ENTROPY-BASED ACTIVATION FUNCTION OPTIMIZATION: A METHOD ON SEARCHING BETTER ACTIVATION FUNCTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

The success of artificial neural networks (ANNs) hinges greatly on the judicious selection of an activation function, introducing non-linearity into network and enabling them to model sophisticated relationships in data. However, the search of activation functions has largely relied on empirical knowledge in the past, lacking theoretical guidance, which has hindered the identification of more effective activation functions. In this work, we offer a proper solution to such issue. Firstly, we theoretically demonstrate the existence of the worst activation function with boundary conditions (WAFBC) from the perspective of information entropy. Furthermore, inspired by the Taylor expansion form of information entropy functional, we propose the Entropy-based Activation Function Optimization (EAFO) methodology. EAFO methodology presents a novel perspective for designing static activation functions in deep neural networks and the potential of dynamically optimizing activation during iterative training. Utilizing EAFO methodology, we derive a novel activation function from ReLU, known as Correction Regularized ReLU (CRReLU). Experiments conducted with vision transformer and its variants on CIFAR-10, CIFAR-100 and ImageNet-1K datasets demonstrate the superiority of CRReLU over existing corrections of ReLU. Extensive empirical studies on task of large language model (LLM) fine-tuning, CRReLU exhibits superior performance compared to GELU, suggesting its broader potential for practical applications.

1 INTRODUCTION

Flourishing development of artificial intelligence is predominantly attributable to rapid advancements in artificial neural networks (ANNs) observed in recent years. Activation functions (AFs) play a critical role in the performance of ANNs due to their fundamental role in enabling nonlinear representations. Despite continuous development of novel activation functions and their empirical success in improving network performance, theoretical analysis towards these activation functions remain scarce in the research literature. In other words, proposal of improved activation functions is often based on empirical evidence without theoretical validations, which greatly hinders the search for better activation functions. Hence, a theoretically reliable methodology on searching better activation functions holds significant value for the machine learning community and future research.

In our work, we initiate our exploration from the correlation between information entropy and Bayesian error rate. Subsequently, we establish the connection between activation function and information entropy, ultimately deriving the specific form that the worst activation function does exist under boundary conditions. Based on the derivation, we propose a novel method for optimizing activation functions, namely **the Entropy-based Activation Function Optimization(EAFO)** methodology. Utilizing EAFO methodology, we derive a novel activation function known as **Correction Regularized ReLU (CRReLU)** with the beginning of conventional ReLU (Hahnloser et al., 2000; Jarrett et al., 2009; Nair & Hinton, 2010). The derived CRReLU activation function possesses several desirable properties, including the avoidance of neuron death, the preservation of neuron sparsity, and so on. Experiments involving the vision transformer (Dosovitskiy et al., 2020) and its variants (Touvron et al., 2021; Han et al., 2021), conducted on CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009) and ImageNet-1K (Deng et al., 2009) datasets, have consistently demonstrated the superior performance of CRReLU compared to other activation function baselines. Extensive ex-

perimental studies on the task of large language model (LLM) fine-tuning with direct preference optimization (DPO) method (Rafailov et al., 2023) also demonstrate that CRReLU surpasses GELU in performance, suggesting the wider applicability of CRReLU in practical scenarios. Moreover, the EAFO methodology also shows potential to further optimize activation functions during the iterative training of ANNs, although the specific optimization techniques remain a topic of ongoing research.

In summary, our main contributions are as follows:

- We theoretically prove the existence of the worst activation function with boundary conditions from the perspective of information entropy; and starting from the worst activation function, performance of activation functions always improves.
- We propose the Entropy-based Activation Function Optimization (EAFO) methodology, which provides a novel perspective for designing static activation functions in deep neural networks and the potential of dynamically optimizing activation during iterative training.
- We derive a novel activation function known as Correction Regularized ReLU (CRReLU) starting from ReLU utilizing the EAFO methodology. Experiments across several mainstream architectures, datasets and tasks demonstrate that the proposed CRReLU exceeds existing activation functions, exhibiting exceptional performance.

2 Related Work

With the development of deep learning, deep neural networks (DNNs) have gained significant prominence and achieved notable success across various domains. Recent advancements in the field of natural language processing, exemplified by large language models such as GPT-4 (OpenAI, 2023), LLama-3 (Hugo Touvron, 2023), and Gemini (Team, 2024), have propelled machine understanding and generation of natural language to unprecedented levels of accuracy. Additionally, deep neural networks have also achieved important applications in computer vision (Dosovitskiy et al., 2020; Touvron et al., 2021; Han et al., 2021), deep reinforcement learning (Schulman et al., 2017), autonomous driving(Pan et al., 2024), and many other areas.

The nonlinearity of activation functions in neural networks is crucial for both enabling the efficient learning of complex patterns, and facilitating the extraction of intricate and hierarchical representations from input data, thus allowing them to capture more complex relationships between input and output variables. In contrast, however, the nonlinear activation functions of deep neural networks also presents challenges during training, encompassing challenges like gradient vanishing (Bengio et al., 1994), gradient exploding (Larochelle et al., 2009), and so on.

To address these challenges, researchers have explored alternative approaches for improvement, including the enhancement of activation functions. In the nascent stages of activation function development, scholars predominantly focused on rudimentary thresholding functions, initially directing their attention towards squashing functions such as the Sigmoid function and the Tanh function (Hornik, 1991). In order to mitigate the issues of vanishing and exploding gradients, various non-squashing functions have been proposed. Notably, ReLU (Hahnloser et al., 2000; Jarrett et al., 2009; Nair & Hinton, 2010) has played a pivotal role in the remarkable success of deep learning. The derivative of ReLU for positive inputs is one, thereby preventing the gradient from vanishing; however, negative values are mapped to zero, leading to two main issues: (1) The absence of information flow for negative values, known as dying ReLU ; (2) The shift in subsequent layers due to positive bias maintained by activation.

Given the aforementioned challenges, researchers have dedicated significant efforts to improving the effectiveness of activation functions. The Leaky ReLU (Maas, 2013) activation function permits a small negative slope, ensuring some gradient can still be propagated even when input is less than zero. The Parametric ReLU (PReLU) (He et al., 2015) is an extension of the Leaky ReLU, where α is considered a learnable parameter that is learned from data rather than being predetermined. The Exponential Linear Unit (ELU) (Clevert et al., 2016) outputs a negative value when x is less than 0, leading to the advantageous property of the average output approaching 0. The Continuously Differentiable Exponential Linear Unit (CELU) (Barron, 2017) proposes an alternative parameterization that simplifies analysis of the rectifier function and facilitates the tuning process of parameters in ELU. The Swish (also known as SiLU) (Ramachandran et al., 2017) has been shown to enhance training stability and performance in deep learning models due to its smooth nature and improved

gradient propagation. In Mish (Misra, 2020) activation function, unboundedness of positive values avoids the saturation led by a plateau, slight allowance for negative values enables better gradient flow, and the smoother activation function allows better information to flow deep into neural networks, thus resulting in better accuracy and generalization in performance.

3 MOTIVATION

In Section 2, it is apparent that researchers have dedicated substantial efforts to the exploration of improved activation functions, which are widely acknowledged to hold considerable significance for the advancement of deep learning. However, it has also come to our attention that proposals for these activation functions lack a theoretical framework, indicating such searches are, to some extent, inefficient and aimless.

GELU(Gaussian Error Linear Unit)(Hendrycks & Gimpel, 2023) was first proposed in 2016 and has since gained significant success in a variety of fields, especially with the emergence of large language models in recent years. It has been successfully incorporated into several cutting-edge neural network architectures, such as BERT(Devlin et al., 2019) , ViT (Dosovitskiy et al., 2020) , GPT-4(OpenAI, 2023) , and so on, demonstrating its versatility and effectiveness. In the work conducted by Lee (2023) (2023), insightful mathematical properties of the GELU are finally unveiled, including its differentiability, boundedness, stationarity, and smoothness. Hence, it is often the case that superior performance exhibited by novel activation functions frequently lacks mathematical explanations for their observed enhancements. Understanding may merely limited to the fact that it exhibits improved performance, which hampers exploration for better activation functions and interpretability of neural networks.

In light of the aforementioned challenges, our work endeavors to propose a methodology for searching better activation functions, not only enabling the discovery of improved activation functions but also elucidating the reasons behind their superior performance at the same time.

4 METHODOLOGY

4.1 PROBLEM SETUP

4.1.1 BAYESIAN ERROR RATE AND INFORMATION ENTROPY

A deep neural network can be simplified as comprising a feature extraction layer, which is subsequently followed by a fully connected layer for final classification. From a probabilistic perspective, in binary classification, the feature extraction layer can be conceptualized as transforming the shape of mixture distribution, thereby enabling the final fully connected layer to separate two distributions with a hyperplane. Hence, the more overlapping two distributions are, the higher Bayesian error rate and the worse classification performance. Furthermore, a lower information entropy corresponds to a higher likelihood of forming two distinct peaks (i.e. the smaller classification uncertainty, the easier to classify); and an increase in the overlap between two distributions also leads to the increase of information entropy (i.e. the greater classification uncertainty, the harder to classify). In addition, the above statements can be extended to multi-class classification, and further elaboration is omitted here.

4.1.2 ACTIVATION FUNCTION AND INFORMATION ENTROPY

Assuming the inverse function of the activation function is $y(x)$, and the activation function is monotonically increasing. Many previous activation functions, such as Sigmoid and Tanh (Hornik, 1991), satisfy the assumption that the function has an inverse function in entire definition domain. Furthermore, when an activation function fails to meet the assumption, we can transform the part of such function satisfying this assumption, as is the case with the positive part of ReLU.

