LayerAct: Advanced Activation Mechanism for Robust Inference of CNNs

Kihyuk Yoon, Chiehyeon Lim

UNIST, Ulsan, Republic of Korea
kh.yoon@unist.ac.kr, chlim@unist.ac.kr

Abstract

In this work, we propose a novel activation mechanism called LayerAct for CNNs. This approach is motivated by our theoretical and experimental analyses, which demonstrate that Layer Normalization (LN) can mitigate a limitation of existing activation functions regarding noise robustness. However, LN is known to be disadvantageous in CNNs due to its tendency to make activation outputs homogeneous. The proposed method is designed to be more robust than existing activation functions by reducing the upper bound of influence caused by input shifts without inheriting LN's limitation. We provide analyses and experiments showing that LayerAct functions exhibit superior robustness compared to ElementAct functions. Experimental results on three clean and noisy benchmark datasets for image classification tasks indicate that LayerAct functions outperform other activation functions in handling noisy datasets while achieving superior performance on clean datasets in most cases.

Code & Appendix —
https://github.com/KihyukYoon/LayerAct

Introduction

Most existing activation functions follow *element-level activation (ElementAct)* mechanisms, meaning the functions apply independently to each element of the input. However, we identified a limitation in the method: their robustness varies across samples. This is because their robustness relies on the saturation state, where the activation output for a specific range converges to a certain value. For example, in a layer with ReLU (Nair and Hinton 2010), the elements of the output remain unaffected by input shifts only when they are in the saturation state, specifically when the input and shifted input of the elements are smaller than zero. Consequently, existing activation functions can ensure robustness only when a sufficient number of elements are in the saturation state, not when there are fewer elements in that state.

We found that Layer Normalization (LN; Ba, Kiros, and Hinton (2016)) can address this limitation of ElementAct functions. Our theoretical and experimental analyses reveal that placing LN right before the activation (i.e., using the output of LN as the input to the activation) can enhance the robustness of networks. However, previous studies have reported that Convolutional Neural Networks (CNNs) with LN tend to perform poorly on clean datasets, mainly due to LN's tendency to lead to homogeneous activation. These claims are supported by our empirical results on the representation of the last layer in CNNs with LN. Therefore, our goal is to develop a method that captures the noise-robustness benefits of LN without inheriting its homogeneity limitations.

To achieve this goal, we propose a novel activation method, the Layer-level Activation (LayerAct) mechanism. Unlike the ElementAct mechanism, our proposed method utilizes layer-direction normalization for activation, but the activation output is far different from that of ElementAct with LN. This enables LayerAct to absorb LN's robustness benefit for activation while avoiding its homogeneity limitation. Additionally, LayerAct functions are not subject to the trade-off issue between two important properties of activation, one-side saturation and zero-like activation mean (Clevert, Unterthiner, and Hochreiter 2016; Qiu, Xu, and Cai 2018), which are faced by ElementAct functions (for details, see Appendix A). These two benefits of LayerAct functions, enhanced robustness and freedom from the trade-off problem, underscore their potential to outperform existing ElementAct functions on both noisy and clean datasets.

Experimental analysis with the MNIST dataset demonstrates that LayerAct functions have the following properties: (1) the mean activation of LayerAct functions is zero-like, and (2) the output fluctuation due to noisy input is smaller with these functions than with ElementAct functions. We compared the performance of the residual networks (ResNets; He et al. (2016)) with LayerAct functions to those with other ElementAct functions on three image classification tasks.

Our contributions can be summarized as follows:

- We identify a previously unrecognized limitation in existing activation mechanisms, namely a large variance of noise-robustness across samples.
- We present analysis and experimental evidence showing that while LN can address this limitation, it also causes the final layer representation in CNNs to be less distinguishable between samples with different labels.
- We introduce LayerAct that exhibit greater robustness than existing ElementAct functions, without the homo-

geneity limitation of LN.

- We empirically evaluate LayerAct functions, and they show superior performance on both clean and noisy datasets in most cases.

Backgrounds

In this section, we provide the background for the analyses presented in the remainder of this paper, including the definition of *activation scale* and a review of normalizations.

Activation Scale

In some activation functions, a function bounded between one and zero characterizes their non-linearity. We define this function, denoted as s, and its output as the *activation scale function* and *activation scale*, respectively.

Consider a layer in a multi-layer perceptron with linear projection and an activation function. Given a r-dimensional vector $x = (x_1, ..., x_r)^T$, a weight matrix $W \in \mathbb{R}^{r \times D}$, where d is the dimension of activation input (e.g., $D = C \times H \times W$ for image), and a non-linear activation function f, the activation output is $a = f(y)$, where $y = W^T x$. With an activation scale function s, the activation output and gradient of forward and backward passes are:

$$a_i = y_i s(y_i), \quad \frac{\partial a_i}{\partial y_i} = s(y_i) + y_i \frac{\partial s(y_i)}{\partial y_i}, \tag{1}$$

where s is increasing and $s(y_i) > 0$ if $y_i > 0$. For example, the activation scale function for the i^{th} element in the SiLU (Elfwing, Uchibe, and Doya 2018) is presented as follows:

$$s^{SiLU}(y_i) = sigmoid(y_i) = \frac{1}{1 + e^{-y_i}}, \tag{2}$$

where $sigmoid$ and s^{SiLU} represent the Logistic Sigmoid function, and the activation scale functions of SiLU, respectively. For further discussion, we define a specific activation scale function s.

