University of Ibadan Journal of Science and Logics in ICT Research (UIJSLICTR)

ISSN: 2714-3627

A Journal of the Faculty of Computing, University of Ibadan, Ibadan, Nigeria

Volume 16 No. 1, January 2026

journals.ui.edu.ng/uijslictr
http://uijslictr.org.ng/
uijslictr@gmail.com

University of Ibadan
Journal of Science and Logics in ICT Research

Theoretical Analysis of the Universal Approximation Properties of GELU in Neural Networks

[1]Ohamuo V. U., [2]✉Adinya I. and [3]Udomboso C.

[1,2]*Department of Mathematics, University of Ibadan, Ibadan, Nigeria*
[3]*Department of Statistics, University of Ibadan, Ibadan, Nigeria*
iniadinya@gmail.com, vohamuo050@stu.ui.edu.ng, cgudomboso@gmail.com

Abstract
The choice of activation function is critical to a neural network's expressive power. The Rectified Linear Unit (ReLU) became a widely adopted standard due to its computational efficiency and effectiveness in mitigating vanishing gradients. However, ReLU also possesses well-known theoretical limitations, including non-differentiability at zero and the "Dying ReLU" problem, which can impede training. As an alternative, the smooth C^∞ Gaussian Error Linear Unit (GELU) has seen increasing adoption in state-of-the-art models. This paper provides a rigorous theoretical analysis of GELU's universal approximation properties. We formally prove that GELU satisfies the necessary and sufficient conditions of the Universal Approximation Theorem (UAT) by demonstrating that its C^∞ smoothness ensures its membership in the required function class $\mathcal{M}$, and that its non-terminating Taylor series expansion proves its essential non-polynomial nature. To support this theoretical analysis, we present a series of targeted empirical validations that visually and quantitatively demonstrate the practical consequences of these properties. Our experiments confirm that GELU's smoothness provides a tangible advantage over ReLU in approximating C^∞ functions, especially in deep neural networks; its non-zero negative gradient prevents the neuron death seen in ReLU; and its unbounded nature is superior to Tanh for modeling non-saturating functions. This work provides a complete theoretical explanation for GELU's power as a universal approximator, bridging the abstract UAT framework with the function's specific mathematical properties.

Keywords**:** *Gaussian Error Linear Unit (GELU), Universal Approximation Theorem, Activation Functions, Neural Network, Neurons, Compact Domain*

1.0 Introduction

Artificial neural networks (ANNs) demonstrated remarkable efficacy in modeling complex functions, a capability theoretically underpinned by the Universal Approximation Theorem (UAT). The UAT guarantees that a feedforward network can approximate any continuous function on a compact domain, provided it employs a suitable non-linear activation function [1]. This non-linearity is the essential component that distinguishes ANNs from traditional linear models [4].

Historically, the Rectified Linear Unit (ReLU), defined as $\sigma_{\text{ReLU}}(x) = \max(0, x)$, became a *de facto* standard. Its widespread adoption was driven by its computational simplicity and its practical success in mitigating the vanishing gradient problem prevalent in earlier sigmoidal functions [2]. However, ReLU also possesses well-known theoretical limitations. It is non-differentiable at the origin and, more critically, can suffer from the "dying ReLU" phenomenon, where neurons become permanently inactive and cease to learn, thereby impeding the training process [2].

In response to these limitations, the Gaussian Error Linear Unit (GELU) was introduced by Hendrycks and Gimpel [3]. Defined as $\sigma_{\text{GELU}}(x) = x \cdot \Phi(x)$, where $\Phi(x)$ is the Gaussian Cumulative Distribution Function, GELU is a smooth, C^∞ (infinitely differentiable) function. This smoothness, combined with its non-zero gradient for negative inputs, prevents

Ohamuo V. U., Adinya I. and Udomboso C. (2026). Theoretical Analysis of the Universal Approximation Properties of GELU in Neural Networks. *University of Ibadan Journal of Science and Logics in ICT Research (UIJSLICTR)*, Vol. 16 No. 1, pp. 100 - 113

the neuron death observed in ReLU. Its empirical advantages have been validated by its rapid adoption in state-of-the-art architectures, including the Transformer model like BERT [5], GPT [7] and Vision Transformers (ViT) [6].

Despite this widespread practical success, the theoretical foundations of GELU's expressive power remain less characterized than those of classical activations. While its empirical benefits are well-documented, a cohesive theoretical analysis formally linking GELU's specific mathematical properties to the UAT framework is an emerging area of research.

This paper bridges this gap by providing a rigorous theoretical analysis of GELU's universal approximation properties. We formally prove that GELU satisfies the necessary and sufficient conditions to be a universal approximator. Our theoretical contribution is twofold: first, we demonstrate that GELU's C^{∞} smoothness ensures its membership in the requisite function classes for UAT. Second, we prove its essential non-polynomial nature by analyzing its non-terminating Taylor series expansion, a key condition for universal approximation.

To substantiate this theoretical framework, we present a series of targeted empirical validations designed to visually and quantitatively demonstrate the practical consequences of these properties. We show that (i) GELU's smoothness provides a tangible advantage over ReLU in approximating complex C^{∞} functions; (ii) its non-zero negative gradient robustly prevents the "Dying ReLU" problem; and (iii) its unbounded nature offers superior modeling capacity for non-saturating functions compared to Tanh. This work thus provides a complete theoretical account for GELU's power as a universal approximator, connecting the abstract UAT framework directly to the function's specific mathematical properties.

2. Related Study and Background

We adopt the formal definition of a neural network as a parameterized family of functions composed of affine transformations and non-linear scalar maps.

2.1 Artificial Neural Networks

Following the standard mathematical formalism [1], let $L \in \mathbb{N}$ be the depth of the network, and $d_0, \dots, d_{L+1} \in \mathbb{N}$ be the widths of the layers. A function $\mu: \mathbb{R}^{d_0} \to \mathbb{R}^{d_{L+1}}$ is called a neural network if there exist weight matrices $\mathbf{W}^{(\ell)} \in \mathbb{R}^{d_{\ell+1} \times d_\ell}$ and bias vectors $\mathbf{b}^{(\ell)} \in \mathbb{R}^{d_{\ell+1}}$ for $\ell = 0, \dots, L$, such that:

$$\begin{aligned} \mathbf{x}^{(0)} &:= \mathbf{x} \textit{(Input Layer)} \\ \mathbf{x}^{(\ell)} &:= \sigma(\mathbf{W}^{(\ell-1)}\mathbf{x}^{(\ell-1)} + \mathbf{b}^{(\ell-1)}) \\ &\quad \text{for} \ell \in \{1, \dots, L\} \textit{(Hidden Layer(s))} \\ \mu(\mathbf{x}) &:= \mathbf{W}^{(L)}\mathbf{x}^{(L)} + \mathbf{b}^{(L)} \textit{(Output Layer)} \end{aligned} \quad (1)$$

Here, $\sigma: \mathbb{R} \to \mathbb{R}$ is the non-linear activation function applied element-wise. This architecture enables the network to model complex, hierarchical relationships between inputs and outputs

2..2. The Role and Evolution of Activation Functions

The activation function σ is the critical component that introduces non-linearity into the model. Without it, a neural network of any depth would collapse mathematically into a single affine transformation of the form $W_2(W_1x + b_1) + b_2 = (W_2W_1)x + (W_2b_1 + b_2)$, restricting its expressive power to linear models, which are often insufficient for approximating the complex, non-linear behaviors found in real-world functions or problems [4].