Then we set data distribution before passing through the activation function obeys the distribution $p(x)$. Thus, data distribution after passing through activation function is : $q(x) = p\left(y(x)\right)y'(x)$, where $y'(x)$ represents the derivative of $y(x)$. Hence, we can express the information entropy as:

$$\mathbb{H}(y(x)) = -\int q(x)\log q(x)\mathrm{d}x = -\int p(y(x))y'(x)\log(p(y(x))y'(x))\mathrm{d}x = \int \mathbb{G}(y'(x), y(x))\mathrm{d}x$$

Therefore, the information entropy can be deemed as a **functional**, which takes a function $y(x)$ as input and produces a real number as output.

4.2 Worst Activation Function with Boundary Condition (WAFBC)

Firstly, we would like to determine the extremum (whether it is a maximum or minimum) of the functional $\mathbb{H}(y(x))$. For further deductions, taking the simplest functional into consideration, e.g. setting $\mathbb{H}(y(x)) = \int \mathbb{G}\left(y'(x), y(x), x\right) \mathrm{d}x$.

In order to research the influence brought by variations of function $y(x)$, we apply a small perturbation $\varepsilon\eta(x)$ to function $y(x)$, and then the functional $\mathbb{H}\left(y(x)+\varepsilon\eta(x)\right)$ takes the form as:

$$\mathbb{H}\left(y(x)+\varepsilon\eta(x)\right) = \int \mathbb{G}(y'(x)+\varepsilon\eta'(x), y(x)+\varepsilon\eta(x), x)\mathrm{d}x$$

We apply **Taylor expansion** to functional $\mathbb{H}\left(y(x)+\varepsilon\eta(x)\right)$, we can obtain the following equation:

$$\begin{aligned}\mathbb{H}\left(y(x)+\varepsilon\eta(x)\right) &= \int \left[\mathbb{G}(y'(x), y(x), x) + \varepsilon\frac{\partial\mathbb{G}}{\partial y'}\eta'(x) + \varepsilon\frac{\partial\mathbb{G}}{\partial y}\eta(x) + \mathcal{O}(\varepsilon)\right]\mathrm{d}x \\ &= \mathbb{H}(y(x)) + \varepsilon\int\left[\frac{\partial\mathbb{G}}{\partial y}\eta(x) + \frac{\partial\mathbb{G}}{\partial y'}\eta'(x)\right]\mathrm{d}x + \mathcal{O}(\varepsilon)\end{aligned} \tag{1}$$

As illustrated in Section 4.1.2, $q(x) = p\left(y(x)\right)y'(x)$ is the data distribution after passing through activation function. We can easily get that for the inverse function $y(x)$ of activation function, when x approaches the lower bound (e.g. the initial activation function value approaches lower bound), $y(x)$ should approaches negative infinity; and when x approaches the upper bound (e.g. the initial activation function value approaches upper bound), $y(x)$ should approaches positive infinity. And for the sake that $\varepsilon\eta(x)$ is a small perturbation applied to $y(x)$, we can draw the conclusion that $\eta(x)$ must be 0 at the boundaries.

Utilizing the method of integration by parts and boundary condition towards Equation 1, we can derive the following results:

$$\int\frac{\partial\mathbb{G}}{\partial y'}\eta'(x)\mathrm{d}x = \int\frac{\partial\mathbb{G}}{\partial y'}\mathrm{d}\eta(x) = \eta(x)\frac{\partial\mathbb{G}}{\partial y'}\bigg|_x - \int\eta(x)\frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial\mathbb{G}}{\partial y'}\right)\mathrm{d}x = -\int\eta(x)\frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial\mathbb{G}}{\partial y'}\right)\mathrm{d}x$$

Thus, $\mathbb{H}\left(y(x)+\varepsilon\eta(x)\right)$ has the following expression:

$$\mathbb{H}\left(y(x)+\varepsilon\eta(x)\right) = \mathbb{H}(y(x)) + \varepsilon\int\left[\frac{\partial\mathbb{G}}{\partial y} - \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial\mathbb{G}}{\partial y'}\right)\right]\eta(x)\mathrm{d}x + \mathcal{O}(\varepsilon)$$

In analogy to the extremum of ordinary functions, it is expected that the first-order term should be 0 at the extremum point. Such requirement for arbitrary $\eta(x)$ leads to the Euler-Lagrange equation:

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial\mathbb{G}}{\partial y'}\right) - \frac{\partial\mathbb{G}}{\partial y} = 0 \tag{2}$$

Proposition 1. *If $\mathbb{G}$ is independent of x, i.e. $\mathbb{G} = \mathbb{G}(y, y')$, based on the Euler-Lagrange equation expressed in Equation 2, then we have:*

$$\mathbb{G} - y'\frac{\partial\mathbb{G}}{\partial y'} = C \tag{3}$$

Detailed proof of Proposition 1 can be seen in Appendix A.

Substitute $\mathbb{G} = -p(y(x))y'(x)\log(p(y(x))y'(x))$ into Equation 3 and perform the calculation, the final result is:

$$\frac{\mathrm{d}y}{\mathrm{d}x}p(y(x)) = C$$

Integrating both sides of the equation simultaneously, the final solution is:

$$x = c_1\int p(y)\mathrm{d}y + c_2 \tag{4}$$

Based on the solution we get in Equation 4, for the sake that $y(x)$ is the inverse function of the activation function, the first integral equation can finally be solved to obtain the form of the activation function as:

$$f(x) = C_1 \int_{-\infty}^{x} p(t)\mathrm{d}t + C_2, \tag{5}$$

where C_1 and C_2 are two constants based on the upper bound and lower bound of activation function.

Equation 5 shows the analytical form of the worst activation function with boundary condition. We provide further discussion on this form in Appendix B. Through the above derivation, extremum of the functional is determined. Furthermore, we would like to deduce whether it is a maximum value or a minimum one. Applying Legendre condition to the functional extremum, then we have:

$$\mathbb{G}_{y'y'} = -\frac{p(y(x))}{y'} \leqslant 0$$

Therefore, the derived extremum is a maximum extremum, and is a global maximum extremum actually, meaning the deduced activation function has the worst performance. Actually, the WAFBC possesses some intriguing properties, for example, it inherently has upper and lower bounds, which can explain why bounded activation functions like Sigmoid and Tanh do not perform as well as unbounded functions like ReLU.

4.3 Entropy-based Activation Function Optimization (EAFO)

In Section 4.2, we have derived the extremum of the functional, showing the analytical form in Equation 5. However, the solution obtained is the global maximum, rather than the minimum. The minimum of the functional is needed if we would like to obtain the best activation function. Nonetheless, based on calculation, the actual situation is that this functional only has a global maximum but no global minimum exists. **Hence, there is no best activation function, but only better activation functions.** In this scenario, WAFBC represents a global maximum of the functional, indicating that the performance of activation functions consistently improves from WAFBC to any alternative activation functions. Therefore, we propose the following question: Is there a methodology to begin with an existing, high-performing activation function, and subsequently develop an activation function with superior performance?

Let's reconsider the Taylor expansion of the functional

$$\mathbb{H}(y(x) + \varepsilon\eta(x)) = \mathbb{H}(y(x)) + \varepsilon \int \left[\frac{\partial \mathbb{G}}{\partial y} - \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial \mathbb{G}}{\partial y'}\right) \right] \eta(x)\mathrm{d}x + \mathcal{O}(\varepsilon)$$

To minimize the information entropy of novel activation function, it is advisable to reduce the first-order term of Taylor expansion. In order to ensure that the information entropy of novel activation function has been indeed reduced, we would like to set $\eta(x)$ as the opposite sign to $\frac{\partial \mathbb{G}}{\partial y} - \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial \mathbb{G}}{\partial y'}\right)$, which means we set:

$$\eta(x) = -\left(\frac{\partial \mathbb{G}}{\partial y} - \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial \mathbb{G}}{\partial y'}\right) \right) \tag{6}$$

Substitute the analytical form of functional $\mathbb{G}(y'(x), y(x))$ into Equation 6, perform the calculation, we can derive the following equation:

$$\eta(x) = -\left(p(y(x))\frac{y''(x)}{y'(x)} + p'(y(x))y'(x) \right), \tag{7}$$

where $p(x)$ is the probability density function (PDF) of data distribution before passing through the activation function; $p'(x)$ is the first order derivative of PDF; $y(x)$ is inverse function of the activation function; $y'(x)$ is the first order derivative of $y(x)$; $y''(x)$ is the second order derivative of $y(x)$.

As a result, we have derived a correction term that is capable of decreasing information entropy, expressing its general form in Equation 7. Subsequently, we can obtain the inverse function of the optimized activation function, denoted as $g(x) = y(x) + \eta(x)$. Finally, the optimized activation function can be obtained by deriving the inverse function of $g(x)$.

EAFO methodology outline . In summary, we express the theoretical EAFO methodology as follows: 1) Utilize Equation 7 and derive correction term $\eta(x)$ given data distribution $p(y)$ and inverse function

of activation function $y(x)$. 2) Sum the correction term with the inverse function to obtain the inverse function of the optimized function, i.e. $g(x) = y(x) + \eta(x)$. 3) Derive the rigorous or approximate inverse function of $g(x)$, yielding the optimized activation function.

Furthermore, EAFO methodology has also shown the potential of dynamically optimizing activation during iterative training. We are acknowledged that activation of neural networks with Multi-Layer Perceptrons (MLPs) architecture is typically fixed. Recent studies, such as work done by Liu et al. (2024), have suggested the optimization of activation in innovative network architectures (Kolmogorov-Arnold Networks). Furthermore, across true data distributions $p(y)$, utilizing EAFO methodology, we may continuously optimize activation $y(x)$ practically under Multi-Layer Perceptrons (MLPs) architecture with numerical methods. Moreover, in theory, it is feasible to optimize activation functions using methods such as gradient descent optimization of the information entropy functional through numerical methods; however, we are also aware that this would result in an explosion of computational complexity in large neural networks, which calls for practically efficient algorithms. Hence, the EAFO methodology is still in the theoretical stage presently, providing guidance for calculating the analytical form of better activation functions.

