CSO 211 PROJECT REPORT

Novel Activation Function

Geetha Pitta (23124017)

Sanika Burra (23124013)

Institution: IIT(BHU), Varanasi

Department: Mathematical Sciences

Date of Submission: 30/11/2024

Introduction:

The primary objective of this novel activation function is to provide an alternative to popular activation functions like Relu, sigmoid, and tanh.

- We focused on reducing issues like dying neurons , vanishing gradient, exploding gradient ensuring good computational efficiency and non - linearity.

Problem Statement:

Develop an activation function that satisfies the following requirements:

Non-linearity: Effectively introduces non-linearity to enable the network to capture complex patterns.

Gradient Issues: Addresses the challenges of vanishing or exploding gradients during backpropagation.

Smoothness: Guarantees smoothness and differentiability throughout its domain to support gradient-based optimization.

Bounded Output: Optionally restricts the output within a specific range to avoid excessively large activations.

Efficiency: Ensures computational simplicity for practical implementation.

Methodology:

We chose our custom activation functions based on its properties listed below. We proposed two functions out of which one is best. Unlike ReLU, it avoids the issue of dying neurons by ensuring a gradient exists for all inputs, including negative values. Compared to tanh, it does not compress large positive inputs, allowing better information flow. Additionally, the use of smooth transitions makes it easier for optimization algorithms to learn effectively.

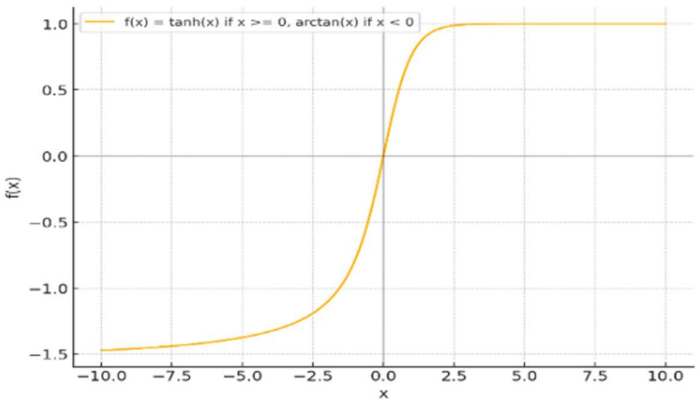
It ensures computational efficiency and stability, making it a better choice for addressing gradient-related problems while retaining flexibility for complex patterns.

Advantages of proposed Function2(RTHIN):

1. No Exploding Gradients: The bounded negative region (tanh + atan) avoids the risk of large negative gradients during training.
2. Smooth Negative Handling: The combination of tanh and atan ensures smoother gradient flow for negative inputs compared to ReLU's zero or ELU's exponential behavior.
3. No Exploding Gradients: The bounded negative region (tanh + atan) avoids the risk of large negative gradients during training.
4. Linear Behavior in Positive Region: Like ReLU, it retains linear behavior for positive values but adds more flexibility in the negative region.
5. Better Convergence for Complex Models: Adding non-linear yet bounded terms in the negative region can help the model explore a richer set of representations while maintaining stability.

Proposed Function 1:

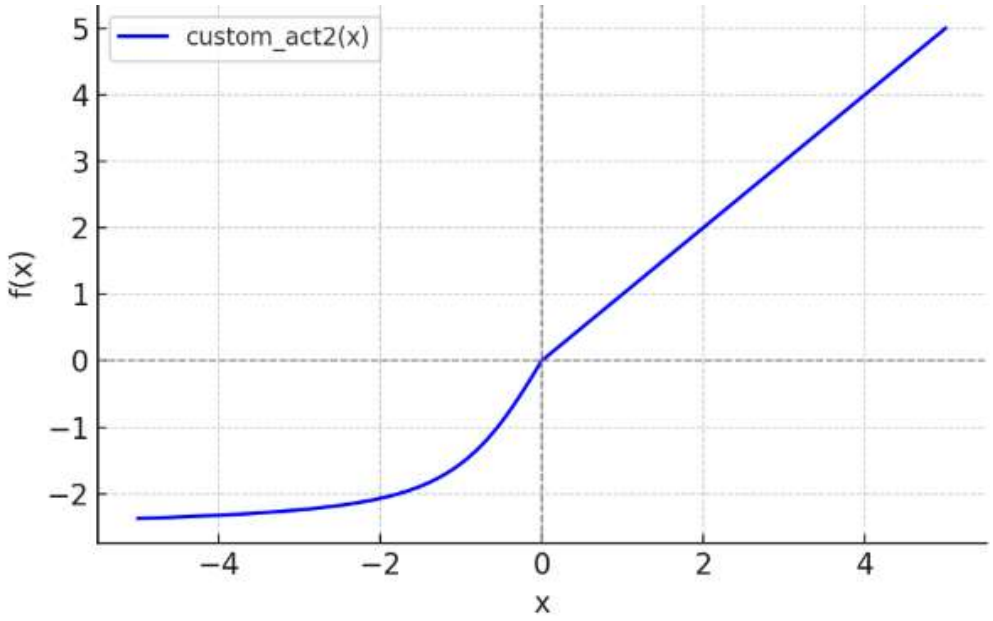
$$f(x) = \begin{cases} \tanh(x) & \text{if } x \geq 0 \\ \tan^{-1}(x) & \text{if } x < 0 \end{cases}$$



