面向 CNN 模型图像分类任务的高效激活函数设计

杜圣杰，贾晓芬，黄友锐，郭永存，赵佰亭

(安徽理工大学 电气与信息工程学院，安徽 淮南 232000)

摘　要：激活函数 (Activation Functions，AF) 对于卷积神经网络学习、拟合复杂函数模型来说具有十分重要的作用，为了使神经网络能更好更快的完成各类学习任务，设计了一种新型高效激活函数 EReLU。EReLU 通过引入自然对数函数有效缓解了神经元“坏死”和梯度弥散问题，通过分析激活函数及其导函数在前馈和反馈过程中的作用对 EReLU 函数的数学模型探索设计，经测试确定 EReLU 函数的具体设计方案，最终实现了提升精度和加速训练的效果；随后在不同网络和数据集上对 EReLU 进行测试，结果显示 EReLU 相较于 ReLU 及其改进函数精度提升 0.12%~6.61%，训练效率提升 1.02%~6.52%，这有力地证明了 EReLU 函数在加速训练和提升精度方面的优越性。

关键词：图像分类；　高效激活函数；　神经元“坏死”；　卷积神经网络

中图分类号：TP391　　**文献标志码：**A　　**DOI：**10.3788/IRLA20210253

High efficient activation function design for CNN model image classification task

Du Shengjie，Jia Xiaofen，Huang Yourui，Guo Yongcun，Zhao Baiting

(School of Electrical and Information Engineering, Anhui University of Science and Technolog, Huainan 232000, China)

Abstract: Activation Functions (AF) play a very important role in learning and fitting complex function models of convolutional neural networks. In order to enable neural networks to complete various learning tasks better and faster, a new efficient activation function EReLU was designed in this paper. By introducing the natural logarithm function, EReLU effectively alleviated the problems of neuronal "necrosis" and gradient dispersion. Through the analysis of the activation function and its derivative function in the feedforward and feedback process of the mathematical model of the EReLU function exploration and design, the specific design of the EReLU function was determined through test, and finally the effect of improving the accuracy and accelerating training was achieved; Subsequently, EReLU was tested on different networks and data sets, and the results show that compared with ReLU and its improved function, the accuracy of EReLU is improved by 0.12%-6.61%, and the training efficiency is improved by 1.02%-6.52%, which strongly proved the superiority of EReLU function in accelerating training and improving accuracy.

Key words: image classification; high efficiency activation function; neurons "necrosis"; convolutional neural network

---

收稿日期:2021−12−10;　**修订日期:**2022−01−25

基金项目:安徽省自然科学基金面上项目 (2108085ME158); 国家自然科学基金面上项目 (52174141); 安徽高校协同创新项目 (GXXT-2020-54); 安徽省重点研究与开发计划 (202004a07020043)

作者简介:杜圣杰，男，硕士生，主要从事图像处理、深度学习方面的研究。

导师简介:贾晓芬，女，教授，博士，主要从事图像处理、深度学习和机器学习方面的研究。

0 引 言

激活函数 (Activation Functions, AF) 的主要作用是将非线性因素引入卷积神经网络 (Convolutional Neural Network, CNN), 为整个模型提供充足的非线性扭曲力, 进而帮助 CNN 更好的理解和拟合复杂函数模型, 完成各项计算机视觉任务。

CNN 早期发展阶段常用的激活函数有 Sigmoid[1]、Tanh[2] 函数, 二者均属于 S 型饱和函数, 这种函数容易导致梯度消失, 使得模型训练困难。为了解决这一问题, Krizhevsky[3] 提出了一种线性整流单元 (Rectified Linear Units, ReLU) 并在当年的 ImageNet ILSVRC 比赛中取得了出色的成绩, 相较于 Sigmoid 和 Tanh 函数, ReLU 函数具有良好的稀疏性和较小的计算量, 它不仅解决了梯度消失问题, 还加快了网络训练速度, 因此 ReLU 函数很快便成为 CNN 网络中常用的主流激活函数, 但 ReLU 函数同样存在着一些缺陷, 即 ReLU 函数容易在训练过程中导致神经元“坏死”[3], 进而使“坏死”的神经元在整个训练过程中失去传递信息的能力, 对模型产生不利的影响。

为了解决上述问题, 基于 ReLu 函数的改进型激活函数出现了。如 Dubey A K[4] 等提出的 LeakyReLU 激活函数在函数负半段设计了泄露单元, 有效缓解了神经元“坏死”问题; He K[5] 等人提出的非线性修正激活函数 PReLU 通过引入额外的可学习参数不仅很好的解决了“坏死”问题, 还有效提高了模型的拟合能力; Clevert D A[6] 等提出的 ELU 激活函数同样取得了比 ReLU 函数更优越的性能; 上述改进型函数均主要针对 ReLU 负半轴的零常函数做出改动, 进而有效的弥补了 ReLU 函数的缺陷, 但它们同样存在新的问题, 如延长了训练时间、提高了训练难度等。