4.4 Correction Regularized ReLU (CRReLU) : From ReLU to Better

As illustrated in Section 4.2, it is theoretically true that the worst activation function exists, and we can determine its exact form. Actually, beginning with the worst activation function, the value of the functional $\mathbb{G}$ consistently decreases, indicating an improvement in the performance of activation function. This reveals the feasibility of searching an improved activation function, which constitutes the crux of "optimization". In Section 4.3, EAFO is proposed as the optimization methodology. Hence, we can easily think of optimizing from WAFBC to get a better-performing activation function. While it is true that such an idea is feasible, we also observe that WAFBC itself takes the form of a variable upper bound integral, which yields a complex form of $\eta(x)$ and renders the deduced result not practically significant. Moreover, commencing optimization from WAFBC also leads to sluggish advancement. Therefore, in practical applications, we are inclined to start from an activation function that already demonstrates relatively good performance.

Here, we would like to take ReLU (Hahnloser et al., 2000; Jarrett et al., 2009; Nair & Hinton, 2010) as the beginning, and show the process of finding a better activation function. Before the deduction, we also notice that ReLU is lack of an inverse function over the entire domain. In this section, we would like to utilize following strategies for mitigating the aforementioned dilemma: the initial activation function only necessitates an inverse function in specific regions where it is required; and when encountering parts without an inverse function, we may employ practical approximations. Therefore, we initially examine the region where x is positive in the case of ReLU. As shown in Equation 7, the derivation of correction term $\eta(x)$ only requires original distribution $p(y)$ and inverse function of the activation function $y(x)$. Knowledge of activation function is easily available, whereas original distribution remains unexplored. However, in real experiments, original distribution of experimental data would surely exhibit a substantial degree of morphological variability, thus lacking a perfect analytical form. Hence, we assume the situation is that networks are large enough, according to the Central Limit Theorem, the data processed by them can be approximated as a Gaussian distribution (Williams, 1996; Lee et al., 2018; Park et al., 2020; Gao et al., 2023)(Huang et al., 2021). Certainly, such assumption may not always hold in networks of real experiments; nevertheless, approximation of the exact solution for inverse function and existence of the learnable parameter ε have significantly mitigated the impact of such assumption, which can also be demonstrated by the insensitivity of CRReLU to data distribution shown in Section 5.

Now, let's consider the derivation from ReLU to CRReLU. For the sake of concise representation, we rewrite the data distribution and the derivative of data distribution as:

$$p(y) = C \cdot e^{-\frac{y^2}{2}}, \; p'(y) = -C \cdot y e^{-\frac{y^2}{2}}$$

Furthermore, ReLU has a mathematical function defined as $y = x$ when x is positive, meaning we have $y(x) = x$, $y'(x) = 1$ and $y''(x) = 0$. Therefore,

$$p'(y(x)) = p'(x) = -C \cdot y e^{-\frac{y^2}{2}} = -C \cdot x e^{-\frac{x^2}{2}}$$

Ultimately, by incorporating $p'(y) = -C \cdot xe^{-\frac{x^2}{2}}$, $y'(x) = 1$ and $y''(x) = 0$ into Equation 7, we can obtain:

$$\eta(x) = -C \cdot xe^{-\frac{x^2}{2}}$$

Furthermore, we make constant C as a learnable parameter ε with the purpose of enabling self-optimization in networks. According to EAFO methodology, we can get the inverse function of revised activation function as follows:

$$g(x) = x - \varepsilon xe^{-\frac{x^2}{2}} \qquad x \geqslant 0 \tag{8}$$

Finally, the optimized activation function CRReLU can be obtained by deriving the inverse function of $g(x)$. However, obtaining the inverse function of Equation 8 presents a challenge using conventional methods; as a consequence, we use the following function as a form of practical approximation.

$$f(x) = x + \varepsilon xe^{-\frac{x^2}{2}} \qquad x \geqslant 0 \tag{9}$$

We show the rationalization and reliability of utilizing Equation 9 as the approximate inverse function of Equation 8 in Proposition 2

Proposition 2. *Known $g(x) = x - \varepsilon xe^{-\frac{x^2}{2}}, f(x) = x + \varepsilon xe^{-\frac{x^2}{2}}$, for $x \geqslant 0$, the absolute value of error between $g(f(x))$ and x is bounded with $\left|e^{-1}\varepsilon^2 + 0.5e^{-\frac{3}{2}}\varepsilon^3\right|$.*

Detailed proof of Proposition 2 can be seen in Appendix C.

As illustrated in Section 4.2, $\varepsilon\eta(x)$ is the small perturbation; hence, from a theoretical perspective, we can set $\varepsilon\eta(x)$ as an infinitesimal. Furthermore, in this case, given the knowledge that $\eta(x)$ is a bounded function, we can easily deduce that ε is also an infinitesimal. Therefore, the absolute value of error between $g(f(x))$ and x is an infinitesimal of higher order. In practice, we typically initialize ε to a small value, such as 0.01 (as described in Section 5), implying that the absolute value of error is a small value.

Finally, let's consider the part where x is negative. When x is negative, the inverse function of ReLU can be visualized as a ray emanating from the origin and extending to infinity, possessing an infinite slope; and when x is positive, it constitutes a ray with the slope of 1. Hence, the correction term solution for both positive and negative values of x can be considered identical, differing only by constant C. In Equation 9 and Proposition 2, it is shown that incorporating the correction term into a linear activation function can have beneficial effects by reducing the information entropy. Therefore, we can obtain the full form of Correction Regularized ReLU as:

$$f(x) = \max(0, x) + \varepsilon xe^{-\frac{x^2}{2}} \tag{10}$$

Discussion on introduced learnable parameter ε. In Section 4.2, we have successfully demonstrated existence of the worst activation function, and from the worst as a starting point, it always moves towards improvement, regardless of the direction taken. However, commencing from a specific activation function, like ReLU here, does not invariably result in improvement across all directions, i.e. certain optimization paths may lead to deteriorated outcomes. Therefore, from the practical perspective, we introduce learnable parameter ε with the aim of enabling self-optimization of networks. From another perspective, in the derivation from ReLU to CRReLU, we assume that data follows Gaussian distribution, which might not be true in real experiments. Existence of the learnable parameter ε also weakens this assumption to some extent.

Finally, we provide further details of CRReLU in Appendix D, including python-like pseudocode of CRReLU in Appendix D.1, and further discussion on properties of CRReLU in Appendix D.2.

5 Experiments

Datasets. In experiments of image classification task, we adopt three datasets, ordered as CIFAR-10 (Krizhevsky et al., 2009), CIFAR-100 (Krizhevsky et al., 2009) and ImageNet-1K (Deng et al., 2009)

Table 1: Test accuracy of experiments conducted on CIFAR-10 for 100 epochs with error bar.

Top-one Accuracy		GELU	ELU	PReLU	CELU	SiLU	Mish	CRReLU (ours)
CIFAR-10	ViT-Tiny	70.4±0.2	66.4±0.5	78.0±0.6	66.5± 0.6	68.6±0.3	68.7±0.3	**80.7±0.3**
CIFAR-10	DeiT-Tiny	72.4±0.7	67.6±0.6	75.4±0.1	67.7±0.8	69.9±0.5	70.2±0.6	**77.0±0.3**
CIFAR-10	TNT-Small	73.7±0.5	69.5±0.6	75.8±0.3	68.7±0.2	71.1±0.7	71.6±0.8	**76.9±0.5**

Table 2: Test accuracy of experiments conducted on CIFAR-100 for 100 epochs with error bar.

Top-one Accuracy		GELU	ELU	PReLU	CELU	SiLU	Mish	CRReLU (ours)
CIFAR-100	ViT-Tiny	32.6±0.8	28.9±0.1	43.2±1.0	28.9±0.2	31.2±0.6	30.6±0.8	**46.6±0.6**
CIFAR-100	DeiT-Tiny	46.6±0.9	40.5±0.5	50.0±0.5	40.5±0.5	43.5±0.6	43.8±1.0	**50.7±0.1**
CIFAR-100	TNT-Small	47.5±0.8	43.6±0.3	49.0±0.7	43.0±0.5	45.0±0.9	45.5±0.8	**50.9±0.4**

in terms of the number of classification categories. In experiments of large language model (LLM) fine-tuning task, we employ two human preference datasets: SHP (Ethayarajh et al., 2022) and HH (Bai et al., 2022).

Baselines. We conduct experiments comparing the performance of CRReLU with several typical existing corrections of ReLU as illustrated in Section 2 and Section 3 : PReLU (He et al., 2015), ELU (Clevert et al., 2016), CELU (Barron, 2017), GELU (Hendrycks & Gimpel, 2023), Swish (SiLU) (Ramachandran et al., 2017) and Mish (Misra, 2020).

Experimental hyperparameters. For all transformer-based architectures, we directly set ε to 0.01 without further optimization. Detailed experimental hyperparameters are provided in Appendix E.

5.1 Task of Image Classification

We conduct all experiments within CIFAR10 and CIFAR100 on 4×RTX3090 and those within ImageNet1K on 4×NVIDIA L20 for 100 epochs using the AdamW optimizer with weight decay of 0.05, truncated normal initialization, gradient clipping norm of 1.0, cross entropy loss function, and cosine annealing learning rate scheduler with linear warm-up. All experiments are conducted three runs, we report the mean and standard derivation.

