Neural Computing and Applications (2024) 36:7595–7607
https://doi.org/10.1007/s00521-024-09538-9

ORIGINAL ARTICLE

Parametric RSigELU: a new trainable activation function for deep learning

Serhat Kiliçarslan[1] · Mete Celik[2]

Received: 1 November 2022 / Accepted: 22 January 2024 / Published online: 28 February 2024

Abstract
Activation functions are used to extract meaningful relationships from real-world problems with the help of deep learning models. Thus, the development of activation functions which affect deep learning models' performances is of great interest to researchers. In the literature, mostly, nonlinear activation functions are preferred since linear activation functions limit the learning performances of the deep learning models. Non-linear activation functions can be classified as fixed-parameter and trainable activation functions based on whether the activation function parameter is fixed (i.e., user-given) or modified during the training process of deep learning models. The parameters of the fixed-parameter activation functions should be specified before the deep learning model training process. However, it takes too much time to determine appropriate function parameter values and can cause the slow convergence of the deep learning model. In contrast, trainable activation functions whose parameters are updated in each iteration of deep learning models training process achieve faster and better convergence by obtaining the most suitable parameter values for the datasets and deep learning architectures. This study proposes parametric RSigELU (P+RSigELU) trainable activation functions, such as P+RSigELU Single (P+RSigELUS) and P+RSigELU Double (P+RSigELUD), to improve the performance of fixed-parameter activation function of RSigELU. The performances of the proposed trainable activation functions were evaluated on the benchmark datasets of MNIST, CIFAR-10, and CIFAR-100 datasets. Results show that the proposed activation functions outperforms PReLU, PELU, ALISA, P+FELU, PSigmoid, and GELU activation functions found in the literature. To access the codes of the activation function; https://github.com/serhatklc/P-RsigELU-Activation-Function.

Keywords Deep learning · Parametric activation function (P+RSigELU) · MNIST · CIFAR-10 · CIFAR-100 · Trainable activation function

1 Introduction

Deep learning models are used in several application domains because of their outstanding achievements in object detection, classification, and prediction [2, 12, 13, 24, 25, 27]. Activation functions are regarded as one of the key functional components of deep learning architectures since they play an important role in determining whether the network will be active by forwarding inputs to the next network layer [14, 15, 22, 23, 25, 28, 30, 31, 34].

However, finding an efficient and suitable activation function for deep learning architectures and datasets is a challenging problem. In the literature, mostly, nonlinear activation functions are preferred since linear activation functions limit the learning performance of the deep learning models. Non-linear activation functions can be classified as fixed-parameter and trainable activation functions whether the activation function parameter is fixed (i.e., user-given) or modified during the training process of deep learning model. In the literature several fixed-parameter activation functions are proposed, such as ReLU, LReLU, ELU, RSigELU, GeLU, FeLU, PReLU, DReLU,

✉ Serhat Kiliçarslan
skilicarslan@bandirma.edu.tr

Mete Celik
mcelik@erciyes.edu.tr

[1] Department of Software Engineering, Bandırma Onyedi Eylül University, 10200 Bandırma, Balıkesir, Turkey

[2] Department of Computer Engineering, Faculty of Engineering, Erciyes University, 38039 Kayseri, Turkey

and PELU [18, 19, 25, 29, 36, 40–42]. However, the parameters of these activations functions are fixed or user-given and so their parameter values are not modified during the training process of the model which leads to the slow convergence of deep learning models and may not be suitable for the dataset. In addition, it takes too much time to determine appropriate function parameter values. Moreover, some of the fixed-parameter activation functions can not take derivative if the input value is negative, which causes the learning process to slow down.

To overcome the limitations of fixed-parameter activations functions, in the literature, several trainable activation functions are proposed, such as, ALISA, PELU, PRELU, Parametric Flatten-T Swish (PFTS), Mexican ReLU (MeLU), P+FELU, PSigmoid, and GELU [2, 8, 14, 19, 20, 35, 42, 44] (Bawa et al. [5]). In order to achieve optimal values during model training, the backpropagation technique in deep learning models is used to update the weights, bias values, and parameters of the trainable activation functions in each neuron. That is, in each iteration of the training process, the parameters of the trainable activation functions are updated to obtain the most appropriate weights for each neuron. In this way, trainable activation functions achieve faster and better convergence by obtaining suitable parameter values for the datasets and deep learning architectures.

In this study, parametric RSigELU (P+RSigELU) trainable activation functions, such as P+RSigELU Single (P+RSigELUS) and P+RSigELU Double (P+RSigELUD), are proposed to improve the performances of the learning process of deep learning models. P+RSigELUS function has single slope parameter to be trained and P+RSigELUD function has double slope parameters to be trained. These functions are extended versions of RSigELUS and RSigELUD fixed-parameter activation functions [25]. The proposed P+RSigELU trainable activation functions work, actively, on both positive and negative input values and their parameter values are updated in each iteration of the model's training process. They perform the learning process as soon as possible by determining suitable slope parameter values. In addition, thanks to the constant gradient parameter added to the negative and positive regions, the flexibility feature of the activation functions have been gained. Thus, it will continue the learning process by avoiding the errors that may occur from the trainable parameters obtained for each neuron. P+RSigELU trainable activation functions inherit merits of smooth activation functions (such as Sigmoid and Tanh) and piecewise activation functions (such as ReLU and its variants), and avoids their deficiencies. Main contributions of this study is listed as following.

1. Parametric RSigELU (P+RSigELU) trainable activation functions, such as P+RSigELUS and P+RSigELUD were proposed.
2. The performances of the proposed P+RSigELU activation functions have been evaluated on MNIST, CIFAR-10, and CIFAR-100 benchmark datasets using a convolutional neural network (CNN) model.
3. The results show that the proposed trainable activation functions outperform the PReLU, PELU, ALISA, P+FELU, PSigmoid, and GELU activation functions.
4. The proposed P+RSigELU activation functions works actively in negative and positive regions and can overcome the problem of vanishing gradient and negative region.
5. The proposed P-RSigELU activation function can adapt very well to the model with little risk of overfitting.

The organization of this article is as follows. The related work is presented in Sect. 2. The proposed activation functions are presented in Sect. 3. The methods and materials used in the experiments are presented in Sect. 4. Experimental results and discussion are presented in Sect. 5. Finally, the conclusion is presented in Sect. 6.

2 Related work

In the literature, several fixed-parameter and trainable activation functions have been developed to improve the performance of the training process of the deep learning models. This study proposes trainable activation functions.

In the literature several fixed-parameter activation functions are proposed. They can be listed as ReLU, LReLU, ELU, RSigELU, GeLU, FeLU, PReLU, DReLU, and PELU [18, 19, 25, 29, 36, 40–42]. ReLU activation function is proposed to overcome the vanishing gradient problem by returning negative values to the positive values [36]. LReLU activation function takes part in training with negative weights [34]. The ELU activation function has been proposed as an alternative to the ReLU activation function to address the issues of vanishing gradient and negative weights [9]. SELU activation function is proposed to enhance the training performance of ELU activation function [29]. The function parameters of fixed-parameter activation functions are given before the training process of the deep learning model which leads to the slow convergence of deep learning models. In addition, determining a suitable parameter values activation function requires numerous trials which is a time-comsuming process.

