号**:0367-6234(2018)04-0117-07

# A novel softplus linear unit for deep CNN

ZHAO Huizhen, LIU Fuxian, LI Longyue

(School of Air and Missile Defense, Air Force Engineering University, Xi'an 710051, China)

**Abstract**: Currently, the most popular activation function for deep convolutional neural network is the rectified linear unit (ReLU). The ReLU activation function outputs zero for negative quadrant, inducing the death of some neurons, and remains the input data for the positive quadrant, inducing a bias shift. According to the theory that "zero means activations improving learning ability", softplus linear unit (SLU) is introduced as an adaptive activation function that can tackle with these two problems. Firstly, negative inputs are processed with the softplus function, pushing the mean of outputs of the activation function to zero and reducing the bias shift. Then, the parameters of the positive component are fixed to control vanishing gradients. Thirdly, to maintain continuity and differentiability at zero, the parameters of the negative part are updated according to the positive quadrant. Several experiments are conducted on the MNIST dataset for supervised learning with deep auto-encode networks, as well as several experiments on the CIFAR-10 dataset for unsupervised learning with deep convolutional neural networks. The experiments have shown faster convergence and better performance for image classification of SLU-based networks compared with rectified activation functions.

**Keywords**: deep learning; deep convolutional neural network; activation function; softplus function; rectified linear unit

深度卷积神经网络(deep convolutional neural network, CNN)利用结构深度的优势在计算机视觉如图像分类[1-2]、目标识别[3]、目标探测[4]等应用中取得了较大的成功,故 CNN 结构的加深是深度学习的重要研究内容.2012 年,Krizhevsky 等[1]首次将 CNN 应用于计算机视觉的模型具有 8 层网络,2014 年,Simonyan 等[5]将网络加深至 19 层,2015 年 Szegedy 等[2]设计了 22 层的深度网络,He 等[6]将网络加深至 152 层.随着网络的加深,梯度消失/溢出的问题越来越突出,网络较浅时,非线性层数较少,传递到网络底层时梯度变化较少,能够驱动参数更新;而网络较深时,非线性层数较多,传递到网络底层的梯度会消失/溢出,无法驱动底层参数[7].传统常用的非线性变换如 sigmoid、tanh 等,容易形成梯度消失/溢出.

2010 年,Nair 等[8]提出线性修正单元(rectified linear unit, ReLU),ReLU 具有单侧抑制、相对宽阔的兴奋边界等特点,不仅能够缓解梯度消失/溢出问题,还能够加速模型收敛速度、提高模型学习精度.但 ReLU 对负数输入的处理造成了部分神经元死

**收稿日期**:2017-03-23
**基金项目**:国家自然科学基金(61601499)
**作者简介**:赵慧珍(1990—),女,博士研究生;
刘付显(1962—),男,教授,博士生导师
**通信作者**:赵慧珍,happy100zhao90@163.com

亡,使得梯度不能在负数部分传播;且当输入为正数时,输出保持输入不变,使函数输出的平均值始终大于零,形成了偏移现象(bias shift),限制了学习速率和学习精度[1,8].在 ReLU 的基础上,Mass 等[9]提出弱修正线性单元(leaky ReLU,LReLU)以避免零梯度问题,但 LReLU 对坡度因子较为敏感;He 等[10]针对损失函数对不同坡度因子不可导的特点提出了参数修正线性单元(parametric ReLU,PReLU),但模型容易过拟合;Xu 等[11]提出随机弱线性修正单元(randomized LReLU, RReLU),使得模型一定程度上减缓偏移现象;Clevert 等[12]提出指数线性单元(exponential linear unit, ELU),但存在左软饱和现象,且其指数级变化使得饱和点更接近零点,而饱和点之后的梯度较难传递[7].

针对 ReLU 函数存在的零梯度问题与偏移现象,本文提出了 SLU(softplus linear unit)函数.实验结果证明,无论是监督学习还是非监督学习,基于 SLU 的模型均具有更好的特征学习能力和更高的分类精度.

## 1 ReLU 及相关改进单元

自 2012 年 Krizhevsky 等[1]首次将 ReLU 作为激活函数应用于神经网络模型并取得突破后,很多学者针对 ReLU 存在的零梯度问题和偏移现象进行了改进,以期进一步提升神经网络性能.对 ReLU 的改进方法主要有 LReLU、PReLU、RReLU 和ELU.

### 1.1 ReLU 函数

ReLU 的定义如[8]

$$y_i = \begin{cases} x_i, x_i \geqslant 0; \\ 0, x_i < 0. \end{cases}$$

式中:$x_i$ 为第 $i$ 个输入,$y_i$ 为相对应的输出.如图 1(a)所示,当输入为正数时,输出保持输入不变,当输入为负数时,输出为零,故 ReLU 函数输出平均值大于零,存在偏移现象,影响模型的收敛速率和学习效果.

