Prometheus AI Final Report

Winter 2022 Lab COMP 396

Denis Aleksandrov

Neural Network Layer (NN):

The Network class

This class is the implementation of the feed forward neural network. The main structure of the network is made up of a matrix of Neurons, the layers of the network are the vectors at each index of the matrix. Each Neuron has an array of weights that it has between every Neuron in the previous layer. Each Neuron also has an output, bias, backpropagation error, and derivative of the output.

This implementation was largely based off of two YouTube tutorials on neural networks done by 3Blue1Brown and Finn Eggers. You can find the tutorials in the references below. I highly recommend to whoever is planning on improving this project to watch the tutorials. 3Blue1Brown and Finn Eggers do an amazing job at conceptually explaining neural networks, you will likely find the tutorials helpful in understanding my implementation.

The CGMoveALine class

Most of this class was written by Professor Joseph Vybihal, this class acts like a data harvester for the network to be trained on. If you run this class, a window will pop up with a little square which you can move around with the arrow keys. This square acts like the robot for which we need to simulate sensor data. This class generates this data and target values for each data point, the file that is generated is the one that is used in the Trainer class to train the network.

I added the distance sensor data recording for when the square moves around the screen and functionality on adding walls to the screen (the lines that you can see on in the

window). The sensors that are accounted for are the; Left diagonal (LD), front (F), right diagonal (RD), and back (B). This class records each positional point of the square as the four distances to any walls and the corresponding target values; Far , near, blocking. This means that there is a one-to-three mapping between each input value and its corresponding output values.
e.g. LD, F, RD, B, LDfar, LDnear, LDblocking, Ffar, Fnear, Fblocking, RDfar, RDnear, RDblocking, Bfar, Bnear, Bblocking

The Trainer class

This class is meant to train network objects. The train method takes input and target matrices, the learning rate (eta), number of training iterations, and the number of times the iterations are rerun as parameters. Each rerun decreases the number of iterations by half. The input and target matrices are generated by the getDataFromFile method. Additionally there is an ORTest method to test the working of the correctness of the network implementation. This method additionally shows the functionality of the Network class.

Discussion

The problem with my implementation is that there is a one-to-three mapping between each input value and output values. This makes training of the network very inefficient. The next person working on this should focus on coming up with a way to record the data in a one-to-one mapping between input and output values. Considering the current implementation, this work would mainly be focused around the CGMoveALine class

(try starting around lines 323 to 372) and the Trainer class. The Trainer class will need to be modified mostly at the getDataFromFile method.

The next steps for this project, besides what I had mentioned above, are to add the training functionality for all of the other sensors

References

3Blue1Brown Tutorial

Finn Eggers Tutorial