PyTorch Coding Exercises

Given: 12 October 2022 **Deadline:** 15 Oct 2022

Activity 1: Learn a Regression Model from Data

With an understanding of the basics of PyTorch, you are ready to start building and training neural network models. In this activity, you will create a simple regression model that can predict a student's grade based on the number of hours they have studied and slept the day before a test is given.

The network you implement should have two input units (the two input dimensions are hours of study time and hours of sleep, as integers between 0-9), three hidden units, and one output unit (the predicted grade on a continuous scale between 0-1). Use a sigmoid function as the non-linear activation function in the hidden units.

The table below contains the training data you should use (scale these values as necessary). Note the trend that is present: more sleep and more study time leads to a higher grade (it's true!). This is the pattern that should be learned by the network if it is working correctly.

Number of Study Hours	Number of Sleep Hours	Grade
5	9	92
4	8	91
3	6	82
5	8	95
1	4	74
2	6	75
8	8	96
7	8	94
2	5	80
6	9	91

Train your network for 1,000 epochs using a mean sum squared loss function. Your training code should incrementally print the loss value as training proceeds. What is the value of the loss in the final epoch?

Now it's time to test the network. Produce predictions of test performance based on the following sequence of study and sleep hours. Your prediction code should print each test instance out to the terminal. Record each prediction in an CSV file.

Number of Study Hours	Number of Sleep Hours
3	9
1	3
7	7
4	5
8	9

Activity 2: Create a Network that can Learn the NAND Operator

Recall from computer architecture that the NAND operator is a universal operator, meaning that any other logical operator can be expressed as a combination of NAND operations. Fundamentally, this operator itself is the combination of the operators NOT and AND. An interesting task from the perspective of AI is the ability to learn such an operator through example.

The network you implement here should have two input units (the two Boolean values that are being processed by the NAND operator), two hidden units, and one output unit (the exact output of the NAND operator). Use a sigmoid function as the non-linear activation function in the hidden units. The output of your code should be the exact answer for any two input bits. Hint: since we are learning NAND from data, some post-processing may be needed on the prediction from the last layer of the network.

Your training data for this activity is the truth table for the NAND operator:

A	B	A NAND B
1	1	0
1	0	1
0	1	1
0	0	1

Train your network for 1,000 epochs using mean sum squared loss. Your training code should incrementally print the loss value as training proceeds. What is the value of the loss in the final epoch? Record the values.

After training, you should be able to reproduce the above truth table exactly by providing A and B as the two inputs. Your program should print both inputs and the output for all possibilities.

Activity 3: Build Three different Multi-Layer Perceptron Models for Bank Fraud Identification

You are given a CSV file with Bank Fraud Identification dataset that contains 20k+ transactions with 112 features (numerical) and 1 Target Feature (targets). In this activity, you will create a simple regression model using three different variants of Multi-layer Perceptron Models can predict a Bank Fraud based on the various features available.

First Spilt your dataset in the ratio of 80% for Training and retraining 20% for testing.

Input of the model will take up 112 features and outputs a Binary value of either 0 or 1.

You are free to use any loss function or optimizers to train your model.

The dataset file (**fraud_detection_bank_dataset.csv**) is available in the same github repository for your reference.