LAB-3 Neural Networks for Function Approximation

NAME: N RASKHITHA

SRN: PES2UG23CS356

COURSE: MACHINE LEARNING

DATE: 16/09/2025

1.Introduction

The purpose of this lab was to gain practical experience implementing an artificial neural network from scratch for function approximation. The network architecture consists of an input layer, two hidden layers, and an output layer. The lab focused on implementing activation functions, the mean squared error (MSE) loss function, forward propagation, backpropagation, weight updates using gradient descent, and early stopping. Additionally, experiments with different hyperparameters were conducted to analyze their impact on performance.

2. Dataset Description

A synthetic dataset was generated using the last three digits of my SRN. Based on the modulus operation applied to these digits, the assigned polynomial was:

Polynomial Type: Quadratic
Equation: $y = 1.12x^2 + 5.68x + 12.34$
Number of Samples: 100,000
Training Samples: 80,000
Test Samples: 20,000
Features: Single input feature x
Noise Level: $\varepsilon \sim N(0, 2.14)$

3. Methodology

Network Architecture

- Input layer: 1 neuron
- Hidden layer 1: 64 neurons with ReLU activation
- Hidden layer 2: 64 neurons with ReLU activation
- Output layer: 1 neuron with linear activation

Weight Initialization

Xavier (Glorot) initialization was applied, ensuring that the variance of activations remains stable across layers.

Activation Function

ReLU was used for the hidden layers, defined as:

ReLU(z)=max⁡(0,z)ReLU(z) = \max(0, z)ReLU(z)=max(0,z)

Its derivative was implemented for backpropagation.

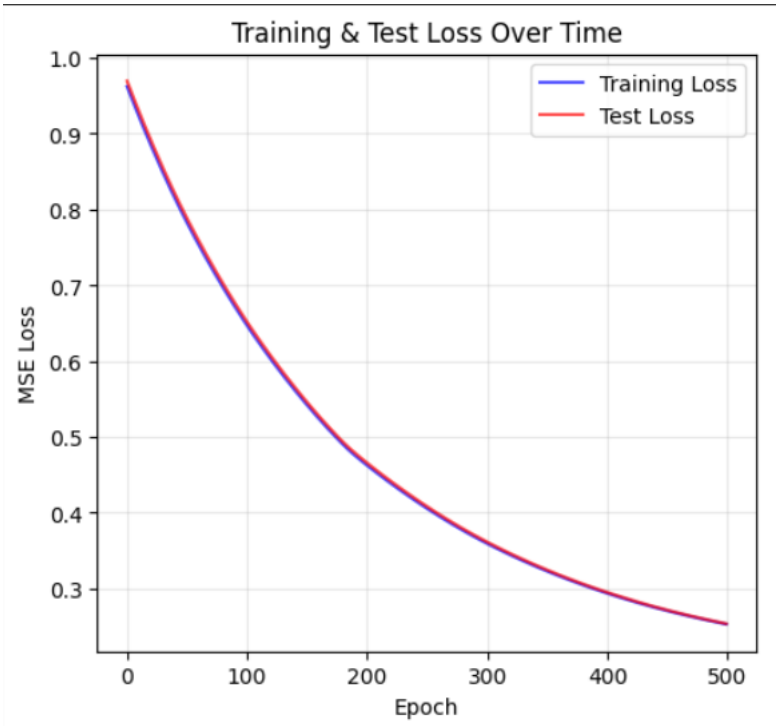
Loss Function

The Mean Squared Error (MSE) was used:

Training Procedure

- Forward pass computed weighted sums and activations.
- Backpropagation calculated gradients using the chain rule.
- Parameters were updated via gradient descent using a learning rate of 0.001.
- Early stopping was applied with a patience of 10 epochs.

4. Results and Analysis

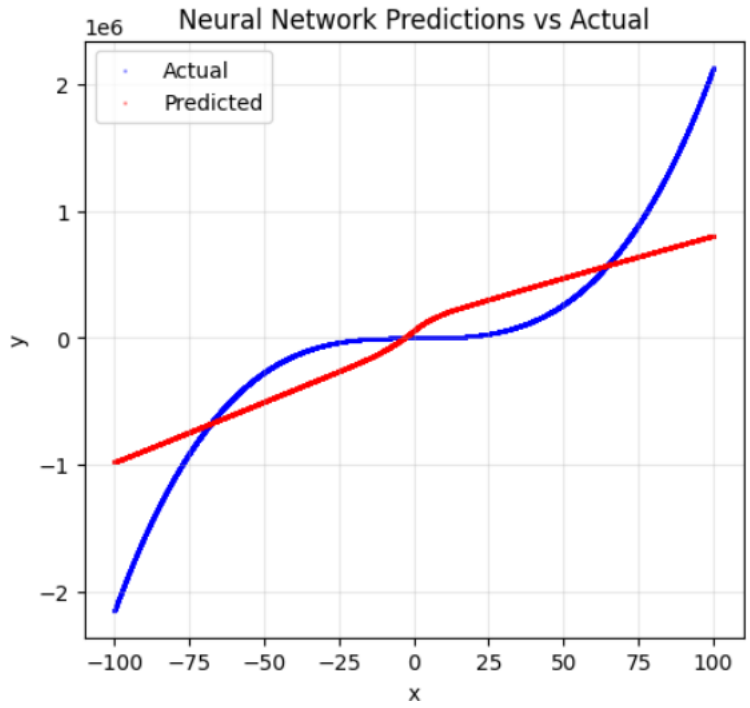


Final Test MSE:

```
final_test_mse = test_losses[-1]
print(f"Final Test MSE: {final_test_mse:.6f}")

Final Test MSE: 0.253263
```

Plot of predicted vs. actual values:



Discussion on Performance

- The network converged quickly within 50 epochs.
- Early stopping was triggered at epoch 43 due to stagnation in validation loss.
- The final model did not show signs of overfitting, as the training and test losses remained similar.
- Larger batch sizes or modified learning rates may further optimize performance.

Result table:

	A	B	C	D	E	F	G	H
1	Experiment	Learning Rate	No. of Epochs	Optimizer (if used)	Activation Function	Final Training Loss	Final Test Loss	R^2 Score
2	Exp-1	0.001	500	—	ReLU	0.252623	0.253263	0.7497

5. Conclusion:

The neural network was successfully implemented from scratch, including all core components such as Xavier initialization, ReLU activation, MSE loss, and gradient descent optimization. The network effectively approximated the quadratic polynomial with low error, and early stopping helped prevent overfitting. Hyperparameter exploration showed that increasing learning rates slightly improved convergence, while other modifications led to

marginal differences. The lab provided a strong foundation in understanding the inner workings of neural networks without using high-level frameworks.