随后, 人们想到通过合并多个函数优点的思路来构建新的激活函数。如石琪[7] 等提出的组合激活函数 ReLU-Softplus, 王红霞[8] 等提出的 ReLU-Softsign 激活函数均获得了比单一激活函数更好的效果, 组合函数采取优势互补的思路实现了更好的性能提升, 但在实际应用中, 组合函数同样存在一些新的问题, 如 ReLU-Softplus 对学习率的设置要求严格, ReLU-Softsign 在速度和精度方面仍有待提升。

综上, ReLU 函数具有良好的稀疏性和运算效率, 但本身存在神经元“坏死”问题; LeakyReLU 等改进函数通过在负半轴引入非零函数改善了 ReLU 的缺陷, 却给自身带来效率慢的问题; 组合函数虽然把不同激活函数优点合并在一起, 但又产生了训练困难的新问题。因此, 对当前 CNN 网络来说设计一个既能解决以往激活函数存在的梯度弥散、神经元“坏死”等缺陷, 又能加速训练、提升性能且不增加训练难度的高效激活函数是非常必要的。

1 激活函数训练过程分析

卷积神经网络 (CNN) 引入激活函数的主要目的是给模型引入非线性因素, 从而使 CNN 模型能实现更好的学习效果, 那么激活函数是怎样对 CNN 模型产生作用的呢? 下面通过推导激活函数在网络训练过程中的具体计算过程分析说明。

CNN 训练通常包括前向和反向传播两个过程[8], 前向传播是指输入信号通过一个或多个网络层进行传递, 最后在输出层得到实际输出的过程; 反向传播则是模型根据实际输出和期望输出计算出误差损失, 然后通过每层的误差损失推导参数的学习规则, 选择更优的参数使实际输出更接近期望值的过程。假设有一个 l 层的网络, 给定一个输入 x_i, 前向传播时上一层的输出经过第 l 层卷积过程如下:

$$y_j^l = \sum_{i \in M_j} x_i^{l-1} k_{ij}^l + b_j^l \tag{1}$$

式中: i 表示第 l-1 层的第 i 个通道; j 表示第 l 层的第 j 个通道; M 表示一个内含 j 个元素的集合; x_i^{l-1}为第 l-1 层的输入; k_{ij}^l为第 l 层的卷积核, b_i^l为第 l 层的偏置项; y_j^l为上层输入经过第 l 层卷积计算的输出。随后把y_j^l代入任意激活函数 f 中去, 计算如下:

$$x_j^l = f(y_j^l) \tag{2}$$

其中, x_j^l是一个完整前向传播的输出结果, 这里 f 的主要作用是将非线性因素引入到网络中去, 使得 CNN 中上下层卷积层不再成线性关系, 进而促进网络更好的学习和拟合函数模型[9], 由于每层网络后都有激活函数参与计算, 所以激活函数计算是否简便, 求导是否方便将对模型的训练速度产生极大的影响[10]。

而反向传播实际上是一个负反馈过程, f 在反向传播过程中参与的计算为:

(1) 计算第 l 层的灵敏度

$$\lambda_j^l = \frac{\partial E}{\partial y_j^l} = \beta_j^{l+1}(f'(u_j^l) \circ up(\lambda_j^{l+1})) \tag{3}$$

式中：λ_j^l为第 l 层的灵敏度；E 为误差损失；β_j^{l+1}为第 l+1 层里下采样层的权值参数；f'为 f 的导函数；$\circ$表示对每个元素作乘法操作；up 表示上采样操作。

(2) 对灵敏度求偏导

$$\begin{aligned} \frac{\partial E}{\partial b_j^l} &= \sum_{u,v} (\lambda_j^l)_{u,v} \\ \frac{\partial E}{\partial k_{ij}^l} &= \sum_{u,v} (\lambda_j^l)_{u,v} (p_i^{l-1})_{u,v} \end{aligned} \tag{4}$$

式中：k 为卷积核参数；b 为偏置项；u, v 表示求偏导过程；$(p_i^{l-1})_{u,v}$为计算x_j^l的过程中与k_{ij}^l对应相乘后的x_i^{l-1}。

(3) 更新卷积层参数 k 和 b

$$\begin{aligned} \Delta k_{ij}^l &= -\eta \frac{\partial E}{\partial k_{ij}^l} \\ \Delta b^l &= -\eta \frac{\partial E}{\partial k_{ij}^l} \end{aligned} \tag{5}$$

式中：η为学习率。由反向传播过程知，调优过程中参数的更新速度和方向均与 f 的导函数 f' 有关，因此导函数 f' 是否计算方便将直接影响信息在网络中的流动速度，结合前向传播可知，激活函数在网络中每一层都会被频繁使用，所以激活函数的导函数求解是否方便、计算是否简单同样对网络的效率有着极大的影响。

结合公式 (1) 可知，改善 ReLU、LeakyReLU 等函数缺陷的方式主要是通过优化激活函数的负半轴实现，而在函数负半轴引入非零函数的同时，还要注意避免因函数本身运算复杂度而导致的计算效率低和训练困难问题。

2 高效激活函数设计

ReLU[3] 函数由于其负半轴函数为零而存在神经元“坏死”缺陷；LeakyReLU[4] 等函数以及组合函数虽然以不同的方式在弥补了 ReLU 的缺陷，但由于函数内部引入了额外的可训练参数或指数型复杂运算而造成了网络效率低、训练难的问题，因此从激活函数负半轴数学模型出发，设计了包括线性型、幂数型、分数型及自然对数型函数在内的四种函数进行实验，以寻找最佳的激活函数设计方案。

