Weighted Shifted S-shaped Activation Functions

Sergiy Popov[1], Dmytro Pikhulya[1]

[1] *Kharkiv National University of Radio Electronics, Nauky Ave. 14, Kharkiv, 61166, Ukraine*

Abstract

In this paper, we propose a family of activation functions (AFs) that can be considered as smooth approximations of bounded ReLU and similar AFs. These AFs are constructed by using a shifted origin-aligned S-shaped function as a basis, and weighing it with another S-shaped function, similar to how SiLU/GELU AFs weigh the $f(x) = x$ function. The use of both regular and adaptive variants of such AFs is explored. The performance of the proposed family of AFs is evaluated in terms of the image classification accuracy with CNN models by comparing their multiple variants with the popular existing AFs on CIFAR-10 and Fashion-MNIST datasets using Adam and stochastic gradient descent (SGD) optimizers with different learning rates. Overall, 28 variants of the proposed AFs are compared with 21 variants of popular existing AFs (including the ReLU-like functions such as ReLU, Leaky ReLU, SiLU, GELU, PReLU, Swish, etc. and some S-shaped AFs), and 6 shifted S-shaped AFs. The experiments have shown that in most cases the adaptive versions of the proposed AFs provide a pronounced image classification accuracy advantage over all existing AFs that were considered when the Adam optimizer is used, and no consistent advantage with the SGD optimizer. Further research regarding the use of these AFs with the SGD optimizer, and the use of their non-adaptive variants is required.

Keywords

Convolutional neural network, Image classification, Activation function, Bounded activation function, Activation function modifications

1. Introduction

Artificial intelligence (AI) as a field that strives to replicate different kinds of cognitive functions pertaining to humans inevitably has to deal with many kinds of computer vision (CV) tasks. The fact that CV-related tasks represent a broad part of AI tasks is a natural consequence of the fact that humans strongly rely on vision in many of their daily routines, which are in turn eventually being targeted for solution by AI. Convolutional neural networks (CNNs) are a class of artificial neural networks that proved to be very effective at solving many CV-related tasks, such as image classification, object detection, semantic segmentation, etc.

The performance of neural network models can vary depending on many factors such as the task being solved, the network's architecture being used, scale of the network, hyperparameters involved in tuning the model, etc. The choice of activation functions (AFs) is one of such hyperparameters that can significantly influence the network's capability to perform a certain task. In case of CNNs, like in many other cases, ReLU AF is a popular choice, along with other AFs that can be said as being its variations, such as Leaky ReLU [1], SiLU [2], GELU [3], ELU [4], etc. In this paper, such functions are called ReLU-like ones for convenience.

The ReLU-like functions effectively solve the vanishing gradient problem [5, 6], which is a typical problem with S-shaped AFs [6]. In many cases, using ReLU-like functions lead to better model's effectiveness for solving the image classification tasks with CNNs than the S-shaped functions like Sigmoid and Tanh [7].

It has been shown that bounding of the ReLU function can be beneficial for training stability and classification accuracy with functions like BReLU, BLReLU [8]. At the same time there are some improved variants of ReLU (LReLU, GELU, SiLU, PReLU) that can show better results, but these

ProfIT AI 2024: 4th International Workshop of IT-professionals on Artificial Intelligence (ProfIT AI 2024), September 25–27, 2024, Cambridge, MA, USA

serhii.popov@nure.ua (S. Popov); dmytro.pikhulia@nure.ua (D. Pikhulya)

0000-0002-1274-5830 (S. Popov); 0009-0003-6280-1890 (D. Pikhulya)

functions are not bounded, hence there's a potential in exploring the bounded versions of such functions to see if this could produce a cumulative improvement effect that would lead to better results than any of these functions.

The way that BReLU or similar (ReLU-6 [9]) functions are bounded makes the function to have a fixed value with zero derivative after a certain value of its argument, which could degrade the network's training process. Alternative functions, which are formally not bounded, but still limit the function's growth after a certain value of its argument as well are functions like BLReLU [8] and PLU [10]. The BReLU, BLReLU, and PLU AFs are piecewise linear functions though. In this work it is assumed that smoothing the transitions in such functions by replacing piecewise linear functions with smoother approximation functions could improve the overall network's approximation capabilities. The intuition behind this assumption is that real-world data would presumably typically be diverse enough to have no or few hard edges in distributions of most of its aspects.

Some of the functions that can be useful for creating smooth approximations of bounded ReLU-like functions are the shifted S-shaped functions. The work [11] shows that the modified version of the Tanh function, which is shifted horizontally and vertically while still maintaining an intersection with the origin, called Shifted Tanh, achieves a better performance than the Tanh function, and can show a performance that is similar or slightly higher than that of the ReLU AF.

In this paper, we take a further look at such shifted S-shaped functions by first evaluating the performance of shifted variants of the Atan and Asinh functions, and then modifying them to represent better smoothed approximations of bounded ReLU-like functions. In this paper the respective shifted AFs conventionally have the "So" prefix added to them (meaning "shifted, origin-aligned"): SoTanh (same as Shifted Tanh in [11]), SoAtan, SoAsinh. Regarding the Asinh function in particular, it is worth noting that unlike most S-shaped functions, it is an unbounded one, which could potentially be a useful property for mitigating the vanishing gradient problem as it tends to have higher first derivative values for a wider range of x than the bounded functions like Tanh and Atan.

More specifically, shifting an S-shaped function to the right is seen to potentially be beneficial due to the following reasons:

- This makes the negative function's part closer to the x axis, similar to the shape of ReLU-like functions. It is informally assumed in this work that the proximity of ReLU-like functions to 0 plays a certain role in their effectiveness with CNNs. One of the explanations might be the assumption that such AF's property encourages learning sparse representations of network's inputs [5, 11].
- This makes the function in its positive part to have a longer range where the functions value is closer to the $f(x) = x$ function, compared to a regular unshifted version of the same S-function, which also makes their shape closer to that of Bounded ReLU or BLReU AFs, while also having smooth transitions. The proximity of the positive function's part to the $f(x) = x$ function is hypothesized to contribute to preventing both vanishing gradients and exploding gradients in deep networks.

After testing the performance of SoTanh, SoAtan, and SoAsinh functions, we introduce their modified versions, which make them closer to bounded ReLU-like functions. One of the notable differences of SoTanh, SoAtan, SoAsinh functions from functions like ReLU, SiLU, GELU is that the negative part of SoTanh, SoAtan, and SoAsinh AFs is notably farther away from the x axis than ReLU/SiLU/GELU. Hence, it is hypothesized that these shifted S-shaped functions might fail to introduce the activations sparsity that can be seen with ReLU/SiLU/GELU. Thus, there could be a potential for improving their performance by "pushing" their negative part closer to the x axis. Since we strive to create smooth approximations of bounded ReLU-like functions, we choose to explore the same method of "pushing" negative part to the x axis as the one used by the SiLU and GELU functions in this work. As a result, we create shifted S-shaped functions, which are weighted by another S-shaped function that has a range of $(0;\ 1)$ and a value of 0.5 at $x = 0$. In this paper we

call such a family of functions as weighted shifted origin-aligned S-shaped functions (WSoS functions). By using two variants of weight functions and three variants of base S-shaped functions in this work we introduce and investigate the following specific WSoS AFs: SiSoTanh, SiSoAtan, SiSoAsinh, GeSoTanh, GeSoAtan, GeSoAsinh (see section 2.2).

Considering the fact that bounded functions are prone to causing the vanishing gradient problem, in this work we try to mitigate the likelihood of this problem by prolonging the range of the function in its positive range where the function has values close to the $f(x) = x$ function. We do this by introducing the scaling parameters. Besides, there's one more adjustable parameter that identifies the amount by which the base S-function is shifted along the x axis.

Finding a good combination among permutations of all AF's parameters can potentially be hard, so we first perform experiments with certain fixed parameter values, and then make these parameters to be trainable by creating adaptive versions of these AFs. In case of adaptive versions of the AFs, we share the parameters across the entire model rather than introducing different trainable parameter sets per each network's neuron.

