Neurocomputing 429 (2021) 110–117

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

PFLU and FPFLU: Two novel non-monotonic activation functions in convolutional neural networks

Meng Zhu [a,1], Weidong Min [b,c,*,2], Qi Wang [a,3], Song Zou [a,4], Xinhao Chen [a,5]

[a] School of Information Engineering, Nanchang University, Nanchang 330031, China
[b] School of Software, Nanchang University, Nanchang 330047, China
[c] Jiangxi Key Laboratory of Smart City, Nanchang 330047, China

ARTICLE INFO

Article history:
Received 14 May 2020
Revised 27 November 2020
Accepted 29 November 2020
Available online 11 December 2020
Communicated by Zidong Wang

Keywords:
Convolutional Neural Network (CNN)
Activation function
ReLU
Power Function Linear Unit (PFLU)
Faster PFLU (FPFLU)

ABSTRACT

The choice of activation functions in Convolutional Neural Networks (CNNs) is very important. Rectified Linear Unit (ReLU) has been widely-used in most CNNs. Recently, a series of non-monotonic activation functions gradually become the new standard to enhance performance of CNNs. Inspired by them, this paper firstly proposes a novel non-monotonic activation function called Power Function Linear Unit (PFLU). The negative part of PFLU is non-monotonic and closer to zero with the negative input decreasing, which can maintain sparsity of the negative part while introducing negative activation values and non-zero derivative values for the negative part. The positive part of PFLU does not use identity mapping but is closer to identity mapping with the positive input increasing, which can bring non-linearity property for the positive part. Next, this paper proposes faster PFLU (FPFLU). A wide range of classification experiments show that PFLU tends to work better than current state-of-the-art non-monotonic activation functions, and FPFLU can run faster than most non-monotonic activation functions.

1. Introduction

The activation function is the key to introduce non-linearity into Convolutional Neural Networks (CNNs) and plays an important role to enhance performance of CNNs. Rectified Linear Unit (ReLU) [1] has been used as the standard activation function in most CNNs due to its simple implementation and great performance. Compared with traditional Sigmoid and Tanh, ReLU uses identity mapping in its positive part. This special design alleviates the vanishing gradient problem [2,3] since the derivative of one for positive activation values is not contractive [4]. Moreover, ReLU always outputs zero in the negative part, which can bring sparsity for negative activation values. Compared with Softplus [5], computational complexity of ReLU is more lightweight and ReLU can produce the exact sparsity in the negative part. These main advantages of ReLU make it possible to train very deep CNNs and accelerate the training process.

Over the years, many monotonic activation functions have been proposed to replace ReLU, which mainly include Leaky Rectified Linear Unit (LReLU) [6], Parametric Rectified Linear Unit (PReLU) [7], Exponential Linear Unit (ELU) [8] and Scaled Exponential Linear Unit (SELU) [9]. Their main optimization directions are all to let the negative quadrant output negative activation values and non-zero derivative values rather than zero activation values and zero derivative values, thus enhance the performance of ReLU. But none has managed to gain the widespread adoption that ReLU enjoys. Many practitioners have favored the simplicity and reliability of ReLU because the performance improvements of the other activation functions tend to be inconsistent across different models and datasets.

This paper firstly proposes a novel non-monotonic activation function called Power Function Linear Unit (PFLU). The negative part of PFLU is non-monotonic and closer to zero with the negative input decreasing, which can maintain sparsity of the negative part while introducing negative activation values and non-zero derivative values for the negative part. The positive part of PFLU does not use identity mapping but is closer to identity mapping with the positive input increasing, which can bring non-linearity property for the positive part. Next, this paper proposes faster PFLU (FPFLU),

* Corresponding author at: School of Software, Nanchang University, Nanchang 330047, China.
E-mail addresses: mengzhu@email.ncu.edu.cn (M. Zhu), minweidong@ncu.edu.cn (W. Min), 351029018003@email.ncu.edu.cn (Q. Wang), songzou@email.ncu.edu.cn (S. Zou), 411014518286@email.ncu.edu.cn (X. Chen).
[1] ORCID: 0000-0002-4900-8973
[2] ORCID: 0000-0003-2526-2181
[3] ORCID: 0000-0003-0445-5603
[4] ORCID: 0000-0002-5744-8792
[5] ORCID: 0000-0002-8272-8884

https://doi.org/10.1016/j.neucom.2020.11.068
0925-2312/

which uses identity mapping in its positive part, and maintains non-monotonic in its negative part.

The main contributions of this paper are summarized as follows: 1) this paper proposes a novel non-monotonic activation function called PFLU; 2) this paper proposes another novel non-monotonic activation function called FPFLU; 3) a wide range of classification experiments show that PFLU tends to work better than current state-of-the-art non-monotonic activation functions, and FPFLU can run faster than most non-monotonic activation functions.

The remaining of this paper is organized as follows. Section 2 reviews a series of non-monotonic activation functions. Section 3 introduces the proposed PFLU and FPFLU. Then experiments are introduced in Section 4. Section 5 presents conclusion and future works.

2. Related works

ReLU has been the widely-used activation function in CNNs. Many various alternatives to ReLU have been proposed, such as LReLU [6], PReLU [7], ELU [8] and SELU [9]. But none has managed to replace ReLU due to inconsistent gains. Recently, a series of non-monotonic activation functions are proposed, such as Gaussian Error Linear Unit (GELU) [10], Swish [11], HardSwish [12], Rectified Exponential Unit (REU) [13], and Mish [14], which have similar mathematical properties and showcase strong and improved results across different models and datasets. They gradually become the new standard to enhance performance of CNNs. As shown in Fig. 1, it can be clearly found that the biggest difference between monotonic activation functions and non-monotonic activation functions is non-monotonic property in the negative quadrant.