Historically, the choice of activation function has evolved to address optimization challenges:

a) Sigmoid and Tanh:
Early networks utilized bounded, smooth functions like the Sigmoid,$\sigma(x) = (1 + e^{-x})^{-1}$, and Hyperbolic Tangent, $\sigma(x) = \tanh(x)$. While theoretically sound, these functions suffer from the vanishing gradient problem in deep networks, where gradients approach zero for large inputs [2].

b) ReLU:
The Rectified Linear Unit, defined as $\sigma_{\text{ReLU}}(x) = \max(0, x)$, became the standard for deep learning due to its simplicity and computational efficiency. It mitigates vanishing gradients for positive inputs but introduces non-differentiability at the origin and the "dying neuron" problem, where negative inputs result in a zero gradient, effectively deactivating neurons [2].

c) GELU:
To address these limitations, Hendrycks and Gimpel (2016) introduced the Gaussian Error

Linear Unit. It is defined as the product of the input and the standard Gaussian Cumulative Distribution Function (CDF), $\Phi(x)$:

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2}\, dt \qquad (2)$$

$$\sigma_{\text{GELU}}(x) = x\,\Phi(x) = \frac{x}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2}\, dt \qquad (3)$$

GELU is a smooth, C^{∞} function that processes negative inputs probabilistically rather than hard-gating them.

2.3. The Universal Approximation Theorem

The theoretical capability of these networks to model any function is grounded in the Universal Approximation Theorem (UAT). We consider the version of the theorem that highlights the necessity of the activation function's non-polynomial nature.

Theorem 1 (Universal Approximation [1]). *Let $d \in \mathbb{N}$ and $\sigma \in \mathcal{M}$ (as defined in (4)). The class of shallow neural networks with scalar output, $\mathcal{N}_d^1(\sigma; 1)$, is dense in $C(K)$ for every compact $K \subseteq \mathbb{R}^d$ if and only if σ is not a polynomial.*

Here

$$\mathcal{M} := \{\sigma \in L^{\infty}_{\text{loc}}(\mathbb{R}) \mid \text{there exist intervals } I_1, \ldots, I_M \text{ partitioning } \mathbb{R}, \text{ s.t. } \sigma \in C^0(I_i) \text{ for all } i = 1, \ldots, M\}.$$

In other words, $\mathcal{M}$ contains activation functions σ such that:

- They are locally bounded (their output magnitude stays finite within any finite input interval).
- They are piecewise continuous, meaning they are continuous everywhere except possibly at a finite number of jump discontinuities.

This theorem implies that functions like GELU, which are smooth but non-polynomial (as proven by their infinite Taylor series), theoretically guarantee universality.

Normalization

We note that unlike Sigmoid or Tanh, modern activations like ReLU and GELU are unbounded. While this prevents gradient saturation, it can theoretically lead to exploding activations in deep networks. In practice, this is managed via normalization techniques such as Batch Normalization [9], which stabilize the distribution of layer inputs. While crucial for practical training, the mechanics of normalization are outside the scope of this paper, which focuses on the approximation properties of the activation function itself.

3. Methodology and Theoretical Properties of GELU

We now delve into the theoretical properties of the Gaussian Error Linear Unit (GELU). We begin by verifying its fundamental analytical properties which serve as the mathematical basis for our analysis. Subsequently, we leverage its analytic properties to demonstrate GELU's theoretical capability by formally proving that it satisfies the necessary and sufficient conditions of the Universal Approximation Theorem.

3.1 Analytical Properties

In this subsection, we examine the fundamental mathematical characteristics of the Gaussian Error Linear Unit (GELU). We specifically investigate its continuity, differentiability, smoothness (C^{∞}) boundedness, and non-polynomial property which distinguish it from piecewise-linear functions and ensure well-behaved gradients. Furthermore, we analyze its boundedness properties to characterize its asymptotic behavior and stability in deep networks.

3.1.1 Continuity

The exact definition, given previously in Equation (3), expresses GELU as the product of the identity function, x, and the Gaussian cumulative distribution function (CDF): $\sigma_{\text{GELU}}(x) = x\Phi(x)$. Since both the identity function $f(x) = x$ and the Gaussian CDF $\Phi(x)$, given in Equation (2) are continuous functions for all $x \in \mathbb{R}$, their product, the exact GELU function, is also continuous everywhere.

In practice, a computationally efficient approximation involving the hyperbolic tangent function is often used. It is defined in as:

$$\sigma_{\text{GELU}}(x) \approx 0.5x\left(1 + \tanh\left(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)\right)\right). \qquad (5)$$

The hyperbolic tangent function itself is defined as:

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}.$$

This function is continuous for all real inputs z, as its denominator is always positive. The approximate GELU formula is constructed from compositions and products of continuous functions (identity, polynomials, constants, and tanh). Since compositions and products of continuous functions remain continuous, this approximate form of GELU is also continuous for all $x \in \mathbb{R}$.

Having verified the continuity of both forms of GELU, we next investigate its differentiability and derive its derivatives.

3.1.2 Differentiability

Recall the definition in (3):

$$\sigma_{\text{GELU}}(x) = x\,\Phi(x)$$

where $\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-t^2/2}\,dt$, is the standard normal CDF defined in Equation (2). Thus,

$$\frac{d}{dx}(\sigma_{GELU}(x)) = \frac{d(x\Phi(x))}{dx}, \text{ by product rule}$$

$$\frac{d}{dx}(\sigma_{GELU}(x)) = x \cdot \frac{d(\Phi(x))}{dx} + \Phi(x) \cdot \frac{d(x)}{dx} \quad (6)$$

Now,

$$\frac{d}{dx}(\Phi(x)) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}, \quad (7)$$

and

$$\frac{d}{dx}(x) = 1. \quad (8)$$

Substituting (7) and (8) into (6), we obtain,

$$\frac{d}{dx}(\sigma_{GELU}(x)) = \sigma'_{\text{GELU}}(x) = \Phi(x) + \frac{x}{\sqrt{2\pi}} e^{-x^2/2} \quad (9)$$

Recall also, the GELU approximation from Equation (5):

$$\sigma_{\text{GELU}}(x) \approx 0.5x\left(1 \tanh\left(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)\right)\right).$$

For simplicity, let $k = \sqrt{2/\pi}$ and $c = 0.044715$. Let the argument of the tanh function be $A(x) = k(x + cx^3)$. Then the approximation is

$$\sigma_{\text{GELU}}(x) \approx 0.5x(1 + \tanh(A(x)))$$

We use the product rule $(uv)' = u'v + uv'$, with $u(x) = 0.5x$ and $v(x) = 1 + \tanh(A(x))$. So,