2. Method

2.1. Shifted origin-aligned S-shaped AFs

The notion of shifted origin-aligned S-shaped functions is not new (the Shifted Tanh function was explored in [11]), but we include them into the comparison to see how the AFs proposed in this work stack up against them along with other existing AFs. Besides, in addition to the Shifted Tanh function (which is called SoTanh in this work for brevity and consistency with the proposed AFs), this work also explores the shifted versions of Atan and Asinh functions, which follow the same pattern, and evaluates the performance of their adaptive variants.

The general form of such shifted origin-aligned S-shaped functions $So(x)$ used in this work can be described with the following formula:

$$So(x) = S(x - \alpha) + S(\alpha), \tag{1}$$

where

S is an arbitrary S-shaped function, which is also called a base function in this paper, and

α is a value by which the function is shifted horizontally.

This results in the following three AFs, which represent the shifted versions of Tanh, Atan, and Asinh functions (see Table 1). Examples of such AFs can be seen in Figure 1.

Table 1
Shifted origin-aligned AFs evaluated in this work

Base function	Shifted, origin-aligned AF formula
Tanh	$SoTanh(x) = Tanh(x - \alpha) + Tanh(\alpha)$
Atan	$SoAtan(x) = Atan(x - \alpha) + Atan(\alpha)$
Asinh	$SoAsinh(x) = Asinh(x - \alpha) + Asinh(\alpha)$

2.2. Weighted shifted origin-aligned S-shaped AFs

The family of AFs proposed in this work (by convention called as WSoS AFs family in this paper) contains the modifications of shifted origin-aligned S-shaped functions (see section 2.1), whose negative part is softly pushed closer to the x axis by weighing them with another S-shaped function, as can be described in a general form by this formula:

$$WSo(x) = \gamma W(x)\beta So(\frac{1}{\beta}x), \tag{2}$$

where

γ is a function's vertical scaling parameter.

$W(x)$ is any S-shaped function in range (0; 1) symmetric with respect to the point (0, 0.5).

β is a horizontal and vertical scaling parameter for the So function.

$So(x)$ is a shifted origin-aligned S-shaped function (1).

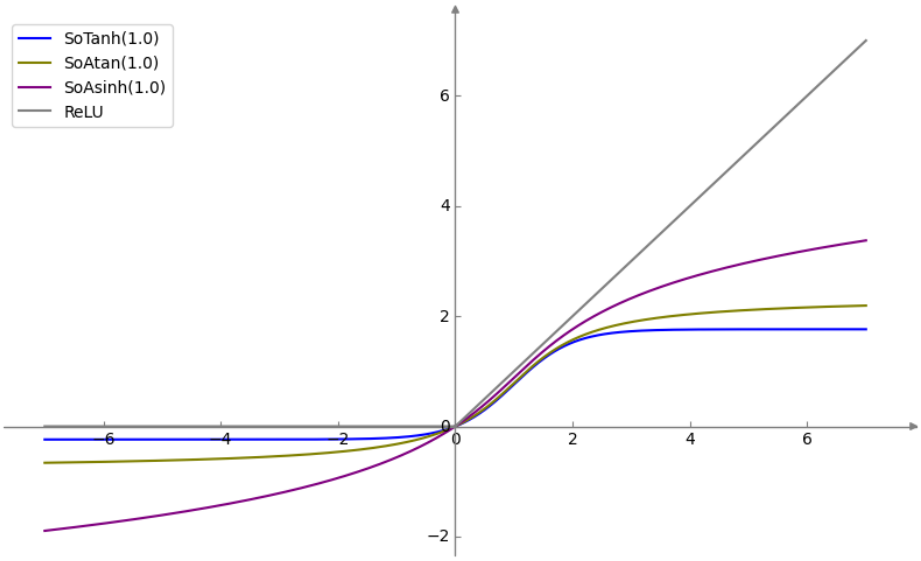

Figure 1: Shifted origin-aligned S-shaped functions with $\alpha = 1.0$

The way of weighing a shifted origin-aligned S-shaped function, which was chosen in this work, is similar to the way of weighing the $f(x) = x$ function in the SiLU and GELU AFs, where the logistic sigmoid and Gauss error functions are used respectively as the weight function $W(x)$. The respective variants used in this work, informally called Si and Ge, are shown in Table 2.

Table 2
Variants of the weight function W(x) used in this work

Weight function	Formula
Si	$Si(x) = \sigma(x) = \frac{1}{1+e^{-x}}$
Ge	$Ge(x) = \frac{1}{2}(Erf(x) + 1)$

With the three variants of shifted S-shaped functions listed in Table 1, this results in 6 specific AFs that belong to the class of WSoS AFs, which are explored in this work (see Table 3).

In an attempt to identify some concrete efficient AF variants, each of these functions is tested with several different sets of α, β, and γ parameters. Some examples of these functions with different values of α, β, γ parameters can be seen in Figure 2 and Figure 3.

Table 3
The proposed WSoS AFs

Activation function	Formula
SiSoTanh	$SiSoTanh(x) = \gamma\beta\sigma(x)\left(Tanh\left(\frac{1}{\beta}x-\alpha\right)+Tanh(\alpha)\right)$
SiSoAtan	$SiSoAtan(x) = \gamma\beta\sigma(x)\left(Atan\left(\frac{1}{\beta}x-\alpha\right)+Atan(\alpha)\right)$
SiSoAsinh	$SiSoAsinh(x) = \gamma\beta\sigma(x)\left(Asinh\left(\frac{1}{\beta}x-\alpha\right)+Asinh(\alpha)\right)$
GeSoTanh	$GeSoTanh(x) = \frac{1}{2}\gamma\beta(Erf(x)+1)\left(Tanh\left(\frac{1}{\beta}x-\alpha\right)+Tanh(\alpha)\right)$
GeSoAtan	$GeSoAtan(x) = \frac{1}{2}\gamma\beta(Erf(x)+1)\left(Atan\left(\frac{1}{\beta}x-\alpha\right)+Atan(\alpha)\right)$
GeSoAsinh	$GeSoAsinh(x) = \frac{1}{2}\gamma\beta(Erf(x)+1)\left(Asinh\left(\frac{1}{\beta}x-\alpha\right)+Asinh(\alpha)\right)$

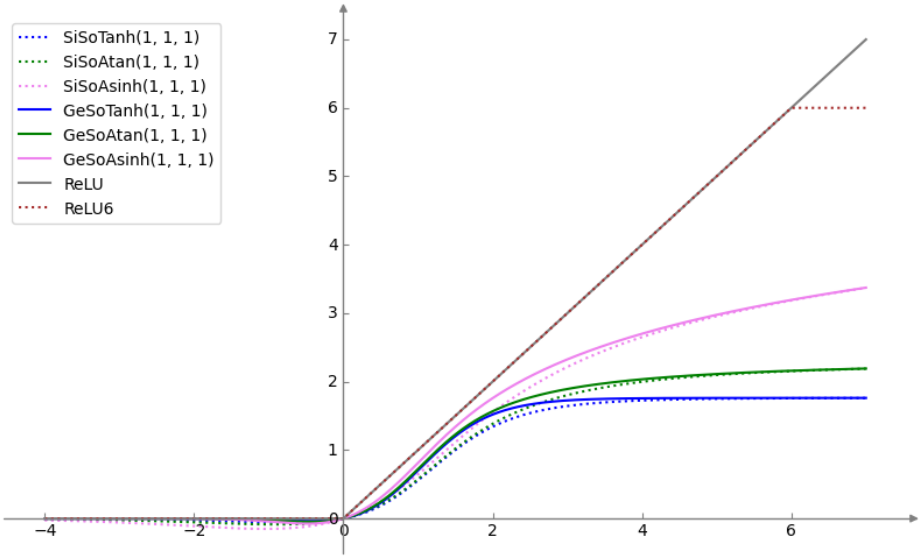

Figure 2: Weighted shifted origin-aligned S-shaped functions with α, β, γ all equal to 1.0