ReLU [1] is described as Eq. (1):

$$\mathrm{f}(x) = \max(x, 0) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leqslant 0 \end{cases} \tag{1}$$

Here x is the input and $\mathrm{f}(x)$ is the output (activation values). ReLU uses an identity mapping in its positive quadrant and outputs zero in its negative part. Identity mapping can alleviate the problem of vanishing gradients. Compared with traditional sigmoid or tanh function, there are two main changes:

- unbounded in the positive part;
- sparse in the negative part.

GELU [10] is described as Eq. (2):

$$\begin{aligned} \mathrm{f}(x) &= x \cdot \Phi(x) = x \cdot \mathrm{P}(X \leqslant x) = \frac{x}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2}} \mathrm{d}t \\ &= \frac{x}{2}\left[1 + \mathrm{erf}\left(\frac{x}{\sqrt{2}}\right)\right] \end{aligned} \tag{2}$$

Here $\Phi(x) = \mathrm{P}(X \leqslant x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2}}$ is the cumulative distribution function of the standard normal distribution. GELU is a continuously differentiable function. Due to its complex computability, GELU has another two forms of approximate calculation, described as Eqs. (3) and (4), respectively:

$$\mathrm{f}(x) = x \cdot \sigma(1.702x) \tag{3}$$

$$\mathrm{f}(x) = \frac{x}{2} \cdot (1 + \tanh(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3))) \tag{4}$$

Swish [11] is described as Eq. (5):

$$\mathrm{f}(x) = x \cdot \sigma(\beta x) = \frac{x}{1 + e^{-\beta x}} \tag{5}$$

Different from other hand-designed activation functions, Swish is discovered by leverage automatic search techniques. Here β can be fixed or trainable. Moreover, it is always a default constant of 1.0.

HardSwish [12] is described as Eq. (6):

$$\mathrm{f}(x) = x\frac{\min(6, \max(x+3, 0))}{6} \tag{6}$$

Sigmoid σ function is much more expensive to compute, so HardSwish replaces Sigmoid function with its piece-wise linear hard analog: $\frac{\min(6,\max(x+3,0))}{6}$. HardSwish is continuous but not derivable.

REU [13] is described as Eq. (7):

$$\mathrm{f}(x) = \begin{cases} x & \text{if } x > 0 \\ x \cdot e^{x} & \text{if } x \leqslant 0 \end{cases} \tag{7}$$

An identity mapping is still used in the positive part of REU. To get the non-monotonic property in the negative part, REU uses $x \cdot e^{x}$. Meanwhile, REU is derivable.

Mish [14] is described as Eq. (8):

$$\mathrm{f}(x) = x \cdot \tanh(\mathrm{softplus}(x)) = x \cdot \tanh(\ln(1 + e^{x})) \tag{8}$$

Similar to GELU and Swish, Mish also uses multiplication form.

3. The proposed method

This section firstly presents rough explanation why non-monotonic activation functions work better than current monotonic activation functions and introduces how to construct similar non-monotonic activation functions in Section 3.1. Then PFLU and FPFLU are proposed in Sections 3.2 and 3.3, respectively.

3.1. Non-monotonic activation functions review

Non-monotonic activation functions showcase strong and improved results across different models and datasets, which this paper thinks is mainly based on that the negative part is non-monotonic, which can maintain sparsity of the negative part while introducing negative activation values and non-zero derivative values for the negative part.

Observing on current non-monotonic activation functions, they can be abstracted as Eq. (9):

$$\mathrm{f}(x) = x \cdot \mathrm{s}(\beta x) \tag{9}$$

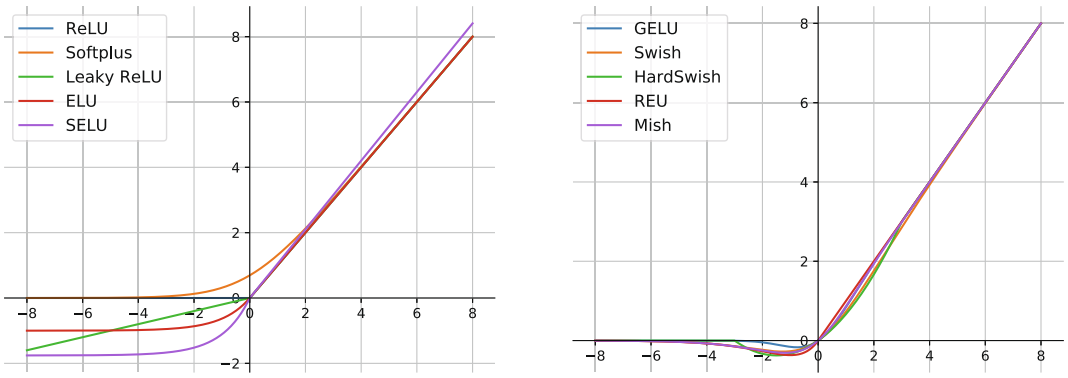
Here $\mathrm{s}(\beta x)$ denotes "S" function and its range is $[0, 1]$. β can be fixed or trainable. Moreover, it is always a default constant of 1.0. The forms of $\mathrm{s}(x)$ can be mainly classified into three categories, as shown in Fig. 2. The first form is like $\mathrm{s}(x)$ of Swish (that is Sigmoid). The second form is like $\mathrm{s}(x)$ of HardSwish (that is HardSigmoid). The third form is like $\mathrm{s}(x)$ of REU.