In the literature, several trainable activation functions are proposed, such as, ALISA, PELU, PRELU, Parametric Flatten-T Swish (PFTS), Mexican ReLU (MeLU),

P+FELU, PSigmoid, and GELU [2, 8, 14, 19, 20, 35, 42, 44] (Bawa et al. [5]). The ALISA activation function has been proposed to address the issues of vanishing gradient and negative weights (Bawa et al. [5]). The PELU activation function has been proposed by adding a trainable scala parameter to the ELU activation function to handle bias shift problem [42]. PReLU activation function has been proposed as an alternative to the ReLU activation function to address the issue of negative weights [19]. In addition, the PReLU activation function makes the parameter value defined as constant in the LReLU activation function trainable [19]. The Parametric Flatten-T Swish activation function has been proposed as an alternative to the ReLU activation function to address the issue of negative weights [8]. The P+FELU activation function has been proposed to address the issues of vanishing gradient and negative weights [2]. Parametric Sigmoid (PSigmoid) activation function proposed to improve the Squeeze and Excitation (SE) Networks block [44]. GELU activation function has been proposed as an alternative to the ReLU and ELU activation functions and find performance improvements across all considered computer vision, natural language processing, and speech tasks [20]. BLU activation function has been proposed as an alternative to the PReLU and PELU activation function [17]. TanhSoft activation function has been proposed with learnable parameters [6]. El Jaafari et al. [14] proposed a parametric rectified nonlinear unit (PRenu) for deep learning models. In contrast to Relu which returns the same received gradient for all positive values in its back-propagation, the PRenu multiplies it by values between $1 - \alpha$ and 1 depending on the value with which each neuron was involved. Trainable activation functions improves speed and convergence of the deep learning models by determining parameter values.

In the literature, the vanishing gradient problem is encountered in Sigmoid and Tanh activation functions. To overcome the vanishing gradient problem, it can be provided to the ReLU, derivatives, and RSigELU activation functions. Negative weights are ignored in the ReLU activation function. Behaviors of activation functions in the literature are shown in Fig. 1.

RSigELU fixed-parameter activation functions are, recently, proposed and outperformed exising fixed-parameter activation functions. This study extends RSigELU fixed-parameter activation functions and proposes their trainable versions. The proposed activation functions focuse on improving the inputs in both negative and positive regions and can overcome the problems of the negative region, bias shift, and vanishing gradient. The trainable activation function has been developed in order to obtain the most ideal values for each neuron during training. Thus, it is ensured that the training process is continuous by

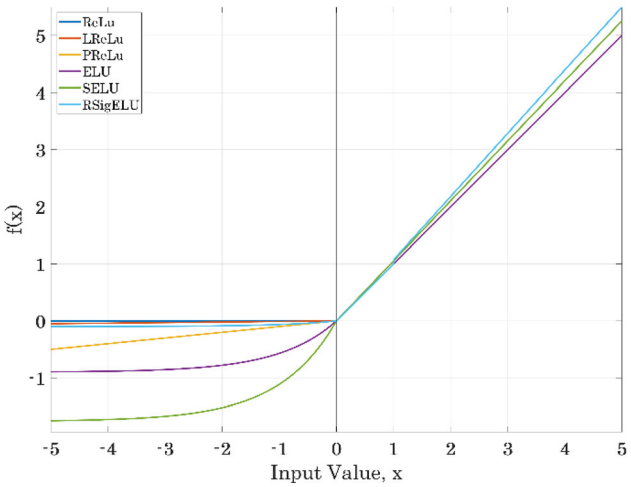

Fig. 1 Behaviors of activation functions

constantly updating the weights suitable for each neuron, thanks to the back-propagation algorithm. The proposed activation function works actively on both positive and negative input values. In addition, thanks to the constant gradient parameter added to the negative and positive regions, the flexibility feature of the activation function has been gained by preventing the errors that occur in the trainable parameters obtained for each neuron and continuing the learning process.

3 Proposed activation functions

In this study, parametric RSigELU (P+RSigELU) trainable activation functions, such as P+RSigELU Single (P+RSigELUS) and P+RSigELU Double (P+RSigELUD), are proposed by extending RSigELU activation functions [25]. The slope parameter values of RSigELU functions are fixed. Finding appropriate slope parameter values require numerous trials and so it is a time-consuming process. To overcome this problem, in this study, the RSigELU activation functions (such as RSigELUS an RSigELUD) are extended by adding trainable parameters to the functions. The proposed P+RSigELUS function has single trainable slope parameter and the proposed P+RSigELUD function has double trainable parameters. The proposed activation functions actively operate in linear, positive, and negative regions. The properties of negative and positive regions are best captured by functions with single and double trainable slope parameters. This represents the slope coefficients of the single and double parameters a^k and β^k [25].

The behaviors of the proposed P+RSigELUS and P+RSigELUD activation functions are shown in Fig. 2. In the proposed P+RSigELU activation functions, the

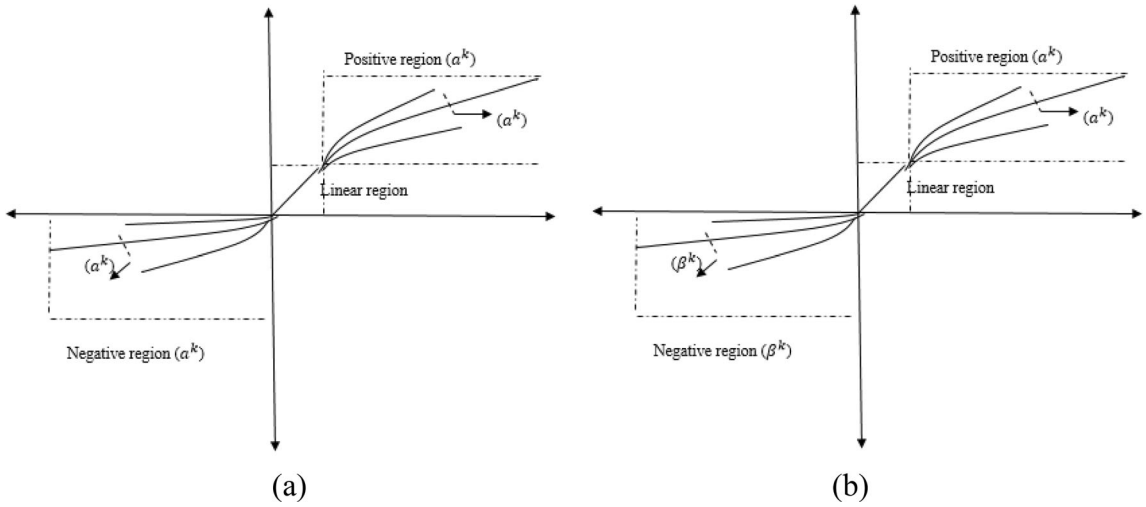

Fig. 2 Behaviors of the proposed activation functions **a** P+RSigELUS and **b** P+RSigELUD

positive and negative region slope parameters of a and β slope are trainable. The proposed P+RSigELU activation functions are a hybrid of ReLU [36] and sigmoid [16] activation functions for the positive region. They act as an ELU [9] activation function for the negative region and as a linear activation function for the linear region. The curve slope is consistent with the P+RSigELU, which ensures that the new activation functions do not change the accuracy advantage of the RSigELU activation functions. The trainable a^k and β^k parameters represents the gradient coefficients. In order to determine the gradient coefficient, it is determined according to the standard deviation value with GlorotNormal as the alpha–beta initializers [16]. Thanks to GlorotNormal, the proposed activation functions reach the global minimum quickly and efficiently. With the controlled start of the activation function proposed in the GlorotNormal process, deep learning architectures work faster and more efficiently [16].