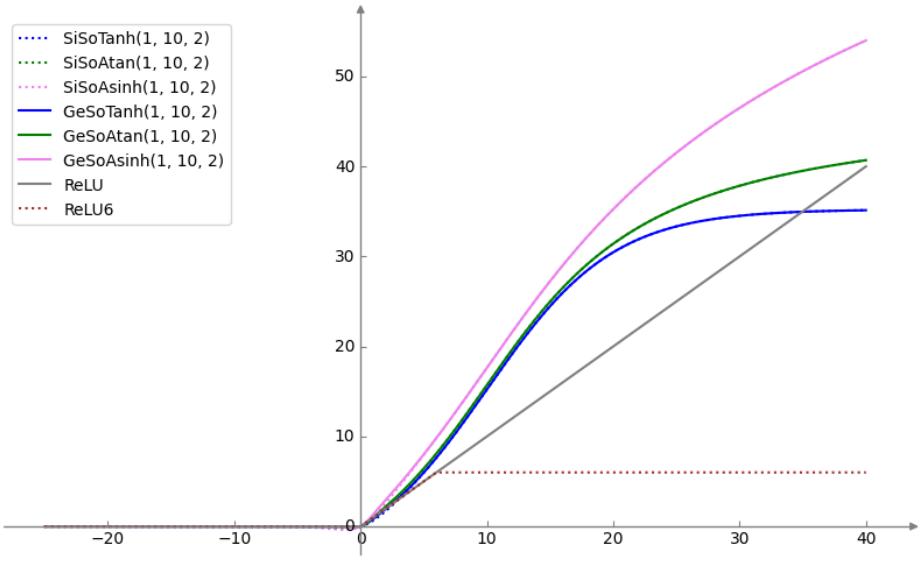

Figure 3: Weighted shifted origin-aligned S-shaped functions with $\alpha = 1, \beta = 10, \gamma = 2$

2.3. Adaptive AF variants

In addition to the functions mentioned in Table 1 and Table 3, this work considers respective adaptive variants of these AFs, which use the same AF formulas, but treat α, β, γ as trainable parameters, which are shared across the whole model. The resulting adaptive variants of shifted origin-aligned S-shaped AFs: ASoTanh, ASoAtan, ASoAsinh. The adaptive variants of the WSoS AFs (later called AWSoS functions for conciseness) are as follows: ASiSoTanh, ASiSoAtan, ASiSoAsinh, AGeSoTanh, AGeSoAtan, AGeSoAsinh.

The ASoTanh, ASoAtan, ASoAsinh functions are tested with one variant of the initial α parameter's value for each AF, and the ASiSoTanh, ASiSoAtan, ASiSoAsinh, AGeSoTanh, AGeSoAtan, AGeSoAsinh AFs are tested with several sets of initial parameter values.

2.4. Experimental setup

All activation functions are tested and compared on the image classification task with various CNN models, datasets, and hyperparameters. Below are the details on the respective experiments that are performed.

2.4.1. Activation functions being compared

In this paper, we compare the proposed activation functions belonging to the WSoS/AWSoS family with a set of existing activation functions by evaluating the average best test accuracy for each AF over several runs. The comparison includes the following functions:

Table 4
Activation functions compared in this work

Category of AFs	AFs	Sets of parameter values/ initial parameter values
The proposed WSoS and AWSoS AFs	SiSoTanh, SiSoAtan, SiSoAsinh, GeSoTanh, GeSoAtan, GeSoAsinh, ASiSoTanh, ASiSoAtan, ASiSoAsinh, AGeSoTanh, AGeSoAtan, AGeSoAsinh	$\alpha = 1, \beta = 1, \gamma = 1$
		$\alpha = 1, \beta = 10, \gamma = 2.6$
	SiSoTanh, GeSoTanh	$\alpha = 1, \beta = 1.5, \gamma = 3.64$
	SiSoAtan, GeSoAtan	$\alpha = 1, \beta = 1.2, \gamma = 3.2$
Shifted origin-aligned S-shaped AFs, and their adaptive variants	SoTanh, ASoTanh	$\alpha = 1$
	SoAtan, ASoAtan	$\alpha = 1$
	SoAsinh, ASoAsinh	$\alpha = 1$
Popular existing AFs	ReLU, ReLU-6, SiLU, GELU, ELU, Softsign, Sigmoid, Tanh, Arctan, Asinh	N/A
	Leaky ReLU	0.01
		0.1
		0.3
		0.6
	PReLU (with per-neuron trainable param.)	0
	PReLU (with trainable parameter shared across the whole model)	0.01
		0.2
		0.4
	Swish (with trainable parameter shared across the whole model)	0.33
		1
		3

All the proposed AWSoS AFs share the trainable α, β, γ parameters across the entire model in this work. This means that using an adaptive variant of the AFs add just a single set of three trainable variables to the entire model, which means that the memory footprint from using these AFs remains practically unaffected.

2.4.2. Testing metrics and configurations

Each of the functions is tested several times in each of the test configurations listed in Table 5. Within the same testing configuration, for every AF, an average image classification accuracy (as well as the standard deviation) across several test runs is evaluated for each training epoch. Only the accuracies obtained from the test dataset (not the training dataset) are used. The maximum average accuracy value that was achieved by a certain AF on any of the training epochs in certain test configuration is considered as the accuracy of this AF in this test configuration. After obtaining accuracies for each AF in each configuration, a common comparison chart for each of the configurations is made, where accuracies of all AFs can be compared with each other within the respective testing configuration.

Table 5
AF testing configurations

Configuration Name	Dataset	Model	Optimizer	Learning Rate	Batch Size	No of epochs	No of runs
CAL	CIFAR-10	CIFAR10-cls	Adam	0.001	32	30	10
CAH	CIFAR-10	CIFAR10-cls	Adam	0.002	32	30	3
CSL	CIFAR-10	CIFAR10-cls	SGD	0.03	32	30	10
CSH	CIFAR-10	CIFAR10-cls	SGD	0.06	32	30	3
FAL	Fashion-MNIST	FMNIST-cls	Adam	0.001	32	30	10
FAH	Fashion-MNIST	FMNIST-cls	Adam	0.01	32	30	3
FSL	Fashion-MNIST	FMNIST-cls	SGD	0.03	32	30	10
FSH	Fashion-MNIST	FMNIST-cls	SGD	0.3	32	30	3

In all cases, CNN kernel weights are initialized with the Glorot uniform weight initialization method, and biases are initialized with zeros.

Besides evaluating AF classification accuracies for each configuration, this work explores whether some AFs tend to have better/worse accuracy across configurations. In order to make this possible while dealing with different datasets, models, and hyperparameters, which can result in different accuracy ranges, a notion of AF accuracy rank is introduced. For any given AF, its accuracy rank $R_{AF,C}$ within a specific configuration C is defined as a 1-based index in a list of AFs sorted by their classification accuracy in an ascending order within this configuration C. Provided that all configurations are performed over the same set of AFs, for any set M of multiple test configurations $C_1 \dots C_n$, a combined rank for each AF $R_{AF,M}$ can be calculated by averaging the respective per-configuration ranks for this AF:

$$R_{AF,M} = \frac{1}{n}\sum_{i=1}^{n} R_{AF,C_i}, \tag{3}$$

In this work, combined AF ranks are calculated for three configuration combinations:
Configurations from Table 5 that use the Adam optimizer.
Configurations from Table 5 that use the stochastic gradient descent (SGD) optimizer.
All configurations from Table 5.

2.4.3. CNN models used

As can be seen in Table 5, each dataset is used with its respective model. The CIFAR-10 dataset is used with the CIFAR10-cls model (see Figure 4), and Fashion-MNIST dataset is used with the FMNIST-cls model (see Figure 5).

