Homework 9

April 18, 2019

1 Problem 1

Compare the average loss of the two models. Provide an explanation for what you observe. (10 points)

```
Running models on data/wine.txt dataset
——— 1–Layer NN ———
Average Training Loss: 0.544947665196195
Average Testing Loss: 0.663365600887563
——— 2–Layer NN ———
Average Training Loss: 0.4943844983085639
Average Testing Loss: 0.6258665537308302
```

The 2 Layer Neural Network has a higher accuracy for both testing and training data. Since the 2 layer NN has a hidden layer, it is able to model more complex data unlike the 1 layer NN.

2 Problem 2

Comment on your parameter choices. These include the learning rate, the hidden layer size and the number of epochs for training. (10 points)

I used learning rate=0.001 (1 layer) and learning rate = 0.01 (2 layer), epochs=25, hidden layer size = 10 (the default parameters).

With a lower learning rate (0.0001) the loss did not change much but it took longer for the program to run and with a higher learning rate (0.1) I got a higher loss. The same trend happened when changing the 2 layer learning loss.

With less epochs (10) the loss was higher, but for more epochs (50) the training loss slightly lower or the same, but the time it took for the program to run was much longer.

A smaller (m=2) hidden layer size increased the loss, but a larger hidden layer (m=20) decreased the training loss but increased the testing loss.

3 Problem 3

(Extra Credit) Train your neural network on every activation function provided on Slide 15 from Lecture 17. Comment on the results.

```
Step function:
```

```
Average Training Loss: 0.6589726892311849
Average Testing Loss: 0.7771729435940152

ReLU:

Average Training Loss: 0.485344256134619
Average Testing Loss: 0.6499353960116795
```

The Step function did worse than both the ReLU and Sigmoid function on both training and testing data. The ReLU had a lower training loss, but higher testing loss than the sigmoid. This could be because the paramaters tuned earlier were tuned for the sigmoid function not the ReLU.

4 Problem 4

(Extra Credit) Construct a fake dataset that has really low training error on the 2-layer neural network and high training error the 1-layer neural network. In a file generate data.py, you should write a function generate data that returns arrays X and Y that can be passed to the train functions of any of the three models. Comment on your approach and the results in your report.

Uncommon the lines in main.py to generated dataset for X and Y instead of loading the wine dataset. The dataset I generated was XOR as it is difficult for 1 layer neural network as seen before in lecture. However the 2 layer NN does much better classifying the XOR as seen by the higher testing and training accuracy.

```
——— 1-Layer NN ———
Average Training Loss: 0.24553583710370366
Average Testing Loss: 0.23994749719412142
——— 2-Layer NN ———
Average Training Loss: 0.007495842447141998
Average Testing Loss: 0.007474206909993705
```