Lab 6 - Neural Networks for Function Approximation

Name: Monisha Sharma
SRN: PES2UG23CS906
Class: 5F
Date: 17/09/2025

1. Introduction

The purpose of this lab was to implement a neural network from scratch to approximate a polynomial curve generated from synthetic data.

Tasks performed:

- Implement key neural network components:
  - Xavier initialization
  - ReLU activation and derivative
  - Mean Squared Error (MSE) loss
  - Forward propagation
  - Backpropagation with gradient descent
- Train the network with early stopping.
- Evaluate performance with training/test loss curves and R^2 score.
- Run hyperparameter experiments and compare results.

2. Dataset Description

Polynomial Type: Cubic - > y=2.23x3−0.12x2+5.32x+10.75

Samples: 100,000 total train-80k test-20k

Noise Level: Gaussian noise, standard deviation = 10

Features: Single input x, single output y

Scaling: Both x and y standardized using `StandardScaler`.

3. Methodology

1. **Network Architecture:**
   - Input (1)
   - Hidden Layer 1 (72 neurons, ReLU)
   - Hidden Layer 2 (32 neurons, ReLU)
   - Output (1, linear).

2. **Weight Initialization:**
   Xavier initialization:

   ~ normal dist (0,sqrt2(fan_in+fan_out))

3. **Forward Pass:**
   Sequentially compute linear transformations + ReLU, ending with linear output.

4. **Loss Function:**
   Mean Squared Error (MSE).

5. **Backpropagation:**
   - Compute gradients using chain rule.
   - Update weights with gradient descent

4. Results

Experiment	Learning Rate	Batch Size	Epochs	Activation	Train Loss	Test Loss	R^2 Score	Observations
Baseline	0.001	64	500	ReLU	0.6255	0.6291	0.3743	Stable, no overfitting, but accuracy is low
Exp 1	**0.01**	**64**	**300**	**ReLU**	**0.2076**	**0.2091**	**0.7921**	**Higher LR gave much faster convergence, best fit**
Exp 2	0.001	128	500	ReLU	0.6255	0.6291	0.3743	Larger batch slowed learning, no gain
Exp 3	0.0005	64	500	ReLU	0.8018	0.8059	0.1984	Smaller LR led to underfitting, poor results
Exp 4	0.001	64	800	ReLU	0.5087	0.5120	0.4908	More epochs improved fit compared to baseline

5. Conclusion

The baseline model (learning rate = 0.001, batch size = 64, 500 epochs) achieved a moderate performance with a **Test Loss of 0.6291 and an R^2 Score of 0.3743,** showing that the network was able to learn some structure but not fit the data with high accuracy.

Through hyperparameter exploration (Part B):

- A higher learning rate (0.01) significantly improved learning speed and accuracy, giving the best performance (Test Loss = 0.2091, R^2 = 0.7921).
- A smaller learning rate (0.0005) led to underfitting, with poor accuracy (R^2 = 0.1984).

- Increasing the batch size to 128 slowed convergence and did not improve results.
- Increasing the number of epochs to 800 moderately improved generalization ($R^2 = 0.4908$) compared to baseline.

Overall, the best model was achieved with a relatively higher learning rate and smaller batch size, which allowed the network to fit the polynomial more effectively.