Therefore, it can be possible to construct similar non-monotonic activation functions by designing $\mathrm{s}(x)$ functions. In addition, it can take advantage of the additional flexibility of trainable parameter to add a trainable parameter of β.

3.2. Power function linear unit (PFLU)

Different from most non-monotonic activation functions whose meta units are exponential function form, the proposed non-monotonic activation functions are construct with new kind of expression form, whoes meta unit is power function form. Firstly, this paper proposes PFLU, which is defined as Eq. (10):

$$\mathrm{f}(x) = x \cdot \frac{1}{2}\left(1 + \frac{x}{\sqrt{1 + x^2}}\right) \tag{10}$$



(a) Monotonic activation functions (b) Non-monotonic activation functions

Fig. 1. Comparison between monotonic activation functions and non-monotonic activation functions. Best viewed in color.

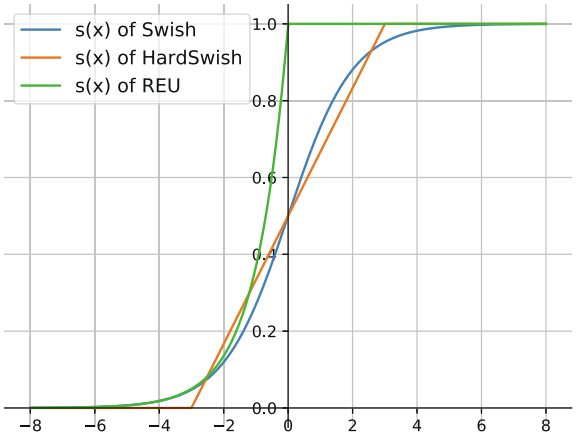


Fig. 2. Three main forms of $s(x)$.

Its first derivative is defined as Eq. (11):

$$f'(x) = \frac{1}{2}\left(1 + \frac{x}{\sqrt{1+x^2}} + \frac{x}{(1+x^2)\sqrt{1+x^2}}\right) \tag{11}$$

Some properties of PFLU are listed as follows:

- Obviously, $f(x)$ is derivable $\forall x \in \boldsymbol{R}$
- Obviously, $f(0) = 0$ and $f'(0) = \frac{1}{2}$
- When $x > 0, 0 < f(x) < x$ and $f'(x) > 0$. As $x \to +\infty, f(x) \to x$

  Proof :

  Obviously, $\forall x \in \boldsymbol{R},\ \frac{1}{2}\left(1 + \frac{x}{\sqrt{1+x^2}}\right) \in (0,1)$

  $\Rightarrow$ When $x > 0,\ 0 < f(x) < x$

  Proof :

  $f(x) = \frac{1}{2}\left(x + \frac{x^2}{\sqrt{1+x^2}}\right)$

  $= \frac{1}{2}\left(x + \sqrt{1+x^2} - \frac{1}{\sqrt{1+x^2}}\right)$

  $\Rightarrow$ As $x \to +\infty,\ f(x) \to \frac{1}{2}(x + x - 0) = x$

- When $x \leqslant 0, f(x) \leqslant 0$. When $x \in [\frac{1-\sqrt{5}}{\sqrt{2(\sqrt{5}-1)}}, 0], f'(x) \geqslant 0$, when $x < \frac{1-\sqrt{5}}{\sqrt{2(\sqrt{5}-1)}}, f'(x) < 0$. As $x \to -\infty, f(x) \to 0$

  Proof :

  Similarly, $\forall x \in \boldsymbol{R},\ \frac{1}{2}\left(1 + \frac{x}{\sqrt{1+x^2}}\right) \in (0,1)$

  $\Rightarrow$ When $x < 0,\ f(x) < 0$

  Proof :

  It imples that only $\exists x_0 \in (-1,0), f'(x_0) = 0$

  $\sqrt{1+x_0^2} \neq 0 \Rightarrow \sqrt{1+x_0^2} + x_0 + \frac{x_0}{1+x_0^2} = 0$

  Let $x_0 = \tan(\theta), \theta \in (\frac{\pi}{4}, 0)$

  then $\frac{1}{\cos(\theta)} + \tan(\theta) + \sin(\theta)\tan(\theta)\cos^2(\theta) = 0$

  $\because \cos(\theta) \neq 0$

  $\Rightarrow 1 + \sin(\theta) + \sin(\theta)\cos^2(\theta) = 0$

  $\Rightarrow 1 + \sin(\theta) + \sin(\theta)(1 - \sin^2(\theta)) = 0$

  $\Rightarrow \sin^3(\theta) - 2\sin(\theta) - 1 = 0$

  $\Rightarrow (1 + \sin(\theta))(\sin^2(\theta) - \sin(\theta) - 1) = 0$

  $\because \theta \in (\frac{\pi}{4}, 0), 1 + \sin(\theta) \neq 0$

  $\Rightarrow \sin^2(\theta) - \sin(\theta) - 1 = 0$

  $\Rightarrow \sin(\theta) = \frac{1-\sqrt{5}}{2}$

  $\Rightarrow x_0 = \tan(\theta) = \frac{1-\sqrt{5}}{\sqrt{2(\sqrt{5}-1)}}$

  Proof :