Definition 1 (Activation Scale Function) *The activation scale function s is an increasing Lipschitz continuous function that is bounded between zero and one:*

$$s(0) = 1/2, \ |s(a) - s(b)| \leq K|a - b| \quad \forall a, b \in \mathbb{R}. \tag{3}$$

For example, the activation scale of SiLU satisfies this, as the activation scale is $sigmoid(y_i)$. Meanwhile, the activation scale also determines the saturation state of the activation functions, which occurs when $s(y_i) \simeq 0$.

In conclusion, the activation scale function plays a crucial role in providing non-linearity during the forward pass, controlling the gradient during the backward pass, and determining the saturation state of the activation function.

Revisiting Normalizations in CNNs

Normalization layers re-scale and re-center both the input and the gradient during forward and backward propagation, respectively. The normalization output n_i from the normalization input y_i is:

$$n_i = \gamma_i \cdot \frac{y_i - \mu_y}{\sigma_y} + \beta_i, \tag{4}$$

where γ_i and β_i are the learnable parameters of the affine function, D is the normalizing dimension, and μ_y and σ_y are:

$$\mu_y = \frac{1}{D}\sum_{i=1}^{D} y_i, \quad \sigma_y = \sqrt{\frac{1}{D}\sum_{i=1}^{D}(y_i - \mu_y)^2}, \tag{5}$$

the mean and variance of the input y, respectively. Normalization layers can stabilize the network training by re-scaling and re-centering as shown in Equation 4. The dimension D varies according to the normalization methods, $D = N \times H \times W$ for Batch Normalization (BN; Ioffe and Szegedy (2015)) and $D = C \times H \times W$ for LN, where N, C, H, and W denote the size of the batch size, channel, height, and width of the image dataset.

BN has achieved great success in computer vision tasks within CNN-based networks and remains dominant in the field. In contrast, LN is less preferred compared to BN, largely due to the inferior performance of networks using LN. Previous studies have attributed this underperformance to LN's tendency to produce homogeneous outputs (Lubana, Dick, and Tanaka 2021; Labatie et al. 2021).

Problems Analysis and Research Motivation

In this section, we present our motivation, specifically, we focus on: (1) an analysis of the limitations of existing activation functions, and (2) theoretical and empirical analyses of the advantages and disadvantages of LN in CNNs.

Large Variance of Activation Robustness

To analyze the robustness, we define *activation fluctuation* that can represent the influence of the shift of inputs.

Definition 2 (Activation Fluctuation) *Let $\epsilon = (\epsilon_1, \epsilon_2, ..., \epsilon_D)^T$ be the input shift vector, which is independent of the input vector y. We define activation fluctuation as $\|f(y + \epsilon) - f(y)\| \leq c$, where the constant c is the upper bound of activation fluctuation.*

The lower the upper bound c is, the lower the variance of robustness across samples. We can define the activation fluctuation of ElementAct functions.

Definition 3 (Activation Fluctuation of Element-level Activation Functions) *Let $\hat{y}_i = y_i + \epsilon_i$ be the input with a shift. The activation fluctuation of an ElementAct function f is given by:*

$$\begin{aligned} \|f(\hat{y}) - f(y)\| &= \sum_{i=1}^{D} |\hat{y}_i s(\hat{y}_i) - y_i s(y_i)| \\ &= \sum_{i=1}^{D} |y_i (s(\hat{y}_i) - s(y_i)) + \epsilon_i s(\hat{y}_i)|. \end{aligned} \tag{6}$$

A sample will exhibit a small $\|f(\hat{y}) - f(y)\|$ if a sufficient number of its elements are in a saturation state. However, ElementAct functions do not ensure that all samples have a sufficient number of elements in the saturation

Data	BN	None	LN	BN, LN	LN, BN
C10	91.29	88.51	88.24	90.65	89.73
C10-C	69.92	67.43	74.5	74.34	72.55
C10	91.45	88.29	87.53	89.74	90.17
C10-C	70.12	68.94	74.96	73.69	72.47

Table 1: Classification performance of original ResNet20 and three variants of ResNet20 with ReLU (upper) and SiLU (lower) on CIFAR10 (C10) and CIFAR10-C (C10-C).

state. More specifically, the activation fluctuation is upper-bounded when not all elements are in the saturation state, where $y_i > 0$ for all i:

$$\|f(\hat{y}) - f(y)\| \leq \sum_{i=1}^{D} \left(y_i \left|s(\hat{y}_i) - s(y_i)\right| + |\epsilon_i| \cdot s(\hat{y}_i)\right). \tag{7}$$

Considering Definition 1, the upper bound of $\|s(\hat{y}) - s(y)\|$ and $\|s(\hat{y})\|$ of ElementAct functions are given by the following, respectively:

$$\|s(\hat{y}) - s(y)\| \leq \sum_{i=1}^{D} K|\epsilon_i|, \quad \|s(\hat{y}_i)\| \leq d. \tag{8}$$

Equation 7 demonstrates that activation scale is closely related to the activation fluctuation. Samples with large $\|s(\hat{y}) - s(y)\|$ and $\|s(\hat{y})\|$ are not robust to the input shift ϵ. Thus, a method that has a lower upper bound for $\|s(\hat{*}) - s(*)\|$ and $\|s(\hat{*})\|$ will reduce the upper bound of activation fluctuation, resulting in a low variance of robustness across samples.

