ErfAct and Pserf: Non-monotonic Smooth Trainable Activation Functions

Koushik Biswas[1], **Sandeep Kumar**[1, 2], **Shilpak Banerjee**[3], **Ashish Kumar Pandey**[3]

[1]Department of Computer Science, IIIT Delhi, New Delhi, India 110020
[2]Department of Mathematics, Shaheed Bhagat Singh College, University of Delhi, New Delhi, India 110017
[3]Department of Mathematics, IIIT Delhi, New Delhi, India 110020
koushikb@iiitd.ac.in, sandeep_kumar@sbs.du.ac.in, shilpak@iiitd.ac.in, ashish.pandey@iiitd.ac.in

Abstract

An activation function is a crucial component of a neural network that introduces non-linearity in the network. The state-of-the-art performance of a neural network depends also on the perfect choice of an activation function. We propose two novel non-monotonic smooth trainable activation functions, called ErfAct and Pserf. Experiments suggest that the proposed functions improve the network performance significantly compared to the widely used activations like ReLU, Swish, and Mish. Replacing ReLU by ErfAct and Pserf, we have 5.68% and 5.42% improvement for top-1 accuracy on Shufflenet V2 (2.0x) network in CIFAR100 dataset, 2.11% and 1.96% improvement for top-1 accuracy on Shufflenet V2 (2.0x) network in CIFAR10 dataset, 1.0%, and 1.0% improvement on mean average precision (mAP) on SSD300 model in Pascal VOC dataset.

Introduction

The choice of activation function in a deep learning architecture can have a significant impact on the training and performance of the trained network. The machine learning community has so far relied on hand-designed activations like ReLU (Nair and Hinton 2010), Leaky ReLU (Maas, Hannun, and Ng 2013) or their variants. ReLU, in particular, remains widely popular due to faster training times and decent performance. However, evidence suggests that considerable gains can be made when more sophisticated activation functions are used to design networks. For example, activation functions such as ELU (Clevert, Unterthiner, and Hochreiter 2016), Parametric ReLU (PReLU) (He et al. 2015b), ReLU6 (Krizhevsky 2010), PAU (Molina, Schramowski, and Kersting 2020), OPAU (Biswas, Banerjee, and Pandey 2021), ACON (Ma et al. 2021), Mish (Misra 2020), GELU (Hendrycks and Gimpel 2020), Swish (Ramachandran, Zoph, and Le 2017), Serf (Nag and Bhattacharyya 2021) etc. have appeared as powerful contenders to the traditional ones. Though ReLU remains a go-to choice in both research and practice, it has certain well-documented shortcomings such as non-zero mean (Clevert, Unterthiner, and Hochreiter 2016), non-differentiability and negative missing, which leads to the infamous vanishing gradients problem (also known as the dying ReLU problem). Worth noting that prior to the introduction of ReLU, Tanh and Sigmoid were popularly used, but performance gains and training time gains achieved by ReLU led to their decline.

Related works and motivation

The newer activation functions are obtained by combining well-known functions with simple forms in various ways, often using hyper-parameters or trainable parameters. In the case of trainable parameters, we optimize them during the training process itself, yielding networks that are better fitted. In the case of trainable parameters, note that the actual activation function curve may change in different layers during backpropagation. For example, SiLU (Elfwing, Uchibe, and Doya 2017) shows good performance over known activation functions while Swish (Ramachandran, Zoph, and Le 2017) is a trainable version of SiLU, which is a non-linear, non-monotonic activation function. Among the activation mentioned in the previous section, Swish, PReLU, PAU, ACON, and OPAU are trainable activation functions. Swish is a non-monotonic activation function and shows promise across a variety of deep learning tasks. Mish is one of the popular functions proposed recently and gained popularity due to its effectiveness in object detection task on COCO dataset (Lin et al. 2015) in Yolo (Bochkovskiy, Wang, and Liao 2020) models. GELU is very similar to Swish and gained attention due to its effectiveness in computer vision and natural language processing tasks. It is also used in popular architectures like GPT-2 (Radford et al. 2019) and GPT-3 (Brown et al. 2020). Apart from using a combination of known functions, a somewhat fundamentally different technique to construct activation functions is to use perturbation or approximations to well-known activation functions to remove some of the shortcomings yet retain the positive aspects. Recent successful examples where this strategy was employed includes PAU and OPAU, which are activations based on approximation of Leaky ReLU by rational polynomials were constructed.

Motivated from these works, we have proposed two activation functions with trainable parameters, we call them ErfAct and Pserf and shown that they are more effective than conventional activation functions like ReLU, Leaky ReLU, PReLU, ReLU6, Swish, Mish or GELU in a wide range of standard deep learning problems. We summarize the paper as follows:

