Neural Networks Lab Manual

Ahsan Saadat, PhD

1 Introduction

I have created this document to summarize my expectations from students by providing detailed aspects of how to perform the lab tasks. The purpose of this lab is to provide you with a practical learning experience, enabling you to gain firsthand knowledge of different artificial intelligence techniques and accomplish various objectives.

- Gain knowledge about various techniques and strategies for implementing neural networks.
- Develop a hands-on understanding of the inner workings of neural networks by implementing them from scratch.
- Grasp the fundamental concepts and rules of matrix multiplication.
- Understand how matrix multiplication is utilized in neural network computations, especially in the calculation of weighted sums and activation functions.

2 Lab Setup

In this lab, the implementation will be done using Python language. Different IDEs can be used for implementing and executing the code. Following are some options.

- Google Colab (https://colab.research.google.com/) (Preferable)
- Jupyter Notebook (https://jupyter.org/)
- Visual Code Studio (https://code.visualstudio.com/)

2.1 Starting with Google Colab

This section is a guide for getting started with Google Colab.

- Sign into your Google account and open the Google Colab website.
- Create a new notebook from the drop-down menu after clicking the files in the menu bar.
- After creating a new notebook, write code in a cell and execute it by clicking the run button at the left side of the cell or by pressing the shift+enter keys.
- Rename your file to the name of the lab and download it as .ipynb file for future use.

3 Preliminary Concept

Neurons in neural networks are the fundamental building blocks. They are inspired by the neurons in the human brain and are responsible for processing and transmitting information. Neurons receive input signals, apply transformations to them, and produce output signals. The neural network architecture also requires other parameters such as weights, bias and activation functions.

- Weights represent the strength or importance assigned to the inputs of a neuron. They determine how much influence each input has on the neuron's output. During training, the neural network adjusts these weights to optimize its performance in making accurate predictions or classifications.
- Biases are additional parameters in neural networks that enable them to make predictions even when all inputs are zero. They provide an additional level of flexibility and help in adjusting the output of neurons. Similar to weights, biases are learned during the training process.
- Activation functions determine the output of a neuron based on the weighted sum of its inputs. They introduce non-linearity into the neural network, allowing it to learn complex patterns and make nonlinear decisions. Common activation functions include sigmoid, tanh, ReLU (Rectified Linear Unit), and softmax.

4 Rules for Matrix Multiplication

- The number of columns in the first matrix should be equal to the number of rows in the second matrix. If the shape of the first matrix is (m x n) and the shape of the second matrix is (n x p), the resulting matrix will be of shape (m x p).
- Each element in the resulting matrix is obtained by taking the dot product of the corresponding row in the first matrix and the corresponding column in the second matrix.

5 Tasks

The following tasks shall be performed for achieving the goals of this lab:

- Implement parts of neural network architecture (Forward Propagation).
  - Use input matrix of size 3x1.
  - Use one hidden layer for the first case and two hidden layers for the second case with 1 neuron each.
  - Use weights and biases w.r.t to that.
- Implement a function for matrix multiplication and understand the rules.

6 Libraries to use

- import numpy as np

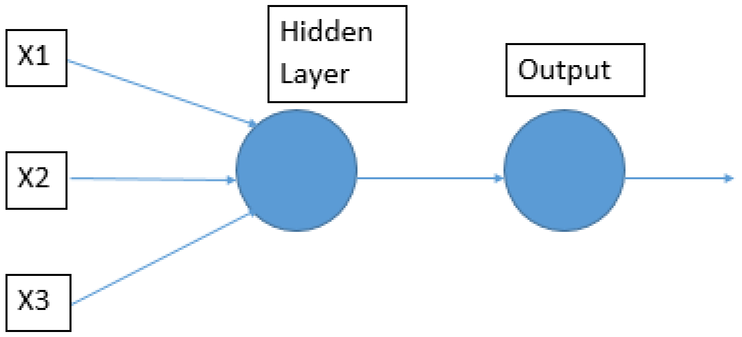

Figure 1: Simple Neural Network