Empirical Analysis on Normalizations in CNNs

To empirically analyze the impact of BN and LN in CNNs, we conducted a series of experiments using the CIFAR10 (Krizhevsky 2009) dataset with the ResNet20 architecture. In the original ResNet20, BN layers are positioned immediately before the activation layers. For our experiments, we designed four variants of ResNet20: (1) eliminating BN layers entirely (ResNet20-None), (2) replacing BN layers with LN layers (ResNet20-LN), (3) adding LN layers before the BN layers (ResNet20-(LN, BN)), and (4) adding LN layers after the BN layers (ResNet20-(BN, LN)).

Table 1 demonstrates the performance of the ResNet20 variants with different normalization on both CIFAR10 and CIFAR10-C (Hendrycks and Dietterich 2019). The networks were trained on CIFAR10. The CIFAR10-C dataset serves as a noise-robustness benchmark, incorporating 19 out-of-distribution (OOD) corruptions of CIFAR10. This dataset includes a total of 19 distinct corruptions, each with five levels of severity, organized into five categories: noise, blur, digital, weather, and extra. We evaluated the models using two activation functions: ReLU and SiLU. The table reports the average mean accuracy over 30 runs, with "Norm" and "Act" denoting the normalization and activation, respectively. The experimental results show that the use of LN alone leads to inferior performance on the clean dataset, CIFAR10. Specifically, networks with LN alone showed inferior results compared to those without normalization, which aligns with findings from previous studies.

LN and other batch-free normalizations have been shown to block the negative effect of BN on robustness of networks (Huang et al. 2022). Interestingly, our experiments reveal a detailed robustness advantage of LN, which can lead to robust activation, and this has not been emphasized in previous studies. On the noisy dataset, CIFAR10-C, the networks incorporating LN demonstrated better performance than those without LN. Notably, when LN is applied immediately before the activation layers, as in ResNet20-LN and ResNet20-(BN, LN), these networks outperformed their counterparts, ResNet20-None and ResNet20-(LN, BN), on noisy datasets, despite underperforming on the clean dataset. This suggests that LN may provide a robustness benefit, which is closely related to activation.

LN and Activation Robustness

Based on the experimental result from the experimental analysis, we analyzed the relationship between LN and activation robustness. Here, we only consider LN without an affine function. In this case, the normalized output n_i, the normalized output with a shift $\hat{n}_i$, the activation output a, and activation output with shift $\hat{a}$ are as follows:

$$\begin{aligned} n_i = \frac{y_i - \mu_y}{\sqrt{\sigma_y^2 + \alpha}}, \quad \hat{n}_i = \frac{\hat{y}_i - \hat{\mu}_y}{\sqrt{\hat{\sigma}_y^2 + \alpha}}, \\ a_i = n_i s(n_i), \quad \hat{a}_i = \hat{n}_i s(\hat{n}_i), \end{aligned} \tag{9}$$

where $\hat{\mu}_y$ and $\hat{\sigma}_y^2$ are the layer-direction mean and variance of $\hat{y}$. The activation fluctuation using Equations 4 and 9 are:

Definition 4 (Activation Fluctuation of ElementAct Functions with LN) *Let ϵ_i, $\hat{y}_i$, and $\hat{n}_i$ of the i^{th} normalization layer be the shift of the input, the input with the shift, and the output from the input. The activation fluctuation of an ElementAct function f is:*

$$\begin{aligned} \|f(\hat{n}) - f(n)\| &= \sum_{i=1}^{D} \left|\hat{n}_i s(\hat{n}_i) - n_i s(n_i)\right| \\ &\lesssim \sum_{i=1}^{D} \left(|n_i| \cdot \left|s(\hat{n}_i) - s(n_i)\right|\right) \\ &\quad + \sum_{i=1}^{D} \left(\frac{|\epsilon - \mu_\epsilon| \cdot s(\hat{n}_i)}{\sqrt{\sigma_y^2 + \alpha}}\right), \end{aligned} \tag{10}$$

where $\sqrt{\sigma_y^2 + \alpha + \sigma_\epsilon^2} \approx \sqrt{\sigma_y^2 + \alpha}$ and $\sigma_y \gg \sigma_\epsilon$.

For the detailed derivation process of the equations, see Appendix B. In the previous section, we discussed that reducing the magnitudes of the terms of the activation scale effectively decreases the upper bound of activation fluctuation, thereby reducing the variance of activation robustness across samples. Considering Definitions 1 and 4, the upper

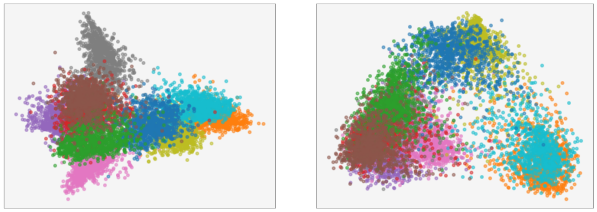

Figure 1: Plots of the last layer representation in ResNet20 with BN (left), and LN (right).