算法复杂度[11] 由算法的时间复杂度和算法的空间复杂度两部分决定，其中时间复杂度与函数阶次相关，影响计算效率；空间复杂度与函数中包含的参数数量相关，决定了占用硬件资源情况。常见的函数复杂度从低到高依次为常数阶<线性阶<对数阶、分数阶、幂数阶<平方阶<指数阶<含超参函数，其中简单的线性函数、幂函数、分数函数及自然对数函数相较于引入了指数运算及额外参数的激活函数在计算上更加方便，这对于加快网络训练效率具有一定的优势。为了方便描述，上述四种激活函数分别用 f_1、f_2、f_3、f_4 表示，对应的导函数分别用 f_1’、f_2’、f_3’、f_4’表示，其中四种激活函数表达式见表 1。

表 1 四种激活函数数学模型

Tab.1 Mathematical models of four activation functions

Function	Function model
f_1	$f_1(x) = \begin{cases} x, x \geqslant 0 \\ -x, x < 0 \end{cases}$
f_2	$f_2(x) = \begin{cases} x \quad, x \geqslant 0 \\ -\frac{2}{3}(-x)^{\frac{3}{2}}, x < 0 \end{cases}$
f_3	$f_3(x) = \begin{cases} x \ , x \geqslant 0 \\ \frac{x}{1-x}, x < 0 \end{cases}$
f_4	$f_4(x) = \begin{cases} x \quad, x \geqslant 0 \\ -\ln^{1-x}, x < 0 \end{cases}$

表中，f_1 函数的设计思路是根据 ReLU 正半轴的线性函数反向延长至整个定义域上的完整线性函数，在继承 ReLU 函数优点的基础上解决了其原来存在的神经元“坏死”缺陷；f_2 函数的设计思路是通过将原 ReLU 函数的零负半轴用简单的幂函数代替后构成的，这里幂参数及系数的设置是为了保证其导函数系数为最简形式；f_3 函数则是通过将原 ReLU 函数的零负半轴用简单的分数函数代替后构成的，分式形式是为了保证其导函数系数为最简形式且在整个定义域上连续；f_4 函数是通过将原 ReLU 函数的零负半轴用自然对数函数代替后构成的，自然对数函数时间复杂度和空间复杂度相对较低，在计算和求导方面均具有一定的优势，其系数设置同样是为了保证其导函数系数最简且在整个定义域上连续。激活函数 f_1、f_2、f_3、f_4 对应的图像如图 1 所示。

f_1、f_2、f_3、f_4 均解决了 ReLU 函数在负半轴上图像为零的问题，有效改善了信息在网络前向和反向传播过程中的完整性，进而提高模型拟合能力，在一定程度上有效提升模型精度。f_1 为线性函数，f_2 为幂函数，

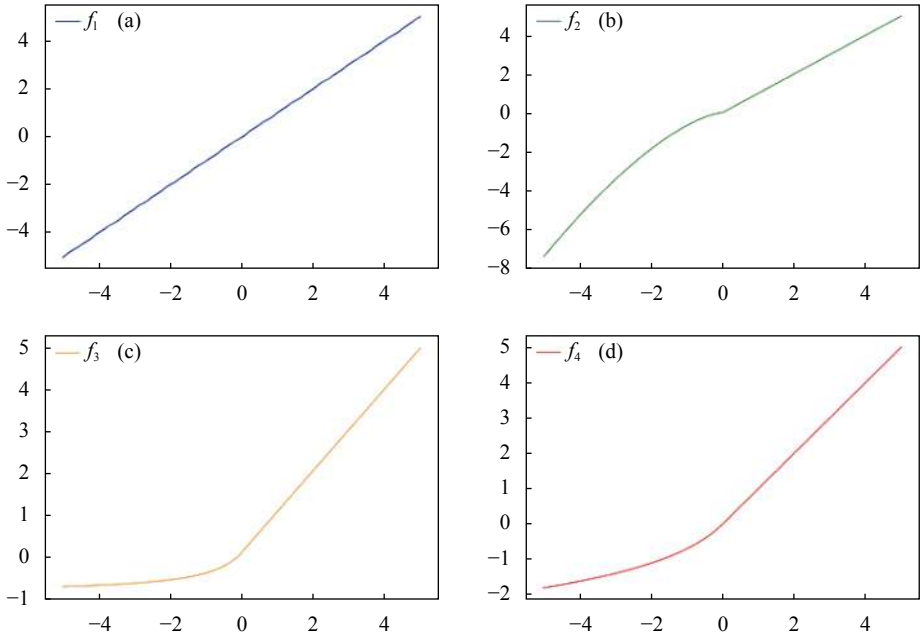

图 1 (a)f_1、(b)f_2、(c)f_3、(d)f_4 函数图像

Fig.1 (a) f_1, (b) f_2, (c)f_3, (d) f_4 functions and images

f_3 为一阶分数函数，f_4 为自然对数函数，时间复杂度上，$f_1 < f_2, f_3, f_4$，相较于包含指数运算 (如 ELU) 的激活函数更低，这决定了上述激活函数在计算方面更高效；在空间复杂度方面，上述函数相较于添加了额外参数 (如 LeakyReLU) 的函数在空也更低，这决定了上述激活函数在占用硬件空间资源方面更少。