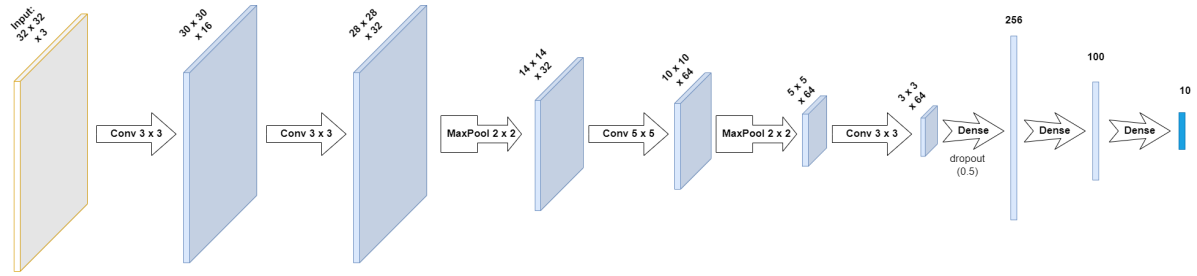

Figure 4: The CIFAR10-cls model used for CIFAR-10 image classification in this work

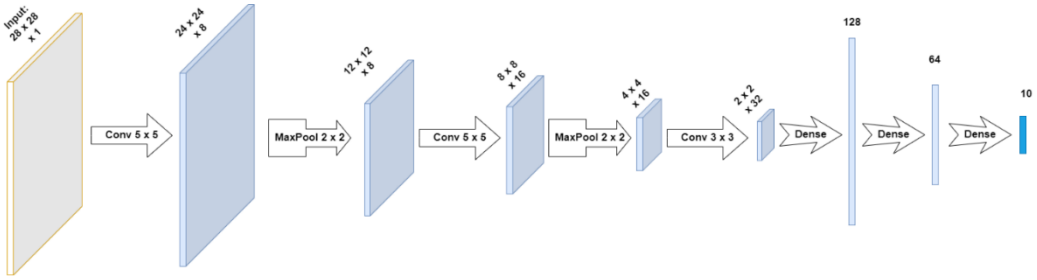

Figure 5: The FMNIST-cls model used for Fashion-MNIST image classification in this work

3. Results

The image classification accuracies that were identified in experiments for each AF in Table 4 in each of the test configurations listed in Table 5 can be seen in

Table **6** (for the proposed WSoS and AWSoS AFs) and Table 7 (for existing AFs).

The AF measurements sorted by classification accuracy for the configurations CAL and CSL in Table 5 are visualized on charts depicted in Figure 6 and Figure 7 respectively.

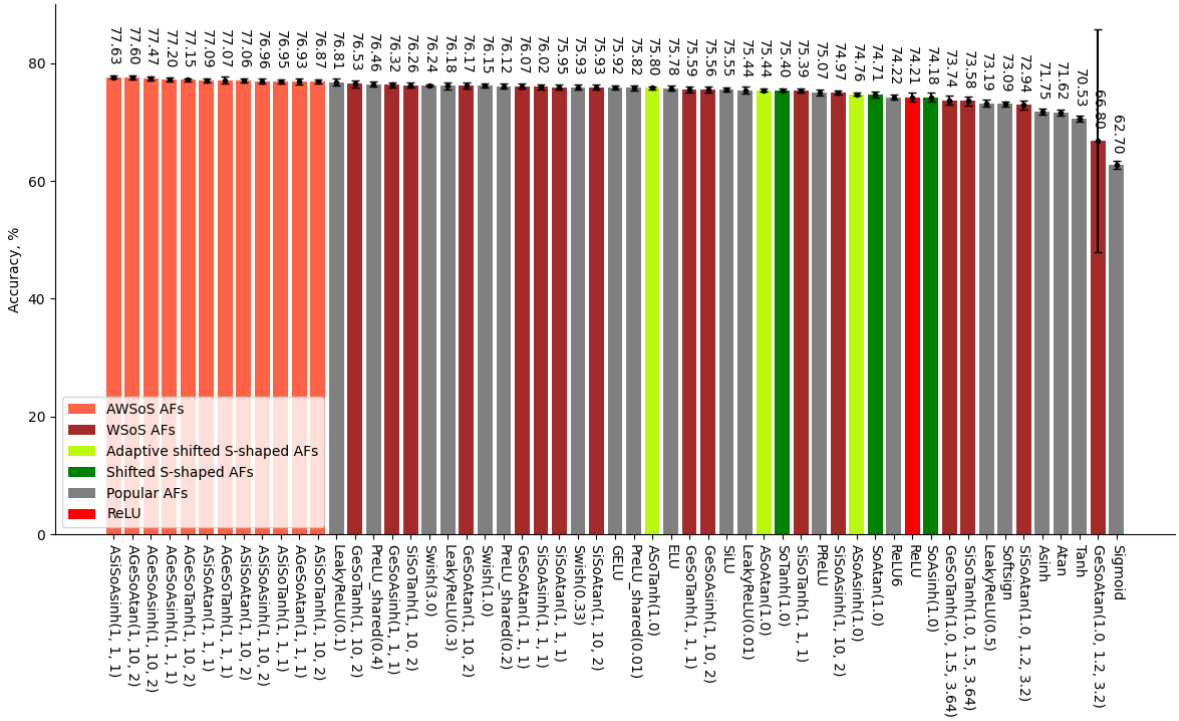

Figure 6: CIFAR-10 classification accuracies for all AFs with the Adam optimizer and learning rate of 0.001—testing configurfation CAL, demonstrates an advantage of AWSoS AFs

Table 6
Image classification accuracies for the proposed WSoS and AWSoS AFs along with std. deviations, %

