Received March 28, 2020, accepted April 7, 2020, date of publication April 14, 2020, date of current version April 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2987829

RMAF: Relu-Memristor-Like Activation Function for Deep Learning

YONGBIN YU[1], KWABENA ADU[1], NYIMA TASHI[2], PATRICK ANOKYE[1], XIANGXIANG WANG[1], (Graduate Student Member, IEEE), AND MIGHTY ABRA AYIDZOE[1]

[1]School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China
[2]School of Information Science and Technology, Tibet University, Lhasa 850000, China

Corresponding author: Kwabena Adu (adukwabena4u@gmail.com)

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61550110248, in part by the Research on Sino-Tibetan Multi-Source Information Acquisition, Fusion, Data Mining and Its Application under Grant H04W170186, and in part by the Sichuan Science and Technology Program under Grant 2019YFG0190.

ABSTRACT Activation functions facilitate deep neural networks by introducing non-linearity to the learning process. The non-linearity feature gives the neural network the ability to learn complex patterns. Recently, the most widely used activation function is the Rectified Linear Unit (ReLU). Though, other various existing activation including hand-designed alternatives to ReLU have been proposed. However, none has succeeded in replacing ReLU due to their existing inconsistencies. In this work, activation function called ReLU-Memristor-like Activation Function (RMAF) is proposed to leverage benefits of negative values in neural networks. RMAF introduces a constant parameter (α) and a threshold parameter (p) making the function smooth, non-monotonous, and introduces non-linearity in the network. Our experiments show that, the RMAF works better than ReLU and other activation functions on deeper models and across number of challenging datasets. Firstly, experiments are performed by training and classifying on multi-layer perceptron (MLP) over benchmark data such as the Wisconsin breast cancer, MNIST, Iris and Car evaluation. RMAF achieves high performance of 98.74%, 99.67%, 98.81% and 99.42% respectively, compared to Sigmoid, Tanh and ReLU. Secondly, experiments were performed on convolution neural network (ResNet) over MNIST, CIFAR-10 and CIFAR-100 data and observed the proposed activation function achieves higher performance accuracy of 99.73%, 98.77% and 79.82% respectively than Tanh, ReLU and Swish. Additionally, we experimented our work on deep networks i.e. squeeze network (SqueezeNet), Dense connected neural network (DenseNet121) and ImageNet dataset, which RMAF produced the best performance. We note that, the RMAF converges faster than the other functions and can replace ReLU in any neural network due to the efficiency, scalability and its similarity to both ReLU and Swish.

INDEX TERMS Activation function, deep learning, memristive window function, muilti-layer perceptron, RMAF.

ACRONYMS

ABReLU	Average Biased Rectifier Linear Unit
CIFAR	Canadian Institute for Advanced Research
CNN	Convolutional Neural Network
CVNN	Complex-Valued Neural Network
DBN	Deep Belief Network
DenseNet	Densely Connected Convolutional Networks
DL	Deep Learning
DFUNet	Diabetic Foot Ulcer Network
DNN	Deep Neural Network
ELU	Exponential Linear Unit
FC	Fully Connected
FNA	Fine Needle Aspiration
GELU	Gaussian Error Linear Unit
ID	Magnetic Resonance Images
Isigmoid	Logistic Sigmoid
LVCSR	Large Vocabulary Continuous Speech Recognition
MFCN	Memristive Fully Convolutional Network
MLP	Multi-Layer Perceptron
MNIST	Modified National Institute of Standards and Technology
NN	Neural Network

The associate editor coordinating the review of this manuscript and approving it for publication was Hongbin Chen.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

PReLU	Parametric Rectifier Linear Unit
RCCNet	Convolutional Neural Network for Routine Colon Cancer Nuclei
ReLU	Rectifier Linear Unit
RelTanh	Rectified Linear Tanh
ResNet	Residual Network
RMAF	ReLU-Memristor-like Activation Function
RReLU	Randomized ReLU
SELU	Scaled Exponential Linear Unit
SqueezeNet	Squeeze Network
Tanh	Hyperbolic tangent

I. INTRODUCTION

Deep learning (DL) has recently shown very good performance on a range of tasks including computer vision, speech processing and natural language processing [1] partly due to the availability of large-scale datasets and high end computational resources [2]. Various deep neural networks have been introduced for different type of problems such as multi-layer perceptrons (MLP) [3] for handling real vector R-dimensional data [4], [5] and Convolutional Neural Networks (CNN) for malware detection, image processing and video processing [6]–[8]. Wen *et al.* proposed a Memristive Fully Convolutional Network (MFCN) for accelerating hardware image-segmentation, where a memory segment was able to perform computations [10]. Several researchers have capitalized the benefits of DL to implement different tasks such as hybrid MLP-CNN classifier which is used for fine resolution remotely sensed image classification [14], DL for solar power forecasting [15], deep CNN for audio-visual speech enhancement [16], DFUNet (a CNN) for diabetic foot ulcer classification [17] and RCCNet (a CNN) for histological routine colon cancer nuclei classification [18]. The deep neural network (DNN) generally transforms input to feature space, where it becomes linearly separable into classes. In other to achieve this, the neural networks depend on units called activation functions [23]. These functions (such as Sigmoid, Tanh and ReLU etc.) are the backbone of any neural network. The work of the activation function is to decide whether a neuron should be activated and also determines the relevant information to be received by the neurons. In addition, the properties of activation functions is the introduction of non-linearity to facilitate learning of connection weights for a precise problem. DL models use several sigmoid hidden layers along with a variety of initialization, optimization and regularization strategies [24]. Previous works have shown that sigmoid non-linearity function is not optimal for DNNs. The observation of this issue brought about a new activation function called hyperbolic tangent (tanh). Which was to leverage the issue of the sigmoid. The tanh could not succeed in solving the problem of the sigmoid but both of the activation functions had the same problem known as vanishing gradient.

Glorot *et al.* [55] found that DNNs with rectifier linear unit (ReLU) in place of traditional sigmoid and tanh can perform much better on image recognition and text classification tasks. Indeed, the advantage of rectified networks was obvious in tasks with more supervised training data, which is the case for DNN model training in large vocabulary continuous speech recognition (LVCSR). DNNs with rectifier non-linearities have played an important role in the top-performing systems for the ImageNet large scale image classification benchmark [7]. Moreover, the nonlinearity used in an unsupervised feature learning neural network played an important role in determining final system performance [22]. ReLU has become popular due to the simplicity of use in various type of DL models. The major drawback with ReLU is the diminishing gradient for the negative values of the inputs. Due to the problem of diminishing gradient, ReLU units can be weak during training and the gradient may die.

In this work, we used memristive window function technique to discover a new activation function, which we call ReLU-Memristor-like activation function (RMAF). The function is proposed to provide effective performance on deep networks and thereby providing researchers chance to replace ReLU. In this work, classification experiments were performed to show the performance improvement over variety of datasets on five types of neural networks. Our experiments show that RMAF outperforms ReLU and other standard activation functions on deep networks applied to variety of challenging domains such as image classification and feature recognition.

The contributions of this paper are as follows,

1) We highlight an activation function named RMAF to improve the performance of neural networks.
2) RMAF function introduces a constant hyperparameters α and p to make the function smooth, non-monotonous and scalable to any given network.
3) To evaluate the RMAF activation function, our study implemented different types of neural networks, including Multilayer Perceptron (MLP) and Convolutional Neural Network (CNN) by residual neural network (ResNet50), Alexnet, SqueezeNet and DenseNet121.
4) Experimental data used for the experiments are 1) 1-dimensional data, thus Wisconsin breast cancer, Car, Iris and MNIST datasets, 2) 2-dimensional image data, thus MNIST, CIFAR-10, CIFAR-100 and ImageNet datasets.

The rest of this paper is organized as follows: Section 2 Introduces the related works. Section 3 outlines the problem statement and the proposed RMAF activation; Section 4 describes the experimental setup; Section 5 presents the experimental results; and in Section 6, we concluded this paper.

II. RELATED WORK

Our focus in this section of the related work was based on the standard activation functions and different variations of the ReLU that has recently been proposed.

A. SIGMOID

The sigmoid activation function defined as $f(x) = \sigma(x) = 1/1 + e^{-x}$, was commonly used in the early days of neural networks. The function serves as the point-wise non-linearity applied to hidden neurons of a deep neural network. The anti-symmetric property of sigmoid is about 0 and has extremely gradual gradient compared a logistic sigmoid. This leads to a more robust optimization when training DNNs. Sigmoid function squashes a large negative number to 0 and a large positive number to 1. Implementing the sigmoid function on a very shallow layer does not give any serious problem. On the other hand, when the architecture becomes increasingly deeper due to the derivative terms that are lesser than 1. This may multiply each other many times and result to smaller values until the gradient tends towards zero, hence vanishing. In addition, if the values are bigger than 1, the opposite result happens with the numbers being multiplied and gradually increased until they approach to infinity and gradient may explode. Hence, the sigmoid function in DNNs could suffer from the problem of vanishing gradient [44].

B. HYPERBOLIC TANGENT

Tanh defined as $(2/2 + e^{-2x}) - 1$, is among the very popular and widely used activation functions by most researchers. This function has the same characteristics as the sigmoid function and can be seen as a scaled sigmoid function. Except that, the output values of tanh lie between -1 and 1 depending on the input real values. The non-linearity nature of tanh also has the capabilities of stacking layers. The range $-1, 1$ of the function gives the property of being capable of solving activation blow up through the network. The gradient for tanh is stronger when compared to sigmoid which has steeper derivatives. Deciding between the sigmoid and tanh will depend on the requirement of gradient strength. The vanishing gradient is the major problem of sigmoid and tanh in both positive and negative directions. The vanishing gradient occurs when the lower layers of a DNN have gradients near to 0. For instance, when the higher layer units are nearly saturated at the point of -1 or 1. The problem can introduce slow optimization convergence and also poor local minimum on the final network trained to converge. Hence, the weight of the hidden unit in the network must be carefully initialized to prevent any significant saturation during the initial stage of training.