- $\tanh(x)$ is the hyperbolic tangent function, defined as:

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

Proposed Function 2(RTHIN):



$$\text{custom\_act2}(x) = \begin{cases} x & \text{if } x \geq 0, \\ \tanh(x) + \tan^{-1}(x) & \text{if } x < 0. \end{cases}$$

Results:

Data set 1- TIME SERIES DATA SET (Battery_RUL)

MODEL 1

This model combines **Temporal Convolutional Network (TCN)** blocks with dilated convolutions for capturing long-term dependencies and a **GRU layer** for sequential modelling, followed by dense layers for regression.

Function	Mean Absolute Error	Mean Squared Error	Root Mean Square Error
Proposed function 2(RTHIN)	2.647822058801626	83.98935330305139	9.164570546569621
Proposed function 1	4.7558883747940675	124.654591766923303	4.96533903846689
Elu	5.075830654060809	196.43295347802166	40.29886827217569
Selu	7.40494887265982	122.04272270882929	11.047294814063273
tanh	8.637200773869017	209.7004474065108	45.79306141835363
Relu	5.864708044168488	173.7580964302667	13.181733437991632

MODEL 2

This model uses a **CNN** layer to find patterns in data, a **Bidirectional GRU layer** to understand sequences in both forward and backward directions, and dense layers to predict the final output.

Function	Mean Absolute Error	Mean Squared Error	Root Mean Square Error
Proposed function 2(RTHIN)	3.1228155356187086	128.87421174238793	11.352277821758413
Proposed function 1	4.640656676153289	34.08639466684065	5.838355476231355
Selu	12.047576765166669	598.0677674420493	24.45542409041498
tanh	11.318339067049305	364.397315269566	19.089193677826362
Relu	12.3549	480.7652	21.9264
Elu	25.910652317482217	122.356884084116	34.97954889419187

Data set 2- TIME SERIES DATA SET (Stock_data_TSLA20)

MODEL 3

This model uses a combination of **CNN** and **Bidirectional LSTM** to analyze patterns in time-series data and predict future values.

Function	Mean Squared Error	Mean Absolute Error
Proposed function2(RTHIN)	4.190934424540955	1.4417399697833595
Proposed functioin1	6.998518581880949	1.5513384054104487
Relu	51.412096973255494	4.4939109772443775
Gelu	9.702612903790861	1.6546513438224795
tanh	16.886911247688165	2.2236484557390215

MODEL 1

Function	Mean Squared Error	Root Mean Square Error	Mean Absolute Error
Proposed function2(RTHIN)	0.0012319764	0.03509952179	0.027311187
Proposed function1	0.0014	0.0379	0.0294
tanh	0.00243975369	0.04939386293	0.036126574
Gelu	0.00141296336	0.03758940490	0.029386946
Relu	0.007562	0.086957	0.078038

MODEL 2

This is a hybrid model combining **a CNN** layer for extracting features, a **Bidirectional GRU** for analyzing time-dependent patterns, and Dense layers for making predictions, primarily designed for time-series forecasting.

Function	Mean Squared Error	Root Mean Square Error	Mean Absolute Error
Proposed function2(RTHIN)	0.0013105382420545298	0.03620135690902387	0.026563275889822376
Proposed function1	0.001696161375715097	0.04118447979172612	0.033941893452032974
tanh	0.0028126674492545273	0.05303458729220514	0.0456155703314018
Gelu	0.0013435769350478269	0.03626438621396701	0.027304250715348588
Relu	0.0025021305942302973	0.05002130140480451	0.03985850994742891

Limitations: The first function had some limitations, particularly in its handling of positive and negative inputs. For positive values, it uses tanh(x), which can cause gradient saturation for large input values, making it harder for the model to learn effectively. For negative values, atan(x) is used, which, while smooth, does not provide a strong gradient flow, potentially hindering optimization. This led to higher error rates (MAE and MSE) compared to established activation functions like SELU and ELU, especially when the input values were large.

We chose the second function(RTHIN), because of its advantages over the first. For positive values, it uses a linear activation (x), allowing the model to retain information without the risk of gradient saturation. For negative values, it combines tanh(x) and atan(x), resulting in a smoother, more gradual response and better gradient flow. This led to improved performance in terms of error metrics, making the model more efficient and accurate**.**

Conclusion: In conclusion, the second function(RTHIN) provided better results by addressing the issues found in the first function, particularly with handling large positive inputs and negative inputs. Its smoother gradient behavior and linear activation for positive

values helped the model learn more effectively, leading to reduced errors and better overall performance.