AF	Dataset	CIFAR-10				Fashion-MNIST			
	Optimizer	Adam		SGD		Adam		SGD	
	Configuration	CAL	CAH	CSL	CSH	FAL	FAH	FSL	FSH
SiSoTanh(1, 1, 1)		75.39 ±0.40	75.73 ±0.43	12.19 ±6.57	54.14 ±20.47	89.81 ±0.29	84.89 ±0.64	89.13 ±0.30	87.93 ±0.67
SiSoTanh(1, 10, 2)		76.26 ±0.39	74.66 ±0.29	74.48 ±0.42	75.62 ±0.90	90.45 ±0.21	84.48 ±0.82	89.43 ±0.49	88.39 ±0.23
SiSoTanh(1.0, 1.5, 3.64)		73.58 ±0.71	56.91 ±1.94	35.59 ±31.35	10.00 ±0.00	90.07 ±0.27	77.25 ±1.86	89.52 ±0.18	10.00 ±0.00
SiSoAtan(1, 1, 1)		75.95 ±0.38	75.51 ±0.66	35.76 ±26.18	60.76 ±7.16	90.22 ±0.31	84.67 ±0.57	89.34 ±0.28	88.19 ±0.35
SiSoAtan(1, 10, 2)		75.93 ±0.45	74.14 ±0.51	74.73 ±1.05	75.40 ±0.23	90.36 ±0.23	83.24 ±0.70	89.65 ±0.33	88.25 ±0.62
SiSoAtan(1.0, 1.2, 3.2)		72.94 ±0.76	57.76 ±1.62	60.01 ±25.03	10.00 ±0.00	90.18 ±0.18	78.38 ±0.81	89.47 ±0.30	35.46 ±36.00
SiSoAsinh(1, 1, 1)		76.02 ±0.41	74.38 ±0.45	72.76 ±0.73	75.19 ±0.35	90.38 ±0.18	85.19 ±0.43	89.19 ±0.26	88.37 ±0.29
SiSoAsinh(1, 10, 2)		74.97 ±0.35	72.50 ±0.47	75.21 ±0.75	73.29 ±0.84	90.37 ±0.30	81.43 ±1.17	89.75 ±0.29	10.00 ±0.00
GeSoTanh(1, 1, 1)		75.59 ±0.52	75.97 ±0.30	41.08 ±23.24	68.89 ±1.25	89.64 ±0.19	83.86 ±0.38	89.13 ±0.45	35.91 ±36.64
GeSoTanh(1, 10, 2)		76.53 ±0.59	74.89 ±0.48	75.09 ±0.45	75.15 ±0.58	90.24 ±0.33	84.02 ±0.25	89.52 ±0.32	86.38 ±0.36
GeSoTanh(1.0, 1.5, 3.64)		73.74 ±0.74	42.08 ±22.77	10.00 ±0.00	10.00 ±0.00	89.90 ±0.24	79.54 ±0.88	89.03 ±0.36	10.00 ±0.00
GeSoAtan(1, 1, 1)		76.07 ±0.41	75.58 ±0.19	68.01 ±3.88	72.02 ±0.23	89.91 ±0.32	84.55 ±0.47	89.22 ±0.26	87.48 ±0.16
GeSoAtan(1, 10, 2)		76.17 ±0.52	74.17 ±0.66	75.19 ±0.76	75.16 ±0.35	90.20 ±0.33	83.31 ±0.72	89.51 ±0.26	58.96 ±34.63
GeSoAtan(1.0, 1.2, 3.2)		66.80 ±18.94	54.14 ±3.70	10.00 ±0.00	10.00 ±0.00	89.90 ±0.28	76.56 ±1.04	89.16 ±0.26	31.54 ±30.46
GeSoAsinh(1, 1, 1)		76.32 ±0.41	74.71 ±0.44	72.97 ±0.74	74.19 ±0.58	90.25 ±0.24	84.50 ±0.25	89.16 ±0.35	87.44 ±0.33
GeSoAsinh(1, 10, 2)		75.56 ±0.46	73.26 ±0.52	75.13 ±0.81	73.51 ±0.20	90.06 ±0.21	82.15 ±0.50	89.48 ±0.14	81.77 ±0.62
ASiSoTanh(1, 1, 1)		76.95 ±0.31	77.38 ±0.03	35.74 ±31.57	10.00 ±0.00	90.48 ±0.18	**90.18 ±0.16**	90.10 ±0.26	35.96 ±36.71
ASiSoTanh(1, 10, 2)		76.87 ±0.31	77.46 ±0.16	49.87 ±32.56	10.00 ±0.00	90.53 ±0.28	89.89 ±0.25	90.06 ±0.22	10.00 ±0.00
ASiSoAtan(1, 1, 1)		77.09 ±0.39	78.16 ±0.18	62.03 ±26.10	75.68 ±1.52	90.45 ±0.27	90.07 ±0.11	90.06 ±0.22	36.20 ±37.05
ASiSoAtan(1, 10, 2)		77.06 ±0.37	77.86 ±0.48	69.75 ±19.93	10.00 ±0.00	90.49 ±0.15	90.10 ±0.04	90.15 ±0.26	36.27 ±37.16
ASiSoAsinh(1, 10, 2)		76.96 ±0.40	77.61 ±0.29	76.37 ±0.13	32.33 ±31.58	90.51 ±0.18	90.05 ±0.16	90.07 ±0.23	10.00 ±0.00
ASiSoAsinh(1, 1, 1)		**77.63 ±0.24**	77.97 ±0.30	**77.01 ±0.27**	**76.83 ±0.40**	**90.64 ±0.17**	90.18 ±0.28	90.13 ±0.24	36.25 ±37.12
AGeSoTanh(1, 1, 1)		77.07 ±0.60	**78.19 ±0.24**	49.42 ±32.19	10.00 ±0.00	90.47 ±0.25	89.95 ±0.45	89.93 ±0.38	35.03 ±35.39
AGeSoTanh(1, 10, 2)		77.16 ±0.29	77.90 ±0.40	62.33 ±26.19	10.00 ±0.00	90.33 ±0.30	89.68 ±0.33	89.98 ±0.21	10.00 ±0.00
AGeSoAtan(1, 1, 1)		76.93 ±0.54	77.32 ±0.04	68.42 ±19.65	55.03 ±31.84	90.51 ±0.25	90.00 ±0.02	89.98 ±0.22	35.99 ±36.76
AGeSoAtan(1, 10, 2)		77.61 ±0.31	78.11 ±0.28	62.82 ±26.42	10.00 ±0.00	90.39 ±0.23	89.91 ±0.09	89.78 ±0.16	10.00 ±0.00
AGeSoAsinh(1, 10, 2)		77.47 ±0.33	77.82 ±0.30	76.00 ±0.56	32.19 ±31.39	90.47 ±0.19	90.05 ±0.36	89.73 ±0.16	35.96 ±36.71
AGeSoAsinh(1, 1, 1)		77.20	77.80	76.89	76.63	90.51	89.92	90.08	61.67

		±0.32	±0.51	±0.49	±0.10	±0.20	±0.27	±0.26	±36.53

Table 7

Image classification accuracies for existing AFs along with std. deviations, %

AF	Dataset	CIFAR-10				Fashion-MNIST			
	Optimizer	Adam		SGD		Adam		SGD	
	Configuration	CAL	CAH	CSL	CSH	FAL	FAH	FSL	FSH
SoTanh(1.0)		75.40 ±0.25	70.38 ±0.31	74.06 ±0.67	75.17 ±0.02	90.30 ±0.23	83.19 ±0.17	89.57 ±0.40	88.24 ±0.15
SoAtan(1.0)		74.71 ±0.49	70.53 ±0.62	74.65 ±0.56	75.37 ±0.02	90.45 ±0.34	83.49 ±0.36	90.10 ±0.31	88.59 ±0.24
SoAsinh(1.0)		74.18 ±0.76	71.36 ±0.65	74.50 ±0.85	74.30 ±0.21	90.21 ±0.18	84.79 ±0.44	89.99 ±0.29	61.12 ±36.15
ASoTanh(1.0)		75.80 ±0.32	75.92 ±0.46	75.48 ±0.49	75.71 ±0.44	90.46 ±0.35	86.95 ±0.06	90.20 ±0.22	87.99 ±0.24
ASoAtan(1.0)		75.44 ±0.28	75.67 ±0.26	75.86 ±0.69	75.89 ±0.15	90.39 ±0.33	89.69 ±0.15	90.14 ±0.24	88.79 ±0.17
ASoAsinh(1.0)		74.76 ±0.26	74.45 ±0.50	75.19 ±0.34	75.41 ±0.59	90.15 ±0.14	89.21 ±0.18	90.06 ±0.20	**90.16 ±0.09**
Sigmoid		62.70 ±0.69	63.72 ±1.34	10.00 ±0.00	10.00 ±0.00	88.57 ±0.33	10.00 ±0.00	37.71 ±34.03	10.00 ±0.00
Tanh		70.53 ±0.54	63.43 ±0.81	72.50 ±0.74	71.56 ±0.14	90.21 ±0.21	80.40 ±0.28	90.14 ±0.27	86.99 ±0.05
Atan		71.62 ±0.58	66.82 ±0.47	72.69 ±0.69	71.81 ±0.55	90.05 ±0.23	82.41 ±0.53	**90.21 ±0.19**	87.82 ±0.17
Asinh		71.75 ±0.52	69.54 ±0.46	71.83 ±0.51	70.44 ±0.59	89.89 ±0.24	84.83 ±0.38	89.99 ±0.21	35.53 ±36.11
Softsign		73.09 ±0.43	70.29 ±0.66	72.65 ±0.83	72.92 ±0.20	90.36 ±0.23	83.95 ±0.28	90.16 ±0.20	89.45 ±0.20
ReLU		74.21 ±0.75	70.51 ±0.35	73.67 ±1.05	73.13 ±0.41	89.94 ±0.41	83.00 ±0.36	89.32 ±0.18	85.90 ±0.15
ReLU6		74.22 ±0.40	68.15 ±0.51	74.35 ±0.53	72.68 ±0.19	90.06 ±0.26	82.47 ±1.31	89.39 ±0.27	85.09 ±0.93
LeakyReLU(0.01)		75.44 ±0.62	73.78 ±0.55	74.08 ±0.44	73.79 ±0.51	89.93 ±0.17	83.12 ±0.11	89.39 ±0.26	86.42 ±0.57
LeakyReLU(0.1)		76.81 ±0.55	75.16 ±0.53	75.77 ±0.84	75.26 ±0.36	90.27 ±0.34	82.76 ±0.81	89.60 ±0.18	61.53 ±36.43
LeakyReLU(0.3)		76.18 ±0.58	73.12 ±0.53	75.16 ±0.75	75.12 ±0.62	90.27 ±0.17	78.58 ±4.18	90.03 ±0.28	35.47 ±36.02
LeakyReLU(0.5)		73.19 ±0.62	70.21 ±0.26	72.91 ±0.89	68.11 ±1.32	90.17 ±0.17	56.23 ±32.69	90.11 ±0.19	10.00 ±0.00
PReLU		75.07 ±0.51	74.70 ±0.45	74.37 ±1.26	74.85 ±0.54	90.58 ±0.15	79.11 ±1.51	89.83 ±0.23	60.68 ±35.84
PreLU_shared(0.01)		75.83 ±0.47	73.80 ±0.96	72.47 ±0.95	55.91 ±3.33	90.28 ±0.25	82.02 ±0.21	89.94 ±0.37	10.00 ±0.00
PreLU_shared(0.2)		76.12 ±0.47	74.69 ±0.59	72.96 ±1.08	59.11 ±0.20	90.25 ±0.31	82.07 ±0.53	89.79 ±0.32	10.00 ±0.00
PreLU_shared(0.4)		76.46 ±0.38	74.91 ±0.20	73.24 ±0.80	59.25 ±3.84	90.30 ±0.32	81.74 ±0.80	89.81 ±0.28	10.00 ±0.00
SiLU		75.55 ±0.34	74.84 ±0.20	75.08 ±0.32	75.69 ±0.18	90.38 ±0.22	83.73 ±0.16	89.46 ±0.23	88.28 ±0.34
Swish(0.33)		75.93 ±0.45	74.95 ±0.77	75.16 ±0.85	76.45 ±0.01	90.31 ±0.19	83.05 ±1.04	89.69 ±0.30	88.02 ±0.32
Swish(1.0)		76.15 ±0.34	75.03 ±0.33	75.76 ±0.42	76.35 ±0.30	90.48 ±0.31	82.79 ±0.32	89.47 ±0.24	88.04 ±0.08
Swish(3.0)		76.24 ±0.20	74.93 ±0.53	75.69 ±0.64	76.30 ±0.38	90.44 ±0.25	84.29 ±0.13	89.70 ±0.22	88.01 ±0.48
GELU		75.92 ±0.34	74.46 ±0.37	75.84 ±0.67	76.14 ±0.83	90.25 ±0.24	83.80 ±0.50	89.70 ±0.29	86.69 ±0.45
ELU		75.78 ±0.48	72.28 ±0.73	76.32 ±0.74	74.99 ±0.29	90.38 ±0.16	82.24 ±1.81	90.20 ±0.25	35.34 ±35.84