C. ReLU

The ReLU activation function ($f(x) = max(x, 0)$) was proposed for training deep networks [6]. The function has breakthrough against the problem termed as vanishing gradient which is in the case of sigmoid and tanh. The ReLU is an identity function for the non-negative inputs and zero function for the negative inputs. Nair and Hinton [28] noted that, the better solution to gradient vanishing from sigmoid and tanh is to maintain the resultant values to 1 to prevent changes when they are multiplied. This is generally the work of ReLU, which has a gradient of 1 for positive inputs and 0 for negative inputs. Having gradient of zero may be an issue at first but can help to make the network sparse by keeping useful links. Several works have shown that, the network becomes less dense and low computation with the help of sparsity. However, in spite of the performance of ReLU, excess amount of sparsity element which it introduce could also be harmful to the networks where negative values are prevented from propagating and once the gradient becomes zero any corresponding nodes would not have influence on the network. As a result, they do not provide any contribution to the improvement of the learning, which is therefore called dying neurons. Hence, negative values are not considered to be important representation. The worse part of this is that, the DNNs would not have any benefits from negative representations. It has been noted from several research that the network can benefit from the negative representations to achieve better performance [25], [27].

Through the adoption of ReLU in deep learning and its dying neurons problem, several variants of ReLU activation function (such as, Leaky ReLU (LReLU) [24], Parametric ReLU (PReLU) [27], Randomized ReLU (RReLU) [51], Exponential linear units (ELU) [26], Gaussian Error Linear Units (GELU) [29] and Scaled Exponential Linear Units (SELU) [28]) have been proposed which allow the propagating of negative values in the network.

D. LReLU

Maas *et al.* [24] proposed LReLU. This function is similar compared to ReLU, except for allowing small, non-negative and constant gradient of input in other to reduce the dying neuron input problem. In other to solve the dying problem, LReLU introduced an alpha (α) parameter which is the leak, so that gradients will be small but not zero. This approach reduces the sparsity but tends to make the gradient more robust for optimization and on the other hand, there is an adjustable weight for the nodes that were not active with ReLU.

E. PReLU

PReLU is an extension of LReLU by training with which trains coefficient of leakage into a parameter instead of fixing it with a constant value He *et al.* [26]. Leakage parameter in PReLU is learned along with the other neural network parameters. Hence, LReLU and PReLU are in between the linear function and ReLU which is a disadvantage in terms of decreased rate in non-linearity.

F. ELU

The function is similar to ReLU in terms of identity functions for non-negative inputs Clevert *et al.* [25]. ELU becomes smooth slowly until its output equals to a constant negative value unlike ReLU which sharply becomes smooth. ELU blows activation which leads to gradient exploding problem for a given positive inputs.

G. SELU

SELU function adds a scaling fixed parameter in ELU that sustains the bad weight initialization Nair and Hinton [28].

H. GELU

GELU randomly applies identity or zero map to neuron input according to Gaussian function Hendrycks and Gimpel [29]. In between ReLU and ELU lies the shape of GELU.

I. ABReLU

ABReLU was proposed which allows prominent negative values after converting it into positive values by biasing it with an average of all activations Dubey and Chakraborty [30]. The function ABReLU could as well not utilize the negative values due to the trimming of values at zero, similarly to ReLU.

J. SWISH

Ramachandran *et al.* proposed a sigmoid-weighted linear unit Swish activation function [33]. In their work, they found Swish a promising activation function by testing several functions. The Swish activation function interpolates between Linear function and ReLU, by learning the parameter. However, the gradient diminishing problem is still in existing in the case of Swish activation function.

K. RelTanh

Wang *et al.* [52] proposed a new activation function called Rectified Linear Tanh (ReLTanh) to improve traditional Tanh function, due to its vanishing gradient problem. In their proposed work, they replaced Tanh's saturated waveforms with two straight lines, which the slope of the lines was calculated by using the tanh's derivatives at two learnable thresholds. The waveform of tanh in the center provides the ReLTanh with ability of nonlinear fitting while the linear contributes to the relief of vanishing gradient problem.

L. ISIGMOID

Qin *et al.* [53] noticed that, ReLU and LReLU functions are not applied to Deep Belief Networks (DBNs), because both functions cannot fully make use of pre-training effects of Restricted Boltzmann Machines. Therefore, they proposed an improved logistic Sigmoid (Isigmoid) units to tackle vanishing gradient problem during back-propagation of DBNs using the conventional sigmoid function. They designed the Isigmoid to combine the advantage of the unsaturation from Leaky Rectified Linear (LReL) units.

M. DReLU

Macedo *et al.* [58] proposed a transfer function called Display Rectifier Linear Unit (DReLU). It was indicated that performing conjecture by the enlarging the identity function of ReLU to the third quadrant can enhance the compatibility of batch normalization.

Recently, an attempt was made to design a nonparametric and complex activation function for complex-valued neural networks (CVNNs) [34]. This depends on the kernel expansion with a fixed dictionary which can be applied on vectorized hardware. The existing activation functions are still weak to replace ReLU due to their inconsistencies.

In this paper, RMAF is proposed which allows scalability to any application and unlike LReLU able to solve dying neurons in the neural networks.

III. PROBLEM STATEMENT

Deep feed-forward Neural Network is composed of more than one hidden nonlinear layer. Given an input vector x, each hidden layer is capable of transforming input vector by applying a linear affine transformation followed by a non-linear mapping. The training of deep neural networks becomes slow in optimization convergence with Sigmoid and Tanh activation functions due to the problem of vanishing gradient. Recently, ReLU has been the popularly used activation function for training deep neural networks which solves vanishing gradient problem [28]. However, ReLU suffers from gradient diminishing problem which leads to the problem of dying neuron in the networks. Therefore, this work proposes RMAF that can replace the ReLU and enhance the performance of given neural networks. In this paper, we explore memristive window function to discover a new activation function for deep neural network. We introduce memristive window function and the proposed activation functions in the next part of this sections.

A. MEMRISTIVE WINDOW FUNCTION

Memristor was proposed by Leon Chua, being the fourth passive circuit element besides resistor, capacitor and inductor. In 2008, a group of researchers announced the production of memristor as a physical device [35]. After, the studies on the memristor increased rapidly. Having a suitable modeling for the memristor element is very important and as a result, various methods of modeling memristor was proposed in the literature. Memristor model called the linear ionic drift model was proposed taking into account the linear drift on the memristive device [35]. However, this model had some boundary effect problem that needed to be tackled. Based on that, it was important to create new memristor models that can achieve scalability, boundary lock, and nonlinear effects at the same time [36]. The method used to model the memristor is a non-linear model called window function. Hence, the window function was an approach to account boundary condition that was not recognised in the linear model. Several window functions available in the literature was used in modeling of the memristor.

This work was inspired by Clevert *et al.* [25] finding a novel window function to solve boundary lock phenomenon.

Given *(4)-(5)* in [36] as a memristive system, having the window function *f(x)* which satisfies *(B.1)-(B.4)*, the internal state variable therefore follows as a sigmoidal curve.

$$x = S(q) = F^{-}1(\alpha(q - q_0)) \tag{1}$$

Here, if the sigmoidal *x=S(q)* is obtained, we compute the corresponding *f(x)* by applying:

$$\frac{1}{f(x)} = \alpha\frac{d}{dx}[S^{-1}(x)] \tag{2}$$

The transformation achieve windows from wide range of sigmoidal function used in modeling most physical and biological processes [37][38]. Sigmoidal is obtain from the window function analytically. Here, *(2)* is simplified as in [36].

$$f(x) = 1 - [(x - 0.5)^2 + 0.75]^p \tag{3}$$

this can be equivalently written as $f(x) = 1 - [0.25(2x - 1)^2 + 0.75]^p$. To solve the boundary lock problem $(2x - 1)$ was replaced by $(x - stp(-i))$. Therefore, the new function is define as

$$f(x) = 1 - [0.25(x - stp(-i))^2 + 0.75]^p \tag{4}$$

where in the design of memristor device, the *stp* is known as short-term plasticity, which is applied to solve any boundary lock effects [56], [57]. While *i* represents the current flowing through the memristive device. The *stp* is defined as

$$stp(i) = \begin{cases} 1, & if\ i \geq 0\ (v \geq 0) \\ 0, & if\ i < 0\ (v < 0) \end{cases} \tag{5}$$

Jinxiang *et al.* [56] modified *(4)* to obtain window function to improve the scalability of the function. Thus, the function can be adjusted to various application.

$$f(x) = j(1 - [0.25(x - stp(-i))^2 + 0.75]^p) \tag{6}$$

where *j* is used as a scale parameter and any *p* scales upward or downward with suitable *j*. In choosing different *j*, the window function can adjust to various applications.

B. THE PROPOSED METHOD: RMAF

Our focus is finding a scalar activation function, which takes in scalar input and output a scalar. This is because scalar activation functions can replace the ReLU function without changing the architectural network.

