Approximating the Runge Function using a Neural Network

Kai-Hung Cheng

2025/9/24

1 Introduction

This experiment aims to train a neural network to approximate the Runge function

$$f(x) = \frac{1}{1 + 25x^2}, \quad x \in [-1, 1].$$

The goal is to construct a neural network model that accurately learns this nonlinear target function through supervised learning. We train and evaluate the model to examine its ability to represent the Runge function and generalize beyond the training data.

2 Method

2.1 Problem Setup

The neural network $\hat{f}(x; \theta)$, parameterized by weights θ, is trained to minimize the mean squared error (MSE) between its prediction and the true function value:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} (\hat{f}(x_i; \theta) - f(x_i))^2.$$

2.2 Data Generation

A total of 300 data points were uniformly sampled from $x \in [-1, 1]$, and their corresponding labels $y_i = f(x_i)$ were computed. The dataset was randomly shuffled and evenly divided into training, validation, and test subsets (each containing 100 samples). All values were normalized to ensure numerical stability during optimization.

2.3 Network Architecture

The model is a fully connected multilayer perceptron (MLP) defined as:

$$x \to \text{Linear}(1, 64) \to \tanh() \to \text{Linear}(64, 64) \to \tanh() \to \text{Linear}(64, 1).$$

It contains two hidden layers with 64 neurons each and uses the tanh activation function.

2.4 Training Setup

We use stochastic gradient descent (SGD) with a learning rate of 0.01 to minimize the MSE loss. The model is trained for 10000 epochs with full-batch updates. Training and validation losses are recorded every epoch to monitor convergence.

2.5 Evaluation Metric

The model's performance is evaluated using the mean squared error (MSE) on the test set, which measures the average squared difference between predictions and ground-truth values.

3 Results and Discussion

3.1 Training Performance

During training, both the training and validation losses decreased smoothly and consistently, reaching values below 3×10^{-4} after 10000 epochs. The loss curves indicate stable optimization without overfitting, as the validation loss closely follows the training loss throughout.

3.2 Function Approximation

The trained model closely matches the true Runge function, accurately capturing the sharp peak near $x = 0$ and the smooth decay toward the boundaries. The resulting test error demonstrates excellent approximation capability:

$$\text{MSE} = 0.00033.$$

The neural network's predictions almost perfectly overlap with the target function, validating its ability to learn nonlinear relationships effectively.

3.3 Discussion

The experiment confirms that a two-layer tanh-based MLP can approximate smooth and bounded functions such as the Runge function with high precision. The tanh activation ensures smooth gradient flow, allowing gradual refinement even after thousands of training epochs. The close alignment of training and validation curves indicates strong generalization, showing that extended training improves fit quality without leading to overfitting.

Table 1: Training and test performance of the neural network.

Epochs	Final Training Loss	Test MSE
10000	0.0003	0.00033

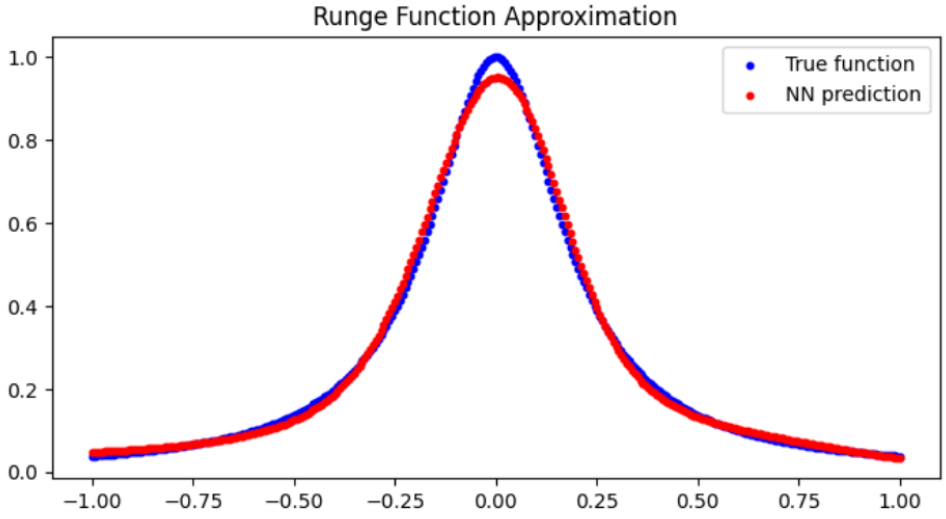

Figure 1: Approximation of the Runge function using a neural network after 10000 epochs. The blue points represent the true function values, and the red points represent the neural network predictions.

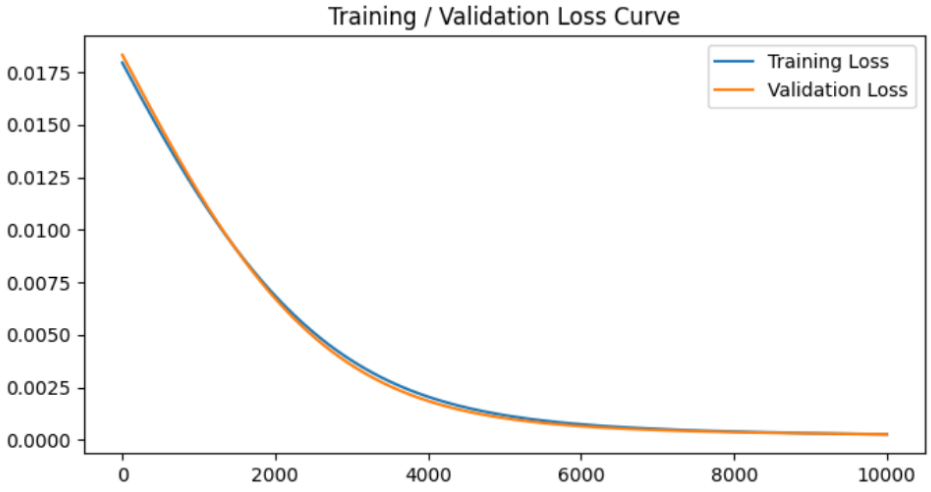

Figure 2: Training and validation loss curves over 10000 epochs, showing smooth convergence and no overfitting.

4 Conclusion

This study demonstrates that a two-layer tanh neural network can effectively approximate the Runge function with very low mean squared error. The model achieves smooth convergence and precise reconstruction of the target function after 10000 epochs, confirming the effectiveness of neural networks in continuous function approximation tasks.