Practice Lab Neural Network 1

Name: Sahil Karne

Roll Number: 37

PRN: 202201040086

Batch: DL2

Date of Submission: 14-01-2025

Neural Network Implementation from Scratch

Objective

Implement a simple feedforward neural network from scratch in Python without using any in-built deep learning libraries. This implementation focuses on basic components such as forward pass, backward propagation (backpropagation), and training using gradient descent.

Problem Definition

Dataset:

The dataset used for this task is a simple binary classification dataset, represented by the "AND problem." The input dataset consists of all possible combinations of binary values (0, 0), (0, 1), (1, 0), (1, 1), with corresponding output labels that represent the AND operation results. The output labels are binary (0 or 1).

Task:

The task is to train a neural network to predict the output of the AND operation. The model should learn the correct classification for each combination of inputs (X = [0, 0], [0, 1], [1, 0], [1, 1]) and produce the corresponding output (Y = [0], [0], [0], [1]).

Methodology

Neural Network Architecture:

- **Input Layer:** The input layer consists of 2 neurons, each representing one of the binary inputs of the AND operation.
- **Hidden Layer:** The hidden layer consists of 3 neurons, with the activation function being the Sigmoid function. This layer computes intermediate activations based on the input features.
- **Output Layer:** The output layer consists of a single neuron that produces the final output (binary classification) after applying the Sigmoid activation function.

Forward Pass:

During the forward pass, the input data is passed through the network in the following steps:

1. The input layer receives the data.
2. The data is multiplied by the weights of the input-to-hidden layer, and the bias is added.
3. The activation function (Sigmoid) is applied to the weighted sum to obtain the hidden layer output.
4. The hidden layer output is multiplied by the weights of the hidden-to-output layer, and the bias is added.
5. The activation function (Sigmoid) is applied again to compute the final output.

Backpropagation:

- The error is computed by subtracting the predicted output from the actual target values.
- This error is propagated backward through the network to adjust the weights using the gradient descent optimization technique.
- The weights and biases are updated based on the calculated gradients, with a learning rate controlling the step size.

Loss Function:

- **Mean Squared Error (MSE):** The loss function used is the Mean Squared Error, which is suitable for regression and binary classification problems. It measures the average squared difference between the actual and predicted values.

Optimization:

- **Gradient Descent:** Gradient descent is implemented to minimize the loss function by updating the weights and biases in the direction of the negative gradient. The learning rate is used to control how much the weights are adjusted during each iteration.

Declaration

I, **Sahil Karne**, confirm that the work submitted in this assignment is my own and has been completed following academic integrity guidelines. The code is uploaded to my GitHub repository, and the repository link is provided below:

GitHub Repository Link:

https://github.com/SahilKarne/Deep-Learning/tree/main/Neural_Network_From_Scratch

Signature:
Sahil Shamrao Karne