RMAF contains properties like both ReLU and Swish, and a threshold *p* and *j* hyperparameter are attached, which could improve the classification accuracy of deep neural networks. The ReLU activation function can mathematically be calculated as [4]:

$$ReLU(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{7}$$

The introduction of ReLU tackles the issues such as gradient vanishing/exploding and squashing problems by the Sigmoid and Tanh activation functions in DNNs [15]–[17]. Sigmoid activation function can be calculated as [18]:

$$Sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{8}$$

Based on the scalar generation and the focus on non-linearity for our model in deep neural networks, the (6) was modified. To construct the RMAF activation function, the (6) was amend by replacing the first minus operator with $(/)$ and $x + stp(-1)^2$ with function $(1 + e^{-}x)$. This can simply be written as

$$RMAF(x) = (j(1/[0.25t(1 + e^{-x}) + 0.75]^p)) \tag{9}$$

In this work, (9) was multiplied with a constant or trainable parameter αx, which in this work, α was initialized to 1 (i.e. $\alpha = 1$). Finally, the RMAF activation function can be defined as

$$RMAF(x) = \left[j\left(\frac{1}{(0.25 \cdot (1 + e^{-x}) + 0.75)^p}\right)\right] \cdot \alpha x \tag{10}$$

where parameter $p > 0$, controls the flatness of region. With adjustment of parameters *j* and *p* make the function like ReLU and capable of scaling to any given network. Fig. 1 shows the plot of the proposed function and its derivative function.

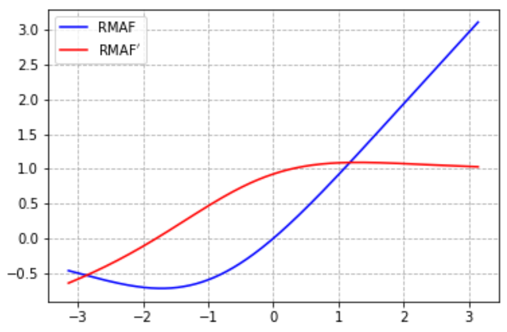


FIGURE 1. The proposed RMAF function and its derivative.

This work has noticed that, the RMAF at x $\geq$ 0 has a similar property to "Swish" which was proposed recently by Google Brain [19]. Fig. 2 shows plot of RMAF function and derivative with different initialized *p*. Moreover, we visualized the comparison of RMAF with ReLU and Swish in Fig. 3, which their derivative plots are shown. Swish shows much superiority over ReLU on several deep models in image classification and machine translation problems [19]. Meanwhile, the derivative of Swish is composed with high portion of nonsparse property and hence, increase the computational complexity of the given network. However, RMAF can retain its hard zero property which on the other side ReLU which often deactivated most of the neurons during both forward and backward propagation.

The RMAF function can be any point utilized when fitting data and giving knowledge to the basic forms dependable on its dynamics.

It is noted that deep neural network is composed of several differentiable functions [50], hence, during the backward propagation [51], neural network updates its parameters (notably the weights and biases) by computing the derivative (or gradient). The derivative of the RMAF function in (10)

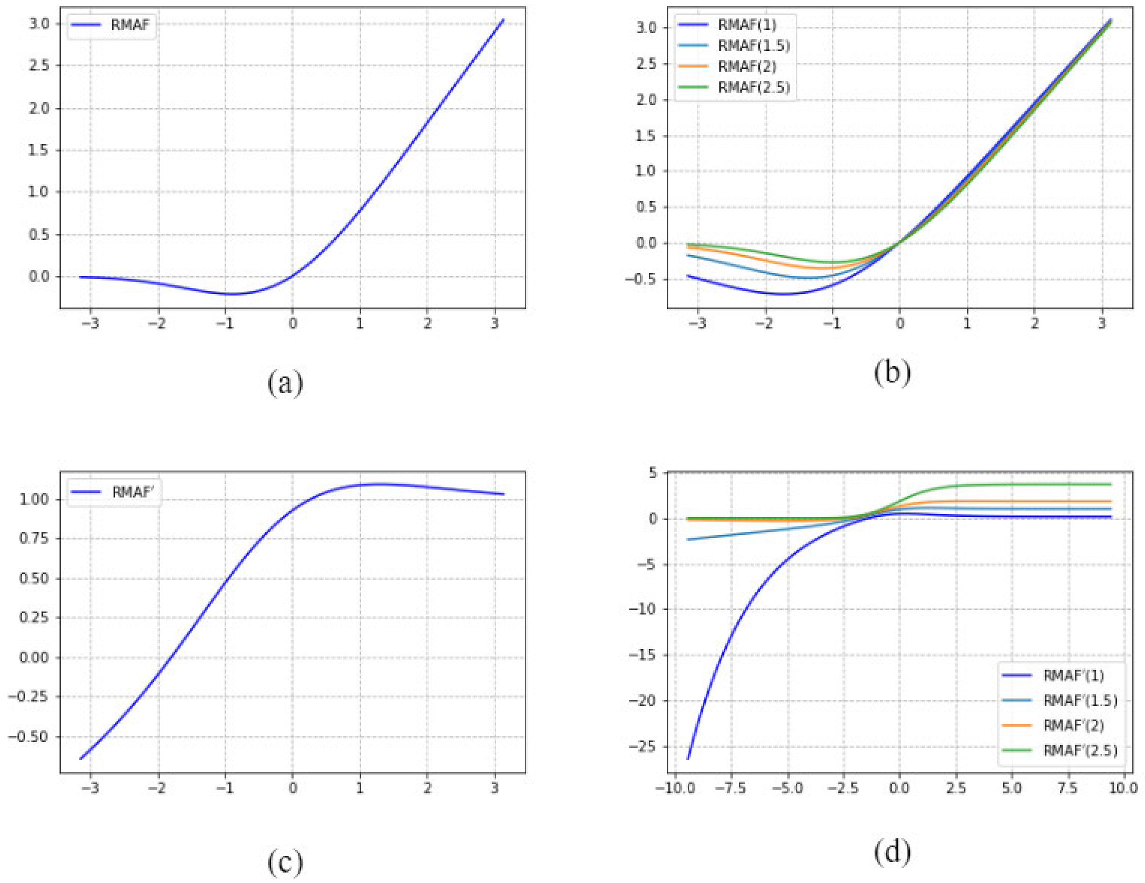


FIGURE 2. **RMAF function and its derivative with different p. In (a) the plot represents the RMAF with the parameter $p = 1$. (b) shows a RMAF function with different selected p. (c) represent the derivative of RMAF, while (d) shows the derivative of RMAF with different p. In this work, all curves were set with the same alpha (i.e. $\alpha = 1$). Best view in color.**

can be formulate as: when $j = 1$,

$$f(x) = \frac{\alpha x}{[0.25(1+e^{-x})+0.75]^p} \tag{11}$$

for convenience sake, let $f_1(x) = \alpha x$ and

$$f_2(x) = [0.25(1+e^{-x})+0.75]^p$$

we can derive that;

$$\frac{d[f_1(x)]}{dx} = \alpha \tag{12}$$

and

$$\begin{aligned}\frac{d[f_2(x)]}{dx} &= p[0.25(1+e^{-x})+0.75]^{p-1} \\ &\quad \cdot \frac{d[0.25(1+e^{-x})+0.75]}{dx} \\ &= -0.25pe^{-x}[0.25(1+e^{-x})+0.75]^{p-1}\end{aligned} \tag{13}$$

According to the derivation rules for compound function, $\frac{df(x)}{dx}$ can be described as

$$\frac{df(x)}{dx} = \frac{d[\frac{f_1(x)}{f_2(x)}]}{dx} = \frac{\frac{d[f_1(x)]}{dx}f_2(x) - \frac{d[f_2(x)]}{dx}f_1(x)}{f_2^2(x)} \tag{14}$$

Substitute (12) (13) and (14) into equation (15) such that we can obtain

$$\begin{aligned}\frac{df(x)}{dx} &= \frac{\alpha f_2(x) + 0.25p\alpha x e^{-x}[0.25(1+e^{-x})+0.75]^{p-1}}{[f_2(x)]^2} \\ &= \frac{\alpha}{[0.25(1+e^{-x})+0.75]^p} \\ &\quad + \frac{0.25p\alpha x e^{-x}[0.25(1+e^{-x})+0.75]^{p-1}}{[0.25(1+e^{-x})+0.75]^{2p}}\end{aligned} \tag{15}$$

we can simply re-write the above as

$$\begin{aligned}\frac{d}{dx}RMAF(x) &= \frac{\alpha}{[0.25(1+e^{-x})+0.75]^p} \\ &\quad + \frac{0.25p\alpha x e^{-x}}{[0.25(1+e^{-x})+0.75]^{p+1}}\end{aligned} \tag{16}$$

In this work, the experiments show that RMAF matches or performs higher than ReLU on varieties of deep learning models. In Fig. 1, RMAF is unbounded above and bounded below unlike ReLU and Swish transfer functions

Theoretically, it is importance to achieve unboundedness to avoid the saturation when training is slow because of near-zero gradient Glorot & Bengio, 2010 [54]. The ReLU transfer

