Multilayer Perceptron for Function Approximation

Karol Korszun

July 28, 2024

1 Introduction

Function approximation using neural networks is a fundamental problem in machine learning. Given a function

$$f(x) = x^2 \cdot \sin(x) + 100 \cdot \sin(x) \cdot \cos(x) \tag{1}$$

we implement a multilayer perceptron (MLP) to approximate this function. The goal is to investigate how different configurations of the neural network and optimization strategies affect the quality of the approximation.

2 Implementation

The network consists of an input layer with a single neuron to accept the input x. This is followed by four hidden layers consisting of 200, 100, 100, and 50 neurons, respectively. Finally, the output layer has a single neuron to produce the function value $f(x)$.

Weights were initialized using the He initialization method:

$$W_i \sim \mathcal{N}(0, \sqrt{2/n_i}) \tag{2}$$

where n_i is the number of input units in the i-th layer. Biases were initialized to zero.

For training the network, we employed two optimization strategies: the gradient descent method and the evolution strategy (1+1). Gradient clipping was applied to prevent the exploding gradient problem, with a clip value of 1.0. Additionally, learning rate decay was used to improve convergence stability.

3 Analysis

The analysis focuses on two main aspects: the impact of the number of neurons in each layer on the quality of the approximation, and the differences in approximation quality depending on whether the gradient descent method or the evolution strategy is used for weight optimization.

3.1 Impact of Neurons and Layers

To evaluate the impact of the network architecture, several experiments were conducted with varying numbers of neurons and layers. The configurations tested include: Configuration A with 50, 25, 10 neurons; Configuration B with 100, 50 and 25 neurons; and Configuration C with 200, 100 and 50 neurons. The performance of each configuration was assessed based on the mean squared error (MSE) between the predicted and actual function values after 20,000 epochs with a learning rate of 0.001. The results are summarized in Table 1.

Configuration	**Test Loss (MSE)**
[50, 25, 10]	1258.03
[100, 50, 25]	932.54
[200, 100, 50]	1541.25

Table 1: Mean squared errors for different configurations after 20,000 epochs

While Configuration C initially had the highest loss in this experiment, proper tuning of the hyperparameters allowed it to eventually converge and achieve a relatively low loss.

3.2 Gradient Method vs Evolution Strategy

A comparative study was performed between the gradient descent method and the evolution strategy. The gradient descent method generally showed faster convergence, reducing the error significantly more quickly.

4 Results

The experiments revealed several key trends. Increasing the number of neurons in each layer generally improved the quality of the approximation, as indicated by lower MSE values. The gradient descent method tended to converge faster than the evolution strategy.

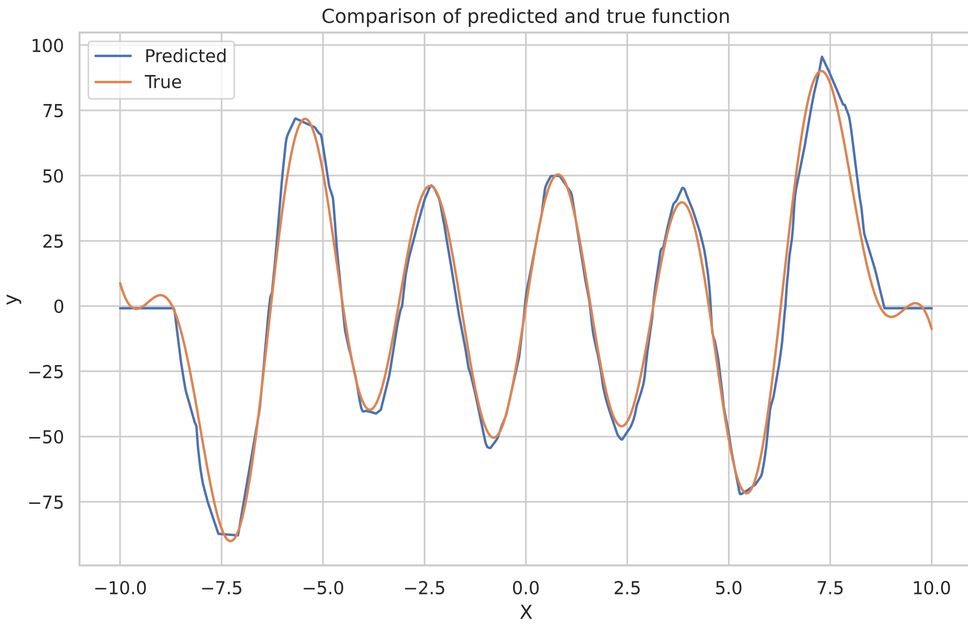

Figure 1: Comparison of function approximation results using different methods

5 Conclusion

In conclusion, the architecture of the multilayer perceptron, particularly the number of neurons in each layer, significantly impacts the quality of function approximation. The gradient descent method offers faster convergence compared to evolution strategy which is significantly slower.