3.1 Proposed P+RSigELUS trainable activation function

This section presents proposed P+RSigELUS trainable activation function. Its formulation.

is given in Eq. (1).

$$f(x) = \begin{cases} x^k * \left(\frac{1}{1+e^{-x^k}}\right) * a^k + x^k, & if \quad 1 < x < \infty \\ x^k, & if \quad 0 \le x \le 1 \\ a^k * \left(e^{x^k} - 1\right), & if \quad -\infty < x < 0 \end{cases} \tag{1}$$

The P+RSigELUS trainable function is active in positive, negative, and linear regions. In the Eq. (1), a^k represents the positive and negative region slope coefficient, x^k represents the input of the activation function, and $f(x)$ represents the output of the activation function. In addition, $k-th$ is number of channels. The alpha value in activation functions are determined the $k-th$ channel by the input signal x and output y. The slope parameter of the P+RSigELUS activation function is trained together with the model parameters during the training process of deep learning models. The proposed P+RSigELUS activation function behaves as shown in Fig. 2a.

The derivative of the proposed P+RSigELUS trainable activation function is given in the Eq. (2).

$$\frac{df(x)}{dx} = \begin{cases} \frac{kx^{k-1} * \left(\left(a^k+1\right) * e^{2x^k} + \left(a^k x^k + a^k + 2\right) * e^{x^k} + 1\right.}{\left(e^{x^k}+1\right)^2}, & if \quad 1 < x < \infty \\ k * x^{k-1}, & if \quad 0 \le x \le 1 \\ a^k * kx^{k-1} e^{x^k}, & if \quad -\infty < x < 0 \end{cases} \tag{2}$$

3.2 Proposed P+RSigELUD trainable activation function

This section presents proposed P+RSigELUD trainable activation function. Its formulation.

is given in Eq. (3).

$$f(x) = \begin{cases} x^k * \left(\frac{1}{1+e^{-x^k}}\right) * a^k + x^k, & if \quad 1 < x < \infty \\ x^k, & if \quad 0 \le x \le 1 \\ \beta^k * \left(e^{x^k} - 1\right), & if \quad -\infty < x < 0 \end{cases} \tag{3}$$

The P+RSigELUD trainable function is active in positive, negative and linear regions. In the Eq. (3),

a^krepresentst he positive region slope coefficient, β^k represent the negative region slope coefficient, x^k represents the input of the activation function, and $f(x)$ represents the output of the activation function. The the a^k and β^k slope parameters of the P+RSigELUD activation function are trained together with the model parameters during the training process of deep learning models. The proposed P+RSigELUD activation function behaves as shown in Fig. 2b.

The derivative of the proposed P+RSigELUD trainable activation function is given in the Eq. (4).

$$\frac{df(x)}{dx} = \begin{cases} \frac{kx^{k-1} * \left(\left(a^k+1\right) * e^{2x^k} + \left(a^k x^k + a^k + 2\right) * e^{x^k} + 1\right.}{\left(e^{x^k}+1\right)^2}, & if \;\; 1 < x < \infty \\ k * x^{k-1}, & if \;\; 0 \le x \le 1 \\ \beta^k * kx^{k-1} e^{x^k}, & if \;\; -\infty < x < 0 \end{cases} \tag{4}$$

In deep learning architectures, backpropagation architecture is used to update parameters during learning [27]. In addition, it is an important feature that the proposed activation functions can be derivatived. It is seen that the proposed activation function is active on both positive and negative regions after derivative. The additional parameter a^k and β^k are learned jointly with the whole model using classical gradient-based methods with backpropagation without weight decay to avoid pushing α to zero during the training. a^k and β^k are parameters learned during the network training. Also, weights are initialized by randomly sampling from a normal distribution with appropriate variance across all layers. Therefore, the value generation range and steepness of the function are not constant like logistic sigmoid and tangent hyperbolic, but vary. While creating the network architecture, architectures are generally created in 2 different ways with this type of free parameter functions. The first is the structures in which all neurons in the network except the last layer have their own training parameters, and the second is the structures in which neurons in each hidden layer have common fixed parameters. In this study, the first method was preferred.

Advantages of the proposed P+RSigELU activation functions, it updates the slope parameters of the activation function during training to obtain the optimal values for each neuron in each iteration. They also overcomes the problems of vanishing gradient and negative region to improve the learning process. In addition, proposed P+RSigELU activation functions provide higher accuracy and faster convergence than fixed parameter ones. In addition, the proposed activation function include parametric, monotonic, and bounded features.

4 Materials and method

To evaluate the performances of the proposed P+RSigELU activation functions, several benchmark datasets and VGG-based CNN architecture have been used. In this section, first, the benchmark datasets are introduced and then VGG-based CNN architecture is presented. In the study, experimental evaluations were carried out on the proposed P+RSigELU activation functions with VGG-CNN consisting of determined architectures and parameters in the study conducted by Kiliçarslan [22, 23, 25, 39, 43].

4.1 Benchmark datasets

In this study, experimental evaluations were performed on MNIST, CIFAR-10, and CIFAR-100 benchmark datasets [25, 32, 33].

The MNIST dataset consists of 70,000 hand written digits in total [33]. It consists of 10 classes, and each image is 28 × 28 pixels in size. In this study, 60,000 of the images contained in the dataset were used for training and 10,000 of the images were used for testing.

The CIFAR-10 dataset consists of 60,000 32 × 32 colour images in 10 classes, with 6000 **images per class. There are 50,000 training images and 10,000 test images [32].

The CIFAR-100 dataset is a subset of the Tiny Images dataset and consists of 60,000 32 × 32 color images. The 100 classes in the CIFAR-100 are grouped into 20 superclasses [32].

4.2 Convolutional neural network

Convolutional neural network (CNN) is widely preferred in many fields such as signal processing, object identification, disease detection, classification [7, 21, 27, 37]. Although it has been used successfully in many fields, it gives better results in image processing [1, 10]. CNN, developed by LeCun inspired by the visual center of animals [33]. CNN architectures consist of convolution, pooling, activation, dropuout, flatten, fully-connected, and softmax layers. The main purpose of CNN is to do more efficient learning by layer-by-layer applying different operations to the input data [25]. In this study, experimental evaluations of the proposed P-RSigELU activation functions were made in VGG-based CNN architecture. The VGG architecture used in the study is shown in Fig. 3 [25].

