Basic & Application of Machine Learning

Homework 3 – Neural Network



Student Name : Jason Melvin Tedjokusumo

Student ID : 32205061

Major : SW 융합모바일시스템공학과

Professor : 이규행교수님

Submission Date : October 29th, 2022

* Background Introduction

A neural network is a network or circuit of biological neurons, or, more recently, an artificial neural network made up of artificial neurons or nodes. [1] Thus, a neural network can be either biological (made up of biological neurons) or artificial (used to solve artificial intelligence (AI) problems). Artificial neural networks model biological neuron connections as weights between nodes. A positive weight indicates an excitatory link, while a negative weight indicates an inhibitory connection. All inputs are weighted and added together. This activity is known as a linear combination. Finally, an activation function regulates the output amplitude. For example, an acceptable output range is often between 0 and 1, or it could be -1 and 1.

However, in this homework the goal is to design a neural network to approximate a function that can describe the given data. The dataset has two values, one (the first column in the given file) is x, and the other one (the second column) is y (y=f(x)). After designing the model and training, find its best parameter set and provide both the model and weights

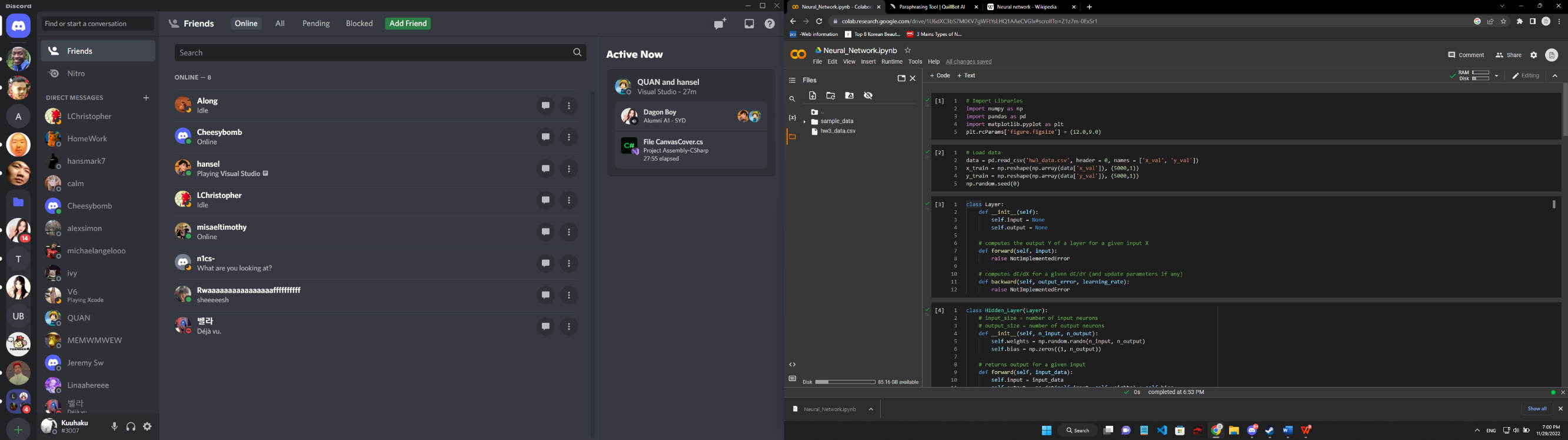
* How to Run

I use Google Colabs on a web browser to run the program. First, open Google Colabs, when prompted to open files, you can first upload the files to google drive then open it from google drive section, or you can upload it from local drive immediately.

Graphical user interface, application

Description automatically generated

After that, you need to upload the data files too, which are “hw3\_data.csv”, by using the upload button just like in the picture.



Finally, you can run the program by pressing “Ctrl + f9” or just click run on each bar from top to bottom in orderly.

* Design  
  The main idea to design this neural network is using a basic MLP (Multilayer perceptron), we provide the basic structure of the layer, the hidden layer, the activation function for each layer (except the input layer), the loss function, and building the network itself.

Text

Description automatically generatedText

Description automatically generatedSo, the inputs will be fed to the next layer, by doing forward propagation, where we calculate the result with the initialized weights and biases for each perceptron on each hidden layer. After that, we use the result to the activation function in that hidden layer, then back to propagate forward by calculating the result with different weights and biases, and so on until we reach the output layer, then we got our final results which is the predicted value.

After that, we need to calculate the loss, which is the difference between the predicted y value and the exact y value. We use that loss to update the parameters which are fed to this outer layer or output layer. By doing the partial derivatives of the loss function with respect to each weight and bias and using the chain rule. Then, the results for each of these losses in all perceptron in previous hidden layer will be their loss to update the parameters as it propagates backwards. Finally, we iterate this process so that the machine can learn and find the best parameters for the given data.

Text

Description automatically generated

* Text

  Description automatically generatedEvaluation Results  
  Text

  Description automatically generated  
  So, I make class object for the Network class, and I add 3 hidden layers in which every time after the activation function is occur, so in total there is 6 layers, where the last hidden layer is the output layer. I make 1-10-10-1 structure.  
    
  And then, I train the data, with 10 iterations only, with 0.1 learning rate. However, the results are not what I expected. I don’t really know what went wrong, and I kind of lost it when trying to implement different way of backward propagation. I feel that the backward propagation is wrong, but I don’t know the mistake.  
    
    
    
    
    
    
    
    
    
    
    
  I try to change the iterations and the learning rate, but it doesn’t matter because only after a few run, the loss doesn’t decrease anymore, and the results always 0, maybe this is because of the ReLU activation function, but there should have been another result other than 0. But I couldn’t find it.

Chart, line chart

Description automatically generated

* Conclusion  
  I am sorry that I couldn’t finish this homework, I took a really long time researching for this, and also, I am not that proficient in Python which makes me harder to solve and understand this neural network, I just understand the concept behind this, however, to write the algorithm manually by myself is very difficult. Thank you.