CS573 DM HW3 SOLUTION

Q1 Deep Learning

Question 1.1

Forward Pass

First hidden layer intermediate value (z_1), (x is input, W_1 is the first hidden layer weights, b_1 is the bias for first layer)

$$z_1 = xW_1 + b_1$$

Pass it through ReLU Activation

$$h_1 = ReLU(z_1)$$

Similarly for second hidden layer,

$$z_2 = xW_2 + b_2$$

$$h_2 = ReLU(z_2)$$

Finally for output layer,

$$z_3 = xW_3 + b_3$$

$$\hat{y} = softmax(z_3)$$

Back propagation

As explained in the lecture, for i^{th} training example error is $(y_i - \hat{y_i})$ and score is

$$L(i) = \sum_j y_i(j) log \hat{y}_i(j)$$

where $y_i(j)$ is the element j of the vector representing the one-hot encoding of the class of training example i and $\hat{y}_i(j)$ is the output of the j-th output neuron for training example i

$$\frac{\partial L(i)}{\partial z_3} = y_i - \hat{y}_i$$

The derivative of last layer of parameters

$$\frac{\partial L(i)}{\partial W_3} = h_2^T(y_i - \hat{y}_i)$$

$$\frac{\partial L(i)}{\partial b_3} = (y_i - \hat{y}_i)$$

Similarly derivatives for other layers,

$$\frac{\partial h_2(i)}{\partial z_2} = \begin{cases} 1 & h_2 > 0 \\ 0 & otherwise \end{cases}$$

$$\frac{\partial L(i)}{\partial W_2} = h_1^T \frac{\partial h_2(i)}{\partial z_2}(y_i - \hat{y}_i)W_3^T$$

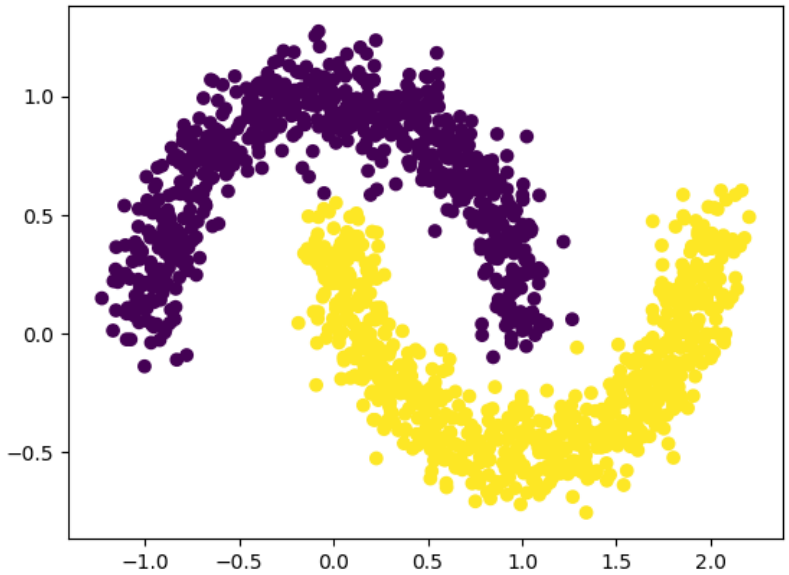
$$\frac{\partial L(i)}{\partial b_2} = (y_i - \hat{y}_i)W_3^T$$

$$\frac{\partial L(i)}{\partial W_1} = x^T \frac{\partial h_1(i)}{\partial z_1} W_2^T \frac{\partial h_2(i)}{\partial z_2}(y_i - \hat{y}_i)W_3^T$$

$$\frac{\partial L(i)}{\partial b_1} = \frac{\partial h_1(i)}{\partial z_1} W_2^T \frac{\partial h_2(i)}{\partial z_2}(y_i - \hat{y}_i)W_3^T$$