### 1.2 LReLU 函数

LReLU 的定义为[9]

$$y_i = \begin{cases} x_i, & x_i \geqslant 0; \\ x_i/\alpha_i, x_i < 0. \end{cases} \tag{1}$$

式中 $\alpha_i$ 为坡度因子,取值范围为 $(1, +\infty)$,取值越大,对负数部分的修正越小,如图 1(b)所示,图中取 $\alpha_i = 10$.坡度因子的存在一定程度上可以减缓偏移现象,但 LReLU 对坡度因子较为敏感,$\alpha_i$ 的取值影响模型的收敛速率和学习效果.

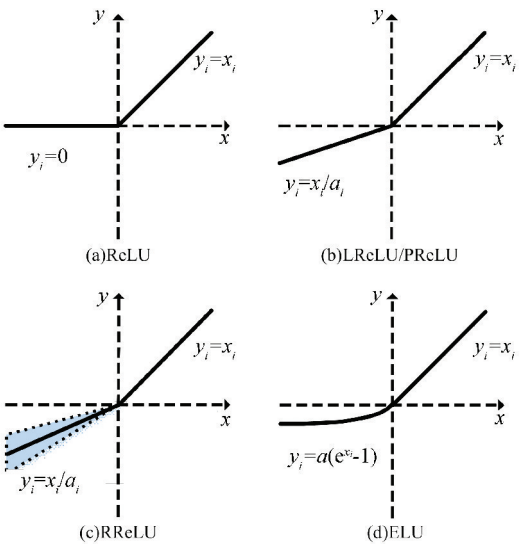


**图 1 ReLU 及其改进函数示意**

Fig.1 Shape of ReLU and variants

### 1.3 PReLU 函数

PReLU 是 LReLU 的改进形式[10],其表达式与 LReLU 一致,如式(1),但其坡度因子 $\alpha_i$ 并非固定值,而是通过后向传播学习得到,灵活可变,其取值范围同样为 $(1, +\infty)$.PReLU 在修正偏移现象的同时调整函数在零点的值,使函数在整个定义域内连续可导,但变化的坡度因子在增加计算量的同时容易导致模型的过拟合.

### 1.4 RReLU 函数

RReLU 是 LReLU 的随机形式[11],其定义为

$$y_i = \begin{cases} x_i, & x_i \geqslant 0; \\ x_i/\alpha_i, x_i < 0. \end{cases}$$

式中坡度因子 $\alpha_i$ 从标准正态分布中随机取值,如

$$\alpha_i \sim U(l,u).$$

RReLU 在较小数据集上可一定程度上防止过拟合,但其在较大数据集上的表现并不理想.

### 1.5 ELU 函数

ELU 的定义为[12]

$$y_i = \begin{cases} x_i, & x_i \geqslant 0; \\ \alpha(\exp(x_i) - 1), x_i < 0. \end{cases}$$

ELU 对正数部分的处理与 ReLU 一致,而对负数部分的处理与 sigmoid 函数相似,Clevert 等[12]证明了 ELU 传递的常规梯度更接近自然梯度(nature gradient),而自然梯度是利用 Fisher 信息矩阵、Hessian 矩阵或者 Gauss-Newton 矩阵梯度求取的更新方向;神经网络后向传播中的梯度更接近自然梯度意味着迭代次数更少、学习速率更快、精度更高.但 ELU 在处理负数部分时存在软饱和现象,且其指数级变化使得饱和点更接近零点,梯度在饱和点之后的负数部分较难传递,梯度可传递范围较窄.

## 2　改进的 SLU 函数

由 ReLU 及相关改进单元可知,ReLU 存在零梯度问题和偏置现象,影响神经网络的收敛速度和学习效果;根据文献[12]所证“输出均值接近零的激活函数能够提升神经网络性能”原理,本文对 ReLU 进行改进,提出 SLU 单元,并在理论上说明 SLU 的优越性.文献[12]中对“输出均值接近零的激活函数能够提升神经网络性能”原理进行了详细的证明,因其过程较为晦涩繁琐,此处并不进行赘述.

Softplus 函数可以看作是 ReLU 函数的平滑校正[13-14],定义为

$$\mathrm{softplus}(x)=\log(\exp(x)+1), \tag{2}$$

对式(2)求导可得

$$\sigma(x)=\frac{1}{\exp(-x)+1}. \tag{3}$$

式(3)为饱和函数 Sigmoid 的定义.Softplus 与 ReLU 相似,具有单侧抑制、相对宽阔的兴奋边界等优点,如图 2 所示,相比于 ReLU,其优势在于其在定义域内连续可导,使得梯度可在整个定义域内传播,更接近生物特性.