bound of $\|s(\hat{n}) - s(n)\|$ and $\|s(\hat{n})\|$ is given as follows:

$$\|s(\hat{n}) - s(n)\| < \sum_{i=1}^{D} \frac{K|\epsilon_i - \mu_\epsilon|}{\sqrt{\sigma_y^2 + \alpha}}, \quad \|s(\hat{n}_i)\| \ll d, \quad (11)$$

which are lower than those of ElementAct functions without normalization (see Equation 8), specifically considering that $\frac{|\epsilon_i - \mu_\epsilon|}{\sqrt{\sigma_y^2+\alpha}} < 1$, where $\sigma_y \gg \sigma_\epsilon$. This reveals that LN can improve activation robustness not only by re-scaling and re-centering, but also by affecting the activation scale. The experimental results in Table 1 support our analysis, demonstrating that networks with LN are much more robust compared to those without normalization.

Homogeneity Limitation of LN across Labels

It is well-known that LN produces homogeneous outputs across samples by normalizing the mean and variance of the input. However, this homogeneity extends beyond its output: LN causes activations to become similar across samples. Specifically, when LN is placed right before the activation function, producing outputs $ns(n)$, it loses the mean and variance statistics of input y. Here, we provide experimental results showing that LN produces homogeneous representations across samples, even those with different labels.

Figure 1 illustrates the last layer representation in ResNet20 with BN and LN on CIFAR10, visualized using Isomap embedding (Tenenbaum, de Silva, and Langford 2000). The figure demonstrates that the representation of the network with LN is less distinguishable between data points with different labels compared to the network with BN, which provides evidence of how LN leads to inferior performance on clean datasets. This reduced distinguishability is due to the similar output statistics of LN across samples. The variances of the output mean and variance in networks with LN are 0.0001 and 0.00072, respectively, which are substantially lower than those observed with BN (0.00073 and 0.00803, respectively). These experimental results align with previous studies, suggesting that LN's tendency to produce homogeneous outputs is a key limitation.

Summary

In this section, we presented analyses and empirical results highlighting that existing activation functions have a limitation regarding noise robustness, which LN can address. However, LN is not a preferred normalization method for CNNs due to its homogeneity limitation. Motivated by this, our goal is to develop a method that captures the robustness benefit of LN without inheriting its limitation.

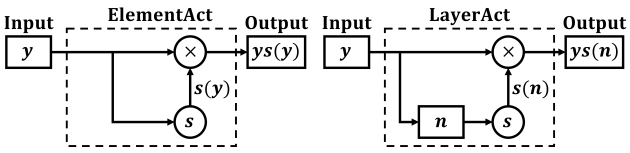

Figure 2: The mechanisms of the ElementAct (left) and proposed layer-level activation (right).

Layer-level Activation

In this section, we introduce LayerAct mechanism, a novel approach that operates differently from existing ElementAct functions, as illustrated in Figure 2.

LayerAct Mechanism

A function that follows LayerAct mechanism is defined as the product of the input y_i and the activation scale $s(n_i)$, which uses the layer-normalized input n_i. The forward and backward pass of a LayerAct function are given by:

$$a_i = y_i s(n_i), \quad n_i = \frac{(y_i - \mu_y)}{\sqrt{\sigma_y^2 + \alpha}}, \quad (12)$$

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mu_y} &= \sum_{i=1}^{D} \frac{\partial \mathcal{L}}{\partial a_i} \cdot \frac{\partial s(n_i)}{\partial n_i} \cdot \frac{-y_i}{\sqrt{\sigma_y^2 + \alpha}}, \\
\frac{\partial \mathcal{L}}{\partial \sigma_y^2} &= \sum_{i=1}^{D} \frac{\partial \mathcal{L}}{\partial a_i} \cdot \frac{\partial s(n_i)}{\partial n_i} \cdot \frac{-y_i \cdot n_i}{2(\sigma_y^2 + \alpha)}, \\
\frac{\partial \mathcal{L}}{\partial y_i} &= \frac{\partial \mathcal{L}}{\partial a_i} s(n_i) + \frac{\partial \mathcal{L}}{\partial a_i} \cdot \frac{\partial s(n_i)}{\partial n_i} \cdot \frac{y_i}{\sqrt{\sigma_y^2 + \alpha}} \\
&\quad + \frac{1}{D} \cdot \frac{\partial \mathcal{L}}{\partial \mu_y} + \frac{2(y_i - \mu_y)}{D} \cdot \frac{\partial \mathcal{L}}{\partial \sigma_y^2},
\end{aligned} \quad (13)$$

respectively, where $\alpha > 0$ is a constant for stability, and μ_y and σ_y^2 are layer-direction mean and variance, respectively.

For the stability of learning and inference, ensuring the continuity of activation outputs across the entire space is essential. While traditional ElementAct functions, such as ReLU, LReLU (Maas et al. 2013), and PReLU (He et al. 2015), do not require a continuous activation scale at zero (given that the activation output $y_i s(y_i)$ remains continuous at zero), LayerAct functions need special consideration, as the activation output $y_i s(n_i)$ is discontinuous if the activation scale function is not continuous.