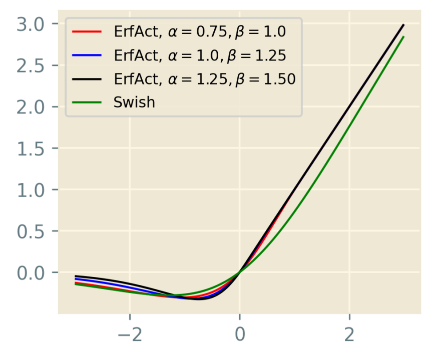

Figure 1: Swish and ErfAct activation for different values of α and β

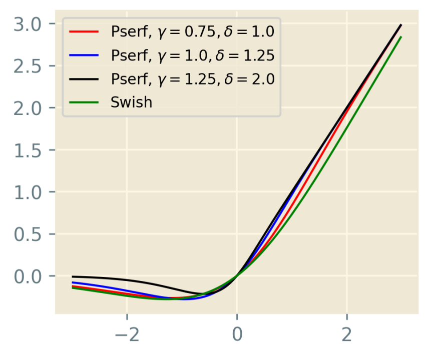

Figure 2: Swish and Pserf activation for different values of γ and δ

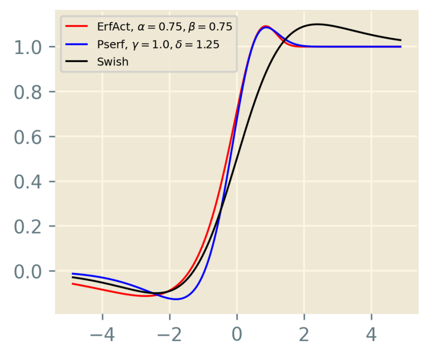

Figure 3: First order derivative of ErfAct, Pserf, and Swish

- We have proposed two new novel trainable activation functions, which are the smooth approximation of ReLU.
- In a wide range of deep learning tasks, the proposed functions outperform widely used activation functions.

ErfAct and Pserf

As motivated earlier, in this paper, we present, ErfAct and Parametric-Serf (Pserf), two novel trainable activation functions which outperforms the widely used activations and has the potential to replace them. ErfAct and Pserf is defined as

$$\text{ErfAct} : \mathcal{F}_1(x; \alpha, \beta) := x \ \text{erf}(\alpha e^{\beta x}), \tag{1}$$

$$\text{Pserf} : \ \mathcal{F}_2(x; \gamma, \delta) := x \ \text{erf}(\gamma ln(1 + e^{\delta x})) \tag{2}$$

where α, β, γ, and δ are trainable parameters (they can be used as hyper-parameters as well) and 'erf' is the error function also known as the Gauss error function and defined as

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} \, dt \tag{3}$$

The corresponding derivatives of the proposed activations are

$$\frac{d}{dx}\mathcal{F}_1(x; \alpha, \beta) = \text{erf}(\alpha e^{\beta x}) + \frac{2x\alpha\beta}{\sqrt{\pi}} e^{\beta x} e^{-(\alpha e^{\beta x})^2} \tag{4}$$

$$\frac{d}{dx}\mathcal{F}_2(x; \gamma, \delta) = \text{erf}(\gamma ln(1 + e^{\delta x})) + \frac{2x\gamma\delta}{\sqrt{\pi}} \frac{e^{\delta x}}{1 + e^{\delta x}} e^{-(\gamma ln(1+e^{\delta x}))^2} \tag{5}$$

where

$$\frac{d}{dx}\text{erf}(x) = \frac{2}{\sqrt{\pi}} e^{-x^2} \tag{6}$$

ErfAct and Pserf are non-monotonic, zero-centered, continuously differentiable, unbounded above but bounded below, and trainable functions. Figures 1 and 2 show the plots for $\mathcal{F}_1(x; \alpha, \beta)$ and $\mathcal{F}_2(x; \gamma, \delta)$ activation functions for different values of α, β, and γ, δ respectively. A comparison between the first derivative of $\mathcal{F}_1(x; \alpha, \beta)$, $\mathcal{F}_2(x; \gamma, \delta)$, and Swish are given in Figures 3, different values of α, β, and γ, δ respectively. From the figures 1 and 2 it is evident that the parameters α, β, and γ, δ controls the slope of the curves for the proposed activations in both positive and negative axis. The proposed functions converges to some known functions for specific values of the parameters. For example, $\mathcal{F}_1(x; 0, \beta)$, $\mathcal{F}_2(x; 0, \delta)$ are zero function while $\mathcal{F}_1(x; \alpha, 0)$, $\mathcal{F}_2(x; \gamma, 0)$ are linear functions. In particular, $\mathcal{F}_2(x; 1, 1)$ share the equivalent form as Serf(Nag and Bhattacharyya 2021) which is a non-parametric form of Pserf. Also, The proposed functions can be seen as a smooth approximation of ReLU.

$$\lim_{\beta \to \infty} \mathcal{F}_1(x; \alpha, \beta) = \text{ReLU}(x), \quad \forall x \in \mathbb{R} \text{ for any fixed } \alpha > 0.$$

$$\lim_{\delta \to \infty} \mathcal{F}_2(x; \gamma, \delta) = \text{ReLU}(x), \quad \forall x \in \mathbb{R} \text{ for any fixed } \gamma > 0.$$

For any K, a compact (closed and bounded) subset of $\mathbb{R}^n$, the set of neural networks with ErfAct (or Pserf) activation functions is dense in $C(K)$, the space of all continuous functions over K (see (Molina, Schramowski, and Kersting 2020)). This follows from the next proposition, as the proposed activation functions are not polynomials.

Proposition (Theorem 1.1 in Kidger and Lyons, 2019 (Kidger and Lyons 2020)) :- Let $\rho : \mathbb{R} \to \mathbb{R}$ be any continuous function. Let N_n^ρ represent the class of neural networks with activation function ρ, with n neurons in the input layer, one neuron in the output layer, and one hidden layer with an arbitrary number of neurons. Let $K \subseteq \mathbb{R}^n$ be compact. Then N_n^ρ is dense in $C(K)$ if and only if ρ is non-polynomial.