Experiments of ViTs on CIFAR-10 and CIFAR-100. Vision Transformer and its variants possess sufficiently complex structure and representational capability, garnering widespread attention from the community. Moreover, the assumption of Gaussian distribution has been theoretically proved as reasonable for sufficiently large MLPs (Williams, 1996; Lee et al., 2018; Park et al., 2020; Gao et al., 2023) and CNNs (Huang et al., 2021); however, the distribution of data under attention mechanism of transformers remains unexplored. Hence, we select vision transformer and its variants as our test model in order to further investigate the insensitivity of CRReLU to data distribution. Phase of experiments on CIFAR-10 and CIFAR-100 involves the selection of Vision Transformer (ViT) (Dosovitskiy et al., 2020), Data-Efficient Image Transformer (DeiT) (Touvron et al., 2021) and Transformer in Transformer (TNT) (Han et al., 2021). We report the top-one accuracy on CIFAR-10 in Table 1 and CIFAR-100 in Table 2, demonstrating CRReLU outperforms other existing corrections of ReLU on CIFAR dataset.

Experiments of ViTs on ImageNet-1K. ImageNet-1K dataset poses a significant challenge to information processing capability of neural networks due to its large image size and extensive range of classification categories. Hence, phase of experiments on ImageNet-1K involves the selection of Vision Transformer (ViT) (Dosovitskiy et al., 2020) and Data-Efficient Image Transformer (DeiT) (Touvron et al., 2021). We report the top-one accuracy on ImageNet-1K in Table 3.

Experiments on ViT clearly demonstrate superiority of CRReLU over other activation functions, and those on DieT, GELU shows 0.4% higher accuracy compared to CRReLU. Such result is attributed to the teacher-student strategy structure of DieT model. We utilize the fine-tuned "deit-tiny-patch16-

Table 3: Test accuracy of experiments conducted on ImageNet-1K for 100 epochs with error bar.

Top-one Accuracy		GELU	ELU	PReLU	CELU	SiLU	Mish	CRReLU (ours)
ImageNet-1K	ViT-Tiny	53.9±0.3	37.2±0.6	56.8±0.3	37.6±0.5	46.1±0.7	46.9±1.1	**57.5±0.4**
ImageNet-1K	DeiT-Tiny	**61.7±0.4**	49.1±0.7	60.8±0.4	48.9±0.8	58.5±0.7	58.9±0.3	**61.6±0.2**

224" model as teacher model, which is trained with GELU. As explained in the work (Abnar et al., 2020), through distillation, transformers will inherit inductive bias. Hence, training a student model with GELU on ImageNet-1K with the help of teacher model, which has already been pre-trained on ImageNet-1K with GELU, is certain to achieve better results than other activation functions.

5.2 Task of Large Language Model (LLM) Fine-tuning

In order to further validate the effectiveness of CRReLU on larger networks and generalization to a richer range of applications, we further perform supplementary experiments on LLM fine-tuning task. We employ the Direct Preference Optimization (DPO) (Rafailov et al., 2023) method to fine-tune GPT-2 (Radford et al., 2019) on Stanford Human Preferences (SHP) dataset (Ethayarajh et al., 2022) and Anthropic HH dataset (Bai et al., 2022). The parameter number of GPT-2 is 137 M, a relatively modest magnitude, hence we conduct full fine-tuning instead of LoRA-based one on 2×RTX3090. Firstly, we carry out supervised fine-tuning (SFT) with the purpose of mitigating distribution shift between the true reference distribution which is unavailable, and the reference policy utilized by DPO. Subsequently, we separately set the penalty coefficient β as **0.1, 1, 2, and 5**, in order to compare the performance of CRReLU and GELU under different penalty coefficients, and then execute DPO. We conduct three runs, reporting mean and standard deviation of evaluation metrics across the fine-tuning process in Table 4, which demonstrates that CRReLU generally outperforms GELU in LLM fine-tuning task.

Table 4: Metrics comparison between CRReLU and GELU in the task of LLM fine-tuning with error bar.

Evaluation Metrics		Evaluation Margin Reward↑	Evaluation Accuracy↑	Evaluation Loss↓
$\beta = 0.1$	CRReLU	**0.1428±0.0002**	**0.6209±0.0001**	**0.6476±0.0000**
	GELU	0.1420±0.0003	0.6197±0.0001	0.6480±0.0000
$\beta = 1$	CRReLU	**0.4627±0.0007**	**0.5757±0.0001**	**0.9202±0.0002**
	GELU	0.4560±0.0006	0.5729±0.0003	0.9387±0.0008
$\beta = 2$	CRReLU	**0.7757±0.0021**	**0.5631±0.0003**	**1.4610±0.0008**
	GELU	0.7178±0.0015	0.5606±0.0001	1.4814±0.0005
$\beta = 5$	CRReLU	**1.8473±0.0032**	**0.5635±0.0002**	**3.2677±0.0006**
	GELU	1.6538±0.0069	0.5568±0.0003	3.3034±0.0021

6 Discussion

Pursuit of better activation functions has been a longstanding and fundamental topic in the realm of machine learning. However, prior research has consistently concentrated on empirical search, without an emphasis on understanding the underlying mathematical mechanisms. This work aims to offer a proper solution to such issue. Our investigation into the relationship between activation functions and information theory concepts reveals that information entropy can be represented as a functional. Existence of the worst activation function with boundary condition (WAFBC) furnishes a solid theoretical basis for exploring better activation functions. In the process of solving WAFBC, we draw inspiration from the Taylor expansion form, leading us to propose Entropy-based Activation Function Optimization (EAFO) methodology. EAFO methodology presents a novel perspective for designing static activation functions in deep neural networks and shows the potential of dynamically optimizing

activation during iterative training. Utilizing EAFO methodology, we derive a novel activation function from ReLU, called Correction Regularized ReLU (CRReLU). Experiments involving image classification task and large language model (LLM) fine-tuning task demonstrate that CRReLU is comparable to or surpasses existing corrections of ReLU. Overall, the EAFO methodology provides numerous promising avenues for future research on activation functions, and the CRReLU introduces a novel addition to the set of high-performing activation functions.

Limitations and Future Work. Our findings raise several important questions for future work. *Firstly*, how can EAFO framework be systematically generalized to non-invertible activation functions? In the initial setting of EAFO methodology, the choice of activation function is restricted to those with invertible counterparts. Despite ReLU being a prominent example of activation function without an inverse, we derive CRReLU utilizing EAFO; however, the derivation also partly benefits from the simplicity of ReLU's form and several heuristic approaches. *Secondly*, how to effectively implement activation function iteration optimization during neural network training? Notwithstanding the demonstrated feasibility of iterative activation function optimization during neural network training, it is currently hindered by the high computational complexity, particularly in large-scale neural networks. Applicability of the EAFO methodology to optimize activation in alternative network structures, such as Kolmogorov-Arnold Networks (KANs), also deserves further in-depth research. Therefore, the development of practical and efficient algorithms is an exciting direction for future work. *Moreover*, although previous work has theoretically established the reasonability of Gaussian distribution assumptions for MLP and CNN, and we have empirically demonstrated superiority of CRReLU in modern architectures like transformers, the theoretical robustness under a broader range of distributions is still worthy of further research. *Finally*, while we have empirically validated the exceptional performance of CRReLU on image classification task and large language model fine-tuning task, its performance on other tasks remains to be explored, thereby warranting further investigation.

REFERENCES

Samira Abnar, Mostafa Dehghani, and Willem Zuidema. Transferring inductive biases through knowledge distillation, 2020.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022.

Jonathan T. Barron. Continuously differentiable exponential linear units, 2017.

Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus), 2016.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. Understanding dataset difficulty with $\mathcal{V}$-usable information, 2022.

Tianxiang Gao, Xiaokai Huo, Hailiang Liu, and Hongyang Gao. Wide neural networks as gaussian processes: Lessons from deep equilibrium models, 2023.

Richard Hahnloser, Rahul Sarpeshkar, Misha Mahowald, Rodney Douglas, and H. Seung. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*, 405: 947–51, 07 2000.

Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus), 2023.

Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4: 251–257, 1991.

Zhongzhan Huang, Wenqi Shao, Xinjiang Wang, Liang Lin, and Ping Luo. Convolution-weight-distribution assumption: Rethinking the criteria of channel pruning, 2021.

Thibaut Lavril etal. Hugo Touvron. Llama: Open and efficient foundation language models, 2023.

Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th International Conference on Computer Vision*, pp. 2146–2153, 2009.

Grigory Khromov and Sidak Pal Singh. Some fundamental aspects about lipschitz continuity of neural networks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=5jWsW08zUh.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

H. Larochelle, Yoshua Bengio, Jérôme Louradour, and Pascal Lamblin. Exploring strategies for training deep neural networks. *J. Mach. Learn. Res.*, 10:1–40, 2009.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes, 2018.

Minhyeok Lee. Gelu activation function in deep learning: A comprehensive mathematical analysis and performance, 2023.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y. Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks, 2024.

Andrew L. Maas. Rectifier nonlinearities improve neural network acoustic models. 2013.

Diganta Misra. Mish: A self regularized non-monotonic activation function. In *British Machine Vision Conference*, 2020.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning*, 2010.

OpenAI. Gpt-4 technical report, 2023.

Chenbin Pan, Burhaneddin Yaman, Tommaso Nesti, Abhirup Mallik, Alessandro G Allievi, Senem Velipasalar, and Liu Ren. Vlp: Vision language planning for autonomous driving, 2024.

Daniel S. Park, Jaehoon Lee, Daiyi Peng, Yuan Cao, and Jascha Sohl-Dickstein. Towards nngp-guided neural architecture search, 2020.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2023.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2017.