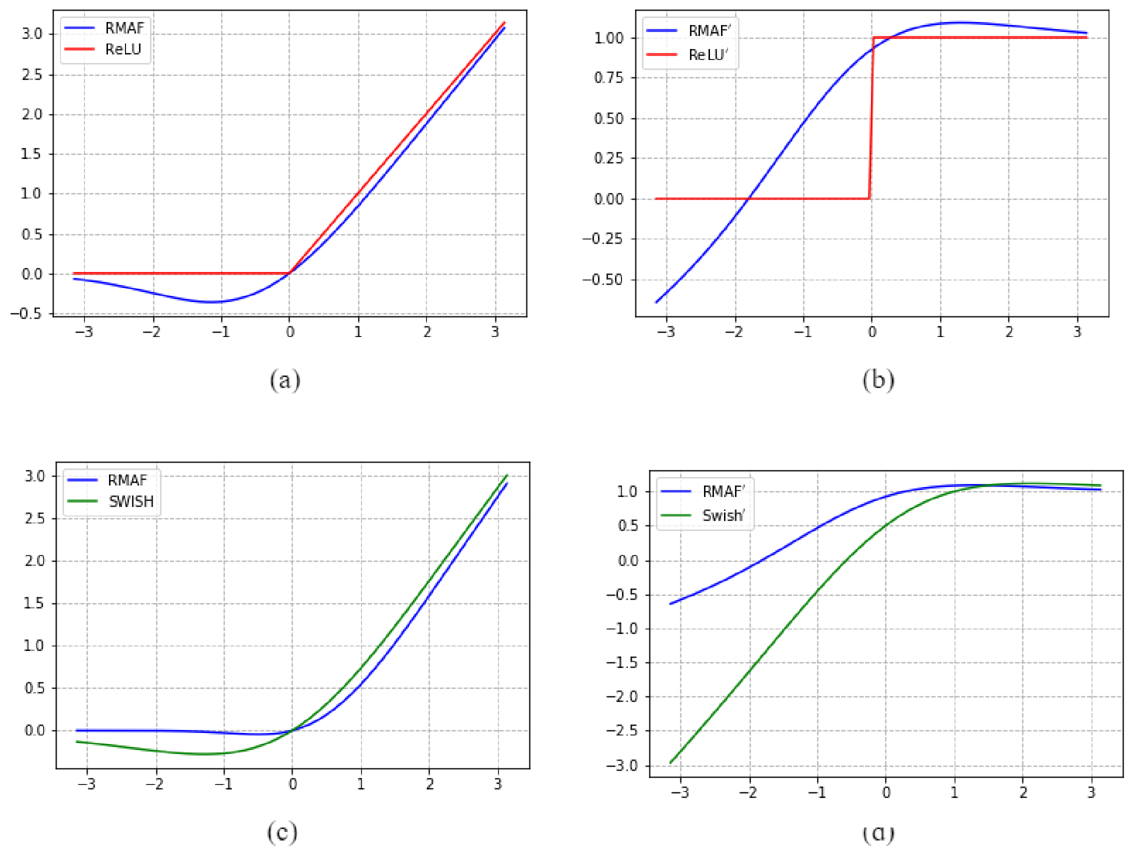


FIGURE 3. **Visualization of RMAF, its derivative compared with ReLU and Swish activation functions. (a) RMAF vs ReLU, (b) The derivative of RMAF and ReLU. (c) RMAF vs Swish and (d) The derivative of RMAF and Swish. The derivative of RMAF is denoted as RMAF′. This is same with the ReLU and Swish as shown at the upper left of each of the figures. Best view in color.**

function was a great improvement over the tanh function due to the property of unbounded above that can avoid saturation when $x > 0$. The unbounded above property is very important and have been applied in almost recently proposed transfer functions. Additionally, the property of bounded below can also be an advantage due to the strong regularization effects. Due the limitation of the function approaching zero, large negative inputs are irrecoverable Ramachandran *et al.* [32]. The RMAF activation function shares the same property having its positive side approaching the linear function as the input becomes increasingly positive. Unlike LReLU [24], RMAF also introduced hyperparameters (α) and (p) which are considered as leaks to make sure gradients will be small but not zero. This idea was applied to resolve the vanishing gradient and dying neuron problem.

Although, it is much difficult to rate an activation function over its existing counterparts, we are convinced that the RMAF function being both unbounded above, bounded below, smooth and non-monotonic are all advantages. The non-monotonic subject to RMAF has the capability of increasing and improving the gradient flow, which can be considered as a great importance focusing on the pre-activations that falls into this range. Furthermore, robustness can also be achieved through different initialization process and learning rates.

IV. EXPERIMENT SETUP

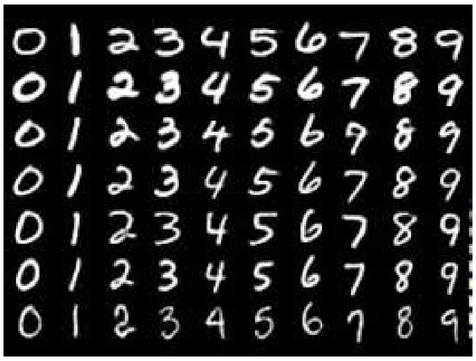
A. DATASETS

The effectiveness of the proposed RMAF function was evaluated on seven benchmark databases, thus Wisconsin breast cancer, MNIST, Iris, Car evaluation, CIFAR-10, CIFAR-100 and ImageNet.

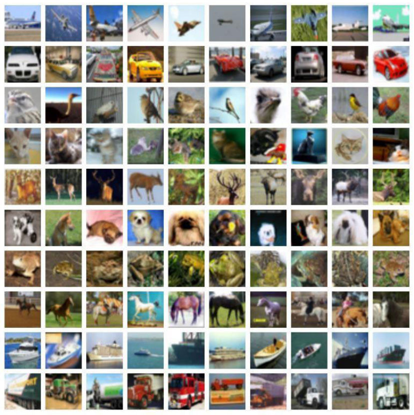
Recently, researches on medicine have shown that, the features of breast cell nuclei can be used to classify benign and malignant breast tumors [40]. In this work, the breast cancer data from Wisconsin Breast Cancer Data repository created by [Dr. William H. Wolberg] was used. The dataset helps scientists, cancer biologists and clinical researchers to evaluate and conduct their research [41]. Features are generated from a digitized image of a fine needle aspiration (FNA) of a breast mass. Which describe characteristics of the cell available in the image. The dataset consists of 569 instances of both benign and malignant cells as shown in Table 1. The attribute of the dataset is made of ID number, Diagnosis and 30 real-valued features which are computed for each cell nucleus. The dataset was split into training and testing set, each involving 357 instances and 212 instances respectively. For training and testing the multi-layer neural network on the dataset in other to distinguish between the benign (non-cancerous) and malignant (cancerous), 70% of the cancer dataset was used for training and 30% for testing.

TABLE 1. Database information of Wisconsin breast cancer, MNIST, Iris Car evaluation, CIFAR-10 and CIFAR-100 showing training, testing, total set and the number of classes for each datasets.

No.	Dataset	Training	Testing	Total	Classes
1	Breast Cancer	398	171	569	2
2	MNIST	60,000	10,000	70,000	10
3	Iris	105	45	150	3
4	Car Eval.	1209	519	1728	4
5	CIFAR-10	50,000	10,000	60,000	10
6	CIFAR-100	50,000	10,000	60,000	100



(a)



(b)

FIGURE 4. Sample data of MNIST and CIFAR.(a) Shows a sample data of MNIST, (b) shows sample data of CIFAR-10 and CIFAR-100. We note that, CIFAR-10 and CIFAR-100 are composed of different total class samples. As the names depict, the total classes of CIFAR-10 is 10 while CIFAR-100 has 100 classes.

In this study, normalization features were performed to have a uniformly scaled value. Moreover, the process was done to prevent gradient descent taking too long to converge. In addition, the idea of normalization regarding each feature was to have the mean of zero and unit variance.

The Car Evaluation dataset used in this paper was to test the performance of activation functions with multi-layer perceptron. The dataset contains total samples 1728 with 4 classes having 6 dimensional features [43]. The Iris Flower dataset [43] is a classic and one of the best-known multivariate datasets that contains a total of 150 samples from 3 different species, including Iris setosa, Iris virginica and Iris versicolor. In Iris dataset, every sample is a vector of length four (i.e., "sepal length", "sepal width", "petal length", and "petal width"). The MNIST dataset is widely used for recognizing handwritten digits from images. This dataset contains a total of 60,000 training samples and 10,000 test samples of dimension 28 × 28 [44]. In this dataset, one image contains a digit from 0 to 9. The Fig. 4(a) shows sample images of the MNIST dataset. The CIFAR-10 dataset consists of 32×32 resolution 60,000 color images from 10 classes, with 6,000 images per class [45]. The 5,000 images with 5,000 images per class were used for the training and 10,000 images with 1,000 images per class were used for the testing. The 10 classes of CIFAR-10 dataset are composed of the 'airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', and 'truck' categories, respectively. The CIFAR-100 dataset contains the same CIFAR-10 dataset images. The dataset was split into two where 50000 was used for training and 10,000 for testing. The training and testing images were categorized into 100 classes in CIFAR-100 dataset. The sample images of CIFAR-10 and CIFAR-100 data is shown Fig. 4(b). The CIFAR datasets is available www.cs.toronto.edu/ kris/cifar.html

B. TESTED NEURAL NETWORKS

A neural network (NN) is an implementation of biological neural network used in solving complex task using rule of programming [46]. A perceptron defined for 2-dimensional space that classifies problems can be expressed as

$$T_n = W_1X_1 + W_2X_2 + \ldots W_{n-1}X_{n-1} \tag{17}$$

$$Y = \phi(\sum_{i=1}^{n} W_i * X_i + B) \tag{18}$$

where ϕ represents an activation function, X represents the input vector, which can also be the output from the previous layer. W is the weight or set of parameters in the layer and B represents bias vector. The activation functions utilized in our experimentation are sigmoid, eq. 7, and ReLU (Rectified Linear Unit) function (eq. 6).