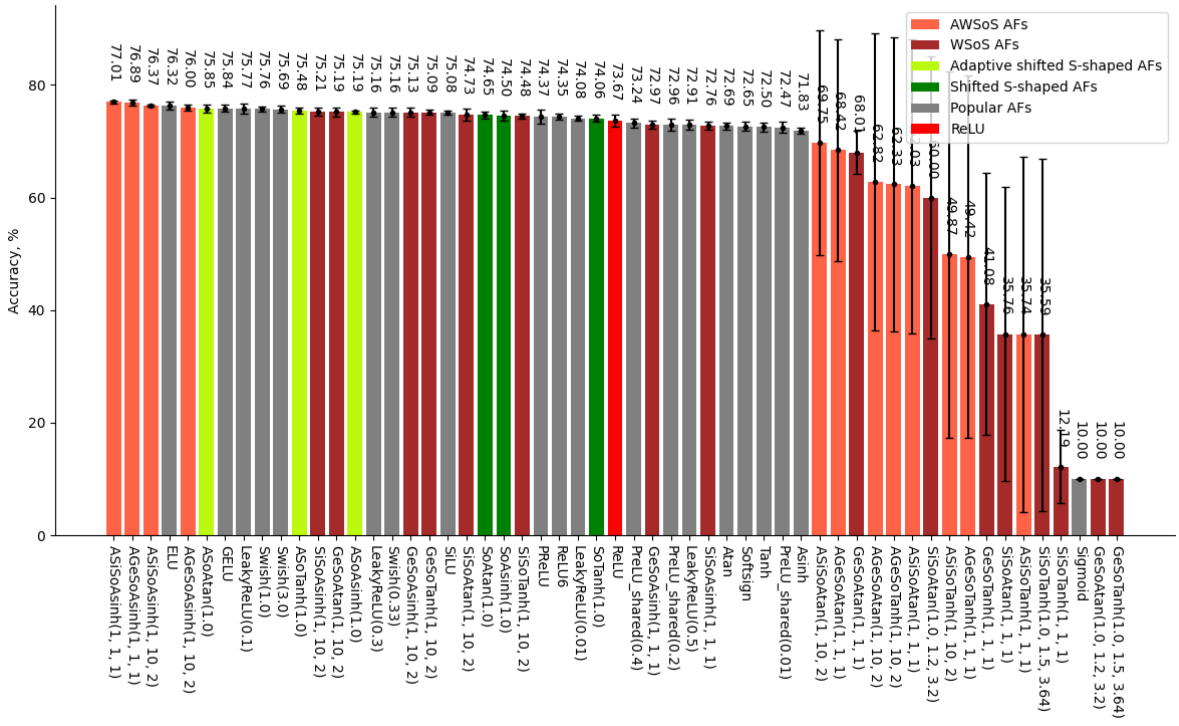

Figure 7: CIFAR-10 classification accuracies for all AFs with the SGD optimizer and learning rate of 0.03—testing configuration CSL, shows that there's no consistent advantage of AWSoS AFs with the SGD optimizer

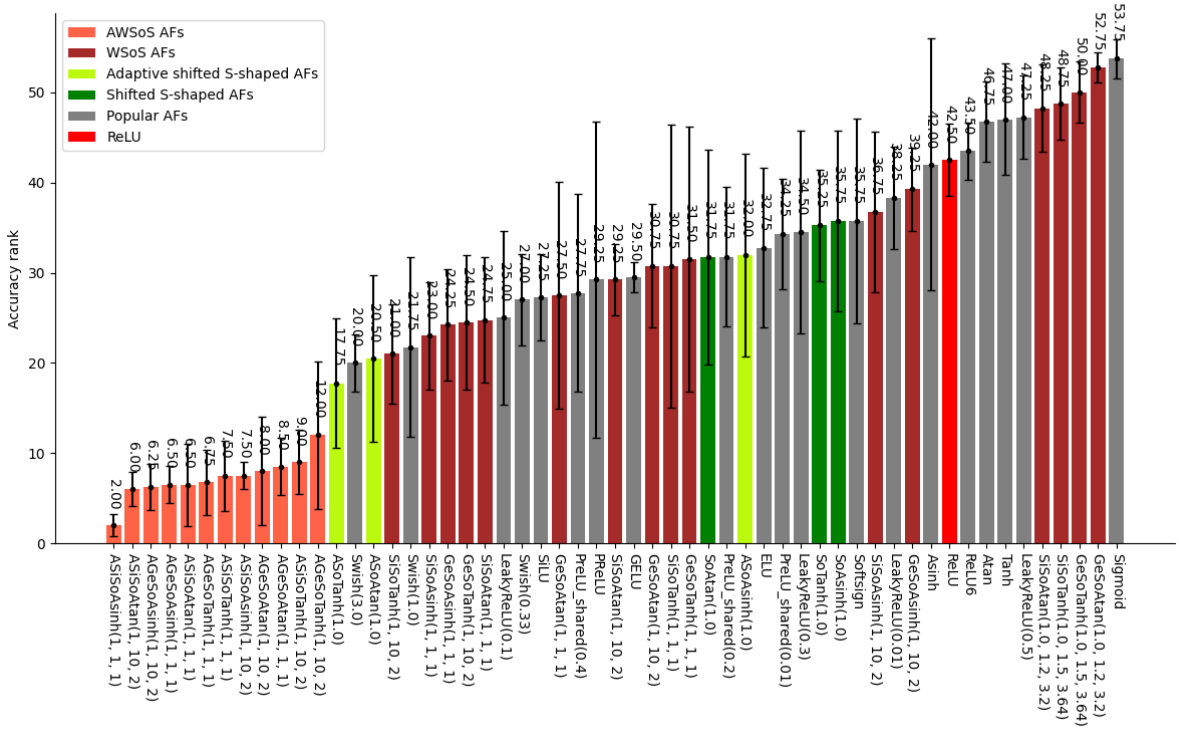

Figure 8: Combined accuracy ranks for all testing configurations using the Adam optimizer—configurations CAL, CAH, FAL, FAH (lower is better), demonstrates a pronounced superiority of AWSoS AFs over existing AFs with the Adam optimizer