In Fig. 3, the VGG-based CNN architecture used in the experiments consists of four convolution blocks, a flatten layer, two fully-connected layers, and a softmax layer. To evaluate the proposed P-RSigELU activation functions, the MNIST, CIFAR-10 and CIFAR-100 datasets are first

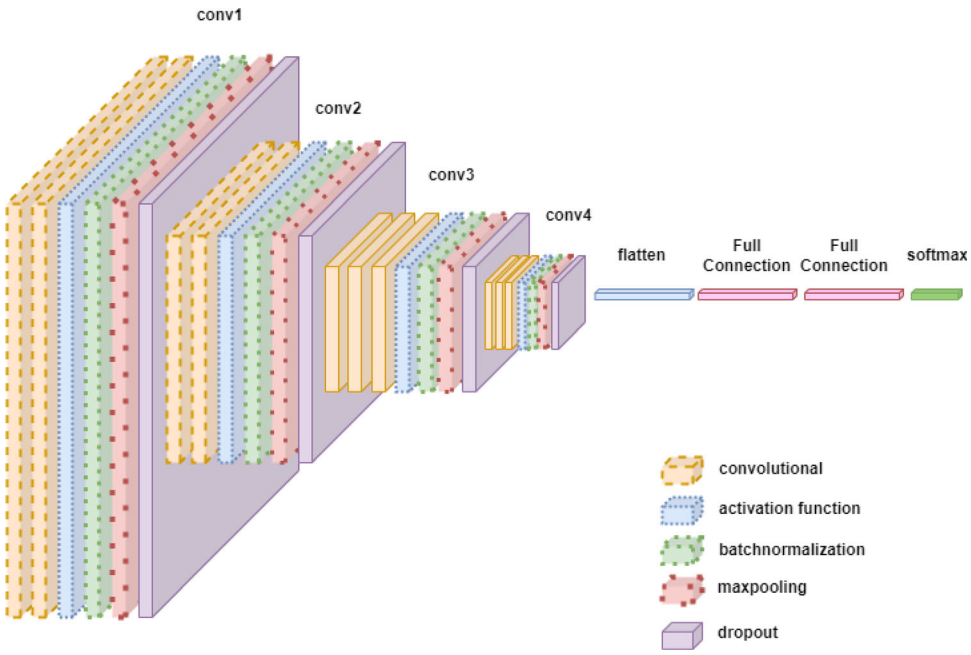

Fig. 3 VGG-based CNN architecture for evaluating the proposed activation functions

applied to the convolution operation in the VGG-based CNN architecture. The filters number, filter size, and steps number parameters in the convolution process are among the significant parameters chosen by the designers. The number of filters is represented by an odd number along with the filter size. The filter size gives information about how much the determined filter should move on the data in each step. In the convolution process, by circulating the 3×3 filter determined on the dataset are provided the discovery of features of data and obtaining the feature map matrix. After the convolution process, the activation functions used in the experimental evaluations are included in the obtained feature map matrix. In this study, PReLU, PELU, ALISA, GELU, P+FELU, PSigmoid, P+RSigELUS, and P+RSigELUD trainable activation functions have been used. In the VGG-based CNN architecture, after activation function block, the normalization method is utilized to adjust for the erratic distribution in the previously produced feature map matrix [11, 25, 26, 38]. Then, the maximum pooling method is applied to the feature map obtained with the values of pool = 2 and stride = 2. By halving the size of the feature map during the maximum pooling process, the model performs better and avoids delivering inaccurate rote results. In order to prevent overlearning after the pooling process, the dropout layer was applied as 0.25. The flatten layer is applied after the convolution blocks to obtain the feature vector suitable for the fully-connected layer. The fully-connected layer architecture is similar to the artificial neural network (ANN) architecture. Therefore, the obtained feature map matrix is performed classification by transforming into vector format. After these processes, a fully connected layer with 512 neurons is applied. Thanks to the fully connected layer, all units are connected to each other and the classification process is performed. Finally, the logistic regression-based softmax layer is used as the output function [3, 4, 12, 13].

The VGG-based CNN architecture used in the study consists of four CNN blocks in total. The first CNN block consists of two $32@3 \times 3$ filters, the second CNN block consists of two $48@3 \times 3$ filters, the third CNN block consists of three $64@3 \times 3$, and the last CNN block consists of $96@3 \times 3$ filters. Following the CNN blocks, experiments are carried out on the VGG architecture by connecting two 512-neuron fully-connected layers.

5 Results and discussion

In this section, first, experimental evaluations of the proposed P+RSigELUS and P+RSigELUD trainable activation functions on MNIST, CIFAR-10, and CIFAR-100 datasets and then discussion were presented.

5.1 Experimental evaluation

In the experiments, for comparison, PReLU, PELU, GELU, P+FELU, PSigmoid and ALISA activation functions were used. Experimental evaluations of the proposed and other activation functions on the MNIST dataset are presented in Tables 1, 2, and 3. Tables shows number of epochs, training loss (Train_Loss), training accuracy (Train_Acc.), validation loss (Val._Loss), and validation accuracy (Val._Acc.) for number of repeats or diffrent activation functions. In the experimental evaluations,

Table 1 Experimental results of the proposed P+RSigELUS for MNIST dataset

Repeat	Epoch	Train_Loss	Train_Acc	Val._Loss	Val._Acc
Repeat 1	40	0.2864	0.9193	0.2160	0.9380
Repeat 2	40	0.1861	0.9429	0.1562	0.9492
Repeat 3	40	0.3235	0.9161	0.2086	0.9444
Repeat 4	40	0.2309	0.9305	0.1745	0.9483
Repeat 5	40	0.2239	0.9347	0.1475	0.9558
Average		0.2501	0.9287	0.1805	0.9471

Table 2 Experimental results of the proposed P+RSigELUD for MNIST dataset

Repeat	Epoch	Train_Loss	Train_Acc	Val._Loss	Val._Acc
Repeat 1	40	0.2557	0.9246	0.1923	0.9453
Repeat 2	40	0.2059	0.9335	0.1523	0.9517
Repeat 3	40	0.1594	0.9546	0.1038	0.9674
Repeat 4	40	0.1778	0.9459	0.1176	0.9658
Repeat 5	40	0.2325	0.9314	0.1633	0.9518
Average		0.2062	0.9380	0.1458	0.9564

datasets were separated as training and testing as stated in Sect. 4.1. Computationally, the training loss is calculated by taking the sum of errors for each example in the training set. The accuracy score in machine learning is a measurement statistic that compares the proportion of accurate predictions made by a model to all predictions made. We determine it by dividing the total number of forecasts by the number of correct guesses. In deep learning, metrics like accuracy, and loss score are frequently used to assess the performance of models. Accuracy: This is the ratio of total forecasts made to total predictions made correctly. The training loss is a metric used to assess how a deep learning model fits the training data. In the loss metric (in Eq. (6)), the m parameter returns the number of training samples, the i parameter returns the training example in a dataset, and the y_i parameter returns the actual value for the ith training sample. It's a straightforward statistic that provides a general sense of how well a model is doing. Each metric has a mathematical formula that is given in the following equations:

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Number of total predictions}} \tag{5}$$

$$loss = -\frac{1}{m}\sum_{i=1}^{m} y_i \log(\widehat{y}_i) \tag{6}$$

Tables 1, 2, and 3 present, the experimental results performed by repeating 5 times on the MNIST dataset. In the experimental results, it can be seen that the best performance value was obtained by the double parameter P+RSigELUD trainable activation function. In experimental evaluations, it was observed that an average of 0.9564 validation accuracy and 0.1458 validation loss values were obtained with P+RSigELUD. In addition to, for the single parameter P+RSigELUS trainable activation function, an average of 0.9471 validation accuracy and 0.1805 validation loss values were obtained. Also, it was observed that the lowest success was achieved with an average of 0.9335 validation accuracy and 0.2275 validation loss by ALISA activation function. In addition, the proposed activation function seems to work faster than other activation functions during the training process.