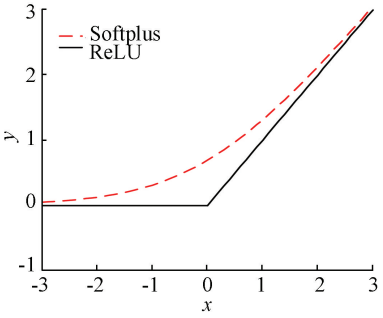


**图 2　Softplus 和 ReLU 函数**

Fig.2　Softplus and ReLU function

输入为负数时,ReLU 单元输出为零,导致零梯度问题;且正数输入时,单元输出为正,使得 ReLU 输出平均值为正数,存在偏移现象.本文利用 softplus 处理 ReLU 负数输入部分,使得负数输入时的输出为负,可以避免的零梯度问题,且一定程度上修正整个函数的输出接近于零.改进的修正线性单元函数 SLU 的一般形式可以定义为

$$\mathrm{SLU}(x)=\begin{cases}\beta x, & x\geqslant 0;\\ \alpha\log(\mathrm{e}^{x}+1)-\gamma, & x<0.\end{cases}$$

式中:$\beta$ 为坡度因子,$\beta$ 越大,正数部分坡度越陡,较陡的坡度在深度学习中容易导致梯度溢出;$\alpha$ 为影响负数部分的饱和位置,$\alpha$ 越大,饱和点越靠近零点,在训练过程中越容易导致梯度消散;$\gamma$ 为调节负数输入时,函数向下偏离横轴的距离,$\gamma$ 越大,偏离横轴的距离越大,一定范围内,激活函数的平均值越接近零.SLU 的导数为

$$\mathrm{SLU}'(x)=\begin{cases}\beta, & x\geqslant 0;\\ \alpha\dfrac{1}{(\mathrm{e}^{-x}+1)}, & x<0.\end{cases} \tag{4}$$

由式(4)可见,$\mathrm{SLU}'(x)$ 的负数部分为 Sigmoid 函数与 $\alpha$ 的乘积.为保证信息前向传播与梯度后向传播的连续平稳,令函数 $\mathrm{SLU}(x)$ 在 0 点处连续且可导,可得

$$\begin{cases}\mathrm{SLU}(x)\big|_{x=0^+}=0,\\ \mathrm{SLU}(x)\big|_{x=0^-}=\alpha\log 2-\gamma,\\ \mathrm{SLU}(x)\big|_{x=0^+}=\mathrm{SLU}(x)\big|_{x=0^-},\\ \mathrm{SLU}'(x)\big|_{x=0^+}=\beta,\\ \mathrm{SLU}'(x)\big|_{x=0^-}=\dfrac{\alpha}{2},\\ \mathrm{SLU}'(x)\big|_{x=0^+}=\mathrm{SLU}'(x)\big|_{x=0^-},\end{cases} \tag{5}$$

解式(5)可得

$$\begin{cases}\gamma=\alpha\log 2,\\ \beta=\dfrac{\alpha}{2}.\end{cases}$$

在 CNN 的后向传播过程中,经过多次非线性变换后,传递到网络底层的梯度会变大/变小,变化的速度是以激活函数导数为底、以非线性变换层数为指数的幂的值[7].为减缓梯度在传递过程中出现的消失/溢出现象,令 $\beta=1$,则 $\alpha=2$,$\gamma=2\log 2$,SLU 可精确定义为

$$\mathrm{SLU}(x)=\begin{cases}x, & x\geqslant 0;\\ 2\log\dfrac{\mathrm{e}^{x}+1}{2}, & x<0.\end{cases} \tag{6}$$

式(6)为 SLU 函数的精确形式,也是本文后续用于训练时的函数.

设参数概率模型为 $p(x;\omega)$,令 $x^+$ 为正数输入,$x^-$ 为负数输入,$\omega$ 为输入 $x$ 的概率.经过 SLU 进行非线性变换后,输出均值 $m_{\mathrm{SLU}}(x)$ 为

$$m_{\mathrm{SLU}}(x)=\sum\omega f_{\mathrm{SLU}}(x)=\sum\omega x^{+}+\sum\omega 2\log\frac{\mathrm{e}^{x^{-}}+1}{2}. \tag{7}$$

经过 ReLU 进行非线性变换后,输出均值 $m_{\mathrm{ReLU}}(x)$ 为

$$m_{\mathrm{ReLU}}(x)=\sum\omega f_{\mathrm{ReLU}}(x)=\sum\omega x^{+}. \tag{8}$$

显然,式(7)、(8)中的权重 $\omega$ 为正,所以 ReLU 的输出均值 $m_{\mathrm{ReLU}}(x)$ 始终为正;式(7)的第 1 项为正数,第 2 项为负数,一定程度上可以修正输出值接近于零.所以,相比于 ReLU 函数,SLU 的输出平均值更接近零.如图 3 所示,SLU 函数为黑色实线,ReLU 函数为红色线段,SLU 函数相比 ReLU 均值更接近于零.