In this paper, two different types of neural networks (i.e. Multi-layer Perceptron (MLP)) with one or more hidden layers, and deep convolutional Pre-activation Residual Network (ResNet-PreAct) [47] were designed.

1) MULTIlAYER PERCEPTRON

We designed a MultiLayer Perceptron (MLP) consisting of three hidden layers for the classification problem. This MLP network is a deep network due to the number of hidden layers implemented, which composed of more than two hidden layers. Fig. 5 shows our simple design structure of MLP.

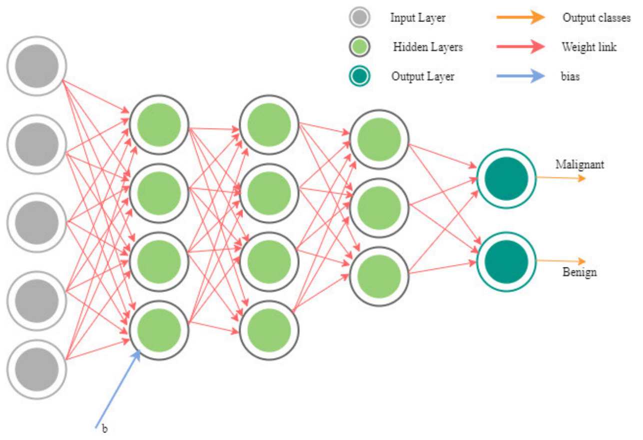


FIGURE 5. A structure of our multi-layer perceptron (MLP).

Evaluating the proposed activation function on the breast cancer classification, our MLP shown in Fig. 5 was used. The MLP architecture is composed of input, hidden, and output layers. The number of neurons initialized in the input layer corresponds to the features of the Wisconsin breast cancer dataset, and two neurons in the output layer (0 or 1) signifying the predicted classes. The network is made of three hidden layers where they constitute 8, 6, and 4 neurons respectively which can be viewed as deep network. The input of a hidden unit *j* in layer l receive value from the previous layer and multiplying weights plus the biases. The hidden unit *j* obtains a single output by following nonlinear activation function *f*.

$$0_j^l = F\left(\sum_{K=1}^{K} w_{k_j} x_k + \theta_j\right) \tag{19}$$

where F is a transfer function and K is the number of hidden units in each hidden layer and w and θ are weights and bias associated with unit j that need to be learned.

For the Car evaluation dataset, the MLP was initialized with 6, 5, and 4 nodes while Iris Flower dataset over MLP consists of 5, 4, and 3 nodes in the input, hidden and output layers respectively. The MNIST dataset images were stretched into one-dimensional (1D) vectors used with multi-layer perceptron neural network. In this work, the MNIST dataset trained on the MLP consists of input, hidden and output layer, and has 784, 512, and 10 nodes respectively.

2) RESIDUAL NEURAL NETWORK

The state-of-the-art for image classification task was the Residual Neural Network (ResNet). The ResNet architecture with pre-activation [47] is accepted as the improved version of ResNet. In this study, the model was used for the image classification over MNIST, CIFAR-10, CIFAR-100 and ImageNet datasets. The pre-activation ResNet was used with 164-layers over the CIFAR-10 and CIFAR-100 datasets.

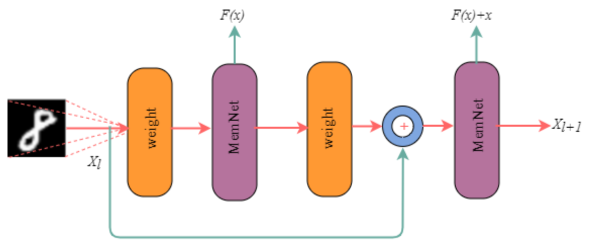


FIGURE 6. Pre-activated residual module: RMAF and Weight represent activation and convolutional layers. The X_l and X_{l+1} are input and output volumes, respectively. Here residual module adds the input volume X_l with output volume of the second convoluional layer *F(x)+x* to obtain the final output volume X_{l+1}.

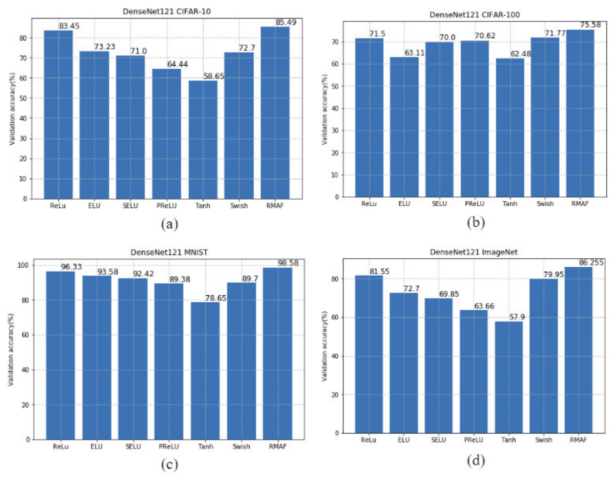


FIGURE 7. DenseNet model validation accuracy on the (a) CIFAR-10, (b) CIFAR-100, (c) MNIST and (d) ImageNet dataset trained with 200 epochs.

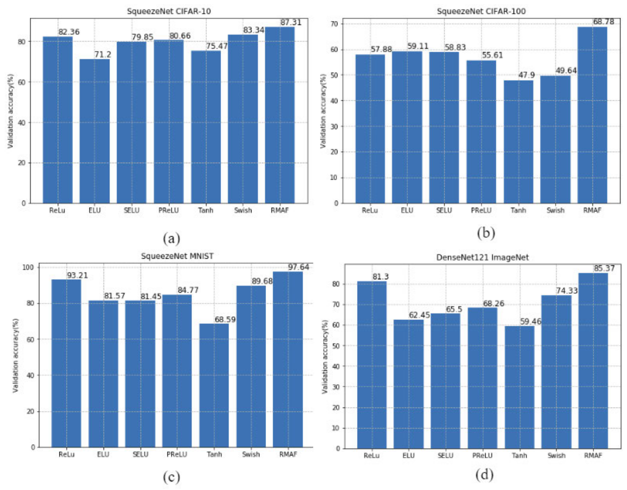


FIGURE 8. SqueezeNet model validation accuracy on the (a) CIFAR-10, (b) CIFAR-100, (c) MNIST and (d) ImageNet dataset trained with 200 epochs.

In this work, we used 20 layers for MNIST dataset. The Fig. 6 illustrates a simple pre-activation residual block.

3) DENSE CONNECTED CONVOLUTIONAL NETWORKS

In other to show how RMAF performs on deep learning structures, we designed and implemented Dense

TABLE 2. Classification performance of MLP over Breast cancer, Cars Evaluation, Iris and MNIST datasets for different activations.

Dataset	Activation Function	Train		Validation	
		Accuracy	Loss	Accuracy	Loss
Breast Cancer	Sigmoid	95.52	0.165	95.22	0.164
	Tanh	96.47	0.103	94.55	0.102
	ReLU	97.44	0.054	96.31	0.053
	Swish	96.47	0.111	95.90	0.110
	RMAF	**98.74**	0.043	**96.34**	0.042
MNIST	Sigmoid	97.83	0.095	96.23	0.091
	Tanh	97.46	0.094	96.26	0.090
	ReLU	99.51	0.019	98.48	0.104
	Swish	99.58	0.016	98.45	0.105
	RMAF	**99.67**	0.013	**98.75**	0.080
Iris	Sigmoid	97.83	0.095	96.23	0.091
	Tanh	97.46	0.094	96.26	0.090
	ReLU	98.33	0.098	96.41	0.089
	Swish	98.50	0.095	96.34	0.099
	RMAF	**98.81**	0.077	**97.83**	0.052
Car Eval.	Sigmoid	98.77	0.025	96.24	0.111
	Tanh	98.84	0.034	96.40	0.099
	ReLU	99.10	0.029	97.40	0.077
	Swish	98.50	0.095	96.34	0.099
	RMAF	**99.42**	0.088	**98.45**	0.049

Convolutional Network (DenseNet) [61] for experimentation on our work, which we trained from scratch. DenseNet is a deep model, connecting each layer to all other layers in a feed-forward manner. Whiles the conventional convolutional networks with number of layers and number connections in between the layers and their preceding layers, we implemented a network with $L_n(L_n+1)/2$ direct connections. In the DenseNet, each layer with their feature-maps of all preceding layers were applied as inputs data. Whiles its existing feature-maps were used as inputs to the subsequent layers. DenseNets have several advantages that improve performance of the network: they resolve the vanishing-gradient problem, feature propagation is strengthen, feature reuse is encourage and finally number of parameters are substantially reduced.

C. TRAINING SETTINGS

In this work, Keras deep learning libraries with tensorflow back-end was used for the implementation and evaluation of the activation function. NVIDIA GeForce GTX 1060 6GB GPU system was used for the different experimentations. In training the neural networks, we used batch size = 128, learning rate of 0.1 and Adam optimizer [49] was used for categorical entropy loss.

V. EXPERIMENTAL RESULTS

In this paper, the effectiveness and performance of the proposed RMAF transfer function was evaluated and compared with state-of-the art activation functions such as sigmoid, Tanh, ReLU, ELU, SELU, PReLU and Swish. In this section of the work, experiment results using multi-layer perceptron (MLP), ResNet50, Alexnet, SqueezeNet and DenseNet121 are presented.