Figure 4 shows the average validation accuracy values (Fig. 4a) and validation loss values (Fig. 4b) of the activation function results for the MNIST dataset. The proposed P+RSigELUS and P+RSigELUD trainable activation functions gave the best average performance values with respect to the other activation functions. Also, as can be seen in Fig. 4a, the average val_accuracy value of the proposed P+RSigELUD activation function is bigger than that of PReLU, PELU, ALISA, GELU, P+FELU, PSigmoid and P+RSigELUS activation funcitons. In addition, the average accuracy performances are between 0.9380 and 0.9558 for P+RSigELUS and 0.9453 and 0.9674 for P+RSigELUD. The average loss performances

Table 3 The average success rates of activation functions for MNIST dataset

Activation	Train_Loss	Train_Acc	Val._Loss	Val._Acc	Time (ms)
PReLU	0.2782	0.9139	0.2011	0.9383	1562
PELU	0.2721	0.9187	0.2134	0.9362	1785
ALISA	0.3018	0.9095	0.2275	0.9335	1465
GELU	0.3726	0.8933	0.2203	0.9244	1864
P+FELU	0.3255	0.8941	0.2127	0.9109	1695
PSigmoid	0.2593	0.9009	0.2207	0.9214	2123
Proposed P+RSigELUS	0.2501	0.9287	0.1805	0.9471	1432
Proposed P+RSigELUD	0.2062	0.9380	0.1458	0.9564	1578

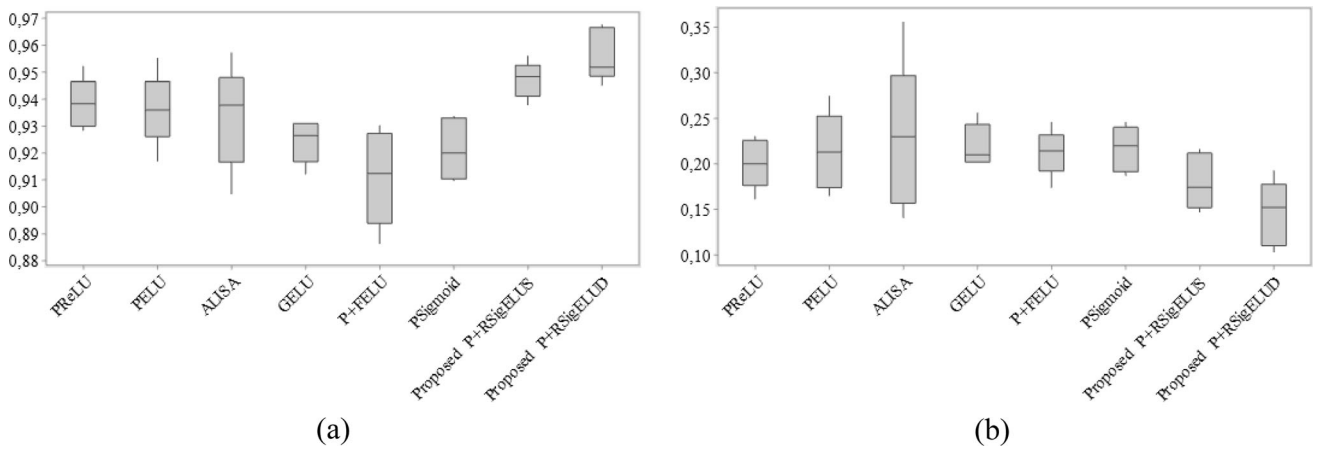

Fig. 4 **a** Average validation accuracy values and **b** average validation loss values of activation function results for MNIST dataset

Table 4 Experimental results of the proposed P+RSigELUS for CIFAR-10 dataset

Repeat	Epoch	Train_Loss	Train_Acc	Val._Loss	Val._Acc
Repeat 1	40	0.5558	0.8548	0.6973	0.8209
Repeat 2	40	0.5317	0.8519	0.6553	0.8165
Repeat 3	40	0.5402	0.8561	0.6835	0.8197
Repeat 4	40	0.5605	0.8509	0.6772	0.8092
Repeat 5	40	0.5541	0.8519	0.7062	0.8056
Average		**0.5484**	**0.8532**	**0.6839**	**0.8143**

Table 5 Experimental results of the proposed P+RSigELUD for CIFAR-10 dataset

Repeat	Epochs	Train_Loss	Train_Acc	Val._Loss	Val._Acc
Repeat 1	40	0.6606	0.8475	0.6999	0.8230
Repeat 2	40	0.5301	0.8510	0.6374	0.8314
Repeat 3	40	0.5451	0.8590	0.6693	0.8246
Repeat 4	40	0.5227	0.8514	0.6702	0.8136
Repeat 5	40	0.5220	0.8401	0.6314	0.8136
Average		**0.5561**	**0.8498**	**0.6616**	**0.8212**

are between 0.1475 and 0.2160 for P+RSigELUS and 0.1038 and 0.1923 for P+RSigELUD. In Fig. 4a, the lowest values of box plot of P+RSigELUS and P+RSigELUD are above that of the rest of the activation functions. Thus, it is observed that the P+RSigELUD activation function gives better results than the others. In Fig. 4, the loss values of the P+RSigELUD activation function are among the lowest values in the box plot compared to other box plots. In this case, when the box plots found in Fig. 4a, b are examined, it can be seen that the activation function of P+RSigELUD gives more consistent results than the other activation functions. Also, the lengths of the whiskers to the box plot are close and the median value is close to the middle of the box plot.

Experimental evaluations of the proposed and other activation functions on the CIFAR-10 dataset are presented in Tables 4, 5, and 6.

Tables 4, 5, and 6 present the experimental results performed on the CIFAR-10 dataset. In the experimental results, it can be seen that the best performance value was obtained by the double parameter P+RSigELUD trainable activation function. In experimental evaluations, it was observed that an average of 0.8212 validation accuracy and 0.6616 validation loss values were obtained with P+RSigELUD. In addition to, for the single parameter P+RSigELUS trainable activation function, an average of 0.8143 validation accuracy and 0.6839 validation loss values were obtained. Also, it was observed that the lowest success was achieved with an average of 0.7762 validation accuracy and 0.8135 validation loss by ALISA activation function. In addition, the proposed activation function seems to work faster than other activation functions during the training process.