函数求导是否方便同样影响模型计算效率，常见函数类型有线性函数、幂函数、分数函数、对数函数、指数函数等，其中线性型函数求导过程为：

$$f_{线}' = (kx)' = k \tag{6}$$

幂型函数求导过程为：

$$f_{幂}' = kx^{\alpha} = \alpha k x^{\alpha-1} \tag{7}$$

分数型函数求导过程为：

$$f_{分}' = \left(\frac{\alpha}{kx}\right)' = -\frac{\alpha}{x^2} \tag{8}$$

自然对数型函数求导过程为：

$$f_{对}' = (\alpha \ln^{kx})' = \frac{\alpha}{x} \tag{9}$$

指数型函数求导过程为：

$$f_{指} = (\alpha e^{kx})' = k\alpha e^{kx} \tag{10}$$

这里，k、α、e 表示函数参数。由分析可知，线性函数求导时主要是对系数 k 的计算，超参数个数为 1；幂函数求导时同时对参数 k 和 α 进行计算，参数个数为 2；分数型函数求导时除了计算参数 α，还要对分式函数二次求导，参数个数为 1；对数函数求导时仅计算参数 α，参数个数为 1；指数函数求导时不仅要计算系数 k、α 和 e，同时需要二次求导，超参个数为 2，而含有额外超参的激活函数中超参个数更多，由此可见，线性函数、幂函数和自然对数函数在求导方面效率较高，分数函数次之，指数函数效率相对最低。对 f_1、f_2、f_3、f_4 依次求导得到相应的导函数 f_1'、f_2'、f_3'、f_4'，四种导函数模型如表 2 所示。

表 2 四种激活函数导函数模型

Tab.2 Four kinds of activation function derivative function model

Derived function	Function model
f_1'	$f_1'(x) = \begin{cases} 1, x \geqslant 0 \\ -1, x < 0 \end{cases}$
f_2'	$f_2'(x) = \begin{cases} 1 & , x \geqslant 0 \\ -\sqrt{(-x)}, & x < 0 \end{cases}$
f_3'	$f_3'(x) = \begin{cases} 1 & , x \geqslant 0 \\ \dfrac{1}{(1-x)^2} & , x < 0 \end{cases}$
f_4'	$f_4'(x) = \begin{cases} 1 & , x \geqslant 0 \\ \dfrac{1}{1-x}, & x < 0 \end{cases}$

可见，f_1'、f_2'、f_3'、f_4'解决了 ReLU 导函数在负半轴上图像为零的问题，促进了信息在网络中传递，进

而达到改善模型精度的目的。f_1'为常函数,f_2'为幂函数,f_3'为二阶分数函数,f_4'为一阶分数函数,时间复杂度上,f_1' < f_2', f_4' < f_3'且均低于包含指数运算 (如 ELU) 的激活函数;空间复杂度上,f_1' < f_2', f_3', f_4'且均低于引入了额外超参数 (如 LeakyReLU) 的激活函数,综上可知,上述导函数相较于 ReLU 及其改进型函数的导函数算法复杂度更低,计算效率更快,相应地占用硬件资源也更少。四种导函数图像如图 2 所示。

