Project Report

Multi-Layer Neural Network

Course: Machine Learning

Department: Bachelors of Artificial Intelligence

Submitted by: Muhammad Arish Khan (22K-4118)

Submitted to: Miss Alishba Subhani

Date: December 9, 2024

Table of Contents

1. Introduction

This project involves implementing a multi-layer neural network from scratch. The aim is to explore the workflow, methodology, and performance of neural networks while addressing challenges in training, evaluation, and optimization. This report documents the complete process, from network setup to final findings and conclusions.

2. Neural Network Setup and Design

The neural network was designed using the following steps:

- Defining the architecture: Input layer, hidden layers, and output layer.
- Selecting activation functions: ReLU for hidden layers and sigmoid for the output layer.
- Initializing weights and biases.
- Configuring the loss function (binary cross-entropy) and optimizer (e.g., SGD or Adam).

3. Data Preprocessing

Before feeding data into the neural network, preprocessing was carried out to ensure optimal performance. Steps included:

- Normalizing the input features.
- Splitting the dataset into training and testing sets.
- Handling any missing values and data anomalies.

4. Model Training and Evaluation

The model was trained using a gradient descent-based optimizer, with the following configurations:

- **Learning rate:** Appropriate tuning was done to avoid vanishing or exploding gradients.

- **Batch size:** Mini-batch gradient descent was used to balance computational efficiency and convergence.
- **Number of epochs:** The training was conducted over multiple epochs until convergence.

Performance evaluation metrics included accuracy, precision, recall, and F1-score. Confusion matrices were used to analyze prediction quality.

5. Findings and Observations

The neural network demonstrated the following:

- Strong performance on training data, achieving high accuracy.
- Challenges in generalizing to unseen data due to overfitting.
- Learning dynamics influenced by hyperparameter choices (e.g., learning rate, activation functions).

6. Tweaks and Improvements

Several improvements were made during experimentation:

- Implemented dropout regularization to mitigate overfitting.
- Fine-tuned learning rate and batch size.
- Augmented data to increase diversity and robustness.
- Enhanced architecture by adding more hidden layers and neurons.

7. Conclusion

This project highlighted the complexities of designing and training neural networks. Through iterative improvements, the model's performance was optimized, demonstrating the importance of careful hyperparameter tuning and regularization.