Figure 5 shows the average validation accuracy values (Fig. 5a) and validation loss values (Fig. 5b) of the activation function results for the CIFAR-10 dataset. The proposed P+RSigELUS and P+RSigELUD trainable activation functions give the best validation accuracy and validation loss performances from other activation functions. Also, the validation accuracy values are are between 0.8056 and 0.8209 for P+RSigELUS and between 0.8136 and 0.8314 for P+RSigELUD. In addition, the validation loss values are between 0.6553 and 0.7062 for P+RSigELUS and 0.6314 and 0.6999 for P+RSigELUD. The highest values of boxes of P+RSigELUD are above that the of rest of the activation functions. In Fig. 5b, the lowest loss values of the activation functions other than ALISA and P+FELU activation functions are close to each other.

Table 6 The average success rates of activation functions for CIFAR-10 dataset

Activation Function	Train_Loss	Train_Acc	Val._Loss	Val._Acc	Time (ms)
PReLU	0.5175	0.8624	0.6815	0.8200	2433
PELU	0.5440	0.8544	0.6814	0.8199	2651
ALISA	0.6093	0.8298	0.8135	0.7762	2502
GELU	0.5460	0.8567	0.6986	0.8132	2713
P+FELU	0.5493	0.8439	0.6957	0.8160	2549
PSigmoid	0.6178	0.8520	0.7789	0.8135	3120
Proposed P+RSigELUS	0.5484	0.8532	0.6839	0.8143	2345
Proposed P+RSigELUD	**0.5561**	**0.8498**	**0.6616**	**0.8212**	**2407**

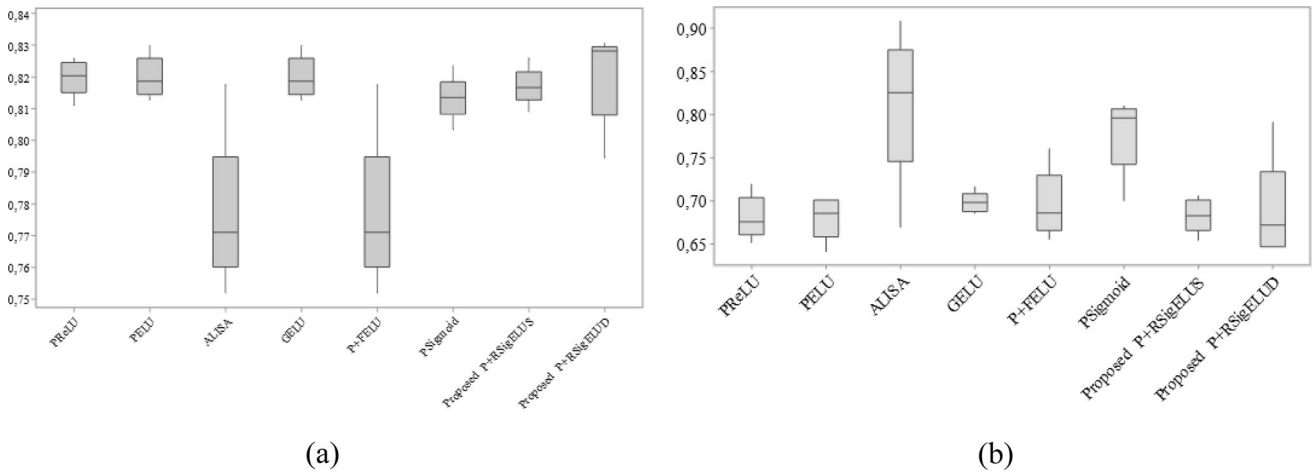

Fig. 5 **a** Average validation accuracy values and **b** average validation loss values of activation function results for CIFAR-10 dataset

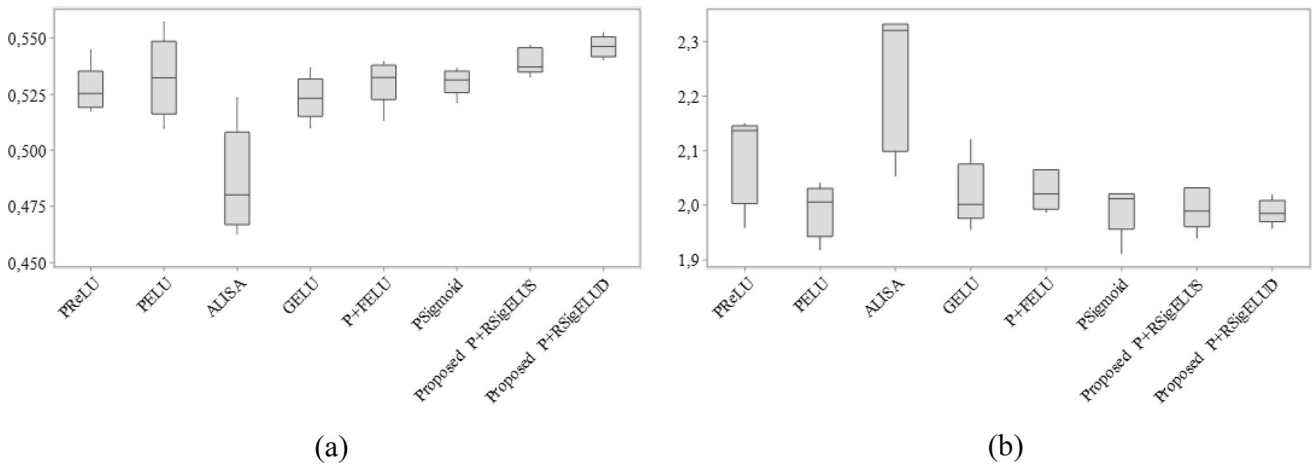

Fig. 6 **a** Average validation accuracy values and **b** average validation loss values of activation function results for CIFAR-100 dataset

In Fig. 5a, b, it can be seen that the ALISA and P+FELU activation functions have poor success compared to other activation functions and there is an inconsistency between the results. In addition, it can be seen that the proposed P+RSigELUS activation function gives more consistent results with a smaller box size than the others. Also, the lengths of the whiskers to the box plot are close and the median value is close to the middle of the box plot. When Figs. 5, 6 are examined, it is observed that the activation functions of PReLU and PELU behave like the proposed functions. However, when the results of ALISA and P+FELU activation functions are compared, too much fluctuation is observed. The reason for this is that there are too many fluctuations and inconsistencies because not most

Table 7 Experimental results of the proposed P+RSigELUS for CIFAR-100 dataset

Repeat	Epoch	Train_Loss	Train_Acc	Val._Loss	Val._Acc
Repeat 1	40	1.3885	0.6531	1.9894	0.5372
Repeat 2	40	1.4114	0.6458	1.9413	0.5467
Repeat 3	40	1.3929	0.6540	1.9804	0.5448
Repeat 4	40	1.3813	0.6563	2.0319	0.5329
Repeat 5	40	1.3729	0.6548	2.0330	0.5368
Average		1.3894	0.6528	1.9952	0.5396

Table 8 Experimental results of the proposed P+RSigELUD for CIFAR-100 dataset

Repeat	Epoch	Train_Loss	Train_Acc	Val._Loss	Val._Acc
Repeat 1	40	1.3268	0.6685	1.9586	0.5524
Repeat 2	40	1.3047	0.6732	1.9993	0.5431
Repeat 3	40	1.3299	0.6647	1.9818	0.5463
Repeat 4	40	1.2988	0.6760	2.0193	0.5404
Repeat 5	40	1.3188	0.6689	1.9852	0.5490
Average		1.3158	0.6702	1.9888	0.5462

activation functions works in accordance with both the deep learning architecture and the dataset.Experimental evaluations of the proposed and other activation functions on the CIFAR-100 dataset are presented in Tables 7, 8, and 9.