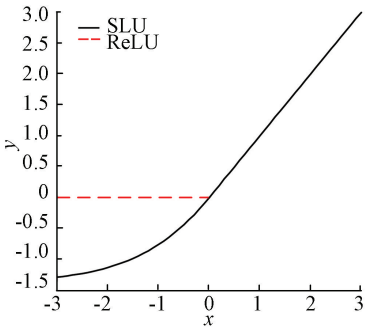


**图 3 SLU 和 ReLU 函数**

Fig.3 SLU and ReLU function

## 3 实验分析

本文分别从监督学习与非监督学习的角度对激活函数进行分析，首先构建深度自编码(deep autoencoder)模型在 MNIST 数据集上进行监督学习，利用模型的重建误差描述其对图像特征的学习能力；其次构建了网中网(network in network, NiN)卷积神经网络模型在 CIFAR-10 上进行非监督学习，利用模型的分类概率损失、top1 误差与 top5 误差描述模型的学习能力.

### 3.1 MNIST

本文构建了深度自编码模型对 MNIST 数据集进行无监督学习.MNIST 手写数字识别数据集含有 0~9 共 10 类手写阿拉伯数字，包括 60 000 个训练数据和 10 000 个测试数据，图像灰度级为 8，分辨率为 28×28[15].深度自编码模型的自编码部分共有 4 层编码层，单元数分别设置为 1 000、500、250 和 30，每层编码层后面对应相应的解码层；利用随机梯度下降法训练模型，对编码后的数据进行批量规范化，每批大小为 128；利用 Dropout 以增强稀疏性和鲁棒性，概率设置为 0.2；学习速率设置为0.001，动量设置为 0.9.模型在 MNIST 数据集上的重建误差如图 4 所示.

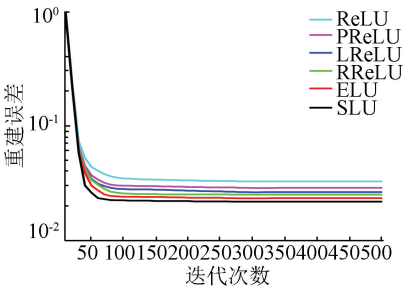


**图 4 不同激活函数下深度自编码重建误差**

Fig.4 Reconstruction error of deep atouencoder with different activations

由图 4 可知，与其他模型相比，基于 SLU 模型的重建误差最小，重建精度最高，误差在迭代 80 余次后趋于平稳，收敛速度稍快，说明其特征学习能力较好；其余模型的重建误差由低到高依次为 ELU、RReLU、LReLU、PReLU 与 ReLU，且 ReLU 在迭代接近 100 次时，重建误差趋于平稳，收敛速度稍慢.在数据集 MINST 上的深度自编码学习验证了基于 SLU 模型的无监督学习能力.

### 3.2 CIFAR-10

本文构建了 NiN[16] 模型在数据集 CIFAR-10[17-18] 上进行监督学习.CIFAR-10 含有 60 000 张彩色图像，包括 50 000 张训练图像和 10 000 测试图像，可被分为 10 类，图片分辨率为 32×32.

#### 3.2.1 模型构建及参数设置

NiN 模型是常用的深度卷积神经网络模型之一.与一般 CNN 模型不同的是，NiN 利用多层感知层进行非线性变换，再与卷积层和池化层共同组成 NiN 区块[16].本文构建的 NiN 模型含有 3 个 NiN 区块与一个分类层，3 个 NiN 区块依次相连学习图像，最后利用 softmax 分类层进行分类.每个 NiN 区块含有 4 层网络，依次为卷积层、两层感知层和池化层，每个 NiN 区块后设置规则化层.NiN 模型架构见表 1.

**表 1 NiN 模型架构**

Tab.1 NiN network structure

| 层次 | 输入尺寸 | 核尺寸 | 补充像素 | 滑动像素 |
|---|---|---|---|---|
| 卷积 | 32×32 | 5×5 | 2 | 1 |
| 感知层-1 | 32×32 | 1×1 | 0 | 1 |
| 感知层-2 | 32×32 | 1×1 | 0 | 1 |
| 最大池化 | 32×32 | 3×3 | 2 | 2 |
| Dropout | 16×16 | 0.5 | 0 | 0 |
| 卷积 | 16×16 | 5×5 | 2 | 1 |
| 感知层-1 | 16×16 | 1×1 | 0 | 1 |
| 感知层-2 | 16×16 | 1×1 | 0 | 1 |
| 平均池化 | 16×16 | 3×3 | 2 | 2 |
| Dropout | 8×8 | 0.5 | 0 | 0 |
| 卷积 | 8×8 | 3×3 | 2 | 1 |
| 感知层-1 | 8×8 | 1×1 | 0 | 1 |
| 感知层-2 | 8×8 | 1×1 | 0 | 1 |
| 平均池化 | 8×8 | 8×8 | 2 | 2 |
| Softmax | 10×1 | 分类 | 0 | 0 |

权重衰减值设置为 0.000 5，最小批规范化大小为 100，动量设置为 0.9；学习速率初始值为 0.5，并在 30 次迭代后降为 0.1，40 次迭代后降为 0.02；非线性变换阶段分别利用 ReLU、LReLU、PReLU、RReLU、ELU、SLU 这 6 种激活函数进行处理，并令 LReLU 的坡度因子 $\alpha_i = 5.5$ [11].