  Similarly, $f(x) = \frac{1}{2}\left(x + \sqrt{1+x^2} - \frac{1}{\sqrt{1+x^2}}\right)$

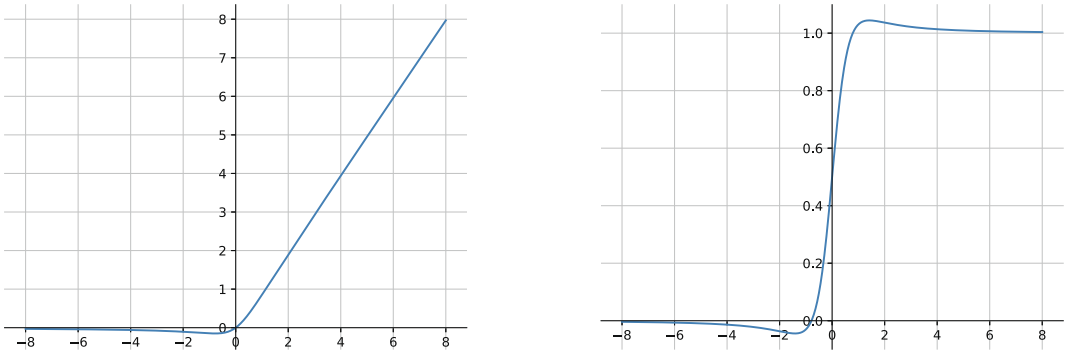
  $\Rightarrow$ As $x \to -\infty,\ f(x) \to \frac{1}{2}(x + |x| - 0) = 0$

Fig. 3a and b plot the shape of PFLU and its first derivative, respectively.

3.3. Faster power function linear unit (FPFLU)

Next, this paper proposes identity mapped PFLU (IMPFLU), which uses identity mapping in its positive part, and maintains non-monotonic in its negative part. IMPFLU is described as Eq. (12):

$$\begin{aligned} f(x) &= x \cdot \max\left(1, 1 + \frac{x}{\sqrt{1+x^2}}\right) \\ &= x \cdot \begin{cases} 1 & \text{if } x > 0 \\ 1 + \frac{x}{\sqrt{1+x^2}} & \text{if } x \leqslant 0 \end{cases} \end{aligned} \tag{12}$$



(a) The shape of PFLU (b) The first derivative of PFLU

Fig. 3. Illustration of PFLU.

However, square root operation is expensive to compute. Therefore, it is expensive to compute for either PFLU or IMPFLU. To alleviate this problem, this paper further proposes FPFLU, which also uses identity mapping in its positive part like IMPFLU, and also maintains non-monotonic in its negative part like IMPFLU. Different from IMPFLU, FPFLU contains none square root operation. Therefore, FPFLU must run faster than IMPFLU. FPFLU is described as Eq. (13):

$$f(x) = \max\left(x, \frac{x}{1+x^2}\right) = x \cdot \begin{cases} 1 & \text{if } x > 0 \\ \frac{1}{1+x^2} & \text{if } x \leqslant 0 \end{cases} \tag{13}$$

Its first derivative is defined as Eq. 14:

$$f'(x) = \begin{cases} 1 & \text{if } x > 0 \\ \frac{1-x^2}{(1+x^2)^2} & \text{if } x \leqslant 0 \end{cases} \tag{14}$$

Some properties of FPFLU are listed as follows:

- $f(x)$ is derivable $\forall x \in \boldsymbol{R}$

  Proof :
  $\begin{cases} f(0^-) = f(0^+) = 0 \\ f'(0^-) = f'(0^+) = 1 \end{cases}$
  $\Rightarrow f(x)$ is derivable $\forall x \in \boldsymbol{R}$

- When $x > 0,\ f(x) > 0$ and $f'(x) = 1$
- When $x \leqslant 0,\ f(x) \leqslant 0$. Obviously, $x \in [-1, 0],\ f'(x) \geqslant 0, x < -1,\ f'(x) < 0$. As $x \to -\infty,\ f(x) \to 0$

  Proof :
  When $x \leqslant 0,\ f(x) = \frac{1}{x+\frac{1}{x}}$
  $\Rightarrow$ As $x \to -\infty,\ f(x) \to 0$

Fig. 4a and b plot the shape of FPFLU and its first derivative, respectively.

4. Experiments

This section shows and analyzes experimental results. It will be demonstrated that PFLU and FPFLU can match or outperform other baseline non-monotonic activation functions rather than trying to achieve the most advanced performance on a certain dataset by all manner of means. All activation functions follow the same experimental setups.

4.1. Hardware and software setups

All of the training tasks are accomplished on a computer with configurations described in Table 1:

4.2. Datasets

Fashion-MNIST. Fashion MNIST dataset [16] consists of 28×28 grayscale images of 70,000 fashion products from 10 categories, with 7,000 images per category. The training set has 60,000 images and the test set has 10,000 images.

CIFAR. CIFAR dataset [17] consists of colored natural images with 32×32 pixels. CIFAR-10 dataset consists of images drawn from 10 and CIFAR-100 dataset from 100 classes. The training and test sets contain 50,000 and 10,000 images, respectively.