Tables 7, 8, and 9 present the experimental results performed on the CIFAR-100 dataset. In the experimental results, it can be seen that the best performance value was obtained by the double parameter P+RSigELUD trainable activation function. The average performance results of the P+RSigELUD activation function were obtained of 0.5462 validation accuracy and 1.9888 validation loss. In addition, for the single parameter P+RSigELUS, an average of 0.5396 validation accuracy and 1.9952 validation loss values were obtained. Also, it was observed that the lowest success was achieved with an average of 0.4861 validation accuracy and 2.2368 validation loss by ALISA activation function. In addition, the proposed activation function seems to work faster than other activation functions during the training process.

Figure 6 shows the average validation accuracy values (Fig. 6a) and validation loss values (Fig. 6b) of the activation function results for the CIFAR-100 dataset. The proposed P+RSigELUS and P+RSigELUD trainable activation functions gave the best average validation accuracy and validation loss performances. The validation accuracy values are between 0.5329 and 0.5467 for P+RSigELUS and between 0.5404 and 0.5524 for P+RSigELUD. In addition, the validation loss values are between 1.9413 and 2.0330 for P+RSigELUS and between 1.9586 and 2.0193 for P+RSigELUD. In Fig. 6a, the lowest box plot values of P+RSigELUS and P+RSigELUD are above the that of ALISA and PELU activation functions and close to that of the rest of the activation functions. Also, as can be seen in Fig. 6a, b, the box sizes of the suggested activation functions are smaller than that of the others. Thus, it is observed that the proposed activation functions offer more successful and consistent results than the others. In addition, the lengths of the whiskers to the box plot are close and the median value is close to the middle of the box plot. When Table 6 and 9 are examined, it is observed validation accuracy of the proposed model is better than the other models and so during the evaluation process, interpretation was made by taking into account the validation values.

5.2 Discussion

In this study, the P-RSigELU trainable activation functions are proposed to improve the learning process of deep learning models. Trainable activation functions update the slope parameters of the activation function to obtain the most appropriate values for each neuron during training. Thus, the weights suitable for each neuron are constantly

Table 9 The average success rates of activation functions for CIFAR-100 dataset

Activation Function	Train_Loss	Train_Acc	Val._Loss	Val._Acc	Time (ms)
PReLU	1.3127	0.6717	2.0874	0.5269	2845
PELU	1.4471	0.6341	1.9911	0.5325	3012
ALISA	1.5408	0.6130	2.2368	0.4861	3145
GELU	1.3399	0.6447	2.0215	0.5234	3245
P+FELU	1.3102	0.6763	2.0277	0.5308	3014
PSigmoid	1.4256	0.6574	1.9935	0.5307	3180
Proposed P+RSigELUS	1.3894	0.6528	1.9952	0.5396	2799
Proposed P+RSigELUD	1.3158	0.6702	1.9888	0.5462	3103

updated by the back propagation algorithm, ensuring that the training process is continuous. The proposed P+RSigELUS and P+RSigELUD trainable activation functions gave the best results in experimental results for MNIST, CIFAR10, and CIFAR-100 datasets as shown in Table 3, 6, and 9, respectively, when compared to the existing trainable activation functions. In this study, the proposed P+RSigELUS and P+RSigELUD trainable activation functions are effective in the positive, negative, and linear activation regions. Also, they can overcome the problems of bias shift, negative region, and vanishing gradient successfully. The curve slope is consistent with the PSigELU, which ensures that the new activation function does not change the accuracy advantage of the RSigELU. In addition, the lengths of the whiskers to the box plot are close and the median value is close to the middle of the box plot. The lower accuracy value of box plots of the proposed P+RSigELU is close to the box than that of the other activation function and upper accuracy values of the proposed P+RSigELU are close to the highest. The proposed P+RSigELU trainable activation functions provide higher accuracy and faster convergence than fixed-parameter activation functions. In addition, when the proposed activation function is compared with the P+RSigELU activation functions, it is seen that it does not complete the training process in a shorter time in all three data sets. All deep learning architectures has different characteristics and therefore like the activation functions proposed in the literature, P+RSigELU may not have consistent result values for all architectures. In addition, the proposed activation function include parametric, monotonic, and bounded features. However, it does not have the smooth feature. In addition, it is observed that the proposed P+RSigELU activation functions gives consistent and stable results on the datasets in the experiments. In addition, it should be noted that the activation functions in the literature do not always give consistent results in all data sets.

6 Conclusion and future work

Activation functions are used to extract meaningful relationships from real world data with the help of deep neural network architectures. For this reason, the development of activation functions to positively affect the performance of deep neural networks is of great interest to researchers. In the study, the P-RSigELU trainable activation functions are proposed to improve the learning process deep learning models. In trainable activation functions, the slope parameters of the activation function are updated in each training iteration of the model in order to obtain the most appropriate values for each neuron of the model. The parameters of the proposed P-RSigELU activation functions (such as P-RSigELUS and P-RSigELUd) are trained in the training process of the deep learning models to best fit to the dataset and the models used. The proposed activation functions include parametric, monotonic, and bounded features. In general, the trainable activation functions show better convergence as it can adapt the datasets faster by learning the parameter from the data. P+RSigELUS and P+RSigELUD trainable activation functions proposed, in this study, can successfully overcome bias shift, negative region, and vanishing gradient problems. The experimental evaluations conducted on the benchmark datasets of MNIST, CIFAR-10, and CIFAR-100 show that the proposed trainable activation functions outperform existing activation functions and they give better results than PReLU, PELU, GELU, P+FELU, PSigmoid, and ALISA activation functions. The proposed P+RSigELUD trainable activation function presents the best validation accuracy values and they are 0.9564, 0.8212 and 0.5462 for the MNIST, CIFAR-10 and CIFAR-100 datasets, respectively.

In the future work, we plan to use intelligent optimization methods [27] to obtain hyperparameters of deep learning algorithms and to apply proposed activation functions on other network structures using different datasets. To accomplish comparable outcomes, a variety of alternative activation functions can be used. There is still considerable research to be done to increase the adaptability of neural networks and their hyper-parameters in terms of necessary output.

Funding Open access funding provided by the Scientific and Technological Research Council of Türkiye (TÜBİTAK).

Data availability The datasets generated during and/or analysed during the current study are available in the [keras.io] repository, [https://keras.io/api/datasets/].

Declarations

Conflict of 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.