Experiments

We have compared our proposed activations against ten popular standard activation functions on different datasets and models on standard deep learning problems like image classification, object detection, semantic segmentation, and machine translation. The experimental results show that ErfAct and Pserf outperform in most networks compared to the standard activations. For all our experiments, we have first initialized the parameters $\alpha = 0.75,\ \beta = 0.75$ for ErfAct and $\gamma = 1.0,\ \delta = 1.25$ for Pserf and then updated via the backpropagation (LeCun et al. 1989) algorithm (see (He et al. 2015b)) according to (7) and for a single layer, the gradient of a parameter ρ is:

$$\frac{\partial L}{\partial \rho} = \sum_x \frac{\partial L}{\partial f(x)} \frac{\partial f(x)}{\partial \rho} \tag{7}$$

where L is the objective function, $\rho \in \{\alpha, \beta, \gamma, \delta\}$ and $f(x) \in \{\mathcal{F}_1(x; \alpha, \beta), \mathcal{F}_2(x; \gamma, \delta)$. For all of our experiments, to make a fair comparison between all the activations, we have first trained a network with hyper-parameter settings with the ReLU activation function and then only replaced ReLU with proposed activation functions and other baseline activations.

Image Classification

We present a detailed experimental comparison on MNIST (LeCun, Cortes, and Burges 2010), Fashion MNIST (Xiao, Rasul, and Vollgraf 2017), CIFAR10 (Krizhevsky 2009), CIFAR100 (Krizhevsky 2009), and Tiny ImageNet (Le and Yang 2015) dataset for image classification problem. We have trained the datasets with different standard models and report the top-1 accuracy.

MNIST, Fashion MNIST, and The Street View House Numbers (SVHN) Database: We first evaluate our proposed activation functions on the MNIST (LeCun, Cortes, and Burges 2010), Fashion MNIST (Xiao, Rasul, and Vollgraf 2017), and SVHN (Netzer et al. 2011) datasets with AlexNet (Krizhevsky, Sutskever, and Hinton 2012) and VGG-16 (Simonyan and Zisserman 2015) (with batch-normalization) models and results for 10-fold mean accuracy are reported in Table 1 and Table 2 respectively. More detailed experiments on these datasets on LeNet (Lecun et al. 1998) and a custom-designed CNN architecture is reported in the supplementary material. We don't use any data augmentation for MNIST or Fashion MNIST, while we use standard data augmentation like rotation, zoom, height shift, shearing for the SVHN dataset. From Table 1 and Table 2, it is clear that the proposed functions outperformed all the baseline activation functions in all the three datasets and the performance are stable clear from mean$\pm$standard deviation.

CIFAR: Next we have considered more challenging datasets like CIFAR100 and CIFAR10 to compare the performance of baseline activations and ErfAct and Pserf. We have reported the Top-1 accuracy for both the datasets for mean of 12 different runs on Table 4 and Table 5 with VGG-16 (with batch-normalization) (Simonyan and Zisserman 2015), PreActResNet-34 (PA-ResNet-34) (He et al. 2016), Densenet-121 (DN-121) (Huang et al. 2016), MobileNet V2 (MN V2) (Sandler et al. 2019), Resnet-50 (He et al. 2015a), Inception V3 (IN-V3) (Szegedy et al. 2015), WideResNet 28-10 (WRN 28-10) (Zagoruyko and Komodakis 2016), and Shufflenet V2 (SF-V2 2.0x) (Ma et al. 2018) models. A more detailed experiments on these datasets with EfficientNet B0 (EN-B0) (Tan and Le 2020), LeNet (LN) (Lecun et al. 1998), AlexNet (AN) (Krizhevsky, Sutskever, and Hinton 2012), PreActResnet-18 (PARN-18) (He et al. 2016), Deep Layer Aggregation (DLA) (Yu et al. 2019), Googlenet (GN) (Szegedy et al. 2014), Resnext (Rxt) (Xie et al. 2017), Xception (Xpt)(Chollet 2017), Squeezenet (SQ-Net) (Iandola et al. 2016), ResNet18 (RN-18) (He et al. 2015a), and Network in Network (NIN) (Lin, Chen, and Yan 2014) is reported in the supplementary section. From all the tables it is evident that the training is stable (mean$\pm$std) and the proposed activations archive 1%-6% higher top-1 accuracy in most of models compared to the baselines. The networks are trained upto 200 epochs with SGD optimizer (Robbins and Monro 1951; Kiefer and Wolfowitz 1952), 0.9 momentum, and $5e^{-4}$ weight decay. We have started with 0.01 initial learning rate and decay the learning rate with cosine annealing (Loshchilov and Hutter 2017) learning rate scheduler. We consider batch size of 128. We consider standard data augmentation methods like horizontal flip, rotation for both the datasets. The Figures 4 and 5 shows the learning curves on CIFAR100 dataset with Shufflenet V2 (2.0x) model for the baseline and the proposed activation functions and it is noticeable that training & test accuracy curve is higher and loss curve is lower respectively for ErfAct and Pserf compared to the baseline activations.

Tiny Imagenet: We consider a more challenging and important classification dataset Tiny Imagenet (Le and Yang 2015) which is a similar type of dataset like ILSVRC and consisting of 200 classes with RGB images of size 64×64 with total 1,00,000 training images, 10,000 validation images, and 10,000 test images. To compare the performance, we have considered WideResNet 28-10 (WRN 28-10) (Zagoruyko and Komodakis 2016) model and Top-1 accuracy is reported in table 3 for mean of 5 different runs.