$u'(x) = 0.5$ and $v'(x) = \frac{d}{dx}(1 + \tanh(A(x))) = \frac{d}{dx}(\tanh(A(x)))$

To find $\frac{d}{dx}(\tanh(A(x)))$, we use the chain rule and the fact that $\frac{d}{dz}(\tanh(z)) = \text{sech}^2(z)$: $\frac{d}{dx}(\tanh(A(x))) = \text{sech}^2(A(x)) \cdot A'(x)$ (10)

Now we need $A'(x)$:

$$A'(x) = \frac{d}{dx}\left(k(x + cx^3)\right) = k(1 + 3cx^2 = \sqrt{\frac{2}{\pi}}(1\ 3\ 0.044715x^2) \quad (11)$$

Substituting (11) into (10) gives $v'(x)$:

$$v'(x) = \text{sech}^2\left(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)\right) \cdot \sqrt{\frac{2}{\pi}}(1 + 0.134145x^2)$$

Now, applying the product rule for $\frac{d}{dx}(0.5x(1 + \tanh(A(x))))$:

$$\sigma'_{\text{GELU}}(x) = u'(x)v(x) + u(x)v'(x) = 0.5 \cdot (1 + \tanh(A(x))) + 0.5x \cdot v'(x)$$

$$= 0.5\left(1 + \tanh\left(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)\right)\right) + 0.5x \cdot \text{sech}^2\left(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)\right) \cdot \sqrt{\frac{2}{\pi}}(1 + 0.134145x^2)$$

This yields the derivative of the approximate GELU function:

$$\sigma'_{\text{GELU}}(x) = 0.5\big(1 + \tanh(A(x))\big) + 0.5x \cdot \operatorname{sech}^2(A(x)) \cdot A'(x)$$

where $A(x) = \sqrt{\frac{2}{\pi}}(x + 0.044715x^3)$ and $A'(x) = \sqrt{\frac{2}{\pi}}(1 + 0.134145x^2)$. Thus,

$$\sigma'_{\text{GELU}}(x) \approx 0.5\left(1 + \tanh\left(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)\right)\right) + 0.5x \cdot \operatorname{sech}^2\left(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)\right) \cdot \sqrt{\frac{2}{\pi}}(1 + 0.134145x^2) \quad (12)$$

Both the exact (9) and approximate (12) forms confirm that GELU is continuously differentiable (C^1) for all $x \in \mathbb{R}$. This property is essential as it ensures gradients required for backpropagation are well-defined everywhere.

In contrast to ReLU, whose derivative vanishes for $x < 0$, GELU's derivative remains non-zero across much of the negative domain and can even take slightly negative values [2]. The consequences of this behavior for the function's overall smoothness will be examined next.

3.1.3 Smoothness (C^∞)

Recall, again, the exact definition (3): $\sigma_{\text{GELU}}(x) = x\Phi(x)$. The components $f(x) = x$ and the Gaussian CDF $\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-t^2/2}\,dt$, are both known to be infinitely differentiable (C^∞). Since standard operations like multiplication preserve infinite differentiability, their product, the exact GELU function, must also be C^∞. Therefore, GELU is classified as a smooth activation function.

This smoothness implies that all higher-order derivatives of GELU exist and are continuous. This level of regularity contributes to a well-behaved function, which can be beneficial for the theoretical analysis and potentially for the stability of optimization landscapes during neural network training. (Similarly, the common 'tanh'-based approximation (5) is also smooth as it is built from C^∞ components.)

3.1.4 Boundedness

To determine if GELU is bounded below, we look for its minimum value. The derivative $\sigma'_{\text{GELU}}(x) = \Phi(x) + x\phi(x)$ equals zero at $x \approx -0.75$. Evaluating the function at this critical point yields $\sigma_{\text{GELU}}(-0.75) \approx -0.17$. Since $\lim_{x\to-\infty}\sigma_{\text{GELU}}(x) = 0$ and $\lim_{x\to\infty}\sigma_{\text{GELU}}(x) = \infty$ (as shown below), this critical point corresponds to the global minimum. Thus, GELU is bounded below by approximately -0.17.

To check if GELU is bounded above, we examine its limit as x approaches positive infinity. We know that $\lim_{x\to\infty}\Phi(x) = 1$. Therefore:

$$\lim_{x\to\infty}\sigma_{\text{GELU}}(x) = \lim_{x\to\infty} x\Phi(x) = \lim_{x\to\infty}(x \cdot 1) = \infty. \quad (13)$$

Since the limit is infinite, GELU is unbounded above.

This theoretical unboundedness could potentially pose challenges during training. Without constraints, if the inputs z'_j to the GELU function in layer j (after the affine transformation $z'_j = W_j z_{j-1} + b_j$) grow large due to increasing weights and biases during learning, the output $z_{j+1} = \sigma_{\text{GELU}}(z'_j)$ could also grow large. This growth can compound across multiple layers ($|z'_j| \leq |W_j||z'_{j-1}| + |b_j|$), potentially leading to very large activation values and contributing to issues like exploding gradients.

However, this theoretical concern is often mitigated in practice through the use of normalization techniques such as Batch Normalization (BN) [9], Layer Normalization (LN) [10], or Group Normalization (GN) [11]. These methods are typically applied after the affine transformation but before the activation function within a layer. The practical upper bound, induced by normalization, helps constrain activation values and mitigates the risk of gradient instability associated with GELU's theoretical unboundedness.

3.1.5 Non-Polynomial Property

To further understand the structure and analytical properties of the GELU activation function, particularly how its smoothness translates into expressive power, it is instructive to analyze its Taylor expansion around the origin, in particular, its Maclaurin expansion. This

expansion is a key property that reveals the polynomial components embedded within the function, which are fundamental to the mechanisms underlying universal approximation.

Now, $\sigma_{\text{GELU}}(x) = x\,\Phi(x)$, where $\Phi(x)$ is the standard normal CDF (2) and its derivative is the standard normal PDF $\phi(x) = \Phi'(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$. To derive the Taylor series for $\sigma_{\text{GELU}}(x)$, we first compute the successive derivatives of $\Phi(x)$ at $x = 0$. Using $\Phi(0) = 1/2$ and repeated differentiation involving $\phi(x)$, we find:

$$\begin{aligned}
&\Phi'(x) = \phi(x),\\
&\Phi''(x) = \phi'(x) = -x\,\phi(x),\\
&\Phi'''(x) = \phi''(x) = (x^2 - 1)\,\phi(x),\\
&\Phi^{(n)}(0) = \begin{cases} \Phi(0) = 1/2 & n = 0 \\ \phi^{(n-1)}(0) & n \geq 1 \end{cases}
\end{aligned} \quad (14)$$