References

1. Adem K, Kiliçarslan S (2019) Performance analysis of optimization algorithms on stacked autoencoder. In: 2019 3rd international symposium on multidisciplinary studies and innovative technologies (ISMSIT) (pp. 1–4). IEEE
2. Adem K (2022) P+FELU: flexible and trainable fast exponential linear unit for deep learning architectures. Neural Comput Appl 34(24):1–12
3. Adem K, Közkurt C (2019) Defect detection of seals in multilayer aseptic packages using deep learning. Turk J Electr Eng Comput Sci 27(6):4220–4230
4. Adem K, Kiliçarslan S, Cömert O (2019) Classification and diagnosis of cervical cancer with stacked autoencoder and softmax classification. Expert Syst Appl 115:557–564
5. Bawa VS, Kumar V (2019) Linearized sigmoidal activation: A novel activation function with tractable non-linear characteristics to boost representation capability Expert Systems with Applications 120346-356 https://doi.org/10.1016/j.eswa.2018.11.042
6. Biswas K, Kumar S, Banerjee S, Pandey AK (2021) TanhSoft—dynamic trainable activation functions for faster learning and better performance. IEEE Access 9:120613–120623
7. Bülbül MA (2023) Kuru Fasulye Tohumlarının Çok Sınıflı Sınıflandırılması İçin Hibrit Bir Yaklaşım. J Inst Sci Technol 13(1):33–43
8. Chieng HH, Wahid N, Ong P (2020) Parametric flatten-T swish: an adaptive non-linear activation function for deep learning. arXiv preprint arXiv:2011.03155
9. Clevert DA, Unterthiner T, Hochreiter S (2015) Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289
10. Çiğdem ACI, ÇIRAK, A. (2019) Türkçe Haber Metinlerinin Konvolüsyonel Sinir Ağları ve Word2Vec Kullanılarak Sınıflandırılması. Bilişim Teknolojileri Dergisi 12(3):219–228
11. Deng L, Yu D (2014) Deep learning: methods and applications. Found trends® in signal process 7(3–4):197–387
12. Diker A (2022) An efficient model of residual based convolutional neural network with Bayesian optimization for the classification of malarial cell images. Comput Biol Med 148:105635
13. Dönmez E (2022) Enhancing classification capacity of CNN models with deep feature selection and fusion: a case study on maize seed classification. Data Knowl Eng 141:102075
14. El Jaafari I, Ellahyani A, Charfi S (2021) Parametric rectified nonlinear unit (PRenu) for convolution neural networks. SIViP 15(2):241–246
15. Elen A, Baş S, Közkurt C (2022) An adaptive Gaussian kernel for support vector machine. Arab J Sci Eng 47(8):10579–10588
16. Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In proceedings of the thirteenth international conference on artificial intelligence and statistics (pp. 249–256). JMLR workshop and conference proceedings
17. Godfrey LB (2019) An evaluation of parametric activation functions for deep learning. In: IEEE international conference on systems, man and cybernetics (SMC). 3006–3011
18. Godin F, Degrave J, Dambre J, De Neve W (2018) Dual rectified linear units (DReLUs): a replacement for tanh activation functions in quasi-recurrent neural networks. Pattern Recogn Lett 116:8–14
19. He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In proceedings of the IEEE international conference on computer vision (pp. 1026–1034)
20. Hendrycks D, Gimpel K (2016). Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415
21. Işık E, Ademović N, Harirchian E, Avcil F, Büyüksaraç A, Hadzima-Nyarko M, Antep B (2023) Determination of natural fundamental period of minarets by using artificial neural network and assess the impact of different materials on their seismic vulnerability. Appl Sci 13(2):2076–3417
22. Kiliçarslan S (2023) A novel nonlinear hybrid HardSReLUE activation function in transfer learning architectures for hemorrhage classification. Multimed Tools Appl 82(4):6345–6365
23. Kiliçarslan S (2023) PSO + GWO: a hybrid particle swarm optimization and grey wolf optimization based algorithm for fine-tuning hyper-parameters of convolutional neural networks for cardiovascular disease detection. J Ambient Intell Humaniz Comput 14(1):87–97
24. Kilicarslan S, Celik M, Sahin Ş (2021) Hybrid models based on genetic algorithm and deep learning algorithms for nutritional anemia disease classification. Biomed Signal Proc Control 63:102231
25. Kiliçarslan S, Celik M (2021) RSigELU: a nonlinear activation function for deep neural networks. Expert Syst Appl 174:114805
26. Kiliçarslan S, Adem K, Celik M (2020) Diagnosis and classification of cancer using hybrid model based on relieff and convolutional neural network. Med Hypothese 137:109577
27. Kiliçarslan S, Adem K, Çelik M (2021) An overview of the activation functions used in deep learning algorithms. J New Result Sci 10(3):75–88
28. Kiliçarslan S, Közkurt C, Baş S, Elen A (2023) Detection and classification of pneumonia using novel superior exponential (supex) activation function in convolutional neural networks. Expert Syst Appl 217:119503
29. Klambauer G, Unterthiner T, Mayr A, Hochreiter S (2017) Self-normalizing neural networks. In Adv neural inf process syst 30:971–980
30. Közkurt C, Kiliçarslan S, Baş S, Elen A (2023) α SechSig and αTanhSig: two novel non-monotonic activation functions. Soft Comput 27(24):1–17
31. Krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. In: adv neural inf process syst 25:1106–1114
32. Krizhevsky A, Hinton G. (2009) Learning multiple layers of features from tiny images. Master's thesis, University of Tront.
33. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
34. Maas AL, Hannun AY, Ng AY (2013) Rectifier nonlinearities improve neural network acoustic models. In Proc Icml 30(1):3
35. Maguolo G, Nanni L, Ghidoni S (2021) Ensemble of convolutional neural networks trained with different activation functions. Expert Syst Appl 166:114048
36. Nair V, Hinton GE (2010). Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th international conference on machine learning (ICML-10) (pp. 807–814)
37. Pacal I (2022) Deep learning approaches for classification of breast cancer in ultrasound (US) images. J Inst Sci Technol 12(4):1917–1927
38. Pacal I, Karaboga D (2021) A robust real-time deep learning based automatic polyp detection system. Comput Biol Med 134:104519

39. Pacal I, Kılıcarslan S (2023) Deep learning-based approaches for robust classification of cervical cancer. Neural Comput Appl 35(25):18813–18828
40. Qiumei Z, Dan T, Fenghua W (2019) Improved convolutional neural network based on fast exponentially linear unit activation function. IEEE Access 7:151359–151367
41. Ramachandran P, Zoph B, Le QV (2017). Searching for activation functions. arXiv preprint arXiv:1710.05941
42. Trottier L, Gigu P, Chaib-draa B (2017). Parametric exponential linear unit for deep convolutional neural networks. In: 16th IEEE international conference on machine learning and applications (ICMLA) (pp. 207–214). IEEE
43. Yılmaz EK, Adem K, Kılıçarslan S, Aydın HA (2023) Classification of lemon quality using hybrid model based on stacked autoencoder and convolutional neural network. Eur Food Res Technol 249:1655–1667
44. Ying Y, Zhang N, Shan P, Miao L, Sun P, Peng S (2021) PSigmoid: improving squeeze-and-excitation block with parametric sigmoid. Appl Intell 51(10):7427–7439