A. MULTI-LAYER PERCEPTRON

In this section, Table 2 report the classification performance of multi-layer perceptron (MLP) over Wisconsin Breast cancer, MNIST, Iris, and Cars datasets. Here, the work was evaluated based on the accuracy and loss over training and validation sets. RMAF transfer activation achieved the least training and validation loss over the other datasets. RMAF activation function achieved the best accuracies of 97.98%, 97.33% and 98.60% over the benchmark Breast cancer, MNIST, Iris and the Car evaluation datasets.

B. RESIDUAL NETWORK

Table 3 and Fig. 9 - 10 summarizes the training accuracies of activations over datasets CIFAR-10, CIFAR-100, MNIST and ImageNet with pre-activation ResNet50. In this work, ResNet50 was given depth of 20 over MNIST while 164 for CIFAR datasets. The training was performed for 200 epochs using the categorical cross-entropy loss. It was observed that our proposed RMAF activation function outperformed the other transfer functions with a record accuracies of 99.73% and 98.77%, and 79.82% over CIFAR-10, CIFAR-100 and MNIST, datasets respectively. The experiment on ImageNet dataset using ResNet50 produced 97.60%. In Table 3, the experiment on ImageNet dataset using ResNet50 produced 97.60% which is higher than the compared functions. Moreover, an improvement has been achieved by RMAF on CIFAR datasets. The nonlinear properties of our proposed RMAF activation function achieved efficient and better training than the other activation functions. The nature of RMAF led to more exploration of weights, negative and positive gradients to improve training and classification performance.

TABLE 3. Classification performance of ResNet50, Alexnet, Squeezenet and Densenet121 on CIFAR-10, CIFAR-100, MNIST and ImageNet dataset for different activation functions.

Model	Dataset	ReLU	ELU	SELU	PReLU	Tanh	Swish	RMAF(Ours)
		Accuracy (%)						
Resnet50	CIFAR-10	95.84	93.50	92.55	94.48	92.24	91.60	**98.77**
Alexnet		80.80	80.58	80.34	80.78	78.06	80.04	**84.58**
SqueezeNet		82.36	71.20	79.85	880.66	75.47	83.34	**87.31**
DenseNet121		83.45	73.23	71.00	64.44	58.65	72.70	**85.49**
Resnet50	CIFAR-100	75.72	74.54	73.45	75.36	64.12	74.46	**79.82**
Alexnet		62.59	61.55	68.79	61.88	56.83	61.17	**69.97**
SqueezeNet		57.88	59.11	58.83	55.61	47.90	49.64	**68.78**
DenseNet121		71.50	63.11	70.00	70.62	62.48	71.77	**75.58**
Resnet50	MNIST	99.56	96.52	96.66	97.80	99.48	99.53	**99.73**
Alexnet		89.44	87.67	87.55	86.23	84.60	88.80	**92.28**
SqueezeNet		93.21	81.57	81.45	84.77	68.59	89.68	**97.64**
DenseNet121		96.33	93.58	92.42	89.38	78.65	89.70	**98.58**
Resnet50	ImageNet	85.85	83.20	83.63	82.55	78.58	81.67	**87.60**
Alexnet		77.60	74.35	72.55	75.30	65.67	73.54	**80.20**
SqueezeNet		81.30	62.45	65.50	68.26	59.46	73.33	**85.37**
DenseNet121		81.55	82.70	69.85	63.66	57.90	79.95	**86.25**

TABLE 4. Statistical features metrics on RMAF validation accuracy.

Model	Dataset	Mean	SD	RMS	MS	Var
Resnet50	CIFAR-10	0.943	±0.056	0.935	0.035	0.025
Alexnet		0.787	±0.050	0.789	0.002	0.003
SqueezeNet		0.763	±0.087	0.768	0.001	0.007
Densenet121		0.845	±0.014	0.847	0.006	0.031
Resnet50	CIFAR-100	0.787	±0.013	0.788	0.043	0.005
Alexnet		0.685	±0.031	0.698	0.050	0.017
SqueezeNet		0.695	±0.018	0.597	0.021	0.009
DenseNet121		0.746	±0.025	0.756	0.007	0.010
Resnet50	MNIST	0.977	±0.019	0.979	0.004	0.025
Alexnet		0.769	±0.017	0.767	0.015	0.009
SqueezeNet		0.967	±0.021	0.968	0.022	0.021
DenseNet121		0.956	±0.019	0.974	0.016	0.005
Resnet50	ImageNet	0.823	±0.025	0.828	0.034	0.055
Alexnet		0.766	±0.022	0.755	0.005	0.060
SqueezeNet		0.816	±0.015	0.813	0.004	0.039
DenseNet121		0.797	±0.031	0.789	0.002	0.004

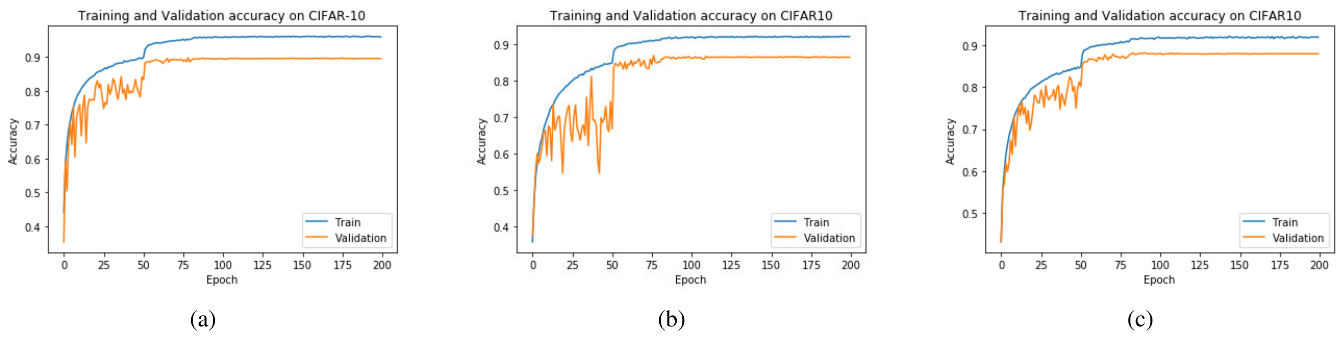


FIGURE 9. Resnet50: Comparison of training and validation accuracy of our proposed RMAF to two baseline (ReLU and Tanh) activation functions on CIFAR10. *(a) Training and Validation accuracy of ReLU achieved 95.9% and 89.5% respectively, (b) Training and Validation of Tanh had 92.2% and 86.6% respectively*. We show that our proposed function *(c)* achieves higher performance training and validation accuracy (i.e. 98.7% and 90.3%) compared to ReLU *(a)* and Tanh *(b)* on CIFAR10 dataset.

C. DENSE CONNCECTED NEURAL NETWORK

In the experiment with DenseNet121, RMAF achieved higher performance compared ReLU and other transfer functions trained on 200 epochs. This result indicates that, RMAF has the fastest training. Table 3 and 4 Fig. presents the various performance on DenseNet121 which RMAF outperformed the other functions. Moreover, Fig. 7 show the performance of RMAF with DenseNet with CIFAR-10, CIFAR-100, MNIST and ImageNet dataset. It could be seen that RMAF achieved the highest performance among the other functions.

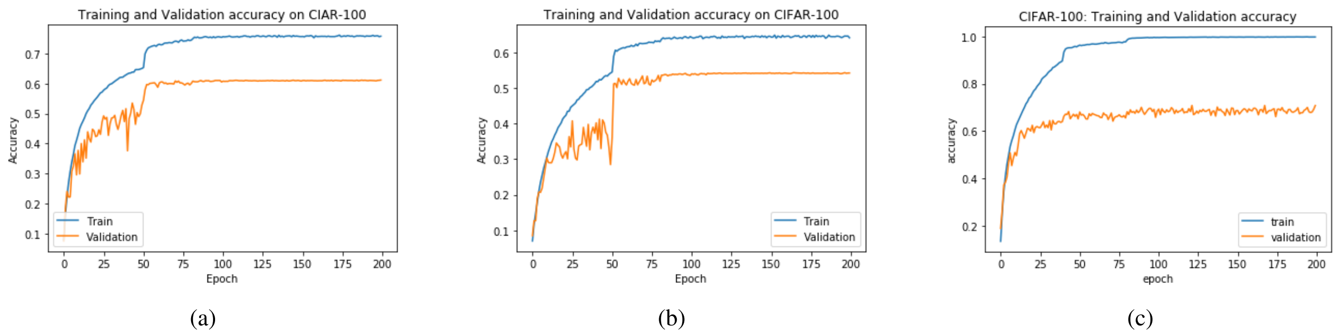


FIGURE 10. **Resnet50: Comparison of training and validation accuracy of our proposed RMAF to two baseline (ReLU and Tanh) activation functions on CIFAR100.** ***(a) Shows training and validation accuracy of ReLU achieving 75.7% and 61.2% respectively, (b) Training and Validation of Tanh had 64.1% and 54.2% respectively.*** **We show that our proposed function *(c)* achieves higher performance training and validation accuracy (i.e. 79.8% and 66.3%) compared to ReLU *(a)* and Tanh *(b)* on CIFAR100 dataset.**

TABLE 5. **Statistic features.**

Name	Formula	Name	Formula
Mean	$t_1 = \frac{1}{N}\sum_i^N x(i)$	Peak	$t_4 = Max\|x(i)\|$
Standard deviation (SD)	$t_2 = \sqrt{\frac{1}{N}\sum_i^N (x(i) - \bar{x})^2}$	Variance (Var)	$t_5 = \frac{1}{N}\sum_i^N (x(i) - \bar{x})^2$
Root Mean Square (RMS)	$t_3 = \sqrt{\frac{1}{N}\sum_i^N x(i)^2}$	Mean Square (MS)	$t_6 = \frac{1}{N}\sum_i^N \|x(i)$