Evaluating these at $x = 0$ yields:

$$\begin{aligned}
&\Phi(0) = 1/2, \quad \Phi'(0) = \frac{1}{\sqrt{2\pi}}, \quad \Phi''(0)\\
&= 0, \quad \Phi'''(0) = -\frac{1}{\sqrt{2\pi}}, \quad \Phi^{(4)}(0)\\
&= 0, \quad \Phi^{(5)}(0)\\
&= \frac{3}{\sqrt{2\pi}}, \ldots
\end{aligned} \quad (15)$$

Applying the Maclaurin expansion for $\Phi(x)$ using these derivatives gives:

$$\begin{aligned}
\Phi(x) &= \Phi(0) + \Phi'(0)\,x + \frac{\Phi''(0)}{2!}x^2 + \frac{\Phi'''(0)}{3!}x^3 + \\
&\quad \frac{\Phi^{(4)}(0)}{4!}x^4 + \frac{\Phi^{(5)}(0)}{5!}x^5 + \cdots\\
&= 1/2 + \frac{1}{\sqrt{2\pi}}x - \frac{1}{6\sqrt{2\pi}}x^3 + \\
&\quad \frac{3}{120\sqrt{2\pi}}x^5 + \mathcal{O}(x^7)\\
&= 1/2 + \frac{x}{\sqrt{2\pi}} - \frac{x^3}{6\sqrt{2\pi}} + \frac{x^5}{40\sqrt{2\pi}} + \mathcal{O}(x^7).
\end{aligned} \quad (16)$$

Multiplying by x gives the Maclaurin expansion for GELU:

$$\begin{aligned}
\sigma_{\text{GELU}}(x) &= x\,\Phi(x)\\
&= \frac{x}{2} + \frac{x^2}{\sqrt{2\pi}} - \frac{x^4}{6\sqrt{2\pi}} + \frac{x^6}{40\sqrt{2\pi}}\\
&\quad + \mathcal{O}(x^8).
\end{aligned} \quad (17)$$

This expansion makes explicit the infinite series of polynomial terms embedded within GELU. Since the series does not terminate (higher odd derivatives of $\Phi(x)$ at 0 are generally non-zero), $\sigma_{\text{GELU}}(x)$ cannot be represented as a finite polynomial. This non-polynomial nature is a critical property satisfying the requirements of the Universal Approximation Theorem.

The presence of these polynomial components demonstrates GELU's ability to generate polynomial structures of various degrees. This capacity is crucial for expressiveness, as linear combinations of shifted and scaled activation functions can leverage these components to approximate arbitrary polynomials on compact domains.

Therefore, the analytical properties derived thus far – continuity, infinite differentiability (C^∞), local boundedness (all ensuring membership in $\mathcal{M}$), and the non-polynomial nature confirmed by the Taylor expansion collectively satisfy the conditions required by the Universal Approximation Theorem. This provides the theoretical foundation for the universal approximation capability of GELU-based neural networks.

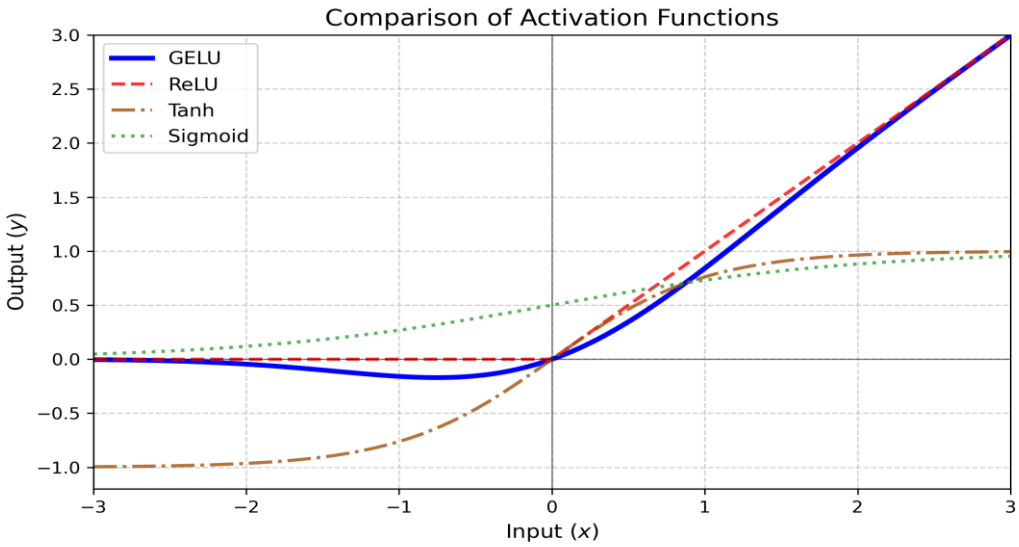

Figure 1: Comparison of GELU with standard activation functions. Note that GELU is smooth (unlike ReLU), non-monotonic, and unbounded above (unlike Tanh and Sigmoid).

3.2 Proof of the Universal Approximation Theorem using GELU

Having derived the key analytical properties of GELU in, including its smoothness (C^∞) and non-polynomial nature, we now formally demonstrate that it satisfies the conditions of the Universal Approximation Theorem. We focus on the standard setting of approximating scalar-valued continuous functions ($m = 1$) on compact domains using shallow networks ($L = 1$) in $\mathcal{N}_d^m(\sigma; L) := \bigcup_{n\in\mathbb{N}} \mathcal{N}_d^m(\sigma; L, n)$.

For;

$$\mathcal{N}_d^m(\sigma; L, n) := \{\mu: \mathbb{R}^d \to \mathbb{R}^m \mid \mu\text{depth}(\mu) \leq L,\ \text{width}(\mu) \leq n\}. \quad (18)$$

Here, $\mathcal{N}_d^m(\sigma; L, n)$ denotes the set containing all possible functions that can be represented by a neural network having d-dimensional input, m-dimensional output, activation function σ, depth of at most L, width of at most n.

First, we require the concept of compact convergence to define density in the space of continuous functions over $\mathbb{R}^d$.

Definition 2 (Compact Convergence [1]). *Let $d \in \mathbb{N}$. A sequence of functions $(f_n)_{n\in\mathbb{N}}$, $f_n: \mathbb{R}^d \to \mathbb{R}$, converges compactly to $f: \mathbb{R}^d \to \mathbb{R}$ (written $f_n \xrightarrow{cc} f$) if for every compact set $K \subseteq \mathbb{R}^d$, $\lim_{n\to\infty} \sup_{\mathbf{x}\in K} |f_n(\mathbf{x}) - f(\mathbf{x})| = 0$. The space $C(\mathbb{R}^d)$ is equipped with the topology induced by this convergence. A set $\mathcal{H} \subseteq C(\mathbb{R}^d)$ is dense if its closure under compact convergence, $\overline{\mathcal{H}}^{cc}$, contains $C(\mathbb{R}^d)$.*

The UAT proof strategy relies significantly on the density of polynomials within continuous functions, established by the Stone-Weierstrass Theorem.