Activation Function	Wide ResNet 28-10 Model
ReLU	61.61 $\pm$ 0.47
Swish	62.44 $\pm$ 0.49
Leaky ReLU	61.47 $\pm$ 0.44
ELU	61.99 $\pm$ 0.57
Softplus	60.42 $\pm$ 0.61
Mish	63.02 $\pm$ 0.57
GELU	62.64 $\pm$ 0.62
PAU	62.04 $\pm$ 0.54
PReLU	61.25 $\pm$ 0.51
ReLU6	61.72 $\pm$ 0.56
ErfAct	**64.20** $\pm$ 0.51
Pserf	**64.01** $\pm$ 0.49

Table 3: Comparison between different baseline activations and ErfAct and Pserf on Tiny ImageNet dataset. Mean of 5 different runs for top-1 accuracy(in %) have been reported. mean$\pm$std is reported in the table.

The model is trained with a batch size of 32, He Normal initializer (He et al. 2015b), 0.2 dropout rate (Srivastava et al. 2014), adam optimizer (Kingma and Ba 2015), with initial

Activation Function	MNIST	Fashion MNIST	SVHN
ReLU	99.09 ± 0.10	93.22 ± 0.21	95.50 ± 0.22
Swish	99.30 ± 0.12	93.29 ± 0.22	95.59 ± 0.20
Leaky ReLU	99.15 ± 0.13	93.30 ± 0.22	95.50 ± 0.28
ELU	99.29 ± 0.13	93.20 ± 0.25	95.60 ± 0.20
Softplus	99.10 ± 0.14	93.18 ± 0.32	95.20 ± 0.37
Mish	99.27 ± 0.14	93.45 ± 0.32	95.60 ± 0.31
GELU	99.22 ± 0.12	93.40 ± 0.25	95.55 ± 0.27
PAU	99.31 ±0.10	93.47 ± 0.23	95.67 ± 0.26
PReLU	99.15 ± 0.16	93.37 ± 0.31	95.42 ± 0.39
ReLU6	99.11 ± 0.10	93.26 ± 0.26	95.47 ± 0.24
ErfAct	**99.51** ± 0.10	**93.79** ± 0.19	**95.87** ± 0.20
Pserf	**99.49** ± 0.10	**93.82** ± 0.19	**95.74** ± 0.22

Table 1: Comparison between different baseline activations and ErfAct and Pserf activations on MNIST, Fashion MNIST, and SVHN datasets in AlexNet. 10-fold mean accuracy (in %) have been reported. mean±std is reported in the table.

Activation Function	MNIST	Fashion MNIST	SVHN
ReLU	99.05 ± 0.11	93.13 ± 0.23	95.09 ± 0.26
Swish	99.09 ± 0.09	93.34 ± 0.21	95.29 ± 0.20
Leaky ReLU	99.02 ± 0.14	93.17 ± 0.28	95.24 ± 0.23
ELU	99.01 ± 0.15	93.12 ± 0.30	95.15 ± 0.28
Softplus	98.97 ± 0.14	92.98 ± 0.34	94.94 ± 0.30
Mish	99.18 ± 0.07	93.47 ± 0.27	95.12 ± 0.25
GELU	99.10 ± 0.09	93.41 ± 0.29	95.11 ± 0.24
PAU	99.07 ± 0.09	93.52 ± 0.24	95.23 ± 0.20
PReLU	99.01 ± 0.09	93.12 ± 0.27	95.14 ± 0.24
ReLU6	99.20 ± 0.08	93.25 ± 0.27	95.22 ± 0.20
ErfAct	**99.37** ± 0.06	**93.81** ± 0.20	**95.67** ± 0.18
Pserf	**99.38** ± 0.09	**93.87** ± 0.22	**95.66** ± 0.20

Table 2: Comparison between different baseline activations, ErfAct, and Pserf activations on MNIST, Fashion MNIST, and SVHN datasets on VGG-16 network. We report results for 10-fold mean accuracy (in %). mean±std is reported in the table.

learning rate(lr rate) 0.01, and lr rate is reduced by a factor of 10 after every 60 epochs up-to 300 epochs. We have considered the standard data augmentation methods like rotation, width shift, height shift, shearing, zoom, horizontal flip, fill mode. From the table, it is clear that the performance for the proposed functions are better than the baseline functions and stable (mean±std) and got a boost in top-1 accuracy by 2.59% and 2.40% for ErfAct and Pserf compared to ReLU.

Semantic Segmentation

Semantic segmentation is an important problem in deep learning. In this section, we present experimental results on the Cityscapes dataset (Cordts et al. 2016). We report the pixel accuracy and mean Intersection-Over-Union (mIOU) on the U-net model(Ronneberger, Fischer, and Brox 2015). The model is trained up to 250 epochs, with adam optimizer (Kingma and Ba 2015), learning rate $5e^{-3}$, batch size 32 and Xavier Uniform initializer (Glorot and Bengio 2010). A mean of 5 different runs on the test dataset is reported in table 6. We got around 1.97% and 1.89% boost on mIOU for ErfAct and Pserf compared to ReLU.

