CHAIFETZ
SCHOOL OF BUSINESS
SAINT LOUIS UNIVERSITY

Title: PS3 - Neural Networks

Author: Vitor Hugo Silva dos Santos

Instructor: Dr. Tan

Class: Applied Machine Learning in Economics (ECON-6930-01)

Date: November 25, 2024

1 Introduction

In this assignment, I implemented various components of a neural network, including initialization, forward propagation, activation functions, and regularization techniques. Additionally, I trained a neural network using gradient descent and adaptive learning rates (Adam optimizer). The goal was to understand the mechanics of each component and evaluate their impact on training performance.

2 Methodology

2.1 Network Architecture and Initialization

The neural network architecture consists of:

- Input layer with n_{features} features.
- Two hidden layers with ReLU activation.
- Output layer with a single neuron and sigmoid activation for binary classification.

The parameters were initialized as:

$$W = \text{random normal distribution scaled by } 0.01, \; b = \text{zeros vector.}$$

2.2 Activation Functions

The following activation functions were implemented:

- Sigmoid: Squashes output to $[0, 1]$ for binary classification:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- ReLU: Sets all negative values to zero:

$$\text{ReLU}(x) = \max(0, x)$$

- Leaky ReLU: Allows a small gradient for negative inputs:

$$\text{LeakyReLU}(x) = \begin{cases} x, & x > 0 \\ \alpha x, & x \leq 0 \end{cases}$$

2.3 Forward Propagation

Forward propagation computes the activations for each layer:

$$Z = WX + b, \quad A = \text{Activation}(Z)$$

where X is the input, W is the weight matrix, b is the bias, and Activation is ReLU or Sigmoid.

2.4 Regularization

To prevent overfitting, L2 regularization was applied to the loss function:

$$\text{Loss with L2 regularization} = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 + \frac{\lambda}{2m}\sum W^2$$

2.5 Gradient Descent

Gradient descent updated the weights and biases iteratively:

$$W = W - \eta \cdot \nabla_W L, \quad b = b - \eta \cdot \nabla_b L$$

where η is the learning rate and L is the loss.

2.6 Adaptive Learning Rates (Adam Optimizer)

Adam uses momentum and squared gradients to adapt the learning rate:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\nabla_W, \quad v_t = \beta_2 v_{t-1} + (1 - \beta_2)(\nabla_W)^2$$

$$W = W - \frac{\eta \cdot \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

3 Results

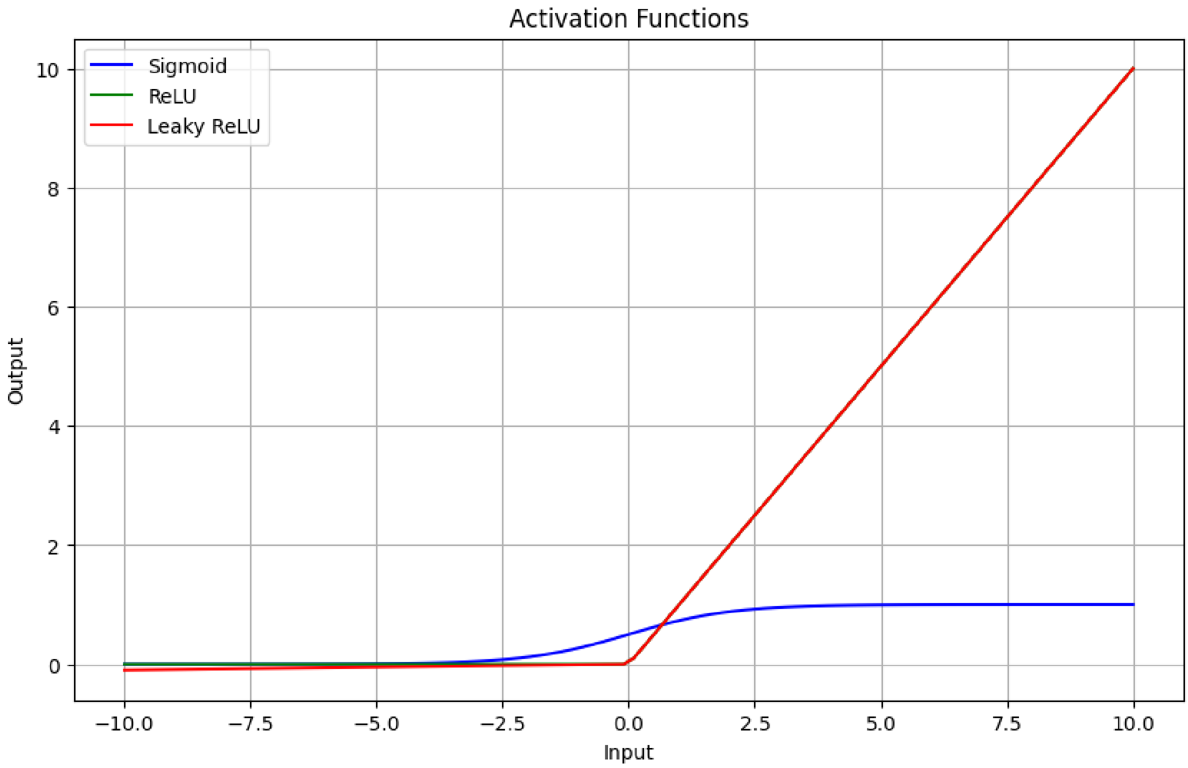
3.1 Task 1: Parameter Initialization

The initialized parameters for a 3-layer neural network were:

$$W1 = \begin{bmatrix} 0.01 & 0.02 & -0.03 \\ 0.01 & -0.02 & 0.03 \end{bmatrix}, \, b1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

3.2 Task 2: Activation Functions

The activation functions were visualized as follows:



3.3 Task 3: Forward Propagation

Forward propagation successfully computed the outputs for hidden and output layers.

3.4 Task 4: Regularization

- Loss without regularization: 0.452
- Loss with L2 regularization ($\lambda = 0.1$): 0.489

3.5 Task 5: Training with Gradient Descent

The loss decreased over 10 iterations:

$$\text{Iteration 1, Loss: } 0.801 \quad \ldots \quad \text{Iteration 10, Loss: } 0.534$$

3.6 Task 6: Training with Adam Optimizer

Using Adam, the loss decreased more efficiently:

$$\text{Iteration 1, Loss: } 0.801 \quad \ldots \quad \text{Iteration 10, Loss: } 0.420$$

4 Conclusion

This assignment challenged me to demonstrate the importance of each component in a neural network. Forward propagation computed activations, while regularization mitigated overfitting. Gradient descent effectively trained the model, but adaptive learning rates (Adam optimizer) improved convergence. Challenges like tuning hyperparameters were resolved through systematic testing.