Theorem 3 (Stone-Weierstrass Theorem [1]). *Let $K \subseteq \mathbb{R}^d$ be a compact set, and let $\mathcal{H} \subseteq C(K)$ be a set of continuous real-valued functions on K such that:*

1. *$\mathcal{H}$ contains the constant functions.*
2. *$\mathcal{H}$ separates points (for all distinct $\mathbf{x}, \mathbf{y} \in K$, there exists $f \in \mathcal{H}$ such that $f(\mathbf{x}) \neq f(\mathbf{y})$).*
3. *$\mathcal{H}$ is an algebra (i.e., closed under addition, multiplication, and scalar multiplication).*

Then $\mathcal{H}$ is dense in $C(K)$ with respect to the supremum norm.

The UAT proof also uses the following key lemmas:

Lemma 4 (Reduction to the Univariate Case [1]). *If $\mathcal{H} \subseteq C(\mathbb{R})$ is dense in $C(\mathbb{R})$ with respect to compact convergence, then for every $d \in \mathbb{N}$, the set*

$$\text{span}\{\mathbf{x} \mapsto g(\mathbf{w}^\top \mathbf{x}) \mid \mathbf{w} \in \mathbb{R}^d,\ g \in \mathcal{H}\}$$

is dense in $C(\mathbb{R}^d)$ with respect to compact convergence.

Lemma 5. *If $\sigma \in C^\infty(\mathbb{R})$ and σ is not a polynomial, then $\mathcal{N}_1^1(\sigma;1)$ is dense in $C(\mathbb{R})$ with respect to compact convergence.*

Remark on Mollification: The general proof of the Universal Approximation Theorem for any $\sigma \in \mathcal{M}$ also requires lemmas involving mollification (convolution with smooth functions) to handle cases where σ is not C^∞ (e.g., ReLU) . However, since we proved that GELU is smooth (C^∞), Lemma 5 applies directly, and the mollification steps are not strictly necessary for this specific case.

We restate the Universal Approximation Theorem in the context relevant here:

Theorem 6 (Universal Approximation). *Let $d \in \mathbb{N}$ and $\sigma \in \mathcal{M}$ (as defined in (4)). The class of shallow neural networks with scalar output, $\mathcal{N}_d^1(\sigma;1)$, is dense in $C(K)$ for every compact $K \subseteq \mathbb{R}^d$ if and only if σ is not a polynomial.*

Proof. We prove the "if and only if" statement.

Sufficiency *("if" part):* We assume $\sigma = \sigma_{\text{GELU}}$, which we have already seen to be smooth and non-polynomial, and show $\mathcal{N}_d^1(\sigma_{\text{GELU}};1)$ is dense in $C(K)$ for every compact $K \subseteq \mathbb{R}^d$. By Lemma 4, the task reduces to demonstrating density in the univariate case; that is, showing $\mathcal{N}_1^1(\sigma_{\text{GELU}};1)$ is dense in $C(\mathbb{R})$ with respect to compact convergence. We also showed that GELU is infinitely differentiable (C^∞), ensuring $\sigma_{\text{GELU}} \in \mathcal{M}$. Furthermore, its Taylor expansion (17) confirms that GELU is not a polynomial. Since σ_{GELU} is both C^∞ and non-polynomial, it directly satisfies the hypotheses of Lemma 5. Therefore, Lemma 5 applies immediately to GELU, allowing us to conclude that $\mathcal{N}_1^1(\sigma_{\text{GELU}};1)$ is dense in $C(\mathbb{R})$ with respect to compact convergence. (Note that the general proof strategy for arbitrary $\sigma \in \mathcal{M}$ typically involves mollification [1] to handle non-smooth cases; however, these steps are unnecessary here due to GELU's inherent smoothness discussed earlier). Finally, reapplying Lemma 4 extends the established univariate density to the multivariate case, proving that $\mathcal{N}_d^1(\sigma_{\text{GELU}};1)$ is dense in $C(\mathbb{R}^d)$ under compact convergence. This implies density in $C(K)$ for any compact $K \subseteq \mathbb{R}^d$.

Necessity *("only if" part):* Assume $\mathcal{N}_d^1(\sigma;1)$ is dense in $C(K)$. We need to show σ is not a polynomial. Suppose, for contradiction, that σ is a polynomial of degree k. Then any function $\mu^* \in \mathcal{N}_d^1(\sigma;1)$ of the form

$$\mu^*(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i \, \sigma(\mathbf{w}_i^\top \mathbf{x} + b_i) + c$$

is also a polynomial in $\mathbf{x}$, with a degree of at most k. The set $\mathcal{N}_d^1(\sigma;1)$ is therefore contained within the finite-dimensional space of polynomials $\mathcal{P}_k(\mathbb{R}^d)$. However, $C(K)$ is infinite-dimensional for any compact K with a non-empty interior. A finite-dimensional subspace cannot be dense in an infinite-dimensional space. This contradiction implies that σ cannot be a polynomial. □

This completes the proof demonstrating that shallow neural networks utilizing the GELU activation function possess the universal approximation property, grounded in its established analytical characteristics.

4. Results and Empirical Validation of GELU

To empirically validate the theoretical properties of GELU discussed above, a clear and reproducible experimental methodology is required.

4.1. Target Function

We select three 1D target functions. Each is chosen to highlight a specific characteristic discussed earlier.

1. Smooth, Periodic Function (f_1): To test the approximation of a smooth (C^∞) function and compare the fit of smooth GELU versus non-smooth ReLU activations, we use:

$$f_1(x) = \sin(2\pi x) + 0.5x, \quad x \in [-1,1]$$

This function combines periodicity with a linear trend.

2. Negative-Biased Function (f_2): To specifically investigate the dying ReLU problem, we use a function whose output is mostly negative, which will likely push neuron pre-activations to be negative:

$$f_2(x) = 0.1x - 1, \quad x \in [-2,2]$$

3. Unbounded Function (f_3)**:** To test the "non-saturating" (unbounded above) property of GELU against the saturating (bounded) property of Tanh, we use a simple parabola:

$$f_3(x) = 0.5x^2, \quad x \in [-2,2]$$

We will train on data from $[-2,2]$ but evaluate on a wider range to observe extrapolation.

4.2. Network Architectures

To test the effect of depth versus width, we define two distinct feedforward network (FNN) architectures, both mapping $\mathbb{R} \to \mathbb{R}$ (i.e., $d_0 = 1$ and $d_{L+1} = 1$):

- Shallow Network: A network with a single hidden layer ($L = 1$). This architecture has a width of 500 neurons ($d_1 = 500$).

- Deep Network: A network with four hidden layers ($L = 4$). This architecture has a uniform width of 125 neurons in each hidden layer ($d_1 = d_2 = d_3 = d_4 = 125$).

This design ensures both architectures have the same total number of hidden neurons (500). For both architectures, the respective activation function (σ) will be applied to all hidden layers as per $\mathbf{x}^{(\ell)} := \sigma(\mathbf{W}^{(\ell-1)}\mathbf{x}^{(\ell-1)} + \mathbf{b}^{(\ell-1)})$ for$\ell \in \{1, \dots, L\}$. The final output layer ($\mathbf{x}^{(L+1)}$) is linear, as is standard for regression tasks $\mu(\mathbf{x}) := \mathbf{W}^{(L)}\mathbf{x}^{(L)} + \mathbf{b}^{(L)}$.

