CS3630 Assignment 1

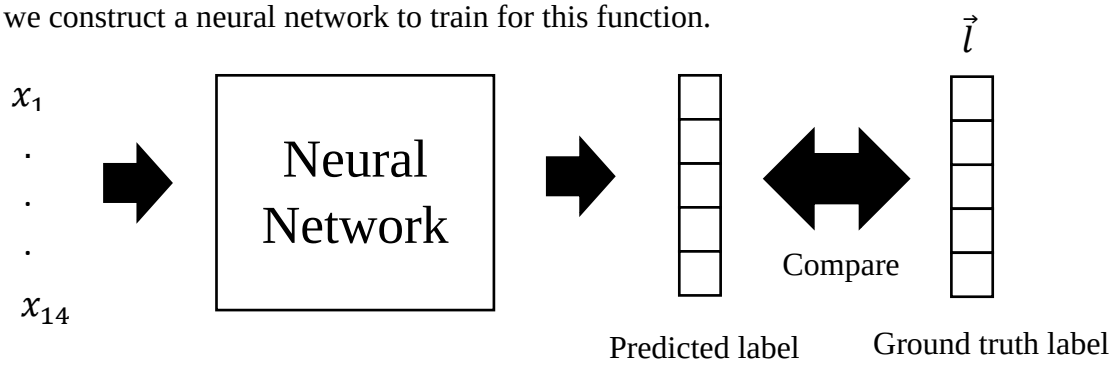
ASSIGNMENT DEADLINE l – October 9th, 2022, 11.59 PM.

You are given a data set that contains 14 attributes $\{x_1, x_2, \dots, x_{14}\}$. Each of these attributes is a binary attribute with a value -1 or 1. The instances in this dataset belong to one of four classes {1,2,3,4}. Hence, we can define the function f that takes these 14 binary attributes and outputs the respective class as follows.

$$f: \{-1,1\}^{14} \rightarrow \{1,2,3,4\}$$

$$f(x_1, \dots, x_{14}) \in \{1,2,3,4\}$$

Suppose we construct a neural network to train for this function.



The neural network takes an instance that consists of 14 attributes and predicts a label. We compare the predicted label with the ground truth label to determine the error made by the network. This error is backpropagated to the inner layers of the neural network to adjust the weights and biases of the network.

Note that we convert the given labels into one-hot vectors when training the neural network.

$$\vec{l} = \begin{cases} (1,0,0,0) & \text{if } f(x_1, \cdots x_{14}) = 0 \\ (0,1,0,0) & \text{if } f(x_1, \cdots x_{14}) = 1 \\ (0,0,1,0) & \text{if } f(x_1, \cdots x_{14}) = 2 \\ (0,0,0,1) & \text{if } f(x_1, \cdots x_{14}) = 3 \end{cases}$$

Suppose we use the architecture shown in Figure 1 for this neural network. Construct that neural network, implement the back-propagation algorithm, and use gradient descent to train the network. Use cross-entropy cost function on a Softmax function for training. All but the last fully connected layers have ReLU as the activation function**.**

Note: Use Python 3.6 for the implementation and you should not use any Python neural network libraries. However, you can use standard Python libraries such as NumPy, Pandas etc.

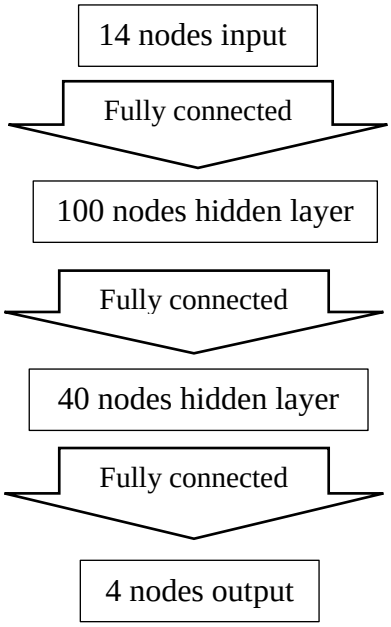


Figure 1: Neural network architecture

Tasks:

1. To verify that your implementation is correct, we give you a data point d, a weight & bias set, W_0 and b_0, together with correct gradients of the output (loss function) computed using the data point d and W_0 $\left(\frac{\partial ouput}{\partial W_0}\right)$ and b_0 $\left(\frac{\partial ouput}{\partial b_0}\right)$. The data point is in the `data_point.txt` file. W_0, , b_0 and correct gradients are in folder `Task_1/a`. To ease your understanding, in the weights and bias CSVs, there's one heading column introducing the weights/biases, while there is **NO** such column for the gradients.
   You are given another weight and bias set, W_1 and b_1, with no corresponding gradients given. Using the data point given above calculate the gradients $\left(\frac{\partial ouput}{\partial W_1}, \frac{\partial ouput}{\partial b_1}\right)$. The new set of weight and bias matrices are in folder `Task_1/b`. Save the gradients into CSV files and submit them for grading.
   Your submission should be in the same format as the given 'true-d*' files. For more details, please check **Appendix I**. Note that you should **NOT** include the headings in your submission CSVs. If possible, use **np.float32** to control the granularity of your gradients; otherwise, round your results to at least **16** digits. **Your submission will be automatically graded using a script. Therefore, be sure to format your output according to the given instructions. The incorrect format will be graded as wrong answers.**

2. Use the data given in `Task_2` folder to train the above neural network. Train three neural networks using 1, 0.1, and 0.001 as the learning rate. For each learning rate
   a. Plot the training cost w.r.t. iterations
   b. Plot the testing cost w.r.t. iterations
   c. Plot the train & test accuracy scores w.r.t. iterations

   Create a short report of two pages including these graphs and discuss the influence of the learning rate on the test accuracy.

Final Submission

- Your submission should be a single zip file renamed with your student ID. Other compressions are NOT acceptable.
- Inside the zip file is a PDF file and two folders; `Task_1` which contains results for task 1 and `code` which has your code.
- Within the `Task_1` folder, there should be two CSV files, dw, and db, corresponding to gradients w.r.t to W_1 and b_1 respectively. Another naming is **NOT** acceptable. The CSV files should use commas (i.e.,',') as the delimiter. Using space or other delimiters are **NOT** acceptable.

Appendix I: More details

Suppose a neural network layer t has N_t number of nodes,

- The weights from layer t to layer $t+1$ form a N_t by N_{t+1} matrix, where the $(ij)^{th}$ entry of this matrix represents the weight, w_{ij} connecting the i^{th} node of layer t to the j^{th} node of layer$(t+1)$.
- The bias is a length-N_t vector for layer t. Note that the input layer has no corresponding bias.
- Softmax is not considered to be a layer in this context, so the output layer output logits.

The given weights files have the following formatting:

- Totally $\sum_{t=0}^{\#\,layers-1} N_t$ rows
- The first N_0 rows are the matrix from input layer 0 to hidden layer 1, the following N_1 rows are the matrix from layer 1 to layer 2, and so on. There's **NO blank line** between matrix(t) and matrix(t+1)
- Then of each matrix, the (ij) item is w_{ij}.
- In the corresponding gradients files, the (ij) item is the derivative w.r.t. w_{ij}, $\frac{doutput}{dw_{ij}}$.

The given bias files also have the same formatting:

- Totally $\sum_{t=0}^{\#\,layers-1} 1$ rows
- The first row is the bias vector for layer 1 and has N_1 items. The second line is the bias vector for layer 2 with N_2 items, and so on.
- Of each row, the i item is either b_i.
- In the corresponding gradients files, the (ij) item is the derivative w.r.t. b_i $\frac{doutput}{db_i}$).

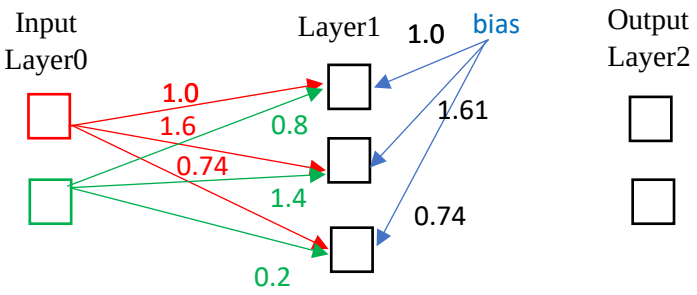


Figure 2: An example fully connected network

For example, the weights and biases file for the network in Figure.2 would be as follows:

Weights	Biases
1.0, 1.6, 0.74	1.0, 1.61, 0.74
0.8, 1.4, 0.2	0.01, −0.99
0.01, -1.87	
0.0.1, -1.23	
0.0, -0.34	

The CSV file you would submit should have the same format as above.

Appendix II: Example Evaluation Script

- Your result will be graded using a script like `sample_evaluation_script.py`. If you run it for verification, you will get an output of 154.
- Do try running your code with this script to adjust your formatting.
- Otherwise, you are likely to receive no points.