Jorge M Santos, Luis A Alexandre, and J Marques de Sá. The error entropy minimization algorithm for neural network classification. In *int. conf. on recent advances in soft computing*, pp. 92–97, 2004.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

Luis M Silva, Joaquim Marques de Sá, and Luis A Alexandre. Neural network classification using shannon's entropy. In *Esann*, pp. 217–222, 2005.

Gemma Team. Gemma: Open models based on gemini research and technology, 2024.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & amp; distillation through attention. In *International Conference on Machine Learning*, volume 139, pp. 10347–10357, July 2021.

Christopher K. I. Williams. Computing with infinite networks. In *Neural Information Processing Systems*, 1996.

Aolin Xu and Maxim Raginsky. Information-theoretic analysis of generalization capability of learning algorithms. *Advances in neural information processing systems*, 30, 2017.

Jirong Yi, Qiaosheng Zhang, Zhen Chen, Qiao Liu, and Wei Shao. Mutual information learned classifiers: An information-theoretic viewpoint of training deep learning classification systems. *arXiv preprint arXiv:2209.10058*, 2022.

A Proof of Proposition 1

Proof. From Equation 2, we know that:

$$\frac{\mathrm{d}}{\mathrm{d}x}(\frac{\partial \mathbb{G}}{\partial y'}) - \frac{\partial \mathbb{G}}{\partial y} = 0$$

Considering the total differential of $\mathbb{G}$:

$$\frac{\mathrm{d}\mathbb{G}}{\mathrm{d}x}\left(y', y, x\right) = \frac{\partial \mathbb{G}}{\partial x} \cdot \frac{\mathrm{d}x}{\mathrm{d}x} + \frac{\partial \mathbb{G}}{\partial y} \cdot \frac{\mathrm{d}y}{\mathrm{d}x} + \frac{\partial \mathbb{G}}{\partial y'} \cdot \frac{\mathrm{d}y'}{\mathrm{d}x} = \frac{\partial \mathbb{G}}{\partial x} + \frac{\partial \mathbb{G}}{\partial y} \cdot y' + \frac{\partial \mathbb{G}}{\partial y'} \cdot y''$$

Thus, we have:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}x}\left(y' \frac{\partial \mathbb{G}}{\partial y'}\right) &= y'' \frac{\partial \mathbb{G}}{\partial y'} + y' \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial \mathbb{G}}{\partial y'}\right) \\
&= \frac{\mathrm{d}\mathbb{G}}{\mathrm{d}x}\left(y', y, x\right) - \frac{\partial \mathbb{G}}{\partial y} \cdot y' - \frac{\partial \mathbb{G}}{\partial x} + y' \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial \mathbb{G}}{\partial y'}\right) \\
&= \frac{\mathrm{d}}{\mathrm{d}x}\mathbb{G}\left(y', y, x\right) - \frac{\partial \mathbb{G}}{\partial x} - y' \cdot \left(\frac{\partial \mathbb{G}}{\partial y} - \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial \mathbb{G}}{\partial y'}\right)\right) \\
&= \frac{\mathrm{d}}{\mathrm{d}x}\mathbb{G}\left(y', y, x\right) - \frac{\partial \mathbb{G}}{\partial x}
\end{aligned}$$

Therefore, we know that

$$\frac{\partial \mathbb{G}}{\partial x} - \frac{\mathrm{d}}{\mathrm{d}x}\left(\mathbb{G} - y' \frac{\partial \mathbb{G}}{\partial y'}\right) = 0$$

For the sake that $\mathbb{G}$ is independent of x, then we have that $\frac{\partial \mathbb{G}}{\partial x} = 0$. Hence,

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(\mathbb{G} - y' \frac{\partial \mathbb{G}}{\partial y'}\right) = 0$$

Finally, we can draw the conclusion that:

$$\mathbb{G} - y' \frac{\partial \mathbb{G}}{\partial y'} = C,$$

which completes the proof. □

B Further Discussion on WAFBC

Let's take several typical boundary conditions into consideration. Firstly, setting $f(x)$ approaches 1, when x tends to positive infinity; and $f(x)$ approaches 0, when x tends to negative infinity. Therefore, the solution takes the form of cumulative distribution function (CDF), which can be expresses as:

$$f(x) = \int_{-\infty}^{x} p(t)\mathrm{d}t$$

Similarly, if fixing the difference between the upper and lower bounds of the activation function to be e, and making the activation function symmetric about the origin, the form can be written as:

$$f(x) = e\int_{0}^{x} p(t)\mathrm{d}t$$

Furthermore, in the event that the input data distribution is assumed to be approximately uniformly distributed, the worst activation function can be approximated as a linear function. Were it to approximate the input data distribution as a normal distribution, then the form of the worst activation function would be closer to Sigmoid and Tanh. We show the comparison of function curves in Figure 1 and Figure 2.

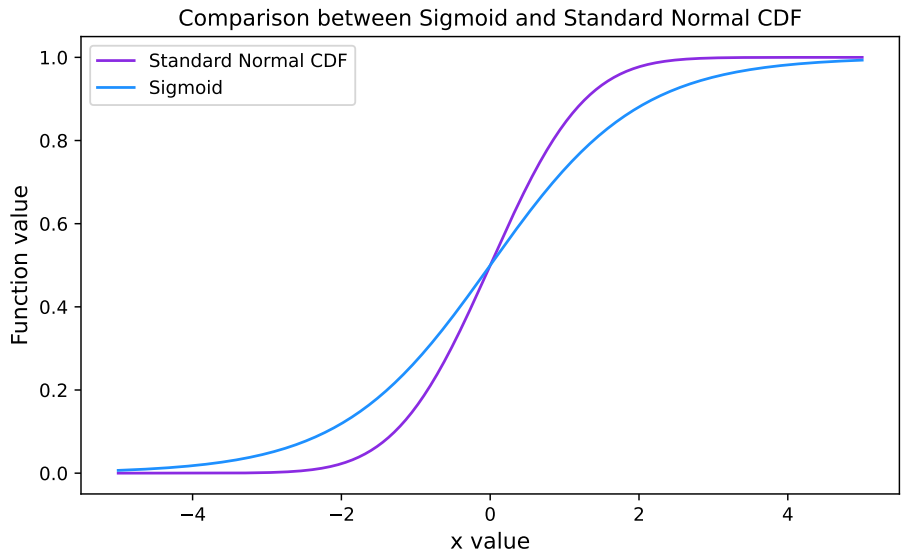

Figure 1: Comparison between Sigmoid and standard normal CDF

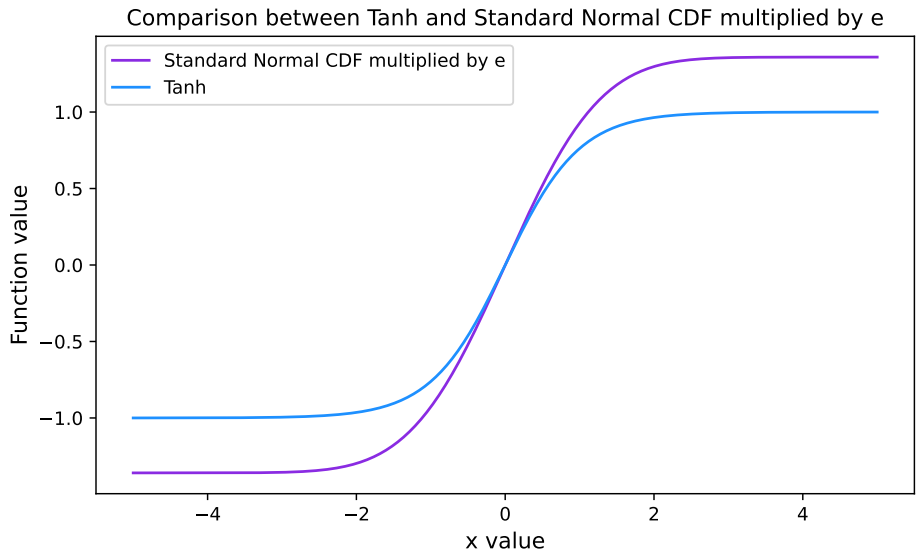

Figure 2: Comparison between Tanh and Standard Normal CDF multiplied by e (has been transformed to achieve symmetry about origin)

C Proof of Proposition 2

Before the proof of Proposition 2, we would like to show four facts without proof.

Fact 1. $f(x) = xe^{-\frac{x^2}{2}}$ *is a bounded function, and range of the function is* $[-e^{-\frac{1}{2}}, e^{-\frac{1}{2}}]$.

Fact 2. $f(x) = x^2 e^{-x^2}$ *is a bounded function, and range of the function is* $[0, e^{-1}]$.

Fact 3. $f(x) = x^3 e^{-\frac{3}{2}x^2}$ *is a bounded function, and range of the function is* $[-e^{-\frac{3}{2}}, e^{-\frac{3}{2}}]$.

Fact 4. $\forall x \in \mathcal{R},\ 1 - e^{-x} - x \leqslant 0$.

We now commence the proof of Proposition 2.