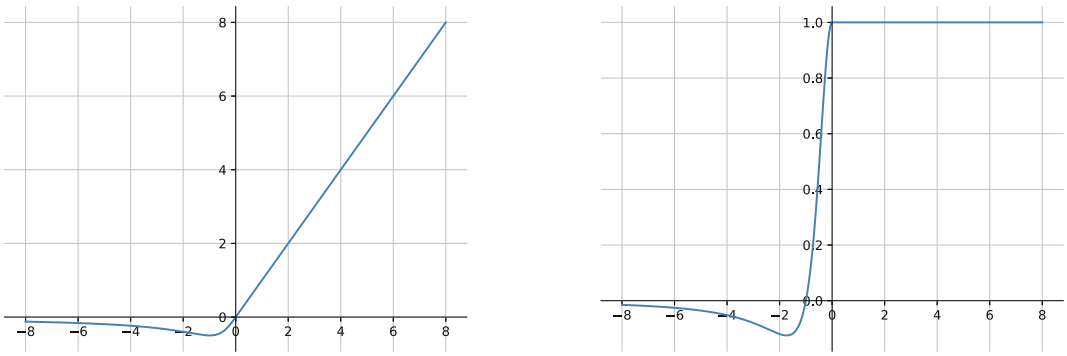
4.3. Models

Many classic image classification models have been proposed, which can be roughly classified into six architectures, as shown in Fig. 5. Representative model architecture of Fig. 5a includes AlexNet [18], VGGNet [19], TinyDarkNet [20] and et al. Representative model architecture of Fig. 5b includes ResNet [21], WideResNet [22], ResNeXt [23] and et al. Representative model architecture of Fig. 5c includes PreResNet [24], DenseNet [25] and et al. Representative model architecture of Fig. 5d includes Xception [26] and et al. Representative model architecture of Fig. 5e includes DarkNet [20], ShuffleNetV2 [27] and et al. Representative model architecture of Fig. 5f includes MobileNetV3 [12], EfficientNet [28] and et al.

This paper selects VGGNet16 (VN), ResNet50 (RN), PreResNet50 (PRN), Xception (XCP), ShuffleNetV2-1.0 (SN) and EfficientNet-B0 (EN) as baeline models from each architecture. In order to be suitable for the input images with 28×28 or 32×32 pixels, all models only retain the last three down sampling. In addition, the number of channels becomes half of the original for the ResNet50, PreResNet50 and Xception, the number of channels keeps the original unchanged for other models.

4.4. Training setups

This paper normalizes the input images using channel means and standard deviations for pre-processing. Then this paper uses data augmentation by random rotation, shifting, flipping horizontally. All models use the categorical cross entropy loss function and are initialized with He initialization [7]. All models are trained with using AdamW [29] on the Fashion-MNIST dataset, and SGDM



(a) The shape of FPFLU (b) The first derivative of FPFLU

Fig. 4. Illustration of FPFLU.

Table 1
Hardware and software setups.

CPU	one Intel(R) Core(TM) i5-8500 CPU
GPU	one Nvidia Quadro RTX 4000 8 GB
RAM	32 GB
Python	3.8.3
PyTorch [15]	1.6.0
Cuda	10.2
Cudnn	8.0.3

[30] on the CIFAR dataset. Train batch size values are 512 on the VGGNet16, 256 on ShuffleNetV2-1.0, and 128 on other models, respectively. The entire training epochs is equal to 60 on the Fashion-MNIST dataset and 150 on the CIFAR dataset, respectively. This paper adopts learning rate decay. On the Fashion-MNIST dataset, learning rate during the first 20 epochs is equal to 0.001, and during the following 40 epochs divided by 10 once after each 20 epochs. And on the CIFAR dataset, learning rate during the first 80 epochs is equal to 0.1, and during the following 70 epochs divided by 10 once after each 35 epochs. This paper applies L2 weight decay which additionally helps stabilize training. The weight decay is 5×10^{-4} for the weights of all convolution layers and all fully connected layers.

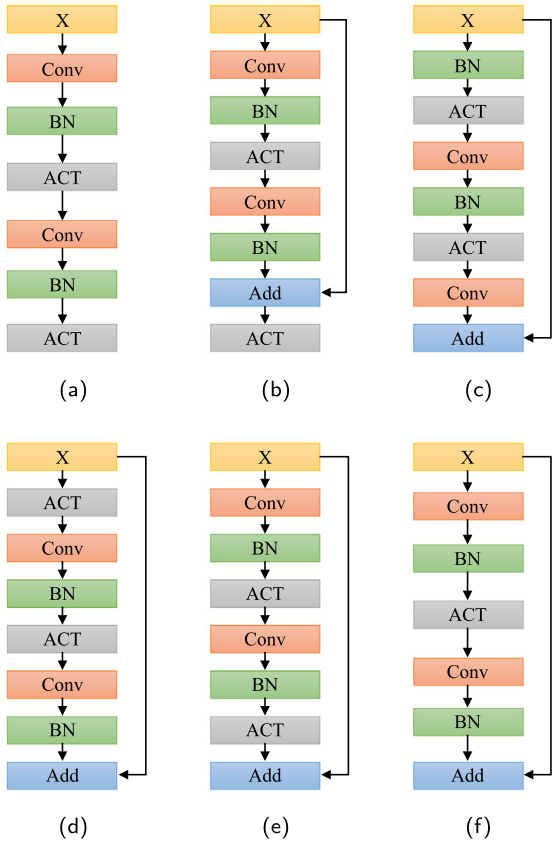


Fig. 5. Comparison between different model architectures. (a) sequential architecture; (b) standard residual architecture; (c) full pre-activation residual architecture; (d) ACT-only pre-activation residual architecture; (e) ACT before addition residual architecture; (f) linear residual architecture.

