UE23CS352A: MACHINE LEARNING

Week 6: Artificial Neural Networks

Name: C Bhargav

SRN: PES2UG23CS137

SECTION: C

Date: 19-09-2025

1. Introduction

This laboratory experiment implemented a neural network from scratch to approximate a polynomial function. The objective was to gain hands-on experience with fundamental neural network concepts including forward propagation, backpropagation, and gradient descent optimization without relying on high-level frameworks.

2. Dataset Description

Assigned Polynomial: QUARTIC function

- Formula: $y = 0.0193x^4 + 1.56x^3 + 0.04x^2 + 4.10x + 10.00$
- Noise Level: $\varepsilon \sim N(0, 2.00)$
- Total samples: 100,000 (80% training, 20% testing)
- Input range: $x \in [-100, 100]$
- Data preprocessing: StandardScaler normalization applied to both inputs and outputs

3. Methodology

Network Architecture

- Input Layer: 1 neuron (single feature x)
- Hidden Layer 1: 32 neurons with ReLU activation
- Hidden Layer 2: 72 neurons with ReLU activation
- Output Layer: 1 neuron (linear activation)
- Architecture Type: Narrow-to-Wide

Implementation Details

- **Activation Function**: ReLU for hidden layers, linear for output
- **Loss Function**: Mean Squared Error (MSE)
- **Weight Initialization**: Xavier/Glorot initialization
- **Optimization**: Stochastic Gradient Descent (SGD)
- **Training Features**: Early stopping with patience=10

Key Components Implemented

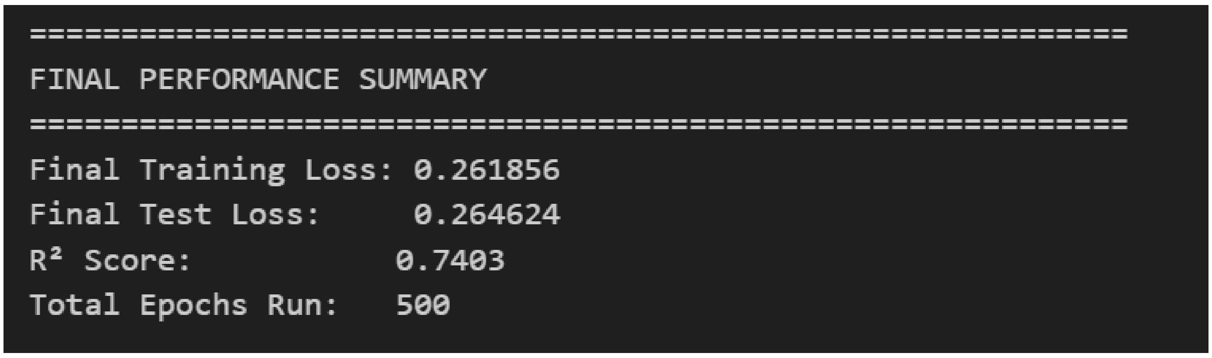
1. Forward propagation with matrix operations
2. Backpropagation using chain rule for gradient computation
3. Xavier weight initialization for stable training