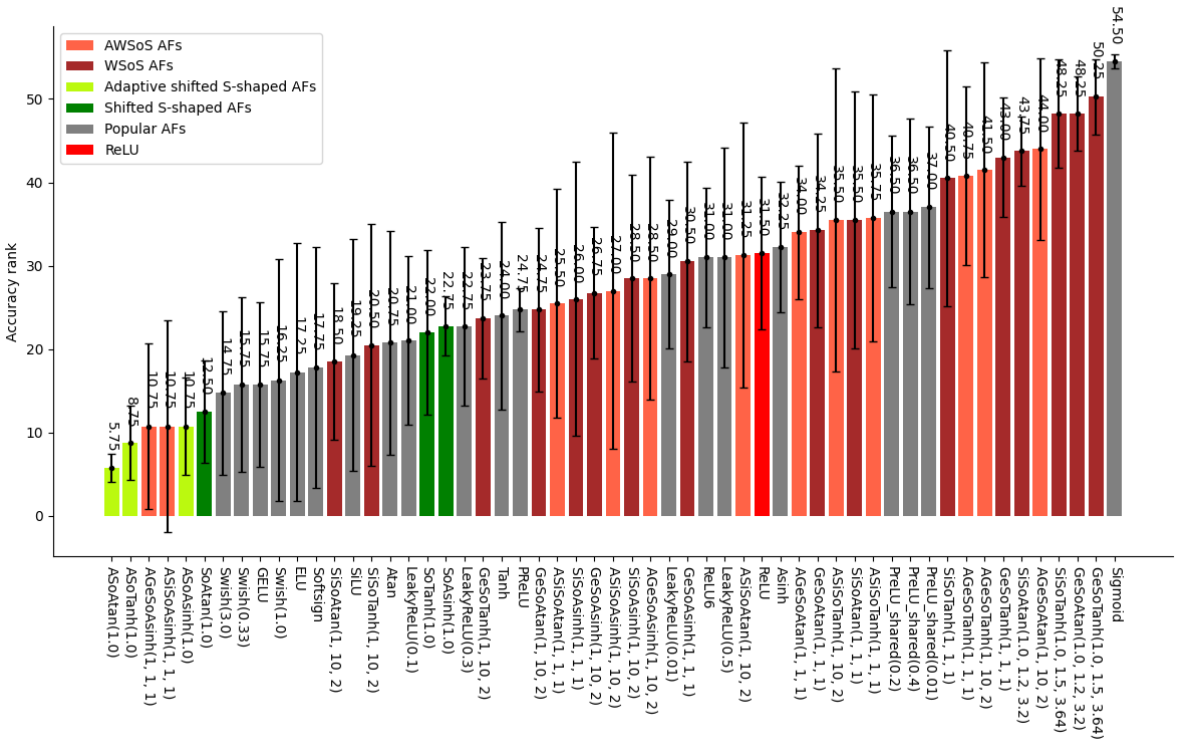

Figure 9: Combined accuracy ranks for all testing configurations using the SGD optimizer—configurations CSL, CSH, FSL, FSH (lower is better), demonstrates poor performance of WSoS AFs with the SGD optimizer

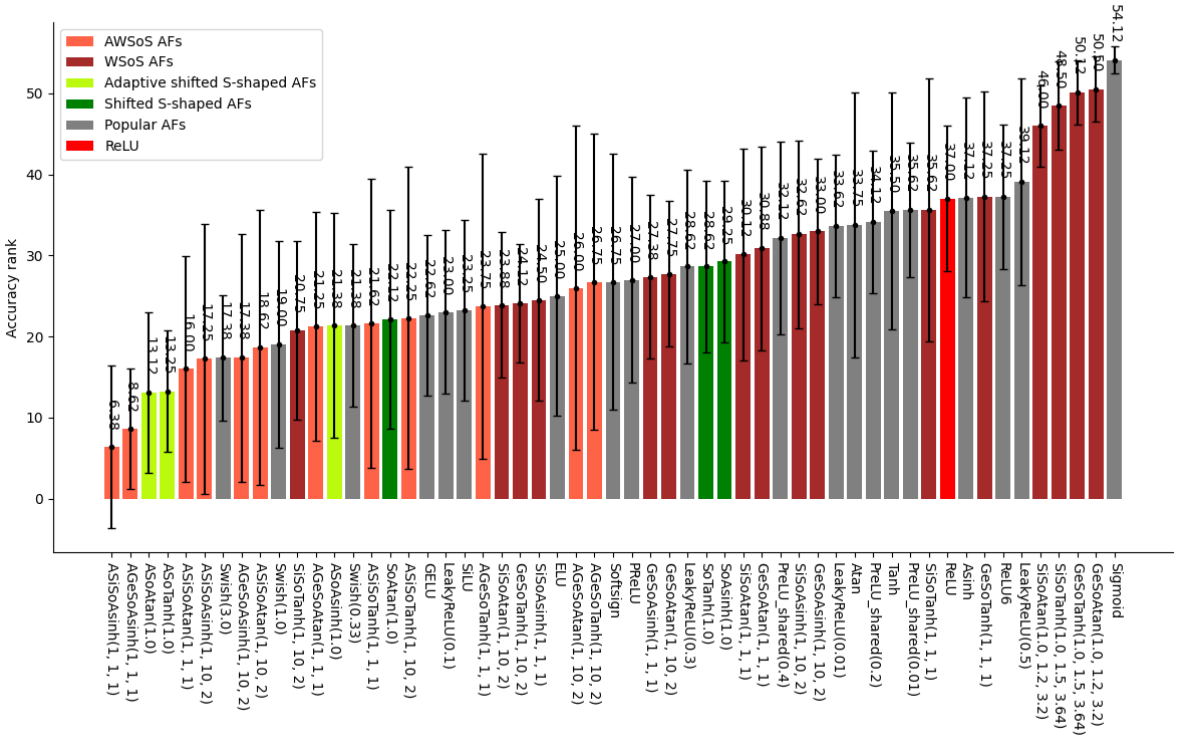

Figure 10: Combined accuracy ranks for all testing configurations in Table 5 (lower is better), shows some AWSoS and shifted S-shaped AFs that are on average better than most Afs

4. Discussion

4.1. Analysis of the results

4.1.1. The advantage of adaptive AWSoS AFs with the Adam optimizer

Overall, reviewing the results from the experiments made in this work shows that the adaptive AWSoS AFs perform notably better than all other AFs when the model is trained with the Adam optimizer. Here are the respective notes that can be made regarding such observations:

- The adaptive AWSoS AF variants have a pronounced advantage in image classification accuracy over existing popular ReLU-like AFs in all testing configurations that use the Adam optimizer. With a few of exceptions all AWSoS AFs have resulted in higher image classification accuracies than all other AFs considered in this work in all testing configurations that use the Adam optimizer. This can in particular be seen by the respective combined AF ranks in Figure 8.
- The classification accuracy advantage of the AWSoS AFs can be seen to be even more pronounced with higher learning rates when using the Adam optimizer. In the CAH testing configuration, the highest-accuracy AF AGeSoTanh(1, 1, 1) shows an accuracy of 78.19%, which is ~3% higher than the highest-accuracy standard AF LeakyReLU(0.1) of 75.16% in this configuration. In comparison, a similar configuration with a lower learning rate CAL shows a lower advantage of ~0.8% which the highest-accuracy AWSoS AF ASiSoAsinh(1, 1, 1) (77.63%) has over the highest-accuracy existing AF LeakyReLU(0.1) (76.81%). A similar tendency can be seen on models trained for Fashion-MNIST image classification with low and high learning rates.
- The choice of initial parameter values for the AWSoS AFs is seen to have no or little decisive effect with the Adam optimizer, and they consistently show higher accuracy than the existing AFs in most cases. Nevertheless, the choice of their parameter values is still important to fine tune the level of accuracy that can be achieved.
- In 6 out of 8 testing configurations the ASiSoAsinh(1, 1, 1) AF has provided a classification accuracy higher than that of all considered existing AFs. Besides, in 4 out of 8 configurations this particular AF had shown an accuracy higher than all other compared AFs.
- In the testing configurations that use the SGD optimizer, the AWSoS AFs don't have a consistent advantage over the existing AFs. Adaptive versions of these AFs are in many cases not stable in the configurations using SGD, where they often fail to converge during training. A few exceptions are the ASiSoAsinh(1, 1, 1) and AGeSoAsinh(1, 1, 1) AFs, which in three of four SGD-related configurations have provided a higher accuracy than most of the standard AFs.