#### 3.2.2 实验过程与结果分析

利用模型构建及参数设置中所述 NiN 模型及其参数在数据集 CIFAR-10 上进行监督训练，记录基于不同激活函数的 NiN 模型的分类概率损失、top1 和 top5 误差，比较 ReLU 与 SLU 的迭代过程如图 5 所示；比较 LReLU 与 SLU 的迭代过程如图 6 所示；比较 PReLU 与 SLU 的迭代过程如图 7 所示；比较 RReLU 与 SLU 的迭代过程如图 8 所示；比较 ELU 与 SLU 的迭代过程如图 9 所示.

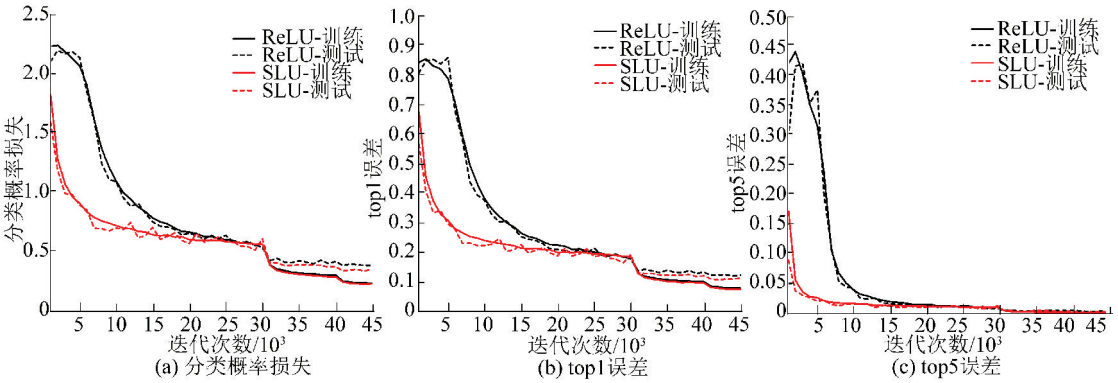


**图 5　基于 ReLU 与 SLU 函数 NiN 模型的迭代过程比较**

Fig.5　Convergence curves for training and test sets of ReLU and SLU activation based models

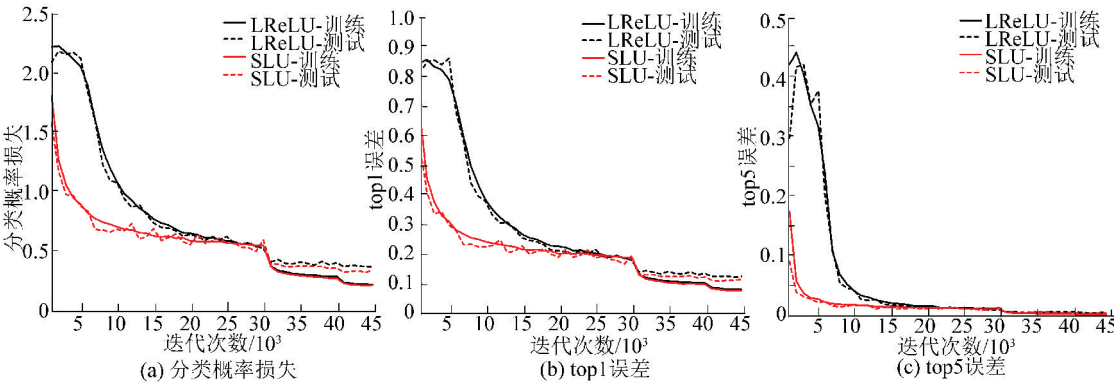


**图 6　基于 LReLU 与 SLU 函数 NiN 模型的迭代过程比较**

Fig.6　Convergence curves for training and test sets of LReLU and SLU activation based models

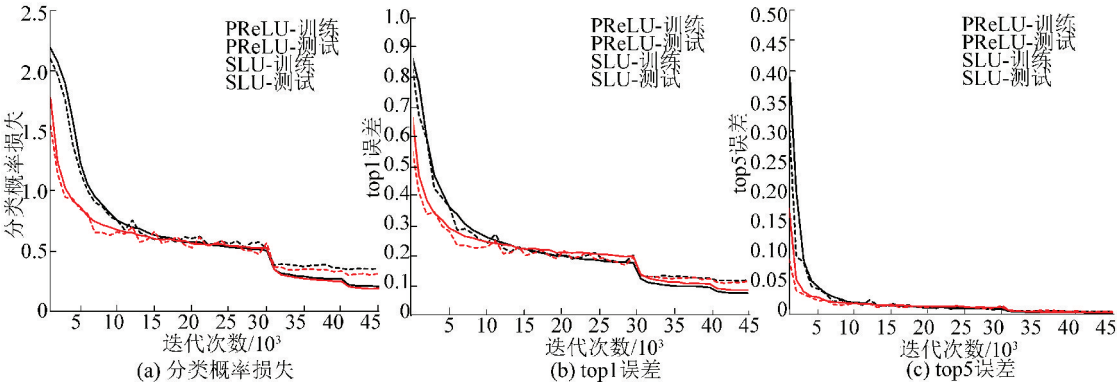


**图 7　基于 PReLU 与 SLU 函数 NiN 模型的迭代过程比较**

Fig.7　Convergence curves for training and test sets of PReLU and SLU activation based models