Given such requirements for the activation scale functions, functions that satisfy Definition 1 are suitable for use as LayerAct functions. In this work, we propose two basic LayerAct functions, LA-SiLU and LA-HardSiLU, which utilize the Sigmoid and HardSigmoid functions as activation

scale functions, respectively. The activation outputs of these functions are as follows:

$$\begin{aligned} a^{LA\text{-}SiLU} =& \frac{y_i}{1+e^{-n_i}}, \\ a^{LA\text{-}HardSiLU} =& min\left(y_i, max\left(\frac{y_i \cdot n_i}{6}+\frac{y_i}{2}, 0\right)\right). \end{aligned} \tag{14}$$

For the HardSigmoid of LA-HardSiLU, we used the function of Howard et al. (2019), which is a good approximation of Sigmoid.

Benefits and Properties of LayerAct

Diverse Activation Output We discussed that LN tends to produce homogeneous activation outputs, as the mean and variance of n_i are similar across all samples. In contrast, LayerAct functions are designed to preserve these diverse representations by transferring the statistics of the activation input y to the activation output a. This difference is from the different activation outputs of activation with LN and LayerAct, $n_i s\left(n_i\right)$ and $y_i s\left(n_i\right)$, respectively. Due its unique output, LayerAct can maintain the input statistics within the activation outputs, resulting in much more diverse representations. For details, see Appendices C and D.

Noise-robustness of LayerAct From here, we begin by establishing that the activation fluctuation of LayerAct is also related to the two terms of the activation scale function, $\|s\left(\hat{*}\right)-s\left(*\right)\|$ and $\|s\left(\hat{*}\right)\|$, as outlined in previous sections. Subsequently, we show that these two terms for LayerAct are bound to be lower than those of ElementAct. We can define the activation fluctuation of LayerAct as follows.

Definition 5 (Activation Fluctuation of LayerAct Functions) *The activation fluctuation of a LayerAct activation function g, where $\hat{n}_i = \left(\hat{y}_i - \mu_{\hat{y}}\right)/\sigma_{\hat{y}}$ denotes the i^{th} noisy normalized input, is defined as:*

$$\begin{aligned} \|g\left(\hat{y}\right)-g\left(y\right)\| =& \sum_{i=1}^{D}\left|\hat{y}_i s\left(\hat{n}_i\right)-y_i s\left(n_i\right)\right| \\ =& \sum_{i=1}^{D}\left|y_i\left(s\left(\hat{n}_i\right)-s\left(n_i\right)\right)+\epsilon_i s\left(\hat{n}_i\right)\right|. \end{aligned} \tag{15}$$

This definition enables us to establish an upper bound for the activation fluctuation of LayerAct functions as follows:

$$\|g\left(\hat{y}\right)-g\left(y\right)\| \leq \sum_{i=1}^{D}\left(\left|y_i\right|\left|s\left(\hat{n}_i\right)-s\left(n_i\right)\right|+\left|\epsilon_i\right| s\left(\hat{n}_i\right)\right). \tag{16}$$

The terms of LayerAct's activation scale functions, $\|s\left(\hat{n}\right)-s\left(n\right)\|$ and $\|s\left(\hat{n}\right)\|$, align precisely with those of the activation with LN, as indicated in Equation 11. Comparing Equations 8 and 11 reveals that LayerAct functions have a lower upper bound of activation fluctuation, similar to activation with LN. This reduction is achieved without the direct re-centering and re-scaling of the activation output. Therefore, it suggests that networks with LayerAct are likely to exhibit improved robustness during the forward pass.

Addressing the Trade-off LayerAct can address another limitation of existing activation functions, namely a trade-off between two important properties of activation: one-side saturation and zero-like activation mean. The key distinction in saturation between the element-level and LayerAct functions is that element-level functions have a fixed saturation state at a certain point of activation output, whereas the saturation state of LayerAct functions depends on layer-dimension normalized inputs. Consequently, LayerAct functions can still reach a saturation state where $s\left(n_i\right) \simeq 0$, yet without constraining the activation output space, enabling larger negative outputs (e.g., consider a layer where $\mu_y \ll 0$). For details of the trade-off, see Appendix E.

Relationship between Normalization

LayerAct functions have unique benefits that ElementAct functions cannot achieve. Therefore, it is important to select normalization methods that preserve the diversity of statistics across samples to maximize the advantages of LayerAct functions. With this consideration, batch-direction normalizations such as BN or Decorrelated Batch Normalization (DBN; Huang et al. (2018)) are identified as the most effective choices for CNNs with LayerAct functions. Conversely, other normalization methods like LN or Switch Normalization (SwitchNorm; Luo et al. (2019)) are less favored. For the details and experiments, see Appendices G, H, and I.

Experiments

In this section, we present the experimental analysis and results of LayerAct. For the detailed experimental setting to ensure reproducibility, see Appendix F.

Experimental Analysis on MNIST

In this subsection, we empirically analyze the properties of LayerAct discussed in the previous section: (1) diverse last layer representation between samples of different labels (2) activation robustness, and (3) zero-like activation mean. We trained a network with a single layer that containing 512 elements on the MNIST training dataset without any noise to observe the behavior of LayerAct functions during training.

Ensuring Diverse Representation One of the key properties of LayerAct is that it does not produce homogeneous outputs, unlike LN, ensuring diverse representations in networks. Figure 3 shows the final layer representation of ResNet20 with LA-SiLU without normalization or BN. LayerAct has more diverse representations than ResNet20 with LN and ReLU (Figure 1), despite utilizing layer-direction normalization. This is due to the activation output, which is the product of the activation input y and the activation scale $s\left(n\right)$ (see Equation 12). This mechanism allows networks with BN and LayerAct to achieve better performance on clean datasets compared to those using LN.