4.1.2. Comparing non-adaptive WSoS functions to existing AFs

In many cases the proposed WSoS AFs provide image classification accuracy similar to the existing ReLU-like functions. Their performance is very sensitive to the choice of the α, β, γ parameter values, so they require respective attention for choosing the suitable parameter values. The results of this work don't provide sufficient data to make recommendations about the potentially more suitable parameter values and this topic requires further research.

4.1.3. Observations related to shifted S-shaped AFs

The regular (non-weighted) shifted S-shaped AFs can be seen to provide an image classification accuracy that is comparable to ReLU-like AFs and typically higher than that of the ReLU AF. This is

in line with the observations made in [11], which was exploring the Shifted Tanh function (named SoTanh in this paper). The experiments made in this work show that the other modifications of such a function, which are based on Atan and Asinh functions, can also significantly improve the classification accuracy compared to their regular unshifted variants.

The experiments also show that the adaptive versions of these AFs (ASoTanh, ASoAtan, ASoAsinh), which use the value of horizontal shift as a trainable parameter, in most cases provide an additional notable improvement in classification accuracy over the non-adaptive forms of these AFs (e.g., see Figure 6–Figure 10).

4.2. Computational performance considerations

This work primarily focuses on investigating the performance of the proposed AFs in terms of the classification accuracy in comparison with the existing ones. The analysis of the computational performance of the proposed functions, which measures the time required to train the model, and the time required to perform a forward pass when using the model in a production environment, was not the target of this work. Nevertheless, preliminary analysis confirms the intuitive assumption that using a function which requires more computations resources, like the AWSoS functions, requires more time. Preliminary measurements show that, when training on CPU, the AWSoS functions can take from ~10% more time than the Swish AF (for ASiSoTanh AF) to ~80% more time than the Swish AF (for ASiSoAsinh AF), but a more thorough study is required to identify the relative cost of using the WSoS/AWSoS AFs relative to the existing ones.

Besides, additional research is needed to evaluate the training speed of the proposed AFs in terms of the number of epochs required to reach certain accuracy, which, in combination with the assessment of the relative computational cost per one epoch, could allow a more realistic evaluation of the actual training speed that the proposed AFs can provide.

Nevertheless, the advantage that the AWSoS AFs can provide in terms of the classification accuracy can be important in some applications by itself regardless of the extra computational cost that might be required to train the model that achieves a higher performance, or use it in a production environment.

4.3. Future work

As was mentioned above, a notable tendency about the AWSoS function is their pronounced advantage over the considered existing AFs with the Adam optimizer, but a not as good performance with the SGD optimizer. This difference requires further research to try to identify ways to improve their performance with the SGD optimizer. One hypothesis that might explain this issue is that the weight initialization method used in this research (Glorot uniform) might lead the model with these AFs to poor convergence while preventing it from finding a global minimum, which is mitigated by the Adam optimizer, but not SGD.

Other directions of further research include exploring the possibility of more computationally efficient variants of WSoS/AWSoS functions, exploring whether some parameter configurations for WSoS functions can be recommended as potentially more efficient ones, and exploring how the WSoS/AWSoS functions perform in significantly deeper networks.

5. Conclusion

This work proposes a class of weighted shifted origin-aligned S-shaped activation functions (WSoS AFs) and explores their performance in image classification tasks using CNNs in comparison with a range of existing AFs. An emphasis is made on comparing the proposed AFs with ReLU-like AFs, which are the most popular choice of AFs with CNNs.

These functions are considered as an evolution of shifted origin-aligned S-shaped functions (e.g. the ones similar to Shifted Tanh in [11]), and are at the same time viewed as softly-bounded versions

of ReLU-like functions GELU and SiLU in this work. The results of experiments show that they can indeed be used to improve the classification accuracy of shifted S-shaped functions and can compete with most of ReLU-like functions, but the classification accuracy that they provide significantly depends on the choice of their three parameters, which can be challenging.

At the same time, a notable result of this work is that the adaptive versions of the WSoS AFs (AWSoS AFs) in most of the tested configurations show a clear advantage over all tested existing AFs including the existing adaptive ones, but this advantage holds only when the training is done with the Adam optimizer, and not the SGD optimizer, where the training is often not stable with these AFs.

Further research is needed to explore ways of achieving similar advantages of AWSoS AFs with the SGD optimizer, which, according to preliminary experiments, could be made with changing the weight initialization method. Besides, more computationally effective forms of WSoS/AWSoS functions can also be explored in the future research. Another line of future research would consider AWSoS AFs in combination with other learning algorithms, including their robust modifications [12], and other neural network architectures [13].

References

[1] A. L. Maas, A. Y. Hannun, A. Y. Ng, Rectifier Nonlinearities Improve Neural Network Acoustic Models, in: Proceedings of the 30th International Conference on Machine Learning, volume 28, 3.

[2] S. Elfwinga, E. Uchibea, K. Doyab, Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning, Neural Networks, volume 107, 2018 3-11. doi:10.1016/j.neunet.2017.12.012.

[3] D. Hendrycks, K. Gimpe, Gaussian Error Linear Units (GELUs), arXiv, 2016. doi:10.48550/arXiv.1606.08415.

[4] D. Clevert, T. Unterthiner, S. Hochreiter, Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs), in: International Conference on Learning Representations, 2015

[5] X. Glorot, A. Bordes, Y. Bengio, Deep Sparse Rectifier Neural Networks, in: International Conference on Artificial Intelligence and Statistics, 2011

[6] T. Szandała, Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks, in: A. K. Bhoi, P. K. Mallick, C. Liu, V. E. Balas (Eds.), Bio-inspired Neurocomputing, Lectures on Embedded Systems, Springer Singapore, 2021. doi:10.1007/978-981-15-5495-7.

[7] S. R. Dubey, S. K. Singh, B. B. Chaudhuri, Activation functions in deep learning: A comprehensive survey and benchmark, Neurocomputing, volume 503, 2022 92-108. doi: 10.1016/j.neucom.2022.06.111. doi:j.neucom.2022.06.111.

[8] S. S. Liew, M. Khalil-Hani, R. Bakhteri, Bounded activation functions for enhanced training stability of deep neural networks on visual pattern recognition problems, Neurocomputing, volume 216, 2016 718–734. doi:10.1016/j.neucom.2016.08.037.

[9] A. Krizhevsky, Convolutional Deep Belief Networks on CIFAR-10, 2012

[10] A. Nicolae, PLU: The Piecewise Linear Unit Activation Function, arXiv, 2018, doi:10.48550/arXiv.1809.09534

[11] D. Kim, W. Kim, S. Kim3, Tanh Works Better With Asymmetry, in: NIPS '23: Proceedings of the 37th International Conference on Neural Information Processing Systems, article no.: 549, 2024 12536-12554

[12] Ye. Bodyanskiy, S. Popov, M. Titov, Robust Learning Algorithm for Networks of Neuro-Fuzzy Units, in: T. Sobh (ed) Innovations and Advances in Computer Sciences and Engineering. Springer, Dordrecht, 2010. doi:10.1007/978-90-481-3658-2_59

[13] Ye. Bodyanskiy, S. Popov, T. Rybalchenko, Feedforward neural network with a specialized architecture for estimation of the temperature influence on the electric load, in Proc. 2008 4th International IEEE Conference Intelligent Systems, Varna, Bulgaria, 2008, pp. 7-14-7-18, doi:10.1109/IS.2008.4670444.