Activation Function	Pixel Accuracy	mIOU
ReLU	79.60 ± 0.45	69.32 ± 0.30
Swish	79.71 ± 0.49	69.68 ± 0.31
Leaky ReLU	79.41 ± 0.42	69.48 ± 0.39
ELU	79.27 ± 0.54	68.12 ± 0.41
Softplus	78.69 ± 0.49	68.12 ± 0.55
Mish	80.12 ± 0.45	69.87 ± 0.29
GELU	79.60 ±0.39	69.51 ± 0.39
PAU	79.95 ± 0.41	69.42 ± 0.46
PReLU	78.99 ± 0.42	68.82 ± 0.41
ReLU6	79.59 ± 0.41	69.66 ± 0.41
ErfAct	**81.41** ± 0.45	**71.29** ± 0.31
Pserf	**81.12** ± 0.42	**71.21** ± 0.34

Table 6: Comparison between different baseline activations and ErfAct and Pserf on semantic segmentation problem on U-NET model in CityScapes dataset. mean±std is reported in the table.

Object Detection

Object detection is a standard problem in computer vision. In this section, we have reported our experimental results on

Activation Function	VGG-16	WRN 28-10	ResNet-50	PA-ResNet-34	DN-121	IN-V3	MN-V2	SF-V2 2.0x
ReLU	71.67 ±0.28	76.32 ±0.25	74.17 ±0.24	73.12 ±0.23	75.67 ±0.28	74.23 ±0.26	74.02 ±0.24	67.49 ±0.26
Leaky ReLU	71.77 ±0.30	76.69 ±0.27	74.11 ±0.27	73.41 ±0.26	75.90 ±0.27	74.40 ±0.28	74.17 ±0.24	67.71 ±0.27
ELU	71.71 ±0.28	76.39 ±0.28	74.51 ±0.24	73.61 ±0.25	75.87 ±0.26	74.71 ±0.26	74.29 ±0.22	67.91 ±0.30
Swish	72.07 ±0.26	77.18 ±0.23	75.10 ±0.24	73.97 ±0.23	76.59 ±0.28	75.31 ±0.27	75.02 ±0.24	70.49 ±0.23
Softplus	71.10 ±0.32	75.36 ±0.37	74.19 ±0.38	73.17 ±0.36	75.08 ±0.36	74.20 ±0.34	74.33 ±0.38	68.93 ±0.36
Mish	72.31 ±0.24	77.40 ±0.25	76.30 ±0.22	75.14 ±0.21	77.11 ±0.25	76.22 ±0.25	75.31 ±0.21	71.79 ±0.22
GELU	71.98 ±0.25	77.35 ±0.25	75.61 ±0.22	74.28 ±0.23	76.79 ±0.27	75.52 ±0.25	75.21 ±0.23	70.35 ±0.27
PAU	71.72 ±0.25	77.20 ±0.26	75.89 ±0.24	74.41 ±0.23	76.59 ±0.28	75.79 ±0.28	75.07 ±0.19	70.68 ±0.26
PReLU	71.77 ±0.30	76.79 ±0.27	74.45 ±0.29	73.32 ±0.27	76.19 ±0.30	74.51 ±0.29	74.31 ±0.32	68.35 ±0.30
ReLU6	72.07 ±0.27	76.62 ±0.28	74.37 ±0.24	73.50 ±0.24	76.07 ±0.26	74.69 ±0.25	74.64 ±0.24	67.93 ±0.26
ErfAct	**72.93** ±0.22	**78.49** ±0.23	**77.09** ±0.20	**76.21** ±0.20	**78.18** ±0.23	**77.12** ±0.24	**76.23** ±0.19	**73.17** ±0.22
Pserf	**72.69** ±0.24	**78.31** ±0.24	**76.97** ±0.20	**75.91** ±0.22	**78.38** ±0.22	**77.01** ±0.25	**76.07** ±0.21	**72.91** ±0.21

Table 4: Comparison between different baseline activations and ErfAct and Pserf on CIFAR100 dataset. Top-1 accuracy(in %) for mean of 12 different runs have been reported. mean±std is reported in the table.

Activation Function	VGG-16	WRN 28-10	ResNet-50	PA-ResNet-34	DN-121	IN-V3	MN-V2	SF-V2 2.0x
ReLU	93.44 ±0.22	95.17 ±0.21	94.35 ±0.18	94.17 ±0.19	94.77 ±0.20	94.15 ±0.20	94.20 ±0.16	91.63 ±0.21
Leaky ReLU	93.65 ±0.21	95.02 ±0.22	94.45 ±0.20	94.33 ±0.18	94.89 ±0.22	94.20 ±0.22	94.32 ±0.19	91.82 ±0.23
ELU	93.70 ±0.19	95.28 ±0.20	94.27 ±0.24	94.30 ±0.25	94.64 ±0.18	94.38 ±0.17	94.27 ±0.18	91.99 ±0.20
Swish	93.77 ±0.18	95.41 ±0.17	94.61 ±0.24	94.47 ±0.25	94.81 ±0.19	94.51 ±0.17	94.40 ±0.20	92.17 ±0.25
Softplus	93.10 ±0.33	94.77 ±0.30	93.91 ±0.30	94.07 ±0.35	94.41 ±0.34	94.21 ±0.32	93.79 ±0.29	91.32 ±0.33
Mish	93.91 ±0.17	95.35 ±0.18	94.78 ±0.22	94.55 ±0.23	95.03 ±0.15	94.64 ±0.18	94.71 ±0.18	92.41 ±0.20
GELU	93.71 ±0.17	95.28 ±0.19	94.64 ±0.23	94.31 ±0.25	94.99 ±0.19	94.57 ±0.21	94.40 ±0.18	92.27 ±0.20
PAU	93.57 ±0.22	95.27 ±0.20	94.67 ±0.23	94.41 ±0.24	94.74 ±0.20	94.57 ±0.19	94.51 ±0.14	92.30 ±0.21
PReLU	93.41 ±0.23	95.02 ±0.24	94.27 ±0.26	94.30 ±0.26	94.51 ±0.24	94.49 ±0.22	94.32 ±0.23	91.80 ±0.25
ReLU6	93.72 ±0.17	95.32 ±0.19	94.30 ±0.24	94.21 ±0.24	94.61 ±0.20	94.42 ±0.20	94.18 ±0.19	91.71 ±0.21
ErfAct	**94.47** ±0.15	**95.88** ±0.12	**95.01** ±0.17	**95.21** ±0.18	**95.71** ±0.15	**95.29** ±0.14	**95.34** ±0.12	**93.74** ±0.18
Pserf	**94.24** ±0.16	**95.71** ±0.13	**95.14** ±0.19	**95.08** ±0.29	**95.62** ±0.17	**95.10** ±0.13	**95.19** ±0.14	**93.59** ±0.18