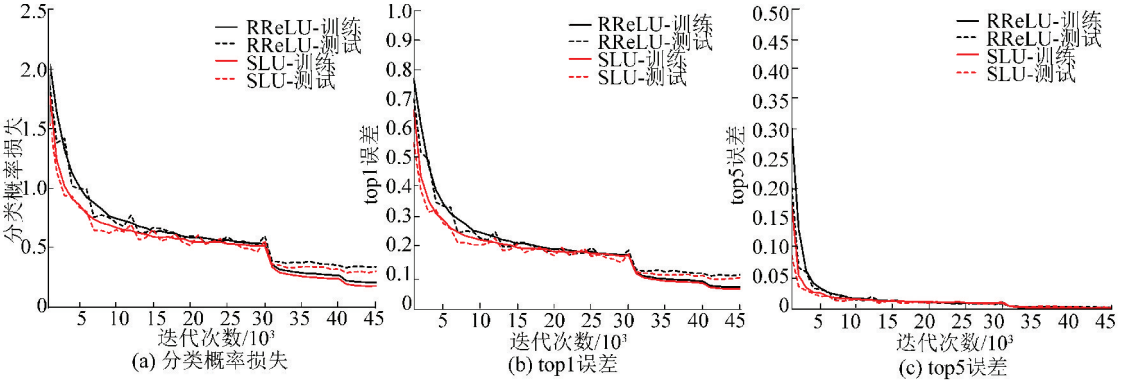


**图 8　基于 RReLU 与 SLU 函数 NiN 模型的迭代过程比较**

Fig.8　Convergence curves for training and test sets of RReLU and SLU activation based models

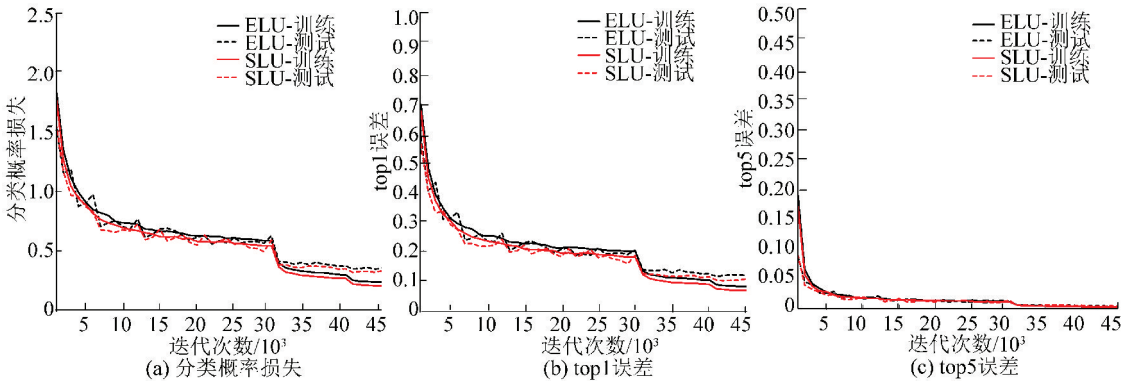


**图 9 基于 ELU 与 SLU 函数 NiN 模型的迭代过程比较**

Fig.9 Convergence curves for training and test sets of ELU and SLU activation based models

由图 5~9 可知,相比于 ReLU、LReLU、PReLU、RReLU 和 ELU 函数,SLU 的收敛速度均有不同程度的加快,图 5 中所示与 ReLU 函数的对比尤为明显;与 ReLU、LReLU、RReLU 和 ELU 函数相比,SLU 分类概率损失、top1 和 top5 的训练误差与测试误差均较小;与 PReLU 函数相比,SLU 分类概率损失、top1 和 top5 的测试误差均较小,但训练误差稍大,这是因为 PReLU 模型容易导致过拟合,所以 PReLU 训练误差较小.

基于不同激活函数 NiN 模型的最终分类概率损失的训练值与测试值见表 2;top1 训练误差与测试误差见表 3;top5 训练误差与测试误差见表 4.