Proof. Substituting the analytic expression into the formula and performing algebraic simplifications, we can obtain:

$$
\begin{aligned}
g\left(f(x)\right) = g\left(x+\varepsilon x e^{-\frac{x^2}{2}}\right) &= x+\varepsilon x e^{-\frac{x^2}{2}} - \varepsilon\left(x+\varepsilon x e^{-\frac{x^2}{2}}\right) e^{-\frac{1}{2}\left(x+\varepsilon x e^{-\frac{x^2}{2}}\right)^2} \\
&= x+\varepsilon x\left(e^{-\frac{x^2}{2}} - e^{-\frac{1}{2}\left(x+\varepsilon x e^{-\frac{x^2}{2}}\right)^2}\right) - \varepsilon^2 x e^{-\frac{x^2}{2}} e^{-\frac{1}{2}\left(x+\varepsilon x e^{-\frac{x^2}{2}}\right)^2} \\
&= x+\varepsilon x e^{-\frac{x^2}{2}}\left[1-e^{-\frac{1}{2}\left(2\varepsilon x e^{-\frac{x^2}{2}}+\varepsilon^2 x^2 e^{-x^2}\right)}\right] - \varepsilon^2 x e^{-\frac{x^2}{2}} e^{-\frac{1}{2}\left(x+\varepsilon x e^{-\frac{x^2}{2}}\right)^2}
\end{aligned}
$$

Thus,

$$
\begin{aligned}
|g(f(x))-x| &= \left|\varepsilon x e^{-\frac{x^2}{2}}\left[1-e^{-\frac{1}{2}\left(2\varepsilon x e^{-\frac{x^2}{2}}+\varepsilon^2 x^2 e^{-x^2}\right)}\right] - \varepsilon^2 x e^{-\frac{x^2}{2}} e^{-\frac{1}{2}\left(x+\varepsilon x e^{-\frac{x^2}{2}}\right)^2}\right| \\
&\leqslant \left|\varepsilon x e^{-\frac{x^2}{2}}\left[1-e^{-\frac{1}{2}\left(2\varepsilon x e^{-\frac{x^2}{2}}+\varepsilon^2 x^2 e^{-x^2}\right)}\right]\right| \\
&\leqslant \left|\varepsilon x e^{-\frac{x^2}{2}}\left[-\frac{2\varepsilon x e^{-\frac{x^2}{2}}+\varepsilon^2 x^2 e^{-x^2}}{2}\right]\right| \\
&= \left|\varepsilon x e^{-\frac{x^2}{2}}\left(-\varepsilon x e^{-\frac{x^2}{2}} - \frac{1}{2}\varepsilon^2 x^2 e^{-x^2}\right)\right| = \left|\varepsilon^2 x^2 e^{-x^2} + \frac{1}{2}\varepsilon^3 x^3 e^{-\frac{3}{2}x^2}\right| \\
&\leqslant \left|e^{-1}\varepsilon^2 + 0.5 e^{-\frac{3}{2}}\varepsilon^3\right|
\end{aligned}
$$

The first inequality is established owing to Fact 1 and the fact that when x is positive, the second term of absolute value must be positive. The second inequality is established owing to Fact 4. The third inequality is established owing to Fact 2 and Fact 3. Hence, we can draw the conclusion that the absolute value of error between $g\left(f(x)\right)$ and x is bounded with $\left|e^{-1}\varepsilon^2 + 0.5 e^{-\frac{3}{2}}\varepsilon^3\right|$, which completes the proof. □

D FURTHER DETAILS OF CRRELU

D.1 CORRECTION REGULARIZED RELU (CRRELU) PSEUDOCODE

Algorithm 1: Correction Regularized ReLU (CRReLU) Pseudocode

```
import torch
import torch.nn as nn
import torch.nn.functional as F

class CRReLU(nn.Module):
    def __init__(self,lr=0.01):
        super(CRReLU,self).__init__()
        self.lr = nn.Parameter(torch.tensor(lr))

    def forward(self,x):
        return F.relu(x)+self.lr*x*torch.exp(-x**2/2)
```

D.2 Further Discussion on Properties of CRReLU

We show the function curves with different ε values for CRReLU in Figure 3. As depicted in the figure, existence of the correction term in CRReLU brings several good properties. It allows propagation of gradient when input is less than zero, serving to alleviate the dying ReLU phenomenon to a certain degree; simultaneously, as x approaches negative infinity, CRReLU also converges to 0, thereby guaranteeing sparsity of models in the negative part.

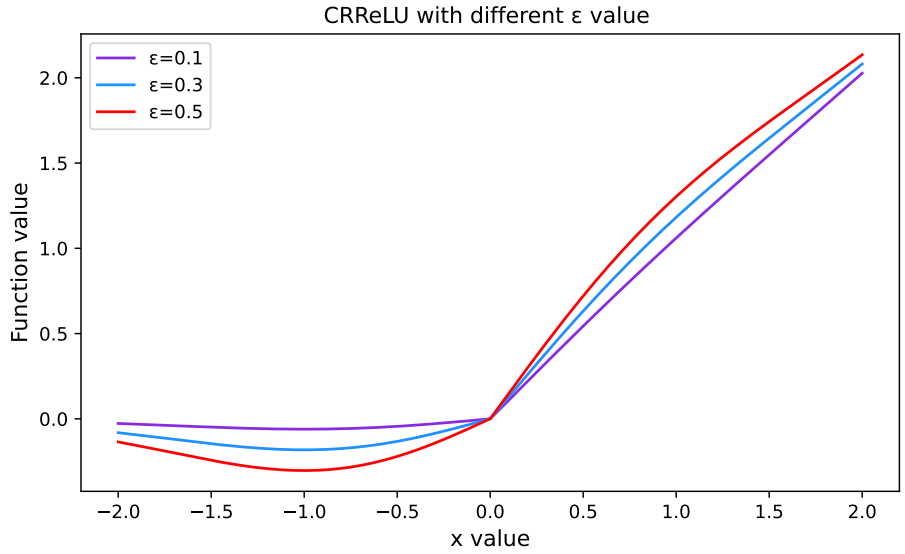

Figure 3: CRReLU with different ε value

E Details of Experimental Settings

E.1 Task of Image Classification

Table 5: Experimental settings of ViT, DeiT and TNT on CIFAR-10 and CIFAR-100 datasets

Image Size	32×32
Patch Size	4
Embedding Dim	192 for ViT-Tiny and DeiT-Tiny ; 384 for TNT-small
Optimizer	AdamW with weight decay = 0.05
Learning Rate	Cosine Annealing Learning Rate Scheduler Initial lr = 2.5×10^{-4} ; lr drop = -1 ; min lr = 1×10^{-5}
Warm up	warmup epochs = 20 ; warmup learning rate = 1×10^{-6}
Gradient Clipping	1.0
Training Epochs	100
Batch Size	256
Loss Function	CrossEntropy Loss
Normalization	Layer Norm
Data Augmentation	True (provided by timm)
Drop Out and Drop Path	False

Table 6: Experimental settings of ViT and DeiT on ImageNet-1K dataset

Image Size	224 × 224
Patch Size	16
Embedding Dim	192
Optimizer	AdamW with weight decay = 0.05
Learning Rate	Cosine Annealing Learning Rate Scheduler Initial lr = 2.5×10^{-4} ; lr drop = -1 ; min lr = 1×10^{-5}
Warm up	warmup epochs = 20 ; warmup learning rate = 1×10^{-6}
Gradient Clipping	1.0
Training Epochs	100
Batch Size	256
Loss Function	CrossEntropy Loss
Normalization	Layer Norm
Data Augmentation	True (provided by timm)
Drop Out and Drop Path	False

Table 7: We record changes in parameter number when employing various activation functions. GELU, ELU, CELU, SiLU (Swish), and Mish are considered activation functions without learnable parameter **(AFs without LP)**, while PReLU and CRReLU are considered activation functions with learnable parameter **(AFs with LP).** The results demonstrate that increase in parameter number introduced by the learnable parameter is negligible.

Parameter Number		CIFAR-10	CIFAR-100	ImageNet-1K
ViT-Tiny	AFs without LP	5399818	5417188	5754472
	AFs with LP	5399830	5417200	5754484
DeiT-Tiny	AFs without LP	5365076	5399816	5910800
	AFs with LP	5365088	5399828	5910812
TNT-Small	AFs without LP	21525298	21559948	/
	AFs with LP	21525322	21559972	/

E.2 Task of Large Language Model (LLM) Fine-tuning

Table 8: Experimental settings of GPT2 fine-tuning task

Batch Size	32
Optimizer	RMSprop (More Memory-Efficient)
Learning Rate	5×10^{-7} with linear warmup steps of 150
Trainer	FSDPTrainer (2 GPUs)
Max Gradient Norm	10.0
Max Length for an Input (Prompt + Response)	512
Max Length for Prompt	256

F Additional Supportive Experiments

In response to reviewers' precious feedback, we conduct additional supportive experiments to address their concerns. These experiments mainly focus on the following aspects: additional architecture (Appendix F.1), additional dataset (Appendix F.2), additional ε initialization (Appendix F.3), entropy calculation after activation (Appendix F.4) and mixed activation function (Appendix F.5).

F.1 Additional Architecture

In this section, we aim to enhance the evaluation of CRReLU's generalization to network architecture. We choose to validate the performance on CIFAR10, CIFAR100, and ImageNet1K with the ConvNeXt-tiny (Liu et al., 2022). For all the experiments, we conduct three runs, and we report the mean and standard deviation in Table 9. Experiments within CIFAR10 and CIFAR100 are conducted on 4×RTX3090 and those within ImageNet1K are conducted on 4×NVIDIA L20.

Table 9: Test accuracy of experiments conducted on ConvNeXt-tiny for 100 epochs with error bar.