4.5. Results and analyses

4.5.1. Top-1 accuracy rate comparison

Table 2–4 report the experimental results. The results show that none of these non-monotonic activation functions can always get the best performance. But these non-monotonic activation functions almost consistently perform better than ReLU.[6]

4.5.2. The number of models comparison

Table 5 shows the proposed each activation function in comparison to each baseline activation function this paper considers. The results in Table 5 are aggregated by comparing the performance of the proposed each activation function to the performance of different activation functions applied into a variety of models across multiple datasets. The improvement of PFLU over other baseline activation functions is statistically significant under a one-sided paired sign test. Similarly, Table 6 shows the proposed FPFLU in comparison to each baseline activation function this paper considers. The results in Table 6 are aggregated by comparing the performance of the proposed FPFLU to the performance of different activation functions applied into a variety of models across multiple datasets. The results show that FPFLU performs worse than GELU, Swish, Mish and PFLU. The performance of FPFLU is similar with HardSwish and REU. Table 7 reports the number of the best results appearing for different activation functions applied into a variety of models across multiple datasets. PFLU and Mish both get the maximum number of the best results appearing.

4.5.3. Average and standard deviation comparison

Table 8 shows average and standard deviation of top-1 accuracy rate for six models on the same dataset. It can be found that Swish, Mish and PFLU are more robust for different models than other

[6] Here β in Swish is a fixed constant of 1.0.

Table 2
Top-1 accuracy rate (%) on Fashion-MNIST dataset [16] for different activation functions. Note that in each column, the red denotes the best result, the green denotes the second good result, and the blue denotes the third good result. Best viewed in color.

	VN	RN	PRN	XCP	SN	EN
ReLU [1]	93.95	94.20	94.47	94.68	93.16	94.21
GELU [10]	93.89	94.43	94.45	94.75	93.87	94.42
Swish[1] [11]	94.07	94.18	94.55	94.77	93.78	94.59
HardSwish [12]	94.08	94.57	94.52	94.72	93.49	94.55
REU [13]	93.87	94.75	94.38	94.82	93.44	94.54
Mish [14]	93.91	94.45	94.71	94.61	93.75	94.66
PFLU (ours)	93.97	94.33	94.33	94.83	93.79	94.61
FPFLU (ours)	93.91	94.37	94.26	94.84	93.64	94.65

Table 3
Top-1 accuracy rate (%) on CIFAR-10 dataset [17] for different activation functions.

	VN	RN	PRN	XCP	SN	EN
ReLU [1]	94.16	93.60	94.28	94.10	92.78	93.66
GELU [10]	94.25	94.11	94.76	94.18	93.24	94.14
Swish [11]	94.18	94.86	94.89	94.22	93.42	94.38
HardSwish [12]	94.16	94.56	94.49	93.96	93.31	93.98
REU [13]	94.29	93.53	94.49	94.22	93.02	94.13
Mish [14]	94.21	94.15	95.31	94.27	93.58	94.23
PFLU (ours)	94.14	94.36	94.64	94.38	93.41	94.39
FPFLU (ours)	94.14	94.12	94.59	94.18	93.15	94.30

Table 4
Top-1 accuracy rate (%) on CIFAR-100 dataset [17] for different activation functions.

	VN	RN	PRN	XCP	SN	EN
ReLU [1]	75.94	74.42	75.86	76.16	71.62	75.90
GELU [10]	75.06	75.02	76.80	76.32	72.47	77.10
Swish [11]	74.96	75.54	77.47	76.43	72.29	76.50
HardSwish [12]	74.25	75.97	76.52	76.16	72.53	76.23
REU [13]	75.13	75.25	76.76	75.42	72.27	75.54
Mish [14]	75.13	75.45	76.73	76.52	72.53	76.36
PFLU (ours)	75.46	76.11	76.69	76.89	72.45	77.12
FPFLU (ours)	75.72	76.05	76.67	75.52	72.13	75.56

Table 5
The number of models on which the proposed PFLU outperforms or underperforms each baseline activation function this paper compared against in the experiments.

	PFLU (ours) > Baseline	PFLU (ours) < Baseline
ReLU [1]	15	3
GELU [10]	11	7
Swish [11]	11	7
HardSwish [12]	12	6
REU [13]	14	4
Mish [14]	10	8
FPFLU (ours)	13	5

Table 6
The number of models on which the proposed FPFLU outperforms or underperforms each baseline activation function this paper compared against in the experiments.

	FPFLU (ours) > Baseline	FPFLU (ours) < Baseline
ReLU [1]	12	6
GELU [10]	7	10
Swish [11]	5	13
HardSwish [12]	9	9
REU [13]	12	6
Mish [14]	4	13
PFLU (ours)	5	13

activation functions. Table 9 shows average and standard deviation of top-1 accuracy rate for three datasets on the same model. It can be found that PFLU is more robust for different datasets than other activation functions. And Swish is also very robust for different datasets. Note that ReLU gets the best performance for three datasets on the VGGNet16 (VN).

Table 7
The number of the best results appearing across different datasets and models.

ReLU [1]	1
GELU [10]	1
Swish [11]	2
HardSwish [12]	2
REU [13]	2
Mish [14]	5
PFLU (ours)	5
FPFLU (ours)	1

Table 8
Average and standard deviation of top-1 accuracy rate for six models on the same dataset. FM denotes Fashion-MNIST. C10 denotes CIFAR-10. C100 denotes CIFAR-100.