Activation Robustness To confirm the robustness of LayerAct functions, we computed the activation fluctuation as defined in Definitions 3 and 5 using the network trained on the clean MNIST dataset. For the noisy input $\hat{y}_i$, we used two different noises with a normal distribution: one with a

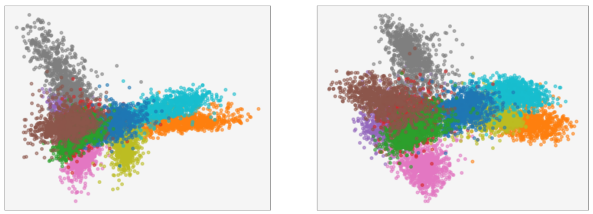

Figure 3: Plots of the last layer representation in ResNet20 with LA-SiLU, with no normalization (left) and BN (right).

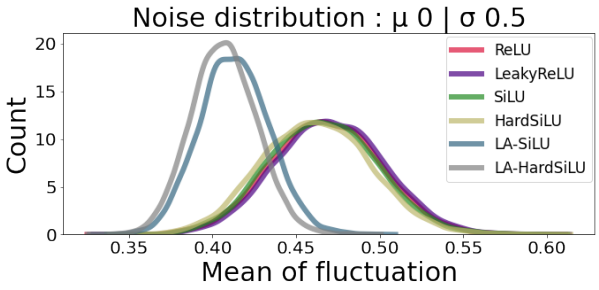

Figure 4: Distribution of activation output fluctuation.

mean of zero and a variance of 0.5^2, and another with a mean of one and a variance of 0.5^2.

Figure 4 shows the distribution of the activation fluctuation with a noise distribution. Although the fluctuation distribution of the activation input was similar (see Appendix K), LayerAct functions have a significantly smaller mean and variance of activation fluctuation among the samples than any other ElementAct function in all cases. The decrease in variance is remarkable, showing that LayerAct functions are robust for all samples. Moreover, the ElementAct functions that ensure a zero-like activation mean with one-sided saturation such as SiLU or HardSiLU showed slightly larger activation fluctuations than those of ReLU or LReLU when the noise had a large mean. However, LayerAct functions maintained lower fluctuations in both cases.

Zero-like Activation Mean Figure 5 demonstrates the distribution of the activation output means of the single-layer network trained on the MNIST dataset at the epoch 40. The distributions did not change after the epoch 40. LayerAct functions maintain a zero-like activation mean for all epochs. Our experimental results indicate that LayerAct allows similar (before epoch 20) or larger (after epoch 40) negative outputs compared to other ElementAct functions. Thus, LA-SiLU and LA-HardSiLU can achieve a more zero-like activation mean than other activation functions.

Classification Performance

We demonstrate the classification performance of LayerAct on CIFAR10, ImageNet (Russakovsky et al. 2015), and a medical image dataset. We used ResNet (He et al. 2016) as the network for our experiments on the three benchmark datasets. In our experiments on CIFAR10 and ImageNet, we utilized BN. We compared LayerAct with ReLU, LReLU, PReLU, SiLU, HardSiLU, Mish (Misra 2020), GELU (Hendrycks and Gimpel 2023), and ELU (Clevert, Unterthiner, and Hochreiter 2016). In the tables, The best results are underlined and bolded, and the second best are bolded. In the main manuscript, we present the results of ResNet20 on CIFAR10 and ResNet50 and ResNet101 on ImageNet. For more experimental results, including on CIFAR100, see Appendix J.

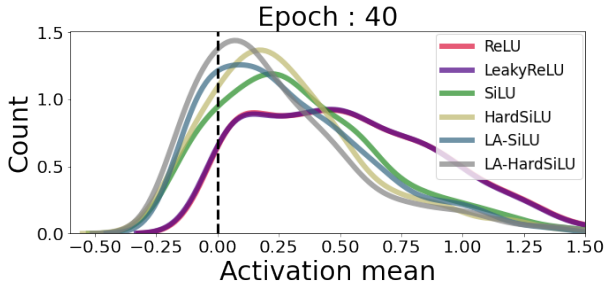

Figure 5: Distribution of the activation output means of the elements in a trained network on MNIST at epoch 40.

On CIFAR10 and CIFAR10-C The column named clean of Table 2 presents the classification performance of ResNet20 with both LayerAct functions and ElementAct functions, benchmarked on the clean CIFAR10. We report the mean accuracy over 30 runs. LA-SiLU achieved the best performance among the activation functions. In other experiments, with ResNet32 and ResNet44 on clean CIFAR10 (Appendix J), networks with GELU achieved better results in two cases: ResNet32 on CIFAR10 and ResNet44 on CIFAR100. In the remaining combinations of networks and datasets, LA-SiLU outperformed in a significant majority of cases, especially in 39 out of 48 cases according to the p-value from a T-test or a Wilcoxon signed-rank test, indicating statistical significance.