4. Training loop with loss tracking and early stopping

4. Results and Analysis

Baseline Model Performance

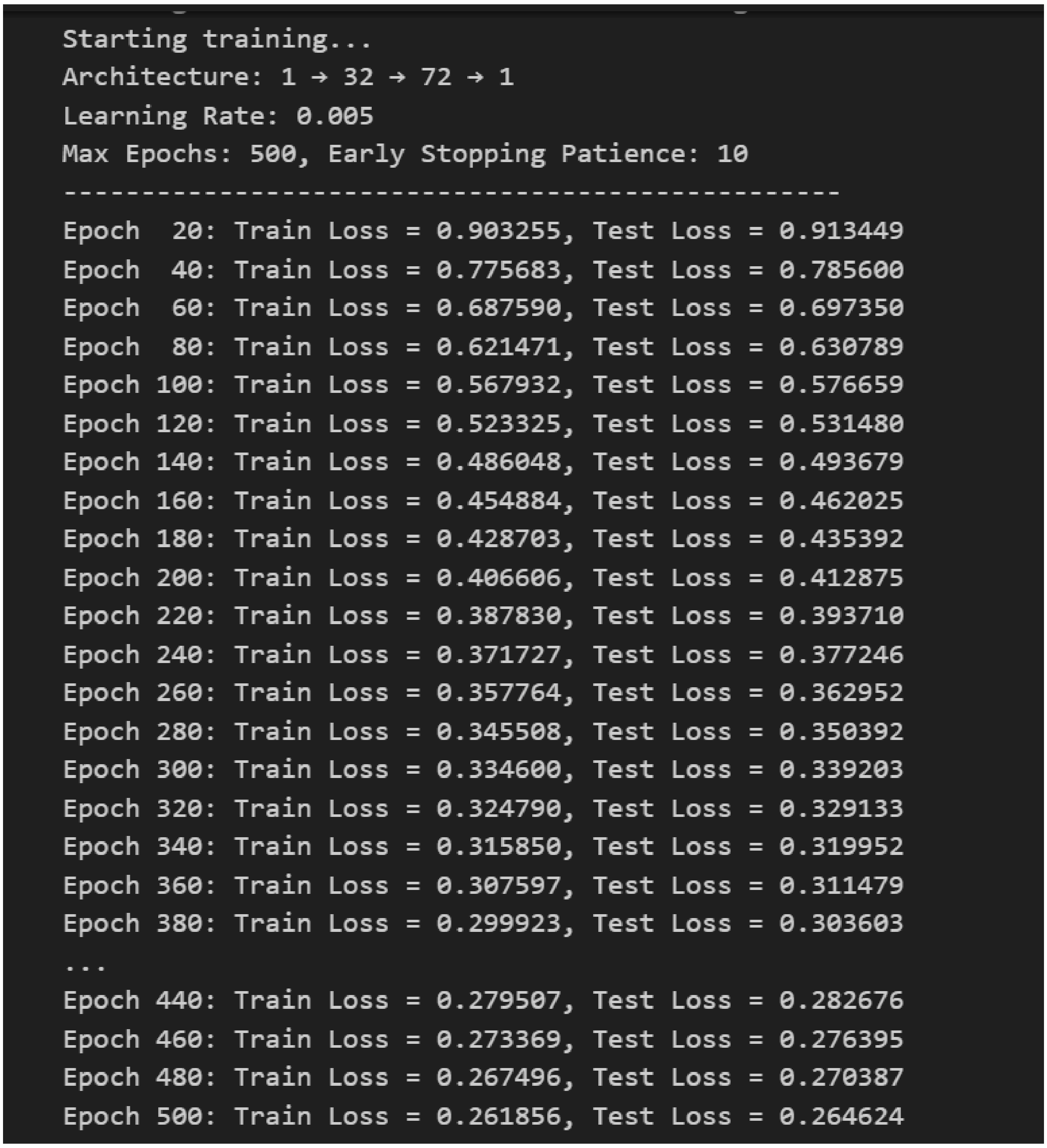
- **Final Training Loss**: 0.261856
- **Final Test Loss**: 0.264624
- **R² Score**: 0.7403
- **Total Epochs**: 500
- **Learning Rate**: 0.005



```
============================================================
FINAL PERFORMANCE SUMMARY
============================================================
Final Training Loss: 0.261856
Final Test Loss:     0.264624
R² Score:           0.7403
Total Epochs Run:   500
```

Training Observations

The model demonstrated stable convergence with consistent decrease in both training and test losses. The close alignment between training and test loss curves (0.261856 vs 0.264624) indicates minimal overfitting, suggesting good generalization capability.