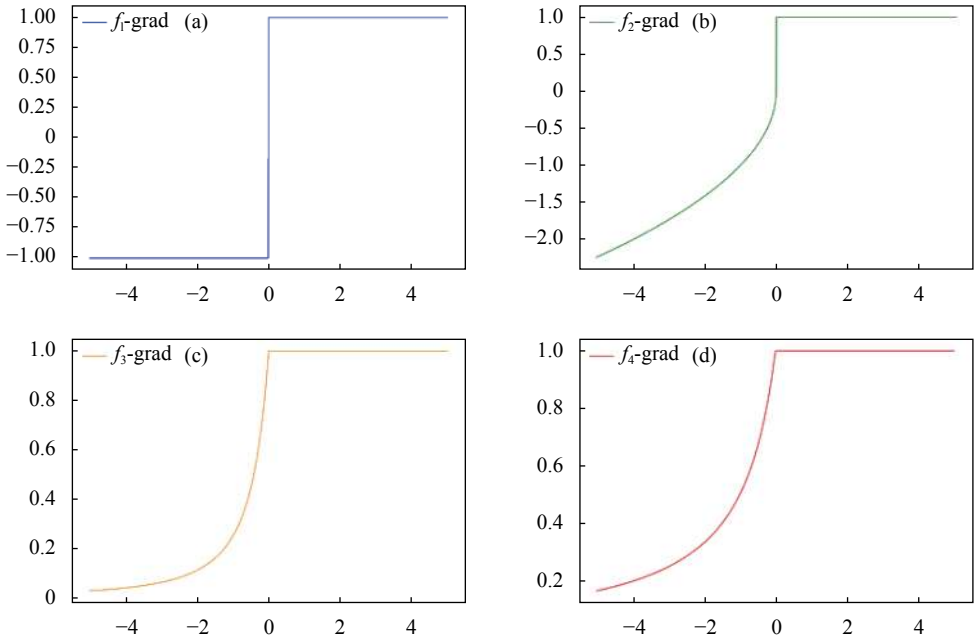

图 2 (a)f_1、(b)f_2、(c)f_3、(d)f_4 导函数图像

Fig.2 Derivatives of (a)f_1, (b)f_2,(c)f_3,(d)f_4 and their graphs

结合 CNN 模型前向和反向传播过程,由于上述四种激活函数及其导函数相较于已有激活函数 (如 ELU、LeakyReLU) 在时间和空间复杂度上更低,因此 f_1、f_2、f_3、f_4 四种函数理论上在促进网络中信息传递和加速训练方面相较于其它基于 ReLU 的改进函数效果更好,接下来通过相应实验来验证各激活函数的具体性能。

3 实验及结果分析

实验在 CIFAR10[12]、CIFAR100[12] 及 Fer2013[13] 数据集上开展。数据集 CIFAR10 和 CIFAR100 均是 32×32 的彩色图像,分别包含 10、100 个类别,Fer2013 为细粒度分类数据集,由 35 886 张 48×48 人脸表情图片组成,共 7 类。实验均在 Pytorch1.5.0、Python3.8、NVIDIA GTX 2080 显卡环境下进行,完整迭代轮数均为 120 个 Epoch,实验中学习率设置为 0.001,优化器采用随机梯度下降+动量法 (SGD+Moument)。使用准确率 ACC(%) 和训练时长 T/h 作为评价指标用以评估不同激活函数的综合性能。

3.1 ERelu 模型确定

为了确定最佳的 EReLU 函数设计方案,按照上述方法使用 VGG16[12] 和 ResNet18[14] 网络分别采用 f_1、f_2、f_3、f_4 及 ReLU 函数在 CIFAR10 和 CIFAR100 数据集上测试,Batchesize 设置为 128,测试结果如表 3、表 4 所示。

表 3 不同激活函数在 ResNet18 网络上的表现

Tab.3 Performance of different activation functions on the ResNet18 network

Results Methods	Datasets			
	CIFAR10		CIFAR100	
	ACC	T/h	ACC	T/h
f_1	93.11%	1.332	74.82%	1.332
f_2	93.03%	1.335	74.27%	1.335
f_3	93.66%	1.290	75.23%	1.290
f_4	93.78%	1.262	75.87%	1.262
ReLU	92.90%	1.325	73.68%	1.325

表 4 不同激活函数在 VGG16 网络上的表现

Tab.4 Performance of different activation functions on the VGG16 network

Results Methods	Datasets			
	CIFAR10		CIFAR100	
	ACC	*T*/h	ACC	*T*/h
f_1	91.31%	1.225	58.91%	1.225
f_2	91.24%	1.248	58.35%	1.248
f_3	91.86%	1.243	59.23%	1.243
f_4	91.98%	1.175	59.95%	1.175
ReLU	91.15%	1.238	56.24%	1.238

实验结果表明，f_4 在不同网络和数据集上综合表现均最好，f_2、f_3 略差，f_1 最差，但都优于 ReLU 函数，这与函数及其导函数本身的时间复杂度及空间复杂度有一定的关系，其中 f_1 为线性函数，f_2 为幂函数，f_3 为一阶分数函数，f_4 为自然对数函数，时间复杂度上，$f_1 < f_2, f_3, f_4$，相较于包含指数运算 (如 ELU) 的激活函数更低，这决定了上述激活函数在计算方面更高效；在空间复杂度方面，上述函数相较于添加了额外参数 (如 LeakyReLU) 的函数在空间复杂度上更低，这决定了上述激活函数在占用硬件空间资源方面更少。由此可见，在负半轴引入上述任意非零函数均对 ReLU 函数的性能有所改善，其中引入了自然对数函数的 f_4 综合表现最佳，因此采用 f_4 作为高效激活函数 EReLU 的最终设计方案，即

$$f_{ERelu}=\begin{cases} x & ,x\geqslant 0 \\ -\ln^{1-x} & ,x<0 \end{cases}$$
$$f'_{ERelu}=\begin{cases} 1 & ,x\geqslant 0 \\ \dfrac{1}{1-x} & ,x<0 \end{cases} \tag{11}$$

3.2 与其他激活函数比较

确定了 EReLU 函数的数学模型后，为了更有效的说明其优越性，使用 VGG16[12] 和 ResNet18[14] 并分别采用 EReLU、ReLU[3]、LerkyReLU[4]、PReLU[5]、ELU[6]、ReLU-softsign[8] 共 6 种激活函数在 CIFAR10[12]、CIFAR100[12] 数据集上按照上述方式进一步实验。

(1) CIFAR10 上实验结果及分析

使用 ResNet18 和 VGG16 网络在 CIFAR10 数据集上的测试结果分别见图 3、图 4，其中图图 3(a) 纵坐标表示分类准确率，横坐标表示不同的激活函数，图 3(b) 纵坐标表示不同的激活函数，横坐标表示训练时长，其中黑色粗体表示最好的结果，蓝色表示第二好结果，后续图中的坐标轴含义均与图 3 相同。