To verify the robustness of LayerAct functions, we evaluated their classification performance on CIFAR10-C. The columns other except clean in Table 2 demonstrates the experimental results. The networks with LayerAct functions achieved remarkable performance compared to those with ElementAct functions. Specifically, LA-SiLU and LA-HardSiLU statistically outperformed other ElementAct functions in 46 out of 48 cases for each, according to the p-value from a T-test or a Wilcoxon signed-rank test. This shows that LayerAct functions exhibit greater robustness to intense noise compared to other functions. Furthermore, the experimental results on ResNets without a normalization method reveal that LA-SiLU can maintain its robustness when inputs are not excessively large.

On ImageNet Table 3 shows the performance of ResNet50 and ResNet101 with LayerAct functions and other ElementAct functions for comparison with clean and noisy ImageNet datasets. For the column named "clean," we report the accuracy of 10-crop testing on the validation dataset. We

	CIFAR10	CIFAR10-C					
Activation	Clean	Total	Noise	Blur	Digital	Weather	Extra
ReLU	91.3 (0.045)	68.8 (1.735)	50.3 (5.967)	65.2 (2.294)	72.7 (0.948)	78.6 (0.523)	72.6 (1.471)
LReLU	91.3 (0.069)	68.7 (1.595)	49.9 (5.175)	65.3 (2.234)	72.7 (0.801)	78.4 (0.481)	72.5 (1.464)
PReLU	90.8 (0.054)	67.9 (2.010)	50.0 (6.180)	64.0 (1.990)	71.9 (1.700)	77.3 (0.669)	71.7 (1.787)
SiLU	**91.5** (0.042)	69.0 (1.225)	50.2 (3.131)	65.6 (1.815)	72.5 (0.763)	78.9 (0.492)	73.0 (1.015)
HardSiLU	91.1 (0.052)	68.5 (1.453)	49.7 (5.605)	65.2 (1.971)	72.0 (0.652)	78.2 (0.418)	72.8 (1.191)
MISH	91.5 (0.053)	69.0 (1.324)	50.0 (4.568)	65.7 (1.720)	72.6 (0.627)	78.9 (0.401)	72.9 (1.267)
GELU	**91.5** (0.035)	68.6 (1.290)	49.8 (3.318)	64.9 (2.311)	72.5 (0.729)	78.7 (0.276)	72.6 (1.276)
ELU	91.0 (0.030)	68.7 (1.363)	48.3 (5.193)	66.5 (1.842)	72.3 (0.605)	78.8 (0.355)	72.5 (1.220)
LA-SiLU	**91.6** (0.027)	**70.4** (1.392)	**51.8** (3.527)	**67.5** (2.155)	**74.0** (0.948)	**79.9** (0.559)	**74.4** (1.014)
LA-HardSiLU	91.2 (0.042)	**70.3** (1.384)	**52.2** (5.707)	**67.5** (1.238)	**73.5** (0.836)	**79.5** (0.501)	**74.3** (1.127)

Table 2: Classification performance of ResNet20 on the CIFAR10 and CIFAR10-C. We present mean and (variance).

		ImageNet	ImageNet-C				
Model	Activation	Clean	Total	Noise	Blur	Digital	Weather	Extra
ResNet50	ReLU	77.71	**43.75**	**34.40**	36.85	**48.37**	47.18	49.60
ResNet50	LReLU	77.83	43.24	32.87	36.33	48.00	47.03	49.38
ResNet50	PReLU	74.99	36.77	23.28	32.27	43.29	39.56	42.05
ResNet50	SiLU	77.85	42.31	29.74	35.94	46.59	47.36	48.77
ResNet50	HardSiLU	76.30	40.56	26.21	35.14	46.01	45.23	46.63
ResNet50	Mish	77.41	42.57	31.24	36.60	46.73	46.83	48.64
ResNet50	GELU	78.01	40.71	27.82	35.35	45.34	44.52	47.28
ResNet50	LA-SiLU	**78.62**	**45.29**	**36.16**	**37.66**	**50.31**	**48.33**	**51.71**
ResNet50	LA-HardSiLU	**78.24**	43.63	32.21	**37.57**	47.69	**47.88**	**49.93**
ResNet101	ReLU	**79.24**	46.7	36.29	39.75	52.16	49.93	52.78
ResNet101	LA-SiLU	79.12	**47.87**	**39.62**	**39.85**	**52.82**	**50.84**	**54.16**

Table 3: Classification performance of ResNet50 and ResNet101 on the clean and noisy ImageNet.

Activation	U-net	Unet++ w/o DSV	Unet++ w DSV
ReLU	84.71	84.94	84.92
SiLU	84.87	85.15	85.01
LA-SiLU	**85.13**	**85.27**	**85.05**

Model	ReLU		SiLU		LA-SiLU	
	Nat	Adv	Nat	Adv	Nat	Adv
RN18	81.79	82.73	83.28	82.93	**83.64**	**84.14**

Table 4: Additional experiments on medical image segmentation (upper) and adversarial robustness (lower).

used the out-of-distribution benchmark dataset, ImageNet-C (Hendrycks and Dietterich 2019), which has the same corruptions as CIFAR-C datasets. The networks with LayerAct functions outperformed those with other activation functions on all datasets. The ResNet50 with LayerAct functions, even with LA-HardSiLU which showed worse performance on the clean CIFAR10 datasets compared to SiLU or LReLU, outperformed other activation functions on clean ImageNet. Based on these results, we trained deeper networks, ResNet101, utilizing ReLU and LA-SiLU as activations. LA-SiLU maintained its superior robustness.