	FM		C10		C100	
	avg	std	avg	std	avg	std
ReLU [1]	94.11	0.48	93.76	0.51	74.98	1.61
GELU [10]	94.30	0.32	94.11	0.45	75.46	1.56
Swish [11]	94.32	0.34	94.33	0.49	75.53	1.65
HardSwish [12]	94.32	0.42	94.08	0.41	75.28	1.43
REU [13]	94.30	0.49	93.95	0.51	75.06	1.36
Mish [14]	94.35	0.38	94.29	0.51	75.45	1.43
PFLU (ours)	94.31	0.35	94.22	0.39	75.79	1.59
FPFLU (ours)	94.28	0.41	94.08	0.45	75.27	1.46

4.5.4. Explanations in theory

From the results and result analyses in Section 4.5.1–4.5.3, it can be found that the proposed PFLU, GELU, Swish or Mish tends to perform better than the proposed FPFLU and REU, which shows that identity mapping in the positive part is no longer an advantage. This paper thinks that the positive part does not use identity mapping but is closer to identity mapping with the positive input increasing, which can bring non-linearity property for the positive part, thus being more robust to data distribution. The proposed PFLU, GELU, Swish or Mish also tends to perform better than HardSwish, which shows that it is also very important to be derivable for the performance of an activation function. The proposed PFLU tends to perform better than GELU, Swish and Mish. This paper thinks that the negative part of PFLU closes to zero more slowly with negative input decreasing, thus maintaining more nonlinearity.

4.5.5. Training and evaluation time comparison

While the non-monotonic activation functions improve accuracy compared with ReLU, they come with non-zero computational cost. Therefore, computational complexity is another index for evaluating the performance of activation function. In the experiments of this paper, training time of the same activation function differ from implemented by this paper and the standard function of the PyTorch. For fair comparison, all activation functions are implemented by this paper. Table 10 compares training and evaluation time between different activation functions. Training and evaluation time are measured by training ResNet50 for an epoch on CIFAR-100 dataset. Time is in second (s). Then computational complexities of these activation functions can be roughly sorted as follows:

$$\text{ReLU} < \text{REU} < \text{HardSwish} < \text{FPFLU (ours)} < \text{Swish} < \text{PFLU (ours)} < \text{GELU} < \text{Mish}$$

Table 9
Average and standard deviation of top-1 accuracy rate for three datasets on the same model.

	VN		RN		PRN		XCP		SN		EN	
	avg	std	avg	std	avg	std	avg	std	avg	std	avg	std
ReLU [1]	88.02	8.54	87.41	9.18	88.20	8.73	88.31	8.60	85.85	10.07	87.92	8.50
GELU [10]	87.73	8.96	87.85	9.08	88.67	8.39	88.42	8.56	86.53	9.94	88.55	8.10
Swish [11]	87.74	9.03	88.19	8.95	88.97	8.13	88.47	8.52	86.50	10.05	88.49	8.48
HardSwish [12]	87.50	9.37	88.37	8.77	88.51	8.48	88.28	8.58	86.44	9.84	88.25	8.50
REU [13]	87.76	8.93	87.84	8.92	88.54	8.33	88.15	9.01	86.24	9.88	88.07	8.86
Mish [14]	87.75	8.92	88.02	8.89	88.92	8.62	88.47	8.45	86.62	9.96	88.42	8.53
PFLU (ours)	87.86	8.77	88.27	8.60	88.55	8.39	88.70	8.35	86.55	9.97	88.71	8.19
FPFLU (ours)	87.92	8.63	88.18	8.58	88.50	8.37	88.18	8.96	86.31	10.03	88.17	8.92

Table 10
Comparison of training and evaluation time between different activation function.

	Train	Eval
ReLU [1]	46.50	2.22
GELU [10]	74.51	4.29
Swish [11]	69.58	3.93
HardSwish [12]	59.07	3.30
REU [13]	55.56	2.98
Mish [14]	123.25	8.24
PFLU (ours)	73.10	4.39
FPFLU (ours)	62.32	3.45

5. Conclusion and future works

This paper proposes tow novel non-monotonic activation functions: PFLU and FPFLU. A wide range of classification experiments show that PFLU tends to work better than current state-of-the-art non-monotonic activation functions, and FPFLU can run faster than most non-monotonic activation functions. If better performance is the demand, PFLU can be chosen. If faster speed is the demand, FPFLU can be selected. PFLU and FPFLU provide more choices for enhancing the performance of CNNs. Future works include to benchmark them on the ImageNet [31] dataset and to extend their theoretical understandings.

CRediT authorship contribution statement

Meng Zhu: Conceptualization, Methodology, Software, Data curation, Writing - original draft, Writing - review & editing. **Weidong Min:** Writing - review & editing, Resources, Supervision, Project administration, Funding acquisition. **Qi Wang:** Writing - review & editing. **Song Zou:** Writing - review & editing. **Xinhao Chen:** Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 62076117, 61762061), the Natural Science Foundation of Jiangxi Province, China (Grant No. 20161ACB20004) and Jiangxi Key Laboratory of Smart City, China (Grant No. 20192BCD40002).