Table 5: Comparison between different baseline activations and ErfAct and Pserf on CIFAR10 dataset. Top-1 accuracy(in %) for mean of 12 different runs have been reported. mean±std is reported in the table.

challenging Pascal VOC dataset (Everingham et al. 2010) with Single Shot MultiBox Detector(SSD) 300 (Liu et al. 2016) with VGG-16(with batch-normalization) (Simonyan and Zisserman 2015) as the backbone network. The mean average precision (mAP) is reported in Table 7 for a mean of 8 different runs. The model is trained with batch size of 8, 0.001 learning rate, SGD optimizer (Robbins and Monro 1951; Kiefer and Wolfowitz 1952) with 0.9 momentum, $5e^{-4}$ weight decay for 120000 iterations. The results are stable on different runs (mean±std). We got around 1% boost in mAP for both ErfAct and Pserf compared to ReLU.

Activation Function	mAP
ReLU	77.2 ± 0.14
Swish	77.5 ± 0.12
Leaky ReLU	77.2 ± 0.19
ELU	75.1 ± 0.22
Softplus	74.2 ± 0.25
Mish	77.6 ± 0.14
GELU	77.5 ± 0.14
PAU	77.4 ± 0.16
PReLU	77.2 ± 0.20
ReLU6	77.1 ± 0.15
ErfAct	**78.2** ± 0.12
Pserf	**78.2** ± 0.14

Table 7: Comparison between different baseline activations and ErfAct and Pserf on Object Detection problem on SSD 300 model in Pascal-VOC dataset. mean±std is reported in the table.

.

Machine Translation

Activation Function	BLEU Score on the newstest2014 dataset
ReLU	26.2 ± 0.15
Swish	26.4 ± 0.10
Leaky ReLU	26.3 ± 0.17
ELU	25.1 ± 0.15
Softplus	23.6 ± 0.16
Mish	26.3 ± 0.12
GELU	26.4 ± 0.19
PAU	26.3 ± 0.16
PReLU	26.2 ± 0.21
ReLU6	26.1 ± 0.14
ErfAct	**26.8** ± 0.11
Pserf	**26.7** ± 0.10

Table 8: Comparison between different baseline activations and ErfAct and Pserf on Machine translation problem on transformer model in WMT-2014 dataset. mean±std is reported in the table.

Machine Translation is a procedure in which text or speech is translated from one language to another language without the help of any human being. We consider the standard WMT 2014 English→German dataset for our experiment. The database contains 4.5 million training sentences. We train an attention-based 8-head transformer network (Vaswani et al. 2017) with Adam optimizer (Kingma and Ba 2015), 0.1 dropout rate (Srivastava et al. 2014), and train up to 100000 steps. We try to keep other hyperparameters similar as mentioned in the original paper (Vaswani et al. 2017). We evaluate the network performance on the newstest2014 dataset using the BLEU score metric. The mean of 5 different runs is being reported on Table 8 on the test dataset(newstest2014). From the table, it is clear that the results are stable on different runs (mean±std), and we got around 0.6% and 0.5% boost in BLEU score for ErfAct and Pserf compared to ReLU.

Baseline Table

The experiment section shows that ErfAct and Pserf beat or perform equally well with baseline activation functions in most cases while under-performs marginally on rare occasions. We provide a detailed comparison based on all the experiments in earlier sections and supplementary material with the proposed and the baseline activation functions in Table 9.

Computational Time Comparison

We present the time comparison for the baseline activation functions and ErfAct, Pserf for the mean of 100 runs for both forward and backward pass on a 32×32 RGB image in PreActResNet-18 (He et al. 2016) model in Table 10. An NVIDIA Tesla V100 GPU with 32GB ram is used to run the experiments. From Table 10 and the experiment section, it is clear that there is a small trade-off between the computational time and the model performance when compared to ReLU as the proposed activations contain trainable parameters. In contrast, the time is comparable with Swish, Mish or GELU & much better than PAU and model performance

Activation Function	Forward Pass	Backward Pass
ReLU	5.39 ± 0.39 μs	5.70 ± 1.56 μs
Swish	8.35 ± 1.44 μs	10.56 ± 2.34 μs
Leaky ReLU	5.50 ± 0.51 μs	5.97 ± 0.75 μs
ELU	6.17 ± 0.50 μs	5.93 ± 0.93 μs
Softplus	6.13 ± 0.49 μs	5.94 ± 0.55 μs
Mish	7.45 ± 2.55 μs	8.89 ± 2.85 μs
GELU	8.87 ± 1.54 μs	9.22 ± 1.75 μs
PAU	19.05 ± 2.69 μs	32.62 ± 3.76 μs
PReLU	6.12 ± 0.90 μs	6.23 ± 1.41 μs
ReLU6	5.77± 0.73 μs	5.73± 0.66 μs
ErfAct	7.41 ± 1.51 μs	10.62 ± 1.53 μs
Pserf	7.53 ± 1.77 μs	10.77 ± 1.78 μs

Table 10: Runtime comparison for the forward and backward passes for ErfAct and Pserf and baseline activation functions for a $32\times$ 32 RGB image in PreActResNet-18 model.