Additional Experiments To evaluate LayerAct's effectiveness, we conducted additional experiments on a nuclei image dataset (Goodman et al. 2018) for segmentation and on CIFAR10 to verify adversarial robustness. Table 4 shows that networks with LA-SiLU outperform those with ReLU or SiLU in both tasks. Results from segmentation tasks with U-Net (Olaf Ronneberger 2015) and UNet++ (Zhou et al. 2018) show LayerAct's potential across different architectures for image tasks. The experimental results on adversarial robustness with ResNet18 (RN18) further underscore LayerAct's robustness. For the detail, see Appendices L and M for experiments on medical image and adversarial robustness, respectively.

Conclusion

In this work, we identified the fundamental limitation of ElementAct in robustness and demonstrated that layer-direction normalization has the potential to enhance the robustness of activation. Inspired by our investigation, we have developed LayerAct mechanism and functions that not only enhance robustness compared to existing activation functions but also resolve the trade-off problem. Moreover, unlike LN, LayerAct does not reduce the performance of CNNs when utilized with BN on clean dataset. Experimental results on benchmark datasets show that LayerAct functions preserve essential activation properties and provide robust performance on both clean and noisy datasets, underscoring their utility and effectiveness in advanced CNN architectures.

Acknowledgements

This research was partly supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Education (RS-2024-00463188) and the Ministry of Science and ICT (RS-2024-00458720), as well as by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grants funded by the Korean government (MSIT) (No.RS-2024-00439932, SW Starlab; No.RS-2020-II201336, Artificial Intelligence Graduate School Program - UNIST; No.RS-2021-II212068, Artificial Intelligence Innovation Hub; No.RS-2024-00443780, Development of Foundation Models for Bioelectrical Signal Data and Validation of Their Clinical Applications: A Noise-and-Variability Robust, Generalizable Self-Supervised Learning Approach).

References

Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer Normalization. arXiv:1607.06450.

Clevert, D.-A.; Unterthiner, T.; and Hochreiter, S. 2016. Fast and accurate deep network learning by exponential linear units (elus). In *International Conference on Learning Representations (ICLR)*.

Elfwing, S.; Uchibe, E.; and Doya, K. 2018. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107: 3–11.

Goodman, A.; Carpenter, A.; Park, E.; jlefman nvidia; Josette_BoozAllen; Kyle; Maggie; Nilofer; Sedivec, P.; and Cukierski, W. 2018. 2018 Data Science Bowl. https://kaggle.com/competitions/data-science-bowl-2018. Accessed: 2024-12-17.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hendrycks, D.; and Dietterich, T. 2019. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. In *International Conference on Learning Representations (ICLR)*.

Hendrycks, D.; and Gimpel, K. 2023. Gaussian Error Linear Units (GELUs). arXiv:1606.08415.

Howard, A.; Sandler, M.; Chu, G.; Chen, L.-C.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V.; Le, Q. V.; and Adam, H. 2019. Searching for MobileNetV3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Huang, L.; Yang, D.; Lang, B.; and Deng, J. 2018. Decorrelated Batch Normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Huang, L.; Zhou, Y.; Wang, T.; Luo, J.; and Liu, X. 2022. Delving Into the Estimation Shift of Batch Normalization in a Network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 763–772.

Ioffe, S.; and Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *International Conference on Machine Learning (ICML)*, volume 37, 448–456. PMLR.

Krizhevsky, A. 2009. *Learning Multiple Layers of Features from Tiny Images*. Ph.D. thesis, University of Toronto.

Labatie, A.; Masters, D.; Eaton-Rosen, Z.; and Luschi, C. 2021. Proxy-Normalizing Activations to Match Batch Normalization while Removing Batch Dependence. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, 16990–17006. Curran Associates, Inc.

Lubana, E. S.; Dick, R.; and Tanaka, H. 2021. Beyond BatchNorm: Towards a Unified Understanding of Normalization in Deep Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, 4778–4791. Curran Associates, Inc.

Luo, P.; Zhang, R.; Ren, J.; Peng, Z.; and Li, J. 2019. Switchable normalization for learning-to-normalize deep representation. *IEEE transactions on pattern analysis and machine intelligence*, 43(2): 712–728.

Maas, A. L.; Hannun, A. Y.; Ng, A. Y.; et al. 2013. Rectifier nonlinearities improve neural network acoustic models. In *International Conference on Machine Learning (ICML)*, volume 30, 3.

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

Nair, V.; and Hinton, G. E. 2010. Rectified linear units improve restricted boltzmann machines. In *International Conference on International Conference on Machine Learning (ICML)*, 807–814.

Olaf Ronneberger, T. B., Philipp Fischer. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 234–241.

Qiu, S.; Xu, X.; and Cai, B. 2018. FReLU: Flexible Rectified Linear Units for Improving Convolutional Neural Networks. In *International Conference on Pattern Recognition (ICPR)*, 1223–1228.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3): 211–252.

Tenenbaum, J. B.; de Silva, V.; and Langford, J. C. 2000. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500): 2319–2323.

Zhou, Z.; Rahman Siddiquee, M. M.; Tajbakhsh, N.; and Liang, J. 2018. Unet++: A nested u-net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support (DLMIA)*, 3–11.