Question 1.2

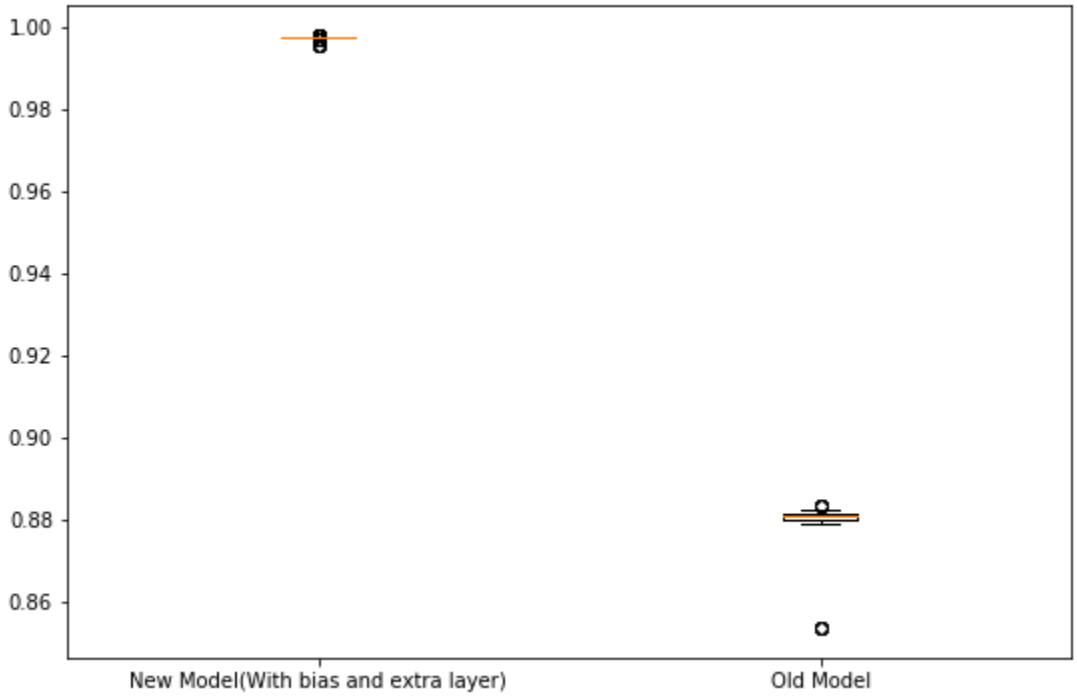
Accuracy :
Iteration (epoch) 0
Iteration (epoch) 1
Iteration (epoch) 2
Iteration (epoch) 3
Iteration (epoch) 4
Iteration (epoch) 5
Iteration (epoch) 6
Iteration (epoch) 7
Iteration (epoch) 8
Iteration (epoch) 9
Accuracy after 10 iterations: 0.9966666666666667
Scatter Plot :



The accuracy has increased in this model as compared to the original model without bias or extra hidden layers. Adding bias and another layer improves the ability of the model to learn the function.

Question 1.3

Original Model : The values corresponding to 20 runs are :
{0.88, 0.882, 0.88133333, 0.88066667, 0.87933333, 0.88, 0.882, 0.88133333, 0.88133333, 0.88066667, 0.87866667, 0.85333333, 0.882, 0.88133333, 0.88, 0.88333333, 0.88066667, 0.88, 0.88066667, 0.87933333}
Average classification accuracy over 20 runs : 0.8794
Standard deviation : 0.00607
New Model(with bias and hidden layers) :
The values corresponding to 20 runs are :
{0.99733333, 0.99666667, 0.998, 0.998, 0.99733333, 0.998, 0.99733333, 0.99733333, 0.99733333, 0.99733333, 0.998, 0.998, 0.99733333, 0.99733333, 0.99733333, 0.99733333, 0.99733333, 0.99666667, 0.99533333, 0.99666667}
Average classification accuracy over 20 runs : 0.9972
Standard deviation : 0.000613
Box plot :



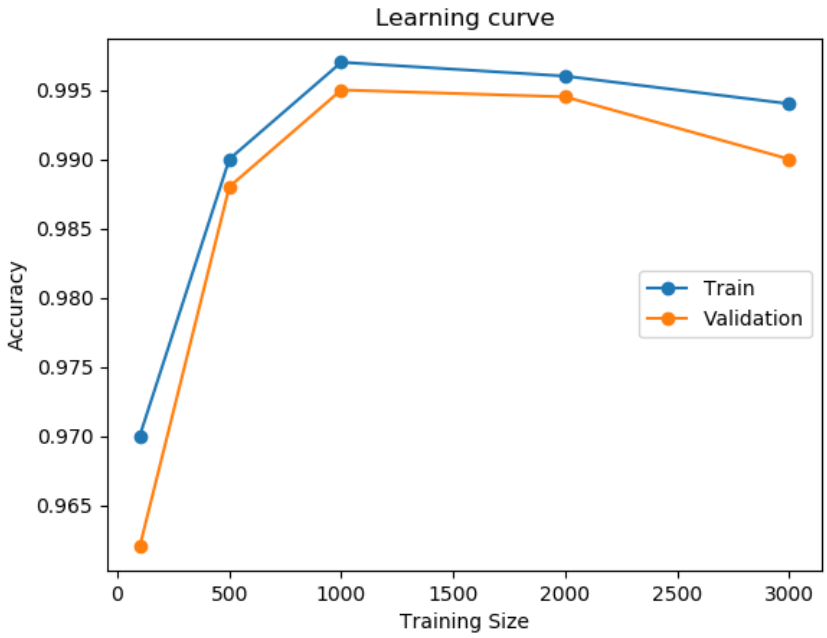
Now we will perform a hypothesis test to compare performance of the two models. Let μ_{NEW} be

the mean accuracy of the new model and μ_{OLD} the mean accuracy of the old model.
Null Hypothesis, H_0: The accuracy of both the models are the same: $\mu_{NEW} = \mu_{OLD}$
Alternate Hypothesis, H_1: The new model has higher accuracy. $\mu_{NEW} > \mu_{OLD}$
We set significance level for the test, α to reject the null hypothesis be 0.05.
We will use the python scipy inbuilt function to perform the t-test. Lets suppose the accuracies for the 20 runs for two models are stored respectively in lists "OLD" and "NEW"
We will use the function:

```
from scipy import stats
t_statistic ,p_value=stats.ttest_rel(OLD,NEW)
```

The p-value above command generates is 1.0810418203534019e-25. It is very less than the significance level we chose for the test. Hence we reject the null hypothesis that the two models mean accuracy is the same iand we can conclude that the according to the alternative hypothesis the NEW model performs better than the OLD model.