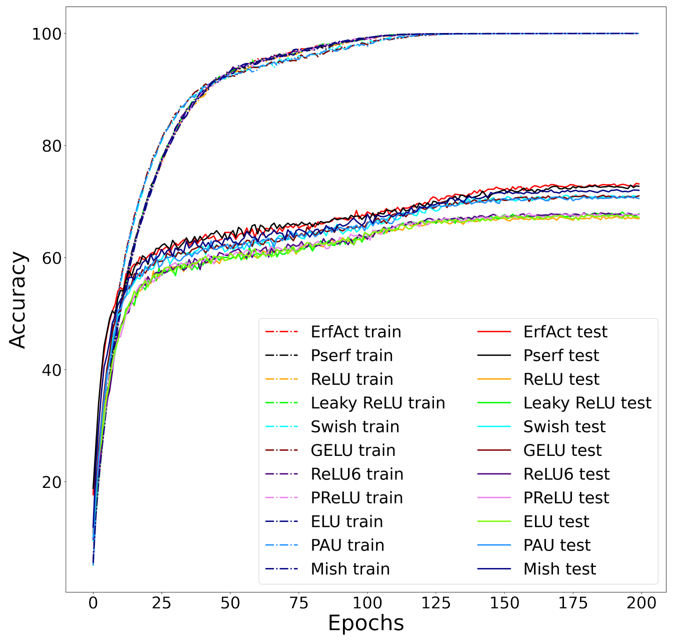

Figure 4: Top-1 Train and Test accuracy (higher is better) on CIFAR100 dataset with Shufflenet V2 (2.0x) network for different baseline activations, ErfAct, and Pserf.

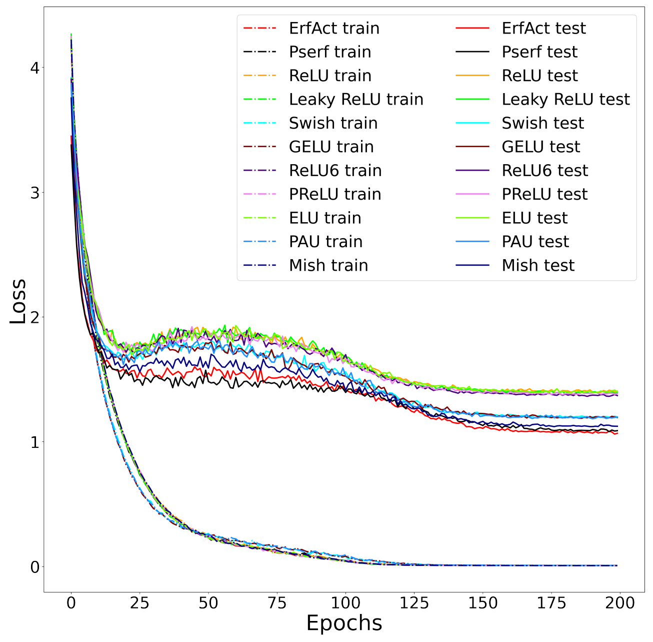

Figure 5: Top-1 Train and Test loss (lower is better) on CIFAR100 dataset with Shufflenet V2 (2.0x) network for different baseline activations, ErfAct, and Pserf.

Baselines	ReLU	Leaky ReLU	ELU	Softplus	Swish	PReLU	ReLU6	Mish	GELU	PAU
ErfAct > Baseline	54	54	54	54	51	53	53	50	52	52
ErfAct = Baseline	0	0	0	0	0	0	0	0	0	0
ErfAct < Baseline	0	0	0	0	3	1	1	4	2	2
Pserf > Baseline	54	54	54	54	51	53	53	50	52	51
Pserf = Baseline	0	0	0	0	0	0	0	0	0	0
Pserf < Baseline	0	0	0	0	3	1	1	4	2	3

Table 9: Baseline table for ErfAct and Pserf. These numbers represent the total number of models in which ErfAct and Pserf underperforms, equal or outperforms compared to the baseline activation functions

comparatively much better than baseline activations in most cases.

Conclusion

In this work, we propose two simple and effective novel activation functions. We call them ErfAct and Pserf. The method of construction is a combination of functions using trainable parameters. The proposed functions are unbounded above, bounded below, non-monotonic, smooth and zero centred. We show that both functions can approximate the ReLU activation function. Across most of the experiments, ErfAct and Pserf are top-performing activation functions from which we can conclude that the proposed functions have the potential to replace the widely used activations like ReLU, Swish or Mish.

References

Biswas, K.; Banerjee, S.; and Pandey, A. K. 2021. Orthogonal-Padé Activation Functions: Trainable Activation functions for smooth and faster convergence in deep networks. arXiv:2106.09693.