Top-one Accuracy		GELU	ELU	PReLU	CELU	SiLU	Mish	CRReLU (ours)
CIFAR10	ConvNeXt	64.9±0.4	59.8±0.5	64.6±1.4	59.8±0.5	60.6±0.2	61.4±0.4	**70.6±1.1**
CIFAR100	ConvNeXt	36.6±0.3	30.3±0.4	35.2±0.5	30.5±0.2	35.0±0.9	35.3±0.7	**42.1±0.7**
ImageNet1K	ConvNeXt	72.9±0.3	71.7±0.5	72.9±0.5	71.8±0.9	72.3±0.7	72.8±0.6	**73.2±0.2**

F.2 Additional Dataset

In this section, we endeavor to elucidate the performance of CRReLU on diverse datasets. We conduct experiments on EuroSAT (Helber et al., 2019) with ConvNeXt-tiny (Liu et al., 2022). All experiments are performed three times, we report the mean and standard deviation. We conduct this part experiments on a single RTX3090 for 25 epochs using the AdamW optimizer, learning rate of 0.0001, cross entropy loss function, batch size of 256.

Table 10: Test accuracy of experiments conducted with ConvNeXt-tiny on the EuroSAT.

GELU	ELU	PReLU	CELU	SiLU	Mish	CRReLU (ours)
83.09±1.06	81.21±0.37	81.33±0.94	81.12±0.27	81.85±1.01	82.23±0.08	**83.26±0.52**

F.3 Additional ε Initialization

In this section, we focus on exploring the impact of different initial values of ε, as well as potential instabilities or failure cases under different initialization schemes. We set ε to -0.5, -0.2, -0.1, -0.05, -0.02, -0.01, 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, and 10, conducting experiments with ViT-tiny on CIFAR10 and CIFAR100. We conduct three runs on 4×RTX3090, reporting the mean and standard derivation in Table 11.

Table 11: Test accuracy of experiments conducted with ViT for 100 epochs under different initializations.

ε	0.01	0.02	0.05	0.1	0.2	0.5	1	10
CIFAR10	80.7±0.4	80.1±0.1	79.7±0.3	79.6±0.2	78.1±0.7	74.1±1.0	68.7±0.3	60.3±0.4
CIFAR100	46.6±0.6	46.0±0.3	45.6±0.4	44.9±0.3	43.6±0.4	36.4±0.9	29.9±0.8	22.7±0.6

ε	-0.01	-0.02	-0.05	-0.1	-0.2	-0.5
CIFAR10	80.1±0.4	80.0±0.3	80.1±0.2	80.6±0.1	80.5±0.3	80.4±0.1
CIFAR100	45.9±0.3	46.0±0.6	46.1±0.6	46.1±0.3	46.0±0.1	45.8±0.5

Discussion: From the above experiments, we can see different initialization strategy does can have an impact on the final result. When the initial values differ significantly from the values we derived in Corollary G.1 (such as 0.5, 1, 10), we have observed that the performance of training will degrade severely, especially at 1 and 10, where the training process becomes extremely unstable.

F.4 Entropy Calculation after Activation

In this section, we focus on comparing the entropy post-training of neural networks trained with CRReLU and GELU. We apply the post-trained ViT-Tiny with CRReLU and GELU on ImageNet1K. By randomly selecting same ten batches of images from ImageNet1K, we compute the information entropy after each of the 12 layers. We present the mean and standard deviation of these values in Table 12.

Table 12: Entropy calculation after activation (GELU and CRReLU) on 12 layers of the trained ViT on ImageNet1K.

Layer	1	2	3	4	5	6
CRReLU	7.594±0.007	7.598±0.003	7.599±0.003	7.595±0.003	7.592±0.003	7.584±0.004
GELU	7.536±0.046	7.541±0.019	7.561±0.011	7.573±0.006	7.580±0.005	7.583±0.004

Layer	7	8	9	10	11	12
CRReLU	7.572±0.005	7.557±0.005	7.540±0.005	7.523±0.007	7.498±0.008	7.461±0.008
GELU	7.585±0.004	7.585±0.004	7.583±0.004	7.580±0.004	7.577±0.004	7.560±0.004

Discussion: From the results presented above, it is evident that for GELU, the entropy after 12 layers of activation exhibits an overall increasing trend, whereas conversely, CRReLU demonstrates a general declining trend. Furthermore, we have noted that the reduction in entropy for CRReLU between layers 1 and 6 is not significant, whereas a marked decline is observed from layers 7 to 12.

F.5 Mixed Activation Function

In Appendix F.4, it is shown that reduction in entropy for CRReLU between layers 1 and 6 is not significant. Hence, in this section, we focus on equipping different activation functions in the initial six layers and subsequent six ones. We employ GELU for layers 1 to 6 and CRReLU for layers 7 to 12, denoting this as "6GELU+6CRReLU". We conduct three runs on CIFAR10, CIFAR100, and ImageNet1K, presenting the mean and standard deviation of the results in Table 13. Experiments on CIFAR10 and CIFAR100 are conducted on 4×RTX3090, and those on ImageNet1K are carried out on 4×NVIDIA L20.

Table 13: Test accuracy of experiments conducted with ViT (12GELU, 6GELU+6CRReLU, 12CRReLU) for 100 epochs.

	12GELU	6GELU+6CRReLU	12CRReLU
CIFAR10	0.704±0.002	0.755±0.008	**0.807±0.003**
CIFAR100	0.326±0.008	0.399±0.004	**0.466±0.006**
ImageNet1K	0.539±0.003	0.512±0.001	**0.575±0.004**

Discussion: From the results, it appears that having only the last few layers equipped with CRReLU is not as effective as utilizing CRReLU throughout the entire network. Especially the results on ImageNet1K, 6GELU+6CRReLU is significantly and stably worsen to all GELU and all CRReLU, which is quite surprising to us. We consider that this may be due to the fact that, while the reduction in entropy is not significantly apparent in the earlier layers, CRReLU's focus on achieving lower entropy still facilitates superior feature extraction. It seems that when using GELU in the earlier layers and CRReLU in the later layers, on small-scale datasets, it is still possible to benefit from the CRReLU mechanism in the later layers (the features learned in the earlier layers are not good enough

yet); however, on large-scale datasets, the features learned in the earlier layers (those equipped with GELU) might even have a negative effect.

G LIPSCHITZ CONTINUITY ANALYSIS

Lipschitz continuity constitutes a stronger form of continuity, which imposes an upper bound on the rate of variation of a function. We would like to begin with the defination of Lipschitz-continuous functions.

Definition G.1 (**Lipschitz Continuous Function**(Khromov & Singh, 2024))**.** *For function f:* $\mathbb{R}^d \to \mathbb{R}^K$*, defined on some domain dom(f)* $\subseteq \mathbb{R}^d$*, is called C-Lipschitz continuous, C > 0, w.r.t. some* α*-norm if,* $\forall\, x, y \in dom(f)$*:* $\|f(x) - f(y)\|_\alpha \leq C\|x - y\|_\alpha$*.*

Generally speaking, we are interested in the **smallest C** that makes the above condition hold. This is what we called the **(true) Lipschitz constant of the function** f. Lipschitz continuity of an activation function is crucial for ensuring a well-behaved optimization landscape, thereby promoting efficient convergence during training; furthermore, a Lipschitz constant closer to 1 indicates stronger Lipschitz continuity. In the work (Lee, 2023), the Lipschitz constant of GELU has been derived by the authors. We refined the derivation in Remark G.1.

Remark G.1 ((Lee, 2023))**.** *Lipschitz constant of GELU is 1.084.*

Proof of sketch. Firstly, compute the derivative of the GELU function as:

$$\frac{d\text{GELU}(x)}{dx} = x \cdot \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} + \Phi(x)$$

To find a constant C > 0 such that for all x$\in \mathbb{R}$, we have that $|\frac{d\text{GELU}(x)}{dx}| \leq C$.

Using the second derivative to establish a tight bound on the derivative. The second derivative of the GELU function is:

$$\frac{d^2\text{GELU}(x)}{dx^2} = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} (2 - x^2)$$

Setting the second derivative equal to zero and solving for x, we obtain two critical points at $x = -\sqrt{2}$ and $x = \sqrt{2}$. Computing the first derivative at $x = \sqrt{2}$:

$$\frac{d\text{GELU}(x)}{dx}|_{x=\sqrt{2}} \approx 1.084$$

Thus, it is found that derivation of GELU is bounded by 1.084, proving its Lipschitz continuity. □

Remark G.2. *Lipschitz constant of SiLU is 1.100.*

Proof of sketch. In the following calculations and derivations, we refer to the derivation in Remark G.1 and make appropriate omissions. Firstly, we write the form of SiLU as :

$$\text{SiLU}(x) = \frac{x}{1 + e^{-x}}$$

We can compute the derivative of the SiLU function as:

$$\frac{d\text{SiLU}(x)}{dx} = \frac{(x+1)e^{-x} + 1}{(1 + e^{-x})^2}$$

and the second derivative as follows:

$$\frac{d^2\text{SiLU}(x)}{dx^2} = \frac{e^x(-e^x(x-2) + x + 2)}{(1 + e^x)^3}$$

We solve the transcendental equation $\frac{d^2\text{SiLU}(x)}{dx^2} = 0$ through a python program, we have $x_1 \approx -2.3994$ and $x_2 \approx 2.3994$. Furthermore, we have the derivative of the SiLU is bounded in the range of [-0.100, 1.100]. Hence, Lipschitz constant of SiLU is 1.100. □

Remark G.3. *Lipschitz constant of Mish is 1.089.*

Proof of sketch. Form of the Mish function, its derivation and its second derivation are performed as:

$$\text{Mish}(x) = x\frac{e^{2x}+2e^{x}}{e^{2x}+2e^{x}+2}$$
$$\frac{d\text{Mish}(x)}{dx} = \frac{e^{x}[4(x+1)+4e^{2x}+e^{3x}+e^{x}(4x+6)]}{(e^{2x}+2e^{x}+2)^2}$$
$$\frac{d^2\text{Mish}(x)}{dx^2} = \frac{4e^{x}(3e^{2x}(x-2)+2e^{3x}(x-1)-2(x+2)-2e^{x}(x+4))}{(e^{2x}+2e^{x}+2)^3}$$

Similarly, we solve the transcendental equation $\frac{d^2\text{Mish}(x)}{dx^2}=0$ through a python program, we then have $x_1 \approx -2.2564$ and $x_2 \approx 1.4906$. Morever, the derivative of the Mish is bounded in the range of [-0.113, 1.089]. Hence, Lipschitz constant of Mish is 1.089. □

Remark G.4. *Under mild assumptions, Lipschitz constant of CRReLU is* $\max(1+\varepsilon, 1-0.446\varepsilon)$.