由图 3、图 4 可见，在标签少、类间距大的 CIFAR10 上使用两种网络对上述 6 种函数测试时，EReLU 表现最佳，ReLU-softsign 第二，PReLU 表现最差，EReLU 在提升精度和效率方面均表现最佳，相较于其他函数在 CIFAR10 上精度提升幅度为 0.85%~1.08%，时长缩短范围为 0.075~0.088 h。

(2) CIFAR100 上实验结果及分析

由图 5、图 6 可见，在标签多、类间距小的 CIFAR100 数据集上使用两种网络对上述 6 种函数测

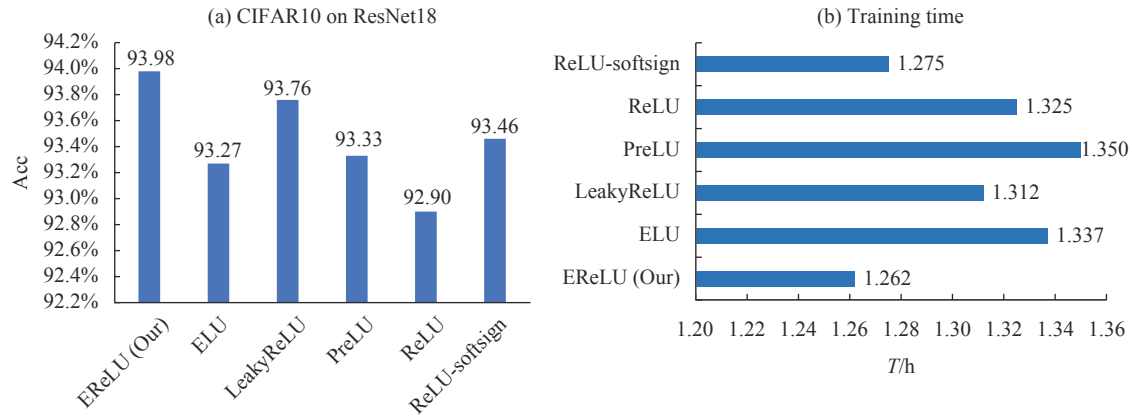

图 3 不同激活函数使用 ResNet18 网络在 CIFAR10 数据集上的测试准确率 (a) 和训练时间 (b)

Fig.3 Test accuracy (a) and training time (b) of different activation functions using ResNet18 network on CIFAR10

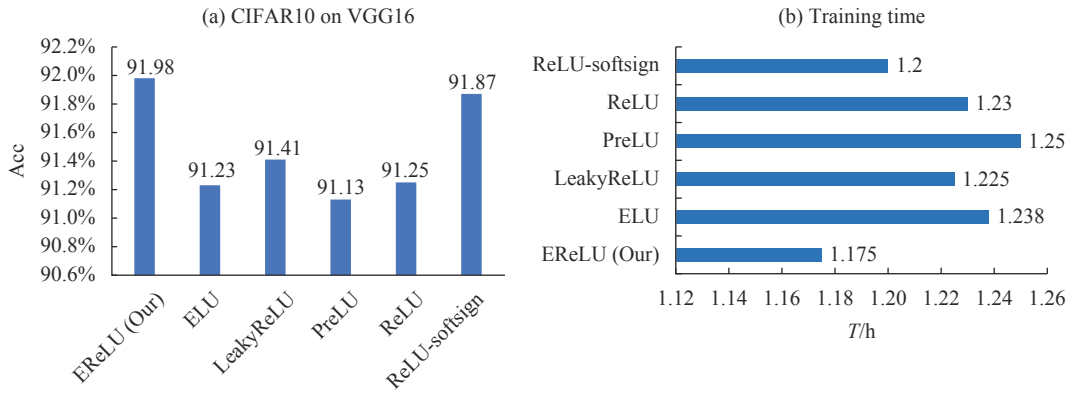

图 4 不同激活函数使用 VGGl6 网络在 CIFARI0 数据集上的测试准确率 (a) 和训练时间 (b)

Fig.4 Test accuracy (a) and training time (b) of different activation functions using VGG16 network on CIFAR10

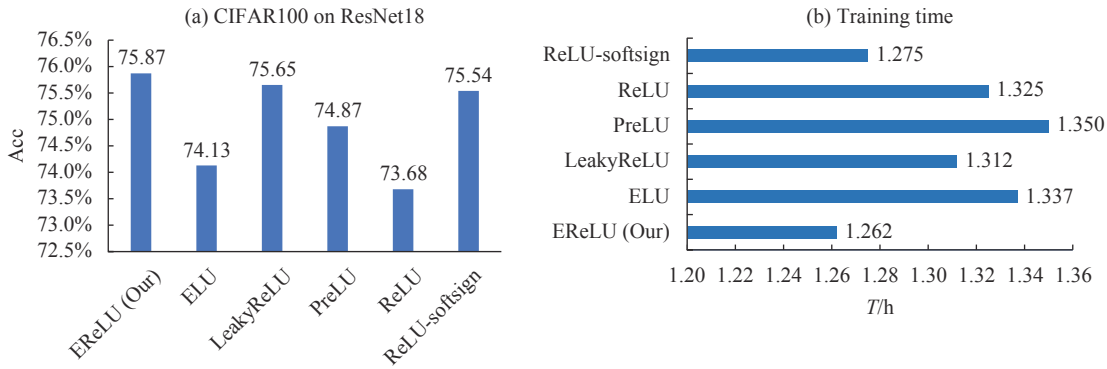

图 5 不同激活函数使用 RcsNct18 网络在 CIFAR100 数据集上的测试准确率 (a) 和训练时间 (b)