Compared Activation Functions

For our experiments, we compare the performance of GELU against two primary baselines: Tanh and ReLU. Each activation function represents a distinct functional class, allowing us to empirically test the theoretical properties, such as smoothness and boundedness.

4.3. Training and Evaluation Setup

All experiments are conducted under a consistent training setup to ensure a fair comparison among activation functions. The entire experimental framework is implemented in Python using the PyTorch deep learning library, which provides automatic differentiation, optimized tensor computations, and reproducible control over model training. Visualization and result analysis are performed using standard Python packages such as matplotlib and pandas.

Each network is trained to approximate a given target function $f_i(x)$ by learning from synthetic data points generated directly from that function. Specifically, we sample $N = 200$ input values x_i uniformly over the prescribed domain (e.g., $[-2,2]$) and compute their corresponding outputs as
$y_i = f_i(x_i)$,

which serve as the reference values the network aims to approximate. Thus, the data are not collected from observation but generated analytically from the target function itself, allowing for controlled and reproducible experiments.

The network is trained to minimize a *loss function*, which quantifies how far the network's predictions are from the true target values. We employ the *mean squared error* (MSE), defined as

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - f_i(x_i))^2,$$

where $\hat{y}_i$ denotes the network's predicted output for input x_i , and θ represents all trainable parameters (weights and biases). The smaller the value of $\mathcal{L}(\theta)$, the closer the network's output is to the desired function.

During training, the parameters are iteratively updated in the direction that reduces the loss. For a neuron with pre-activation
$z = \mathbf{w}^{\top}\mathbf{x} + b$,

the updates at each training step follow the gradient descent rule:

$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} - \eta\,\frac{\partial \mathcal{L}}{\partial \mathbf{w}}, \qquad b_{\text{new}} = b_{\text{old}} - \eta\,\frac{\partial \mathcal{L}}{\partial b},$$

where η (the *learning rate*) determines how large a step is taken in each update. A small η results in gradual learning, while a large value can lead to unstable or divergent updates.

To improve the stability and efficiency of training, we use the *Adam optimizer* (Adaptive Moment Estimation) with a learning rate of 10^{-3}. Adam extends standard gradient descent by maintaining running estimates of the first and second moments (mean and variance) of the gradients for each parameter. This allows it to automatically adapt the effective learning rate

for every parameter, leading to faster and more reliable convergence.

After training, model performance is evaluated on a separate validation set of 1000 clean, uniformly sampled points from the same domain. The final mean squared error on this validation set is reported as the measure of approximation quality. All models are trained for 1000 epochs.

4.4. Experiment 1: Approximation of a Smooth Function

The target function is defined as
$f_1(x) = \sin(2\pi x) + 0.5x,$
which combines a periodic sinusoidal term with a mild linear trend. It is chosen for its smoothness and nonlinearity, making it suitable for testing how well each activation can represent continuous mappings.

Two feedforward neural networks, shallow and deep network, were trained separately using the same dataset.

The goal of this experiment is to compare how closely each activation function can approximate $f_1(x)$ under the same conditions, first using a shallow network and then a deeper network.

4.4.1. Shallow Network Results

Here, we compare how shallow networks using GELU and ReLU activations approximate the target function $f_1(x) = \sin(2\pi x) + 0.5x$.

From Figure 2, we observe distinct behaviors between the two activations. The ReLU-based network achieved a significantly tighter fit to the target function, accurately capturing the full amplitude of the sine wave and converging rapidly to a near-zero MSE.

However, consistent with its theoretical piecewise-linear nature, the ReLU approximation exhibited visible "kinks" at the function's turning points. In contrast, the GELU-based network produced a perfectly smooth, continuously differentiable curve, validating its theoretical smoothness properties. However, in this specific shallow architecture, GELU struggled to capture the full magnitude of the target function, resulting in a higher final loss compared to ReLU. This suggests that while GELU guarantees smoothness, ReLU may offer faster approximation and lower error for simple 1D functions in shallow networks.

4.4.2. Deep Network Results

The network depth is increased to four hidden layers, each with 125 neurons, while maintaining the same data and training setup as in the shallow case. The aim is to examine how increasing depth affects performance for ReLU and GELU activations.

Increasing the network depth to four hidden layers (width 125) significantly enhances the representational capacity of both architectures.

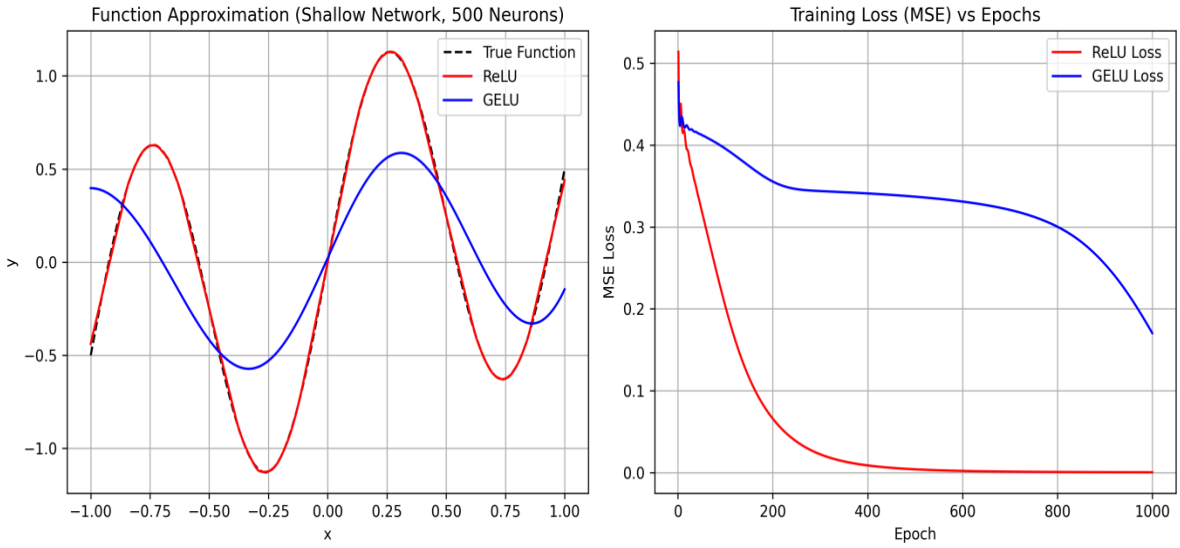

Figure 2: Function approximation and training loss for shallow networks (500 neurons, 1000 epochs) with ReLU and GELU activation functions.