References

[1] V. Nair, G.E. Hinton, Rectified linear units improve restricted boltzmann machines, in: International Conference on Machine Learning, 2010, pp. 807–814..
[2] S. Hochreiter, The vanishing gradient problem during learning recurrent neural nets and problem solutions, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 6 (2) (1998) 107–116, https://doi.org/10.1142/S0218488598000094.
[3] S. Hochreiter, Y. Bengio, P. Frasconi, et al., Gradient flow in recurrent nets: The difficulty of learning long-term dependencies, URL: https://ml.jku.at/publications/older/ch7.pdf (2001)..
[4] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, International Conference on Artificial Intelligence and Statistics (2011) 315–323.
[5] C. Dugas, Y. Bengio, F. Belisle, et al., Incorporating second-order functional knowledge for better option pricing, in: Advances in Neural Information Processing Systems, 2001, pp. 472–478.
[6] A.L. Maas, A.Y. Hannun, A.Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in: International Conference on Machine Learning, 2013..
[7] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: IEEE International Conference on Computer Vision, 2015, pp. 1026–1034.
[8] D.A. Clevert, T. Unterthiner, S. Hochreiter, Fast and accurate deep network learning by exponential linear units (elus), in: International Conference on Learning Representations, 2015..
[9] G. Klambauer, T. Unterthiner, A. Mayr, S. Hochreiter, Self-normalizing neural networks, in: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, 2017.
[10] D. Hendrycks, K. Gimpel, Bridging nonlinearities and stochastic regularizers with gaussian error linear units, URL: https://openreview.net/pdf?id=Bk0MRI5lg (2016)..
[11] P. Ramachandran, B. Zoph, Q.V. Le, Searching for activation functions, in: International Conference on Learning Representations, 2018..
[12] A. Howard, M. Sandler, G. Chu, et al., Searching for mobilenetv3, in: IEEE International Conference on Computer Vision, 2019, pp. 1314–1324.
[13] Y. Ying, J. Su, P. Shan, et al., Rectified exponential units for convolutional neural networks, IEEE Access 7 (2019) 101633–101640, https://doi.org/10.1109/ACCESS.2019.2928442.
[14] D. Misra, Mish: A self regularized non-monotonic neural activation function, URL: http://arxiv.org/abs/1908.08681 (2019)..
[15] A. Paszke, S. Gross, F. Massa, et al, Pytorch: An imperative style, high-performance deep learning library, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. Alche, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems, 2019, pp. 8026–8037..
[16] H. Xiao, K. Rasul, R. Vollgraf, Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms, 2017, URL: http://arxiv.org/abs/1708.07747.
[17] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images, URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.222.9220&rep=rep1&type=pdf (2009)..
[18] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1106–1114.
[19] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: International Conference on Learning Representations, 2015..
[20] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 779–788.
[21] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[22] S. Zagoruyko, N. Komodakis, Wide residual networks, in: R.C. Wilson, E.R. Hancock, W.A.P. Smith (Eds.), Proceedings of the British Machine Vision Conference, 2016, pp. 8701–8712.
[23] S. Xie, R. Girshick, P. Dollar, et al., Aggregated residual transformations for deep neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1492–1500.
[24] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in: European Conference on Computer Vision, 2016, pp. 630–645.
[25] G. Huang, Z. Liu, L. Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.
[26] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1251–1258.
[27] N. Ma, X. Zhang, H.T. Zheng, J. Sun, Shufflenet v2: Practical guidelines for efficient cnn architecture design, in: European Conference on Computer Vision, 2018, pp. 116–131.
[28] M. Tan, Q. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in: K. Chaudhuri, R. Salakhutdinov (Eds.), International Conference on Machine Learning, 2019, pp. 6105–6114..
[29] H. Loshchilow, F. Hutter, Decoupled weight decay regularization, in: International Conference on Learning Representations, 2019..
[30] I. Sutskever, J. Martens, G. Dahl, G. Hinton, On the importance of initialization and momentum in deep learning, in: International Conference on Machine Learning, 2013, pp. 1139–1147..
[31] J. Deng, W. Dong, R. Socher, et al., Imagenet:A large-scale hierarchical image database, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.

Meng Zhu received the B.E. degree in computer science and technology from Nanchang University, China in 2018. He is currently pursuing the M.E. degree at Nanchang University, China. His research interests include computer vision, natural language processing and reinforcement learning.

Weidong Min received the B.E., M.E. and Ph.D. degrees in computer application from Tsinghua University, China in 1989, 1991 and 1995, respectively. He is currently a Professor and the Dean, School of Software, Nanchang University, China. He is an Executive Director of China Society of Image and Graphics. His current research interests include image and video processing, artificial intelligence, big data, distributed system and smart city information technology. Since 2015 he has been a Professor with Nanchang University, China. From 2011 to 2014 he cooperated with School of Computer Science and Software Engineering, Tianjin Polytechnic University, China. From 1998 to 2014 he worked as a Senior Researcher and Senior Project Manager at Corel and other companies in Canada. From 1995 to 1997 he was a Post-Doctoral Researcher at the University of Alberta, Canada. From 1994 to 1995 he was an Assistant Professor at Tsinghua University, China.

Qi Wang received the M.E. degree in computer science and technology from Nanchang University, China in 2017. He is currently pursuing the Ph.D. degree at Nanchang University, China. His current research focuses on computer vision.

Song Zou received the M.E. degree in computer software and theory from Zhejiang University, China in 2011. He is currently pursuing the Ph.D. degree at Nanchang University, China. His research interests include image and video processing, vision-based human action understanding and abnormal behavior detection.

Xinhao Chen received the B.S. degree in information and computing science from Shangrao Normal University, China in 2018. He is currently pursuing the M.E. degree at Nanchang University, China. His research interests include the Internet of Things and computer version.