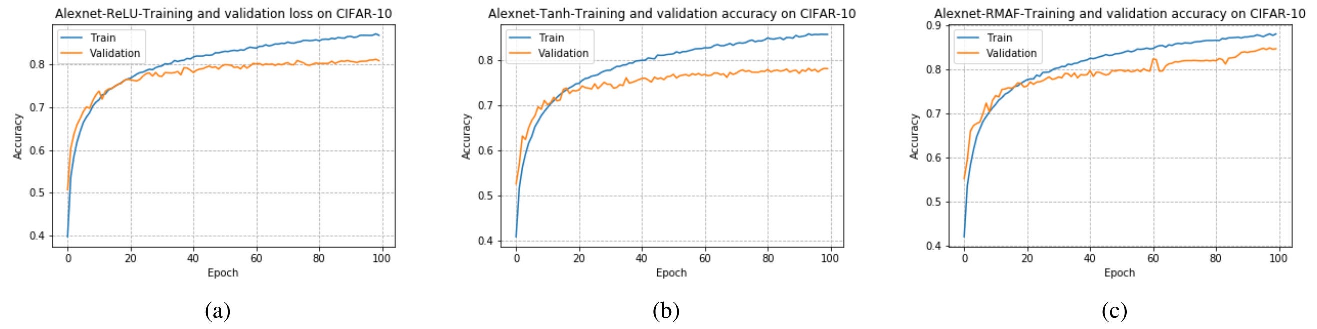


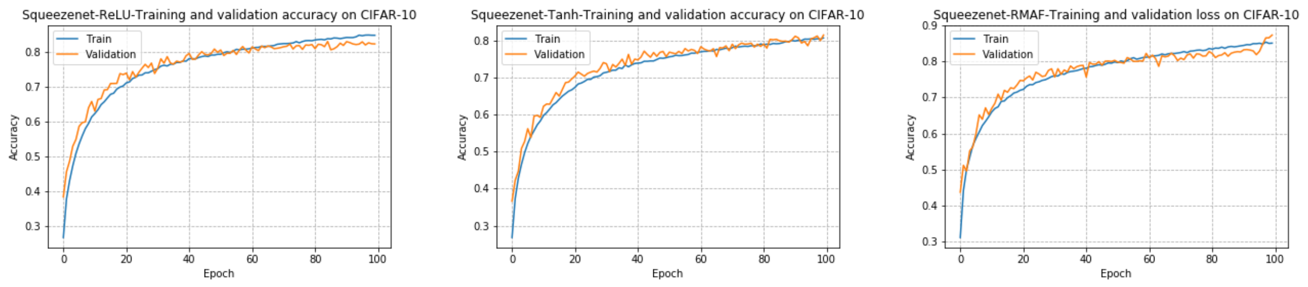
FIGURE 11. **Alexnet: Comparison of training and validation accuracy of our proposed RMAF with two baseline (ReLU and Tanh) activation functions on CIFAR100 based Alexnet.** ***(a) Shows training and validation accuracy of ReLU achieving 86.71% and 80.80% respectively, (b) Training and Validation of Tanh had 85.65% and 78.06% respectively.*** **We show that our proposed function *(c)* achieves higher performance training and validation accuracy (i.e. 94.50% and 84.58%) compared to *(a)* and *(b)* on CIFAR100 dataset. Table 3, presents other variants of ReLU performance.**

D. SQUEEZE NETWORK

In Table 3, 4, we present the accuracy of squeezeNet with RMAF and other nonlinear functions trained on CIFAR-10, CIFAR-100, MNIST and ImageNet dataset. The results indicate that, RMAF performs faster during training compared to the other transfer functions. Fig. 8 compares the performance of RMAF with ReLU, ELU, SELU, PReLU, Tanh and Swish where RMAF produced the highest accuracy. Fig. 12 presents the accuracy curves of RMAF on squeezeNet with ReLU and Tanh.

E. ImageNet CHALLENGE DATASET

In this work, we evaluated RMAF on the 1000-class ImageNet dataset. The data consist of about 1.3M training images which includes 50k images and 100k images meant for validation and testing, respectively. We performed an evaluation of the dataset and the RMAF on Resnet50, Alexnet, SqueezeNet and DenseNet121 networks. To regularize the network, the L2-weight decay term was set to 0.0005, used 40% and 60% drop-out in the two penultimate FC layers. We re-sized the images to 256×256 pixels and per-pixel mean subtracted. Training was performed with 128 × 128 random crops and random horizontal flipping. In other to achieve better regularization, we performed data augmentation during training. In Table 3, we show the learning performance accuracy of ReLU, ELU, SELU, PReLU, Tanh, Swish and RMAF on the adopted networks i.e. ResNet50, Alexnet, Squeezenet and Densenet. In the result provided in Table 3, RMAF



(a) (b) (c)

FIGURE 12. Squeeznet: Comparison of training and validation accuracy of our proposed RMAF with two baseline (ReLU and Tanh) activation functions on CIFAR100 based on Squeezenet. *(a) Shows training and validation accuracy of ReLU achieving 84.85% and 82.36% respectively, (b) Training and Validation of Tanh had 80.80% and 81.58% respectively*. We show that our proposed function *(c)* achieves higher performance training and validation accuracy (i.e. 90.50% and 86.45%) compared to *(a)* and *(b)* on CIFAR100 dataset. Table 3 presents other variants of ReLU performance.

achieved a comparable performance with the other transfer functions. The differences are small because transfer functions have minor influence on the training time (Jia, 2014). We expect to improve RMAF in further implementation, e.g. by faster exponential functions (Schraudolph, 1999).

VI. CONCLUSION AND FUTURE WORK

In this paper, a ReLU-Memristor-like activation function (RMAF) was proposed to improve classification performance of deep neural networks. The RMAF was based on the properties of memristive window function, where the function can adjust to deep networks. The RMAF consist of a constant hyperparameter α and a threshold p making the function smooth and non-monotonous in the network. In addition, the threshold parameter p introduced in the function allows for negative representations to flow through the network during forward propagation and capable of scaling to any given networks. These properties of RMAF enable networks to benefit negative representation, which achieves better performance. We investigated RMAF with state of- the-art activation functions using MLP, ResNet and over some benchmark (i.e. Winsconsin Breast cancer, MNIST, Iris, Car evaluation, CIFAR-10, CIFAR-100 and ImageNet) datasets. The performance of RMAF over MLP achieved high accuracy than Sigmoid, Tanh, ReLU and Swish. RMAF achieved better performance compared to ReLU, ELU, SELU, PReLU, Tanh, Swish over ResNet50, Alexnet, SqueezeNet, DenseNet121 and ImageNet dataset. The experimental results confirmed the effectiveness of non-linear nature of our proposed RMAF activation function, which can improve deep networks performance.

Clearly, our adopted method is subject to some limitations which will be addressed in future studies. Probably the most relevant improvement and extension of this work would be the consideration of further implementation of the learnable alpha and threshold parameters, such as the modifications of the ReLU example the learnable parameter in RelTanh [52].

ACKNOWLEDGMENT

The authors would like to thank the editor and the reviewers for their helpful suggestions and valuable comments.

REFERENCES

[1] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen, "Recent advances in convolutional neural networks," *Pattern Recognit.*, vol. 77, pp. 354–377, May 2018.

[2] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*, vol. 1. Cambridge, MA, USA: MIT Press, 2016.

[3] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, Jan. 1989.

[4] J. Tang, C. Deng, and G.-B. Huang, "Extreme learning machine for multilayer perceptron," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 4, pp. 809–821, Apr. 2016.

[5] L.-W. Kim, "DeepX: Deep learning accelerator for restricted Boltzmann machine artificial neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1441–1453, May 2018.

[6] M. Kalash, M. Rochan, N. Mohammed, N. D. B. Bruce, Y. Wang, and F. Iqbal, "Malware classification with deep convolutional neural networks," in *Proc. 9th IFIP Int. Conf. New Technol., Mobility Secur. (NTMS)*, Feb. 2018, pp. 1–5, doi: 10.1109/NTMS.2018.8328749.

[7] I. S. Krizhevsky and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, p. 770.

[9] T. Mikolov, M. Karafiát, L. Burget, J. Černocky, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. 11th Annu. Conf. Int. Speech Commun. Assoc.*, Nov. 2018, pp. 1–9.

[10] S. Wen, H. Wei, Z. Zeng, and T. Huang, "Memristive fully convolutional network: An accurate hardware image-segmentor in deep learning," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 5, pp. 324–334, Oct. 2018.

[11] X. Wang, A. Bao, Y. Cheng, and Q. Yu, "Multipath ensemble convolutional neural network," *IEEE Trans. Emerg. Topics Comput. Intell.*, pp. 1045–1048.

[12] Y. Chen, H. Chang, J. Meng, and D. Zhang, "Ensemble neural networks (ENN): A gradient-free stochastic method," *Neural Netw.*, vol. 110, pp. 170–185, Feb. 2019.

[13] M. Ravanelli, P. Brakel, M. Omologo, and Y. Bengio, "Light gated recurrent units for speech recognition," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 2, pp. 92–102, Apr. 2018.

[14] C. Zhang, X. Pan, H. Li, A. Gardiner, I. Sargent, J. Hare, and P. M. Atkinson, "A hybrid MLP-CNN classifier for very fine resolution remotely sensed image classification," *ISPRS J. Photogramm. Remote Sens.*, vol. 140, pp. 133–144, Jun. 2018, doi: 10.1016/j.isprsjprs.2017.07.014.