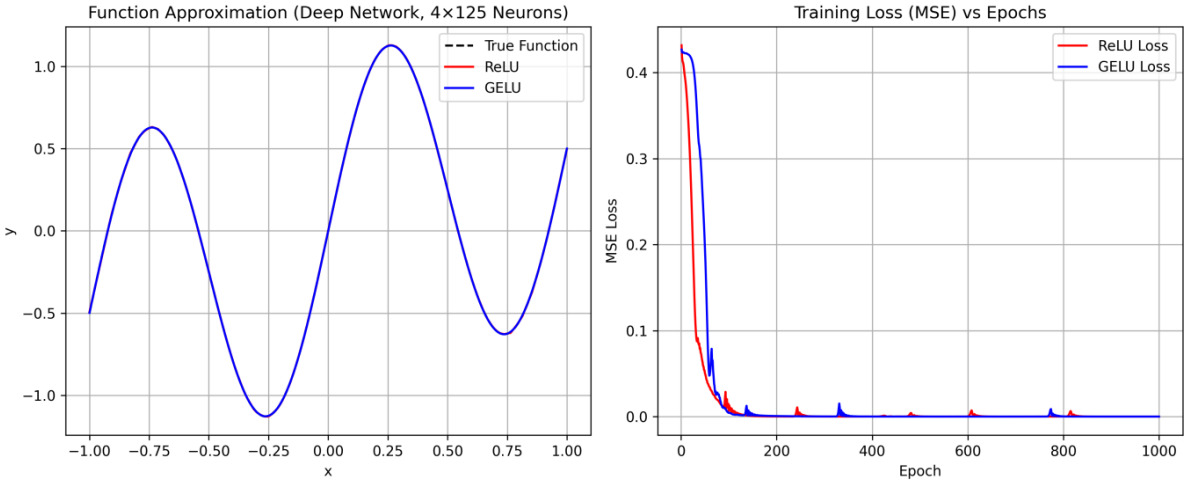

Figure 3: Function approximation and training loss for deep networks using ReLU and GELU activation functions.

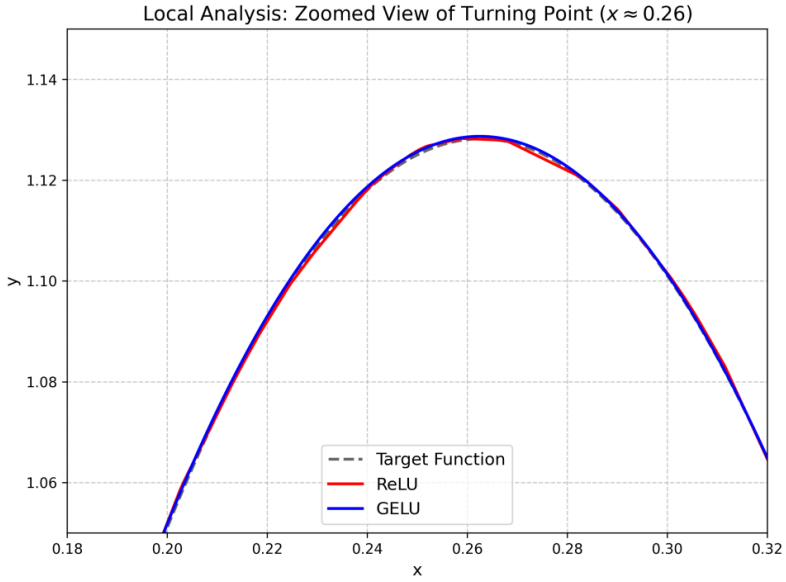

Figure 4: Local analysis of approximation near turning point.

As illustrated in Figure 3, both the ReLU and GELU networks achieve a high-fidelity approximation of the target function $f_1(x)$, with the learned curves closely tracking the target function across the entire domain. Unlike the shallow case, the deep architectures possess sufficient capacity to minimize the Mean Squared Error (MSE) to negligible levels for both activation functions.

However, a critical theoretical distinction remains evident in the local geometry of the approximation. Figure 4 presents a magnified view of the function's local maximum near $x = 0.26$. The ReLU network, constrained by its piecewise-linear nature, approximates the smooth peak using a polygonal chain—a series of linear segments connected at non-differentiable "kinks." While this minimizes the error metric, it fails to capture the intrinsic curvature of the data-generating process. In contrast, the GELU network produces a continuously differentiable curve that naturally models the turning point without geometric artifacts. This confirms that while deep ReLU networks can approximate smooth functions with arbitrary precision, they do so by increasing the density of linear segments, whereas GELU networks fundamentally preserve the smoothness (C^∞) of the target function.

4.5. *Experiment 2: Visualizing the "Dying ReLU" Problem*

To investigate the "dying ReLU" phenomenon, a deep neural network with four hidden layers of 125 neurons each was trained using two activation functions: ReLU and GELU. The network was trained on a simple linear mapping $f_2(x) = 0.1x - 1$,

chosen specifically because it produces mostly negative outputs, which easily trigger ReLU saturation at zero.

After training, the activations of the last hidden layer were collected for all 200 training samples. Since this layer contains 125 neurons, the total number of activation values analyzed was $200 \times 125 = 25{,}000$.

These activations were visualized as histograms, showing how many neurons output near-zero (inactive) values versus nonzero (active) ones.

Figure 5 visually confirms the severity of the "Dying ReLU" phenomenon. The ReLU distribution exhibits a massive spike at zero, indicating that **61.2%** (approximately 15,300) of neurons are inactive. Because ReLU hard-zeros all negative inputs, these neurons effectively cease learning, destroying the gradient signal. In contrast, GELU maintains a dispersed distribution shifted toward its negative minimum (≈ -0.17). By smoothly compressing negative inputs rather than strictly deleting them, GELU preserves essential gradient flow, ensuring that the vast majority of neurons remain active participants in the optimization process.

4.7 *Visualizing Boundedness and Unboundedness.*

We trained two shallow neural networks (one using Tanh, the other using GELU activations) to learn the quadratic function $f_3(x) = 0.5x^2$, using training samples from $x \in [-2,2]$. The learned functions were then evaluated over an extended range $x \in [-4,4]$.

As shown in Figure 6, both networks fit the data well within the training range.