Proof of sketch.

$$\text{CRReLU}(x) = \begin{cases} x+\varepsilon x e^{-\frac{x^2}{2}} & (x>0) \\ \varepsilon x e^{-\frac{x^2}{2}} & (x<0) \end{cases}$$

Furthermore, under mild assumptions, we consider the derivative of CRReLU piecewise (disregarding the potential for non-differentiability at x = 0 temporarily).

$$\frac{d\text{CRReLU}(x)}{dx} = \begin{cases} 1+\varepsilon(1-x^2)e^{-\frac{x^2}{2}} & (x>0) \\ \varepsilon(1-x^2)e^{-\frac{x^2}{2}} & (x<0) \end{cases}$$

Setting the second derviation to be 0

$$\frac{d^2\text{CRReLU}(x)}{dx^2} = \varepsilon e^{-\frac{x^2}{2}}(x^3-3x) = 0$$

Then we have that: $x_1 = 0, x_2 = \sqrt{3}, x_3 = -\sqrt{3}$. For x_2 and x_3, we directly calculate the values at that point:

$$\frac{d\text{CRReLU}(x)}{dx}|_{x_2} = 1-0.446\varepsilon; \quad \frac{d\text{CRReLU}(x)}{dx}|_{x_3} = -0.446\varepsilon$$

For $x_1 = 0$, we calculate the limit of $\frac{d\text{CRReLU}(x)}{dx}$ around 0:

$$\lim_{x\to 0^+}\frac{d\text{CRReLU}(x)}{dx} = 1+\varepsilon; \lim_{x\to 0^-}\frac{d\text{CRReLU}(x)}{dx} = \varepsilon$$

Hence, under mild assumptions, to obtain the upper bound for derivative of CRReLU. we have that:

$$C = \max(1+\varepsilon, \varepsilon, 1-0.446\varepsilon, -0.446\varepsilon)$$

Let's further consider the Lipschitz constant and simplify it. If $\varepsilon > 0$, it is obvious that $C = 1+\varepsilon$; and if $\varepsilon < 0$, we can get $C = 1-0.446\varepsilon$. Hence, we can express the Lipschitz constant of CRReLU as:

$$C = \max(1+\varepsilon, 1-0.446\varepsilon)$$

□

Corollary G.1. *In order to make Lipschitz constant of CRReLU remains lower than that of GELU, the range of* ε *is (-0.188,0.084). We recommend setting the initial value of* ε *within this range.*

Proof of sketch. If $\varepsilon > 0$, considering the following inequality:

$$1+\varepsilon < 1.084, \text{ then we have: } 0 < \varepsilon < 0.084$$

If $\varepsilon < 0$, considering the following inequality:

$$1-0.446\varepsilon < 1.084, \text{ then we have: } -0.188 < \varepsilon < 0$$

□

Corollary G.2. *If* ε *is in (0.084,0.089)∪(-0.198, -0.188), we have the CRReLU's Lipschitz continuity worsen than GELU, but better than Mish.*

Corollary G.3. *If* ε *is in (0.089,0.100)∪(-0.198, -0.188), we have the CRReLU's Lipschitz continuity worsen than Mish, but better than SiLU.*

H Further Discussion on Initialization and Training Stability

In Table 11, we demonstrate that different initialization strategies can have an impact on the final result. Improper initialization can significantly impair performance and lead instability. In this part, we set ε to 0.01 and show the changes of ε in each layer during the 100 epochs' training of ViT-tiny on CIFAR10 and CIFAR100 in Figure 4 and Figure 5. As the figures shown, initializing ε at 0.01 allows its stable change during the training process and could avoid extreme situations.

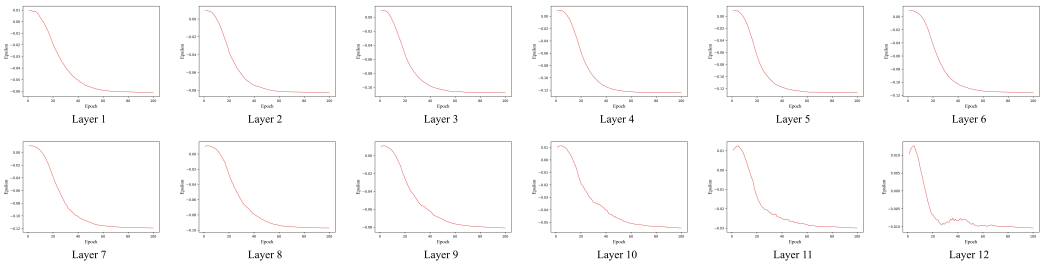

Figure 4: Visualization of ε change in each layer during the training of ViT-tiny on CIFAR10.

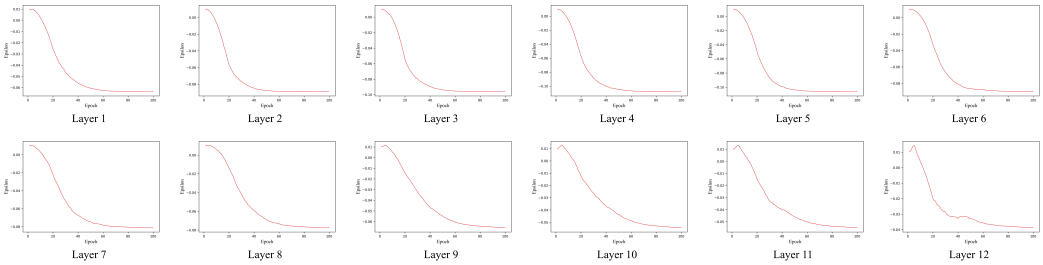

Figure 5: Visualization of ε change in each layer during the training of ViT-tiny on CIFAR100.

Furthermore, we would like to provide some insights on selecting optimal values of ε for practitioners implementing CRReLU in their own networks. Firstly, we suggest looking within the aforementioned scope (-0.188,0.084) (detailed in Corollary G.1). Furthermore, we suggest testing multiple values of it within this range and using k-fold cross-validation to test the performance of model under different initial values. Train and evaluate the model on different folds to find the optimal value. In addition, if the prior knowledge of the dataset and network structure is sufficient, it can also be fully utilized, for example, if the network tends to produce negative outputs, then increasing the value of ε can be considered to enable the model to better capture the features of the samples.

I Further Discussion on Lower Entropy Indicates Better Classification

Our work is largely based on the statement: lower entropy indicates better classification. In this section, we will further elaborate on the reasonability of such a statement. We would like to further elaborate mainly from three perspectives: intuitively, empirically, and theoretically. From the **intuitive** perspective, lower entropy indicates less uncertainty for feature representation, which usually means more information is captured in fewer features. In other words, lower entropy can suggest that features are more discriminative, better able to distinguish different categories or patterns. From the **empirical** perspective, early work (Silva et al., 2005) experimentally showed that minimization of Shannon's entropy of the gap between the output and the desired target could achieve a better performance compared to MSE and CE. In early work (Santos et al., 2004), the authors experimentally illustrated that minimizing entropy of the error between output and desired targets yields exceptionally satisfactory classification performance. From the **theoretical** perspective, work (Yi et al., 2022) proved that for training DNN classifiers essentially learns the conditional entropy of the underlying data distribution of the dataset (the information or uncertainty remained in the labels after revealing the input) and derived the mutual information (between the corresponding feature and the label) bounds for a classification data model (Section 7). Hence, the conditional entropy $\mathbb{H}(\text{output} \mid \text{input})$

will decrease with the process of training. In the work (Xu & Raginsky, 2017), the authors derived upper bounds on the generalization error in terms of the mutual information between its input and output. According to (Xu & Raginsky, 2017), a smaller mutual information means a smaller generalization error upper bound, which in turn suggests better classification performance. We have mutual information $\mathbb{I}$(input,output) = $\mathbb{H}$(output) - $\mathbb{H}$(output | input). Within the process of training, $\mathbb{H}$(output | input) decreases; hence, in order to make the mutual information $\mathbb{I}$(input,output) as small as possible, we should minimize the $\mathbb{H}$(output). Therefore, we consider that a lower entropy signifies better classification performance.

J Further Discussion on Dynamic Optimization

As mentioned, the EAFO methodology has shown the potential for dynamic optimization of activation during iterative training. Although we have not yet obtained an effective dynamic optimization method, we still want to provide some discussion here for future research work.

Dynamic optimization during iterative training might introduce considerable computational complexity. Such a problem of computational complexity might require the utilization of more efficient optimization algorithms (or optimizer) to address. Some insights are provided as follows. Firstly, we suggest conducting such activation optimization at a "batch-level" (gradient updates are typically done at the mini-batch level), which can stabilize the entire training process on one hand, and on the other hand, can reduce the computational complexity of dynamic optimization. That is to say, we can update the network parameters at a mini-batch level; while updating the activation at a batch level. Furthermore, we recommend using techniques similar to momentum methods for the design of optimizers, so that the model can retain information about the speed of gradient descent from the past, thereby accelerating convergence and reducing computing cost overall. Finally, we also would like to consider methods for the adaptive activation learning, similar to Adam, by adjusting the activation learning rates through calculating first and second moment estimates of the gradients.