Question 1.4



Normally we would expect the training accuracy to go down with increasing training size. However, in case of neural networks, they have much higher capacity than normal models to model data, which is why the training accuracy doesn't decrease (or decrease significantly) with increasing training size.

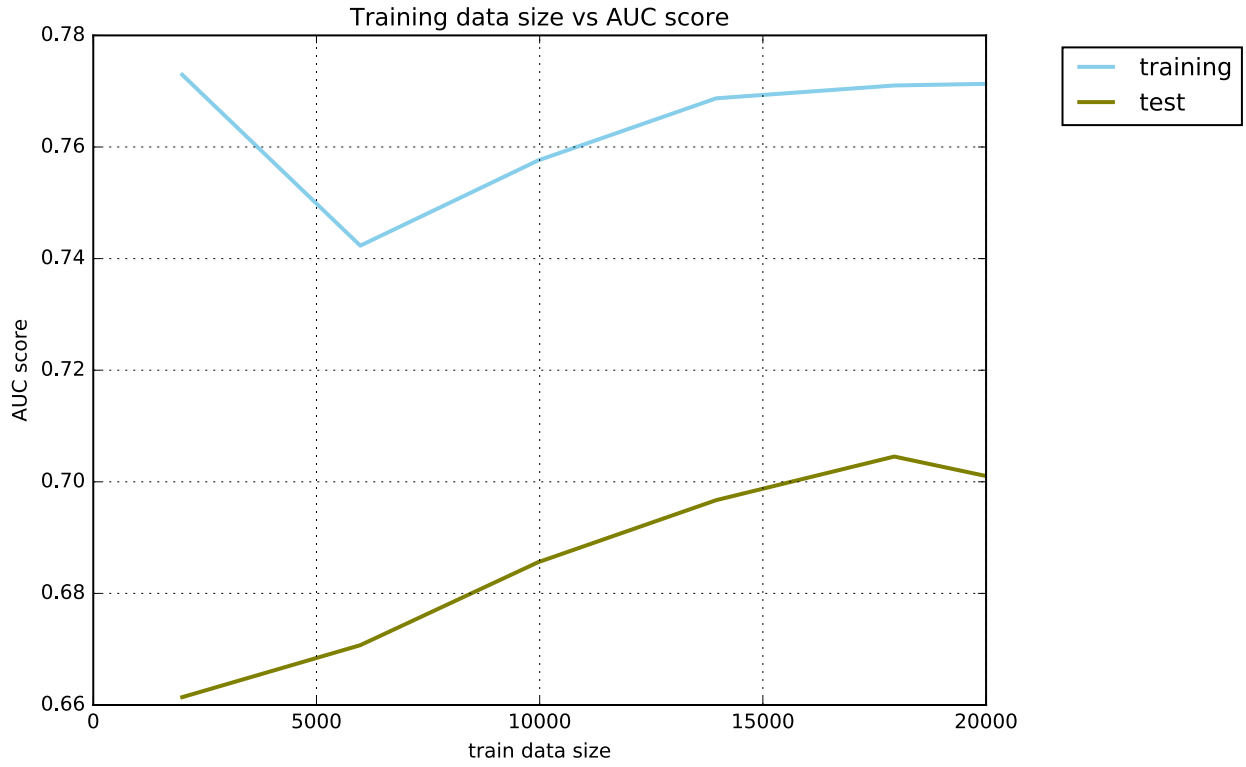
Question 1.5

The condition is all the training examples need to give negative values as input to the ReLU function. The ReLU function will output zero. Hence the gradient will become zero. If all examples give negative values as input then no mini batch can give any non zero value as output and the neuron will always return zero value.

Q2 Cross Validation

Question 2.1

The figure below shows the learning curve (training data size vs. AUC score) of neural networks.



Note that as the training data size increases, training AUC score increases. This is unlike what you have seen in class. With more data, the models with less capacity will have more difficulty in fitting the training data. Hence, you will expected a drop in training accuracy as the training data increases. However, with neural network which has more capacity to model the data, the training accuracy does not drop as we increase the training data. The drop in the figure below comes from statistical fluctuation.

As for test AUC score, it also increase as the training data size increase. With less data, in the search space, there are more candidate models that looks good with the data. When we increase the training data size, some of those candidate models will be dropped.