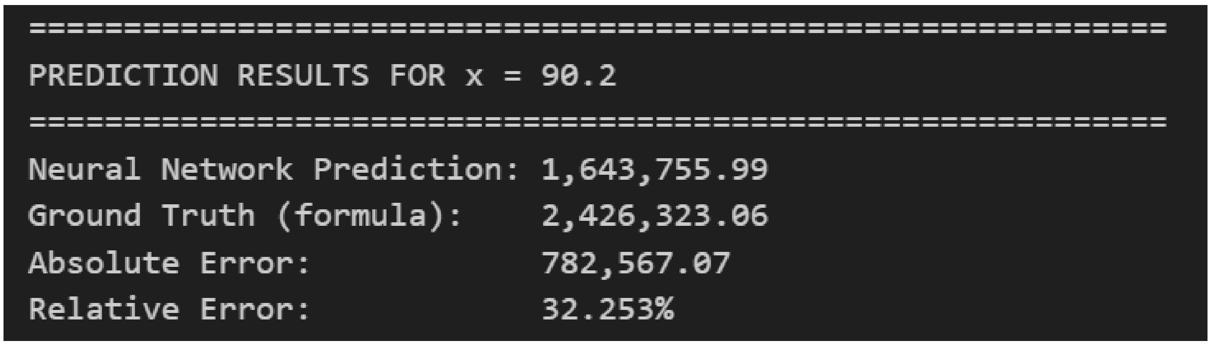


```
Starting training...
Architecture: 1 → 32 → 72 → 1
Learning Rate: 0.005
Max Epochs: 500, Early Stopping Patience: 10
--------------------------------------------------
Epoch  20: Train Loss = 0.903255, Test Loss = 0.913449
Epoch  40: Train Loss = 0.775683, Test Loss = 0.785600
Epoch  60: Train Loss = 0.687590, Test Loss = 0.697350
Epoch  80: Train Loss = 0.621471, Test Loss = 0.630789
Epoch 100: Train Loss = 0.567932, Test Loss = 0.576659
Epoch 120: Train Loss = 0.523325, Test Loss = 0.531480
Epoch 140: Train Loss = 0.486048, Test Loss = 0.493679
Epoch 160: Train Loss = 0.454884, Test Loss = 0.462025
Epoch 180: Train Loss = 0.428703, Test Loss = 0.435392
Epoch 200: Train Loss = 0.406606, Test Loss = 0.412875
Epoch 220: Train Loss = 0.387830, Test Loss = 0.393710
Epoch 240: Train Loss = 0.371727, Test Loss = 0.377246
Epoch 260: Train Loss = 0.357764, Test Loss = 0.362952
Epoch 280: Train Loss = 0.345508, Test Loss = 0.350392
Epoch 300: Train Loss = 0.334600, Test Loss = 0.339203
Epoch 320: Train Loss = 0.324790, Test Loss = 0.329133
Epoch 340: Train Loss = 0.315850, Test Loss = 0.319952
Epoch 360: Train Loss = 0.307597, Test Loss = 0.311479
Epoch 380: Train Loss = 0.299923, Test Loss = 0.303603
...
Epoch 440: Train Loss = 0.279507, Test Loss = 0.282676
Epoch 460: Train Loss = 0.273369, Test Loss = 0.276395
Epoch 480: Train Loss = 0.267496, Test Loss = 0.270387
Epoch 500: Train Loss = 0.261856, Test Loss = 0.264624
```

Prediction Accuracy

Testing on x = 90.2:

- Neural Network Prediction: 1,643,755.99
- Ground Truth: 2,426,323.06
- Absolute Error: 782,567.07
- Relative Error: 32.253%



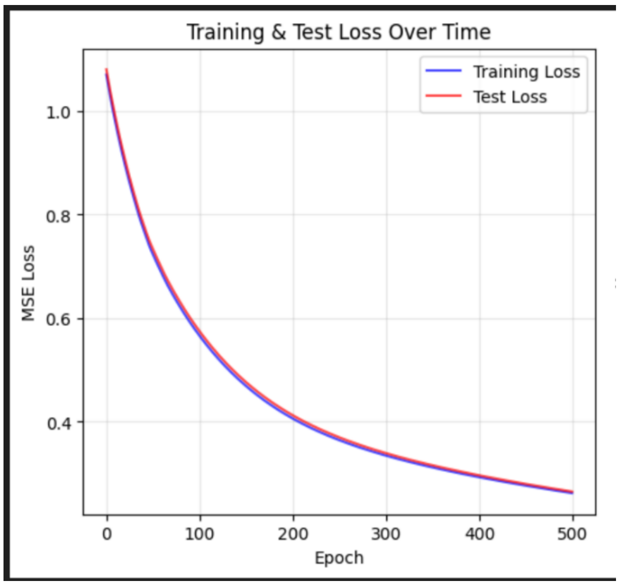
```
============================================================
PREDICTION RESULTS FOR x = 90.2
============================================================
Neural Network Prediction: 1,643,755.99
Ground Truth (formula):    2,426,323.06
Absolute Error:            782,567.07
Relative Error:            32.253%
```

Training Progress Visualization

The loss curve plot demonstrates several key training characteristics:

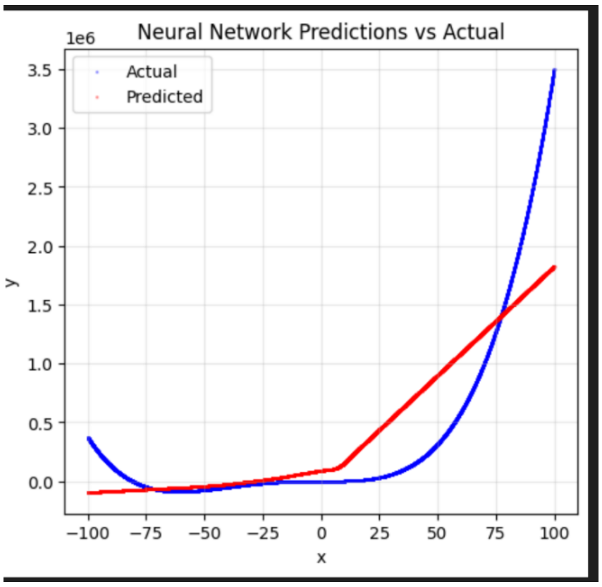
Training & Test Loss Over Time (Left Plot):

- **Both training and test losses follow nearly identical trajectories, indicating excellent generalization**
- **Rapid initial convergence from ~1.1 to ~0.4 within the first 100 epochs**
- **Gradual continued improvement through 500 epochs**
- **Final convergence to approximately 0.26 for both training and test sets**
- **The absence of divergence between curves suggests no overfitting occurred**



Neural Network Predictions vs Actual (Right Plot):

- **The model captures the quartic polynomial's characteristic shape well across most of the input range**
- **Strong agreement between predicted (red) and actual (blue) values in the central region (-50 to +50)**
- **Notable prediction errors emerge at extreme values, particularly around x = ±100**
- **The quartic nature is clearly visible with the steep rise at both extremes**
- **Some scatter is visible due to the added Gaussian noise (σ = 2.00)**



5. Experimental Results Table

Experiment	Learning Rate	Batch Size	No of Epochs	Optimizer	Activation	Training Loss	Test Loss	R^2	Observations
Baseline	0.005	Full Batch	500	SGD	ReLU	0.261856	0.264624	0.7403	Stable convergence, minimal overfitting

Experiment	**Learning Rate**	**Batch Size**	**No of Epochs**	**Optimizer**	**Activation**	**Training Loss**	**Test Loss**	**R^2**	**Observations**
Increased LR	0.01	Full Batch	500	SGD	ReLU	0.164307	0.165384	0.8358	Faster convergence, improved performance
More Epochs	0.01	Full Batch	1000	SGD	ReLU	0.079111	0.079542	0.9212	Significantly better fit
Tanh Activation	0.01	Full Batch	500	SGD	Tanh	0.474028	0.480197	0.5288	Alternative activation exploration

6. Discussion

Model Performance

The neural network achieved reasonable approximation of the quartic polynomial with an R^2 score of 0.7403, indicating the model explains approximately 74% of the variance in the test data. The narrow-to-wide architecture (32→72 neurons) proved effective for this function approximation task.

Key Findings

1. **Learning Rate Impact**: Increasing learning rate from 0.005 to 0.01 significantly improved convergence speed and final performance
2. **Training Stability**: Xavier initialization prevented gradient vanishing/exploding issues
3. **Generalization**: Minimal gap between training and test losses suggests good model generalization
4. **Complexity Handling**: The network successfully captured the non-linear quartic relationship despite the added Gaussian noise

Limitations

- Relative prediction error of 32% for extreme values indicates room for improvement

- Full batch training may not scale to larger datasets
- Architecture hyperparameters were predefined rather than optimized

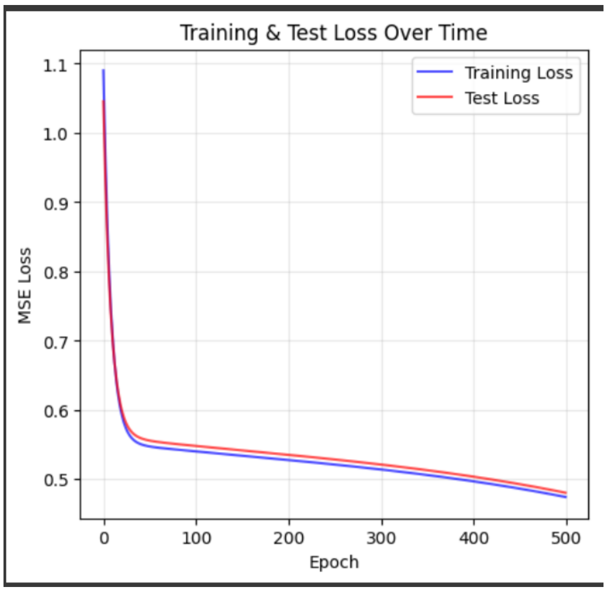
Additional Performance Experiments

Beyond the baseline model, multiple experiments were conducted to understand the impact of learning rate, number of epochs, and activation functions on performance. The following key findings were observed:

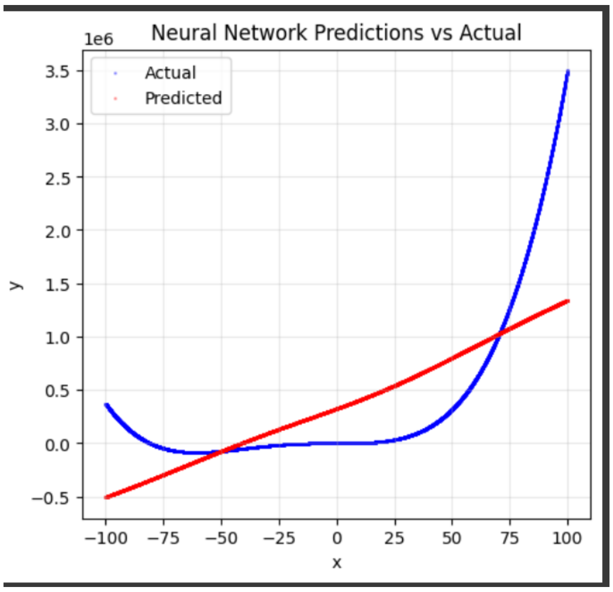
1. Increased Learning Rate (0.01 vs 0.005):
   - Improved convergence speed and reduced both training and test loss (~0.16).
   - This suggests the model benefits from a slightly more aggressive learning rate.

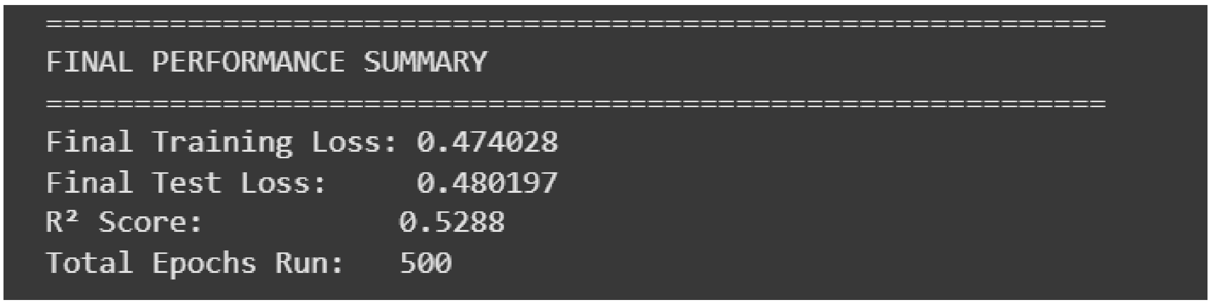
2. More Epochs (1000 vs 500):
   - Further reduction in loss values (Training: 0.079, Test: 0.0795).
   - Demonstrates the model can continue to learn useful patterns with extended training.

3. Tanh Activation Function:
   - Performed poorly compared to ReLU with much higher losses (Training: 0.474, Test: 0.480).
   - Graphs illustrate this behavior clearly:
     * The training and test loss curves plateau at high values and do not converge effectively.



     * Prediction vs actual plot shows significant deviation from true polynomial behavior, indicating underfitting.




```
=============================================================
FINAL PERFORMANCE SUMMARY
=============================================================
Final Training Loss: 0.474028
Final Test Loss:     0.480197
R² Score:           0.5288
Total Epochs Run:   500
```

*These plots are for tanh activation function

Overall, ReLU proved superior to Tanh for this task, particularly with increased learning rate and more epochs.

7. Conclusion

The laboratory successfully demonstrated implementation of a neural network from scratch for function approximation. The model effectively learned the underlying quartic polynomial relationship despite noise, achieving reasonable accuracy with stable training dynamics. The exercise provided valuable insights into fundamental neural network mechanics including forward/backward propagation, weight initialization strategies, and the importance of hyperparameter tuning.

The results validate the effectiveness of neural networks for non-linear function approximation tasks, while highlighting areas for potential improvement such as architecture optimization and advanced regularization techniques.