**表 2 基于不同激活函数 NiN 模型的分类概率损失**

Tab.2 Classification loss of NiN models

| 激活函数 | 训练损失 | 测试损失 |
|---|---|---|
| ReLU | 0.23 | 0.36 |
| LReLU | 0.24 | 0.36 |
| PReLU | **0.18** | 0.35 |
| RReLU | 0.24 | 0.38 |
| ELU | 0.22 | 0.35 |
| SLU | 0.19 | **0.34** |

**表 3 基于不同激活函数 NiN 模型的 top1 误差**

Tab.3 Top1 error rate of NiN models

| 激活函数 | 训练误差/% | 测试误差/% |
|---|---|---|
| ReLU | 3.19 | 12.46 |
| LReLU | 3.62 | 11.21 |
| PReLU | **1.76** | 11.81 |
| RReLU | 5.51 | 11.19 |
| ELU | 2.35 | 10.56 |
| SLU | 3.16 | **10.02** |

**表 4 基于不同激活函数 NiN 模型的 top5 误差**

Tab.4 Top5 error rate of NiN models

| 激活函数 | 训练误差/% | 测试误差/% |
|---|---|---|
| ReLU | 0.15 | 0.42 |
| LReLU | 0.16 | 0.39 |
| PReLU | **0.10** | 0.46 |
| RReLU | 0.18 | 0.35 |
| ELU | 0.14 | 0.33 |
| SLU | 0.13 | **0.28** |

由表 2~4 可知,所有 NiN 网络不同指标下的训练误差均比测试误差小,说明网络在训练过程中有过拟合现象,尤其是 PReLU 函数,其在分类概率损失、top1 误差与 top5 误差的训练指标均为最小,但测试指标相对较高,过拟合现象较其他激活函数严重.利用 ReLU 激活函数的 NiN 网络的训练误差与测试误差均最高,可见其他 5 类激活函数在 ReLU 的基础上都有不同程度的改进,基于文中所提 SLU 函数的 NiN 网络的分类概率损失、top1 误差与 top5 误差的测试指标分别为 0.34%、10.02%、0.28%,均小于基于其他 5 种函数的测试误差,且相比其余最好性能的 ELU 激活函数误差分别减少了 2.9%、5.1%、4.5%.显然,模型增加的参数与步骤提高了模型精度,但也增加了时间消耗.本文记录了基于不同激活函数的 NiN 模型迭代至收敛的时间进行比较,并比较了调用激活函数的次数和时间,见表 5.需要说明的是,实验所用处理器为 Intel(R) Core(TM) i5-4590, CPU 主频为 3.3 GHz,显卡为 AMD 系列,实验并未调用 GPU.实验时间仅作为本文所有实验横向比较参考依据.

**表 5 基于不同激活函数 NiN 模型的时间消耗**

Tab.5 Time-consuming of NiN models s

| 激活函数 | 总消耗时间 | 每千次平均消耗时间 | 激活函数总调用时间 | 激活函数平均调用时间 |
|---|---|---|---|---|
| ReLU | 58 683 | 13 041 | 30 521 | 0.068 5 |
| LReLU | 68 857 | 15 302 | 37 451 | 0.084 1 |
| PReLU | 68 932 | 15 318 | 37 534 | 0.084 3 |
| RReLU | 69 133 | 15 363 | 37 623 | 0.084 5 |
| ELU | 69 090 | 15 353 | 37 579 | 0.084 4 |
| SLU | 69 178 | 15 373 | 37 659 | 0.084 5 |

注:激活函数的调用次数均为 445 500 次

由表 5 可知,与 ReLU 相比,其改进函数 LReLU、PReLU、RReLU、ELU 和 SLU 模型的运行总时间均有不同程度的增加,但 5 种改进的函数相互之间增加的时间与总时间相比并不明显,平均每迭代千次耗时分别为 13 041、15 302、15 318、15 363、15 353、15 373 s;6 种模型对激活函数的调用次数

均为 445 500 次,平均调用时间分别为 0.068 5、0.084 1、0.084 3、0.084 5、0.084 4、0.084 5 s.选取基于不同激活函数的 NiN 模型每迭代千次的时间进行比较,如图 10 所示.

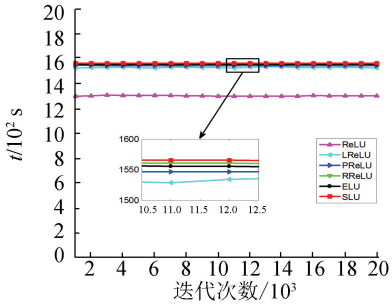


**图 10　基于不同激活函数 NiN 模型时间消耗**

Fig.10　The time of NiN models with different activations

通过表 5 与图 10 可知,相比于其他改进函数,SLU 的消耗时间并没有明显增加.这是因为 SLU 模型虽然增加了参数,带来了一定的时间消耗,但是对比模型的总参数量及时间消耗,增加的时间消耗极微.考虑到模型对精度的提升,本文认为这样的时间消耗是值得的.

## 4　结　论

1)本文针对 ReLU 函数存在的零梯度问题和偏移现象,根据"输出均值接近零的激活函数能够提升神经网络学习性能"原理提出 SLU,利用 softplus 对 ReLU 的负数部分进行改进,再根据 SLU 对正数部分的处理调整负数部分的参数,最后建立了深度自编码模型和网中网模型,分别从无监督和有监督方面验证了改进单元的学习能力.