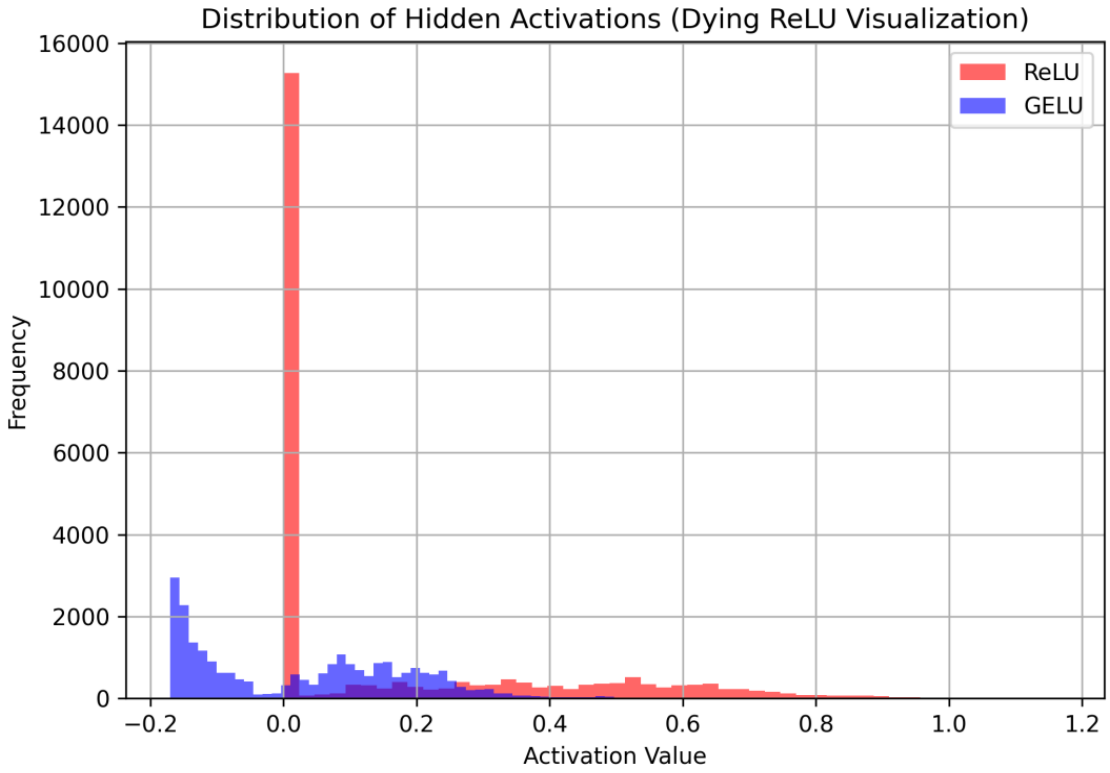

Figure 5: Histogram of hidden-layer activations for ReLU and GELU after training on $f_2(x) = 0.1x - 1$.

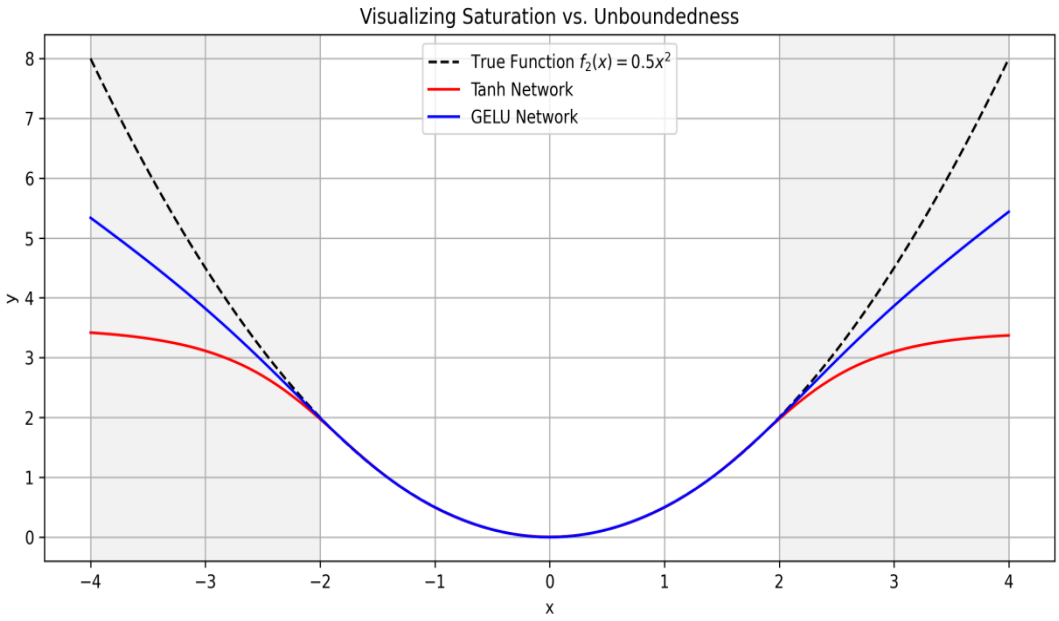

Figure 6: Learned functions for Tanh and GELU networks on $f_3(x) = 0.5x^2$. The shaded region indicates extrapolation beyond the training range.

However, beyond $x = \pm 2$, the Tanh model quickly saturates near $y \approx 3$, due to its boundedness (bounded by $-1\&1$), failing to follow the parabolic rise of the true function. In contrast, the GELU network remains unbounded and continues to increase smoothly, preserving the quadratic growth even in the extrapolation region. This illustrates how the bounded nature of Tanh limits its representational capacity outside the data range, while GELU maintains a more flexible, less saturating response.

5. Conclusion

This study has provided a comprehensive theoretical analysis of the Gaussian Error Linear Unit (GELU), explicitly demonstrating its compliance with the Universal Approximation Theorem (UAT). By deriving the infinite Taylor series expansion of GELU, we verified the essential non-polynomial nature required by the Leshno et al. theorem, confirming that GELU networks possess the theoretical capacity to approximate any continuous function on a compact domain. While the general UAT is a well-established result, this work bridges the gap between abstract approximation theory and the specific analytical properties of GELU, such as its C^∞ smoothness.

Our empirical evaluations further substantiated these theoretical assertions. The experiments confirmed that GELU's smoothness leads to approximation profiles free of the geometric artifacts ("kinks") observed in ReLU networks, while its unbounded nature allows it to model non-saturating trends where functions like Tanh fail. Ultimately, this work confirms that GELU's empirical success in state-of-the-art architectures is underpinned by robust mathematical properties that balance the optimization benefits of unboundedness with the representational density of smooth, non-polynomial functions.

References

[1] Petersen, P., & Zech, J. (2025). *Mathematical theory of deep learning*. arXiv preprint arXiv:2407.18384v3 [cs.LG]. (Original submission July 2024). https://arxiv.org/abs/2407.18384v3

[2] Lee, M. (2023). *GELU Activation Function in Deep Learning: A Comprehensive Mathematical Analysis and Performance*. arXiv preprint arXiv:2305.12073v2 [cs.LG]. https://arxiv.org/abs/2305.12073v2

[3] Hendrycks, D., & Gimpel, K. (2016). *Gaussian error linear units (gelus)*. arXiv preprint arXiv:1606.08415. https://arxiv.org/abs/1606.08415

[4] Sharma, S., Sharma, S., & Athaiya, A. (2020). Activation functions in neural networks. *International Journal of Engineering Applied Sciences and Technology*, *4*(12), 310-316. http://www.ijeast.com/papers/310-316,Tesma412,IJEAST.pdf

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT*, pp. 4171–4186, 2019.

[5] A. Dosovitskiy *et al.*, "An image is worth 16 × 16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations (ICLR)*, 2021.

[7] T. Brown *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[8] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, **323**(6088), 533–536.

[9] Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pp. 448–456.

[10] Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer Normalization. *arXiv preprint arXiv:1607.06450*.

[11] Wu, Y., & He, K. (2018). Group Normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 3–19.