[15] A. Gensler, J. Henze, B. Sick, and N. Raabe, "Deep learning for solar power forecasting—An approach using AutoEncoder and LSTM neural networks," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2016, pp. 2858–2865, doi: 10.1109/SMC.2016.7844673.

[16] J.-C. Hou, S.-S. Wang, Y.-H. Lai, Y. Tsao, H.-W. Chang, and H.-M. Wang, "Audio-visual speech enhancement using multimodal deep convolutional neural networks," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 2, pp. 117–128, Apr. 2018.

[17] M. Goyal, N. D. Reeves, A. K. Davison, S. Rajbhandari, J. Spragg, and M. H. Yap, "DFUNet: Convolutional neural networks for diabetic foot ulcer classification," *IEEE Trans. Emerg. Topics Comput. Intell.*, pp. 1–12, Sep. 2018.

[18] S. H. S. Basha, S. Ghosh, K. K. Babu, S. R. Dubey, V. Pulabaigari, and S. Mukherjee, "RCCNet: An efficient convolutional neural network for histological routine colon cancer nuclei classification," 2018, *arXiv:1810.02797*. [Online]. Available: http://arxiv.org/abs/1810.02797

[19] K. Zheng, W. Q. Yan, and P. Nand, "Video dynamics detection using deep neural networks," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 3, pp. 224–234, Jun. 2018.

[20] V. K. Repala and S. R. Dubey, "Dual CNN models for unsupervised monocular depth estimation," 2018, *arXiv:1804.06324*. [Online]. Available: https://arxiv.org/abs/1804.06324

[21] C. Nagpal and S. R. Dubey, "A performance evaluation of convolutional neural networks for face anti spoofing," 2018, *arXiv:1805.04176*. [Online]. Available: http://arxiv.org/abs/1805.04176

[22] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 41–50, Feb. 2018.

[23] F. Agostinelli, M. Hoffman, P. Sadowski, and P. Baldi, "Learning activation functions to improve deep neural networks," 2014, *arXiv:1412.6830*. [Online]. Available: http://arxiv.org/abs/1412.6830

[24] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, 2013, vol. 30, no. 1, p. 3.

[25] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," 2015, *arXiv:1511.07289*. [Online]. Available: http://arxiv.org/abs/1511.07289

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, p. 1026.

[27] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Selfnormalizing neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, p. 971–980.

[28] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, p. 807–814.

[29] D. Hendrycks and K. Gimpel, "Gaussian error linear units (GELUs)," 2016, *arXiv:1606.08415*. [Online]. Available: http://arxiv.org/abs/1606.08415

[30] S. R. Dubey and S. Chakraborty, "Average biased ReLU based CNN descriptor for improved face retrieval," 2018, *arXiv:1804.02051*. [Online]. Available: http://arxiv.org/abs/1804.02051

[31] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," 2015, *arXiv:1505.00853*. [Online]. Available: http://arxiv.org/abs/1505.00853

[32] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," 2017, *arXiv:1710.05941*. [Online]. Available: http://arxiv.org/abs/1710.05941

[33] S. Scardapane, S. Van Vaerenbergh, A. Hussain, and A. Uncini, "Complex-valued neural networks with non-parametric activation functions," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 4, no. 2, pp. 140–150, Oct. 2018.

[34] S. Kumar Roy, S. Manna, S. R. Dubey, and B. B. Chaudhuri, "LiSHT: Non-parametric linearly scaled hyperbolic tangent activation function for neural networks," 2019, *arXiv:1901.05894*. [Online]. Available: http://arxiv.org/abs/1901.05894

[35] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Netw.*, vol. 107, pp. 3–11, Nov. 2018.

[36] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 453, no. 7191, pp. 80–83, May 2008.

[37] P. S. Georgiou, S. N. Yaliraki, E. M. Drakakis, and M. Barahona, "Window functions and sigmoidal behaviour of memristive systems," *Int. J. Circuit Theory Appl.*, vol. 44, no. 9, pp. 1685–1696, Sep. 2016.

[38] M. E. Turner, E. L. Bradley, K. A. Kirk, and K. M. Pruitt, "A theory of growth math," *Bioscience*, vol. 29, no. 3–4, p. 367–373, 1976.

[39] D. Fekedulegn, M. M. Siurtain, and J. Colbert, "Parameter estimation of nonlinear growth models in forestry," *Silva Fennica*, vol. 33, no. 4, pp. 327–336, 1999.

[40] A. Shukla, R. Tiwari, and P. Kaur, "Knowledge based approach for diagnosis of breast cancer," in *Proc. IEEE Int. Advance Comput. Conf.*, Mar. 2009, pp. 6–12.

[41] M. Ewertz, "Effect of obesity on prognosis after early-stage breast cancer," *J. Clin. Oncol.*, vol. 29, no. 1, p. 25–31, 2011.

[42] N. Choices, "Breast cancer symptoms," in *Proc. WebMD*, 2010, pp. 1–5.

[43] L. Zhang and P. N. Suganthan, "Random forests with ensemble of feature spaces," *Pattern Recognit.*, vol. 47, no. 10, pp. 3429–3437, Oct. 2014.

[44] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[45] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Citeseer, 2009.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, 2016, pp. 630–645.

[47] N. L. Husni, A. S. Handayani, S. Nurmaini, and I. Yani, "Odor classification using support vector machine," in *Proc. Int. Conf. Electr. Eng. Comput. Sci. (ICECOS)*, Aug. 2017, pp. 71–76, doi: 10.1109/ICECOS.2017.8167170.

[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[49] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio, "On the number of linear regions of deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2924–2932.

[50] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, Dec. 1989.

[51] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," 2015, *arXiv:1505.00853*. [Online]. Available: http://arxiv.org/abs/1505.00853

[52] X. Wang, Y. Qin, Y. Wang, S. Xiang, and H. Chen, "ReLTanh: An activation function with vanishing gradient resistance for SAE-based DNNs and its application to rotating machinery fault diagnosis," *Neurocomputing*, vol. 363, pp. 88–98, Oct. 2019.

[53] Y. Qin, X. Wang, and J. Zou, "The optimized deep belief networks with improved logistic sigmoid units and their application in fault diagnosis for planetary gearboxes of wind turbines," *IEEE Trans. Ind. Electron.*, vol. 66, no. 5, pp. 3814–3824, May 2019.

[54] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 315–323.

[55] X. Glorot, A. Bordes, J. Weston, and Y. Bengio, "A semantic matching energy function for learning with multi-relational data," 2013, *arXiv:1301.3485*. [Online]. Available: http://arxiv.org/abs/1301.3485

[56] J. Zha, H. Huang, and Y. Liu, "A novel window function for memristor model with application in programming analog circuits," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 63, no. 5, pp. 423–427, May 2016.

[57] V. Mladenov, *Advanced Memristor Modeling: Memristor Circuits and Networks*. Basel, Switzerland: MDPI, 2019.

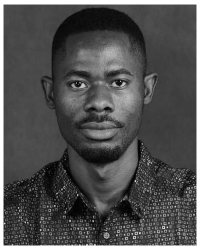
[58] D. Macêdo, C. Zanchettin, A. L. I. Oliveira, and T. Ludermir, "Enhancing batch normalized convolutional networks using displaced rectifier linear units: A systematic comparative study," *Expert Syst. Appl.*, vol. 124, pp. 271–281, Jun. 2019.

[59] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, and R. Girshich, "Caffe: Convolutional architecture for fast feture embedding," *Pattern Recognit.*, vol. 10, pp. 675–678, Nov. 2014.

[60] N. N. Schraudolph, "A fast, compact approximation of the exponential function," *Neural Comput.*, vol. 11, no. 4, pp. 853–862, May 1999.

[61] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

YONGBIN YU received the Ph.D. degree from the University of Electronic Science and Technology of China (UESTC), Chengdu, in 2008. He visited the University of Michigan, Ann Arbor, in 2013, and the University of Carlifornia, Santa Barbara, in 2016, respectively. He is currently an Associate Professor with the School of Information and Software Engineering, UESTC. His research interests include memristor-based neural networks and big data.



KWABENA ADU received the B.S. degree in information technology from the University of Education Winneba–Kumasi, in 2015, and the M.S. degree in software engineering from the University of Electronic Science and Technology of China, in 2018. He is currently pursuing the Ph.D. degree with the School of Information and Software Engineering, University of Electronic Science and Technology of China. His research interests include medical imaging, image processing, computer vision, data science, and machine learning.

NYIMA TASHI received the Ph.D. degree from Sichuan University, Chengdu, in 2009. He is currently a Professor with Tibet University. His research interests include computer networks and information systems.

PATRICK ANOKYE received the B.S. degree in information technology from the University of Education Winneba–Kumasi, in 2015. He is currently pursuing the M.S. degree with the School of Information and Software Engineering, University of Electronic Science and Technology of China. His research interests include computer vision, data science, and machine learning.

XIANGXIANG WANG (Graduate Student Member, IEEE) is currently pursuing the Ph.D. degree with the School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, China. His current research interests include complex dynamical networks, switched systems, and memristive neural networks.

MIGHTY ABRA AYIDZOE received the B.Sc. degree in computing-with-accounting from University for Development Studies–Navrongo, in 2017. She is currently pursuing the master's degree with the School of Information and Software Engineering, University of Electronic Science and Technology of China. Her research interests include deep learning and the Internet of Things.