2)实验结果表明:基于 SLU 的模型的收敛速度和学习精度均比其他几类算法有不同程度的提升;相比于整个模型而言,函数增加的参数并没有消耗过多的时间.基于 SLU 函数的模型具有较好的特征学习能力和较高的分类精度.

## 参考文献

[1]KRIZHEVSKY A, SUTSKEVER I, HINTON G E. Imagenet classification with deep convolutional neural networks [C]// Proceedings of the 25th International Conference on Neural Information Processing Systems. Lake Tahoe, Nevada: Curran Associates Inc., 2012: 1097-1105. DOI: 10.1145/3065386.

[2]SZEGEDY C, LIU Wei, JIA Yangqing, et al. Going deeper with convolutions[C]//Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition. Boston, MA: IEEE, 2015:1-9. DOI: 10.1109/CVPR.2015.7298594.

[3]GIRSHICK R, DONAHUE J, DARRELL T, et, al. Rich feature hierarchies for accurate object detection and semantic segmentation [C]//Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition. Columbus, OH: IEEE, 2014: 580-587.DOI: 10.1109/CVPR.2014.81.

[4]WANG Naiyan, LI Siyi, GUPTA A, et al. Transferring rich feature hierarchies for robust visual tracking[EB/OL].(2015-01-19)[2015-04-23].http://arxiv.org/abs/1501.04587.

[5]SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[EB/OL].(2014-09-04)[2015-04-10]. http://arxiv.org/abs/1409.1556.

[6]HE Kaiming, ZHANG Xiangyu, REN Shaoqing, et al. Deep residual learning for image recognition[EB/OL].(2015-12-10).http://arxiv.org/abs/1512. 03385.

[7]TROTTIER L, GIGUÈRE P, CHAIB-DRAA B. Parametric exponential linear unit for deep convolutional neural networks[EB/OL].(2016-05-30)[2018-01-10]. http://arxiv.org/abs/1605.09332.

[8]NAIR V, HINTON G E. Rectified linear units improve restricted boltzmann machines [C]//Proceedings of the 27th International Conference on Machine Learning. Haifa, Israel:,Omnipress, 2010: 807-814.

[9]MAAS A L, HANNUN A Y, NG A Y. Rectifier nonlinearities improve neural network acoustic models[C]//Proceeding of the 30th International Conference on Machine Learning. Atlanta, GA:[s.n.], 2013.

[10]HE Kaiming, ZHANG Xiangyu, REN Shaoqing, et al. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification[C]//Proceedings of the IEEE International Conference on Computer Vision. Santiago, Chile: IEEE, 2015: 1026-1034. DOI:10.1109/ICCV.2015.123.

[11]XU Bing, WANG Naiyan, CHEN Tianqi, et al. Empirical evaluation of rectified activations in convolutional network [EB/OL].(2015-05-05)[2015-11-27].http://arxiv.org/abs/1505.00853.

[12]CLEVERT D A, UNTERTHINER T, HOCHREITER S. Fast and accurate deep network learning by exponential linear units (elus) [EB/OL].(2015-11-23) [2016-02-22].http://arxiv.org/abs/1511.07289.

[13]GLOROT X, BORDES A, BENGIO Y. Deep sparse rectifier neural networks[C]//Proceeding of the 14th International Conference on Artificial Intelligence and Statistics. Fort Landerdale, FL:[s.n.], 2011:315-323.

[14]SENIOR A, LEI Xin. Fine context, low-rank, softplus deep neural networks for mobile speech recognition[C]// Proceeding of IEEE International Conference on Acoustic, Speech and Signal Processing. Florence, Italy: IEEE, 2014:7644-7648. DOI: 10.1109/ICASSP.2014.6855087.

[15]LECUN Y, BOTTOU L, BENGIO Y, et al. Gradient-based learning applied to document recognition [J]. Proceedings of the IEEE, 1998, 86(11):2278-2324.

[16]LIN Min, CHEN Qiang, YAN Shuicheng. Network in network[EB/OL].(2013-12-16) [2014-03-04] http://arxiv.org/abs/1312.4400.

[17]LEE C Y, XIE S, GALLAGHER P W, et al. Deeply-supervised nets[C]//Proceeding of the 18th International Conference on Artificial Intelligence and Statistics. San Diego,California:[s.n.],2015.

[18]刘凯, 张立民, 范晓磊. 改进卷积波尔兹曼机的图像特征深度提取[J]. 哈尔滨工业大学学报, 2016, 48(5): 155-159. DOI: 10.11918/j.issn.0367-6234.2016.05.025.
LIU Kai, ZHANG Limin, FAN Xiaolei. New image deep feature extraction based on improved CEBM[J]. Journal of Harbin Institute of Technology, 2016, 48(5): 155-159. DOI: 10.11918/j.issn.0367-6234.2016.05.025.

(编辑　张　红)