Fig.5 Test accuracy (a) and training time (b) of different activation functions using ResNet18 network on CIFAR100

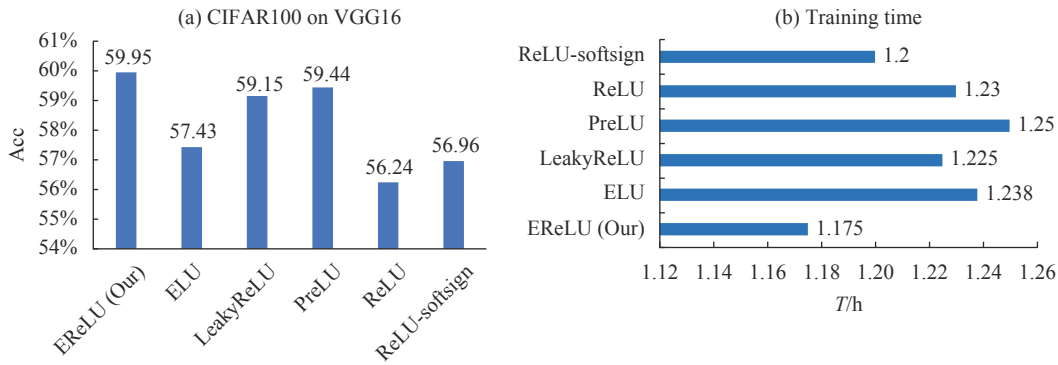

图 6 不同激活函数使用 VGG16 网络在 CIFAR100 数据集上的测试准确率 (a) 和训练时间 (b)

Fig.6 Test accuracy (a) and training time (b) of different activation functions using VGG16 network on CIFAR100

试时, ERelu 函数在精度和效率方面综合表现最佳, PReLU 次之, ReLU 最差, EReLU 相较于其他函数在 CIFAR100 上精度提升幅度为 2.19%~3.71%, 时长缩短范围为 0.075~0.088 h。

(3) Fer2013 上实验结果及分析

由图 7、图 8 可见, 在标签少、类间距小的细粒度数据集 Fer2013 上使用两种网络对上述 6 种函数测试时, EReLU 在提升精度和效率方面同样表现最佳, ReLU-softsign 次之, PReLU 则最差, EReLU 相较于其他函数在 Fer2013 上精度提升幅度为 0.58%~0.78%, 时长缩短范围为 0.015~0.038 h。

综合图 3~图 8 可知, EReLU 和 ReLU-softsign 函

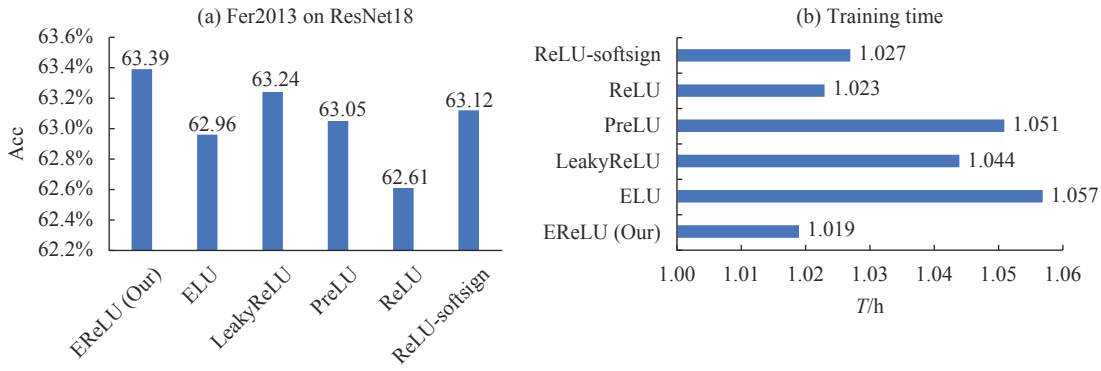

图 7 不同激活函数使用 ResNet18 网络在 Fer2013 数据集上的测试准确率 (a) 和训练时间 (b)

Fig.7 Test accuracy (a) and training time (b) of different activation functions using ResNet18 network on Fer2013