Bochkovskiy, A.; Wang, C.-Y.; and Liao, H.-Y. M. 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv:2004.10934.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165.

Chollet, F. 2017. Xception: Deep Learning with Depthwise Separable Convolutions. arXiv:1610.02357.

Clevert, D.-A.; Unterthiner, T.; and Hochreiter, S. 2016. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). arXiv:1511.07289.

Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; and Schiele, B. 2016. The Cityscapes Dataset for Semantic Urban Scene Understanding. arXiv:1604.01685.

Elfwing, S.; Uchibe, E.; and Doya, K. 2017. Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning. arXiv:1702.03118.

Everingham, M.; Gool, L.; Williams, C. K.; Winn, J.; and Zisserman, A. 2010. The Pascal Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vision*, 88(2): 303–338.

Glorot, X.; and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In Teh, Y. W.; and Titterington, M., eds., *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, 249–256. Chia Laguna Resort, Sardinia, Italy: JMLR Workshop and Conference Proceedings.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015a. Deep Residual Learning for Image Recognition. arXiv:1512.03385.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015b. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. arXiv:1502.01852.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Identity Mappings in Deep Residual Networks. arXiv:1603.05027.

Hendrycks, D.; and Gimpel, K. 2020. Gaussian Error Linear Units (GELUs). arXiv:1606.08415.

Huang, G.; Liu, Z.; van der Maaten, L.; and Weinberger, K. Q. 2016. Densely Connected Convolutional Networks. arXiv:1608.06993.

Iandola, F. N.; Han, S.; Moskewicz, M. W.; Ashraf, K.; Dally, W. J.; and Keutzer, K. 2016. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and ¡0.5MB model size. arXiv:1602.07360.

Kidger, P.; and Lyons, T. 2020. Universal Approximation with Deep Narrow Networks. arXiv:1905.08539.

Kiefer, J.; and Wolfowitz, J. 1952. Stochastic Estimation of the Maximum of a Regression Function. *Annals of Mathematical Statistics*, 23: 462–466.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In Bengio, Y.; and LeCun, Y., eds., *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Krizhevsky, A. 2009. Learning multiple layers of features from tiny images. Technical report, University of Toronto.

Krizhevsky, A. 2010. Convolutional deep belief networks on cifar-10.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, 1097–1105. Red Hook, NY, USA: Curran Associates Inc.

Le, Y.; and Yang, X. 2015. Tiny ImageNet Visual Recognition Challenge.

LeCun, Y.; Boser, B.; Denker, J. S.; Henderson, D.; Howard, R. E.; Hubbard, W.; and Jackel, L. D. 1989. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4): 541–551.

Lecun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

LeCun, Y.; Cortes, C.; and Burges, C. 2010. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2.

Lin, M.; Chen, Q.; and Yan, S. 2014. Network In Network. arXiv:1312.4400.

Lin, T.-Y.; Maire, M.; Belongie, S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C. L.; and Dollár, P. 2015. Microsoft COCO: Common Objects in Context. arXiv:1405.0312.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. SSD: Single Shot MultiBox Detector. *Lecture Notes in Computer Science*, 21–37.

Loshchilov, I.; and Hutter, F. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts. arXiv:1608.03983.

Ma, N.; Zhang, X.; Liu, M.; and Sun, J. 2021. Activate or Not: Learning Customized Activation. arXiv:2009.04759.

Ma, N.; Zhang, X.; Zheng, H.-T.; and Sun, J. 2018. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. arXiv:1807.11164.

Maas, A. L.; Hannun, A. Y.; and Ng, A. Y. 2013. Rectifier nonlinearities improve neural network acoustic models. In *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*.

Misra, D. 2020. Mish: A Self Regularized Non-Monotonic Activation Function. arXiv:1908.08681.

Molina, A.; Schramowski, P.; and Kersting, K. 2020. Padé Activation Units: End-to-end Learning of Flexible Activation Functions in Deep Networks. arXiv:1907.06732.

Nag, S.; and Bhattacharyya, M. 2021. SERF: Towards better training of deep neural networks using log-Softplus ERror activation Function. arXiv:2108.09598.

Nair, V.; and Hinton, G. E. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In Fürnkranz, J.; and Joachims, T., eds., *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, 807–814. Omnipress.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language Models are Unsupervised Multitask Learners.

Ramachandran, P.; Zoph, B.; and Le, Q. V. 2017. Searching for Activation Functions. arXiv:1710.05941.

Robbins, H.; and Monro, S. 1951. A stochastic approximation method. *Annals of Mathematical Statistics*, 22: 400–407.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv:1505.04597.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2019. MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv:1801.04381.

Simonyan, K.; and Zisserman, A. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.*, 15(1): 1929–1958.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2014. Going Deeper with Convolutions. arXiv:1409.4842.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2015. Rethinking the Inception Architecture for Computer Vision. arXiv:1512.00567.

Tan, M.; and Le, Q. V. 2020. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. arXiv:1905.11946.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention Is All You Need. arXiv:1706.03762.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; and He, K. 2017. Aggregated Residual Transformations for Deep Neural Networks. arXiv:1611.05431.

Yu, F.; Wang, D.; Shelhamer, E.; and Darrell, T. 2019. Deep Layer Aggregation. arXiv:1707.06484.

Zagoruyko, S.; and Komodakis, N. 2016. Wide Residual Networks. arXiv:1605.07146.