Introduction to Python Programming
Mini Project

Ali Kashefi
kashefi@stanford.edu

Problem 1. Many modern machine learning models rely on Deep Neural Networks (DNNs) to fit complex functions defined by real-world data sets. In practice, thousands of weights parameterize a DNN, and we train a model by finding optimal values for the machine learning model parameters. The optimal parameter values are determined by minimizing the model error as measured by a given loss function. The following example will motivate the usefulness of neural networks in data fitting. Let us consider the following data set.

input	data 1	data 2	data 3	data 4
x_1	0	0	1	1
x_2	0	1	0	1
x_3	1	1	1	1
output or prediction $\hat{y}$	0	1	1	0

Table 1: Data table

The data in Table 1 represents a sample of $m = 4$ input-output pairs, corresponding to the function $\hat{y}$ defined by

$$\hat{y} = \begin{cases} 0, & \text{if } x_1 + x_2 + x_3 \text{ is odd,} \\ 1, & \text{if } x_1 + x_2 + x_3 \text{ is even.} \end{cases}$$

Each x_i $(1 \leq i \leq 3)$ is either 0 or 1. We would like to design a deep-learning model that takes three inputs (x_i) and outputs (or predicts) 1 if the summation of these three inputs is even and outputs (or predicts) 0 if the summation of these three inputs is odd. Specifically, we use a neural network with one hidden layer (see Fig. 1). The code of the neural network is given to you (`hw4.py` and `hw4.ipynb`). Your task is to investigate the role of different parameters.

- Investigate the role of the number of neurons in the hidden layer.
- Plot the evolution of loss
- Investigate the accuracy of the network prediction for the case of $x_1 = 1, x_2 = 0, x_3 = 0$.
- Investigate the role of adding more data, listed in Table 2 in the accuracy of the prediction.

input	data 5	data 6
x_1	1	1
x_2	1	0
x_3	0	0
output or prediction $\hat{y}$	1	0

Table 2: Data table

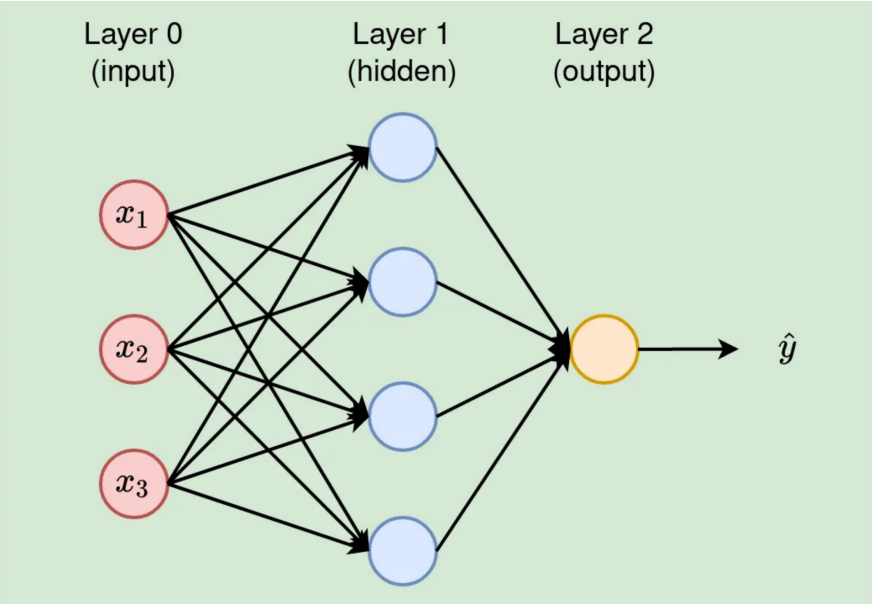

Figure 1: A deep neural network with one hidden layer. The image is taken from `https://towardsdatascience.com`