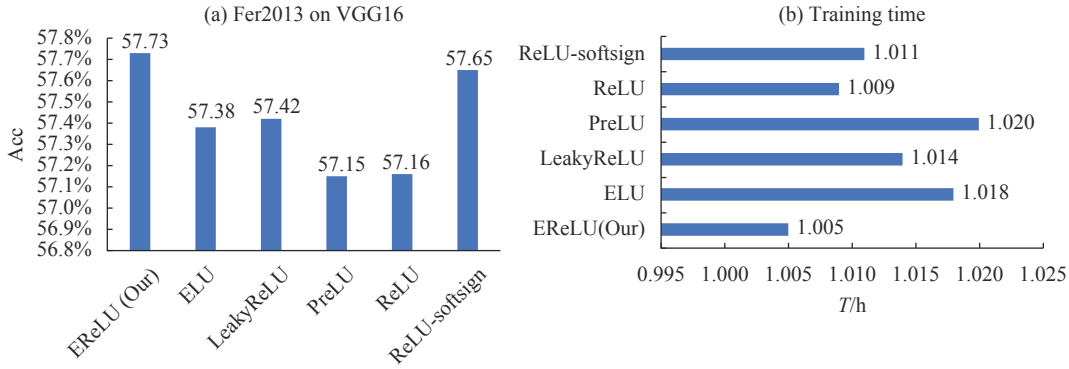

图 8 不同激活函数使用 VGG16 网络在 Fer2013 数据集上的测试准确率 (a) 和训练时间 (b)

Fig.8 Test accuracy (a) and training time (b) of different activation functions using VGG16 network on Fer2013

数在标签少、类间差异小的细粒度数据集上潜力较大, LeakyReLU 和 PReLU 函数在标签多、类间差异大的数据集上表现较好, 而 ELU 仅在标签少、类间差异大的数据集上精度略优于 ReLU, 但效率上有所下降, 而反观 EReLU 函数, 相较于其他函数在提升精度和效率方面表现均最好。由此可见, EReLU 在图像分类任务中相比于其他激活函数更具优越性和竞争力。

4 结 论

文中通过分析激活函数在网络训练过程中的作用提出包括线性型、幂数型、分式型及自然对数型在内的 4 种改进 ReLU 激活函数的方案, 实验表明当在负半轴引入自然对数函数时可以更好的提升网络性能, 由此确定 EReLU 函数的数学模型。随后在 ResNet18 和 VGG16 网络中使用 EReLU、ReLU、LerkyReLU、PReLU、ELU 及 ReLU-softsign 共 6 种激活函数分别在 CIFAR10、CIFAR100 和 Fer2013 数据集上进行测试, 结果表明 EReLU 相较于其他函数精度提升 0.12%~6.61%, 效率提升 1.02%~6.52%, 由此可见, EReLU 函数在提升轻量型网络的精度和效率方面相较于其它激活函数更具竞争力。

在未来的工作中, 将 EReLU 函数应用到大型神经网络或其他领域的数据集进行测试, 以进一步验证 EReLU 函数的优越性和适用性。

参考文献:

[1] Hassell M P, Lawton J H, Beddington J R. Sigmoid functional responses by invertebrate predators and parasitoids [J]. ***The Journal of Animal Ecology***, 1977: 249-262.

[2] Kalman B L, Kwasny S C. Why tanh: Choosing a sigmoidal function[C]//Proceedings 1992IJCNN International Joint Conference on Neural Networks. IEEE, 1992, 4: 578-581.

[3] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks [J]. ***Communications of the ACM***, 2017, 60(6): 84-90.

[4] Dubey A K, Jain V. Comparative study of convolution neural network 's relu and leaky-relu activation functions[M]//

Applications of Computing, Automation and Wireless Systems in Electrical Engineering, 2019: 873-880.

[5] 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. 2015: 1026-1034.

[6] Clevert D A, Unterthiner T, Hochreiter S. Fast and accurate deep network learning by exponential linear units (elus) [J]. ***arXiv preprint arXiv***, 2015: 1511.07289.

[7] Shi Qi. Research and verification of image classification optimization algorithm based on convolutional neural network[D]. Beijing: Beijing Jiaotong University, 2017. (in Chinese)

[8] Wang Hongxia, Zhou Jiaqi, Gu Chenghao, et al. Design of activation functions in convolutional neural networks for image classification [J]. ***Journal of Zhejiang University* (*Engineering Science*)**, 2019, 53(7): 1363-1373. (in Chinese)

[9] Zhang X, Zou Y, Shi W. Dilated convolution neural network with LeakyReLU for environmental sound classification[C]// 2017 22nd International Conference on Digital Signal Processing (DSP). IEEE, 2017: 1-5.

[10] Xu L, Choy C, Li Y W. Deep sparse rectifier neural networks for speech denoising[C]//2016 IEEE International Workshop on Acoustic Signal Enhancement (IWAENC). IEEE, 2016: 1-5.

[11] Liu Jun. Analysis of algorithm design and algorithm time complexity [J]. ***Computer Knowledge and Technology***, 2008, 2(14): 878-879. (in Chinese)

[12] Liu X, Zhang Y, Bao F. Kernel-blending connection approximated by a neural network for image classification [J]. ***Computational Visual Media***, 2020, 6(4): 467-476.

[13] Carrier P L, Courville A, Mirza M, et al. Challenges in representation learning: A report on three machine learning contests[C]//International Conference on Neural Information Processing, 2013: 117-124.

[14] Wang H, Zhou J, Gu C, et al. Design of activation function in CNN for image classification [J]. ***Journal of ZheJiang University* (*Engineering Science*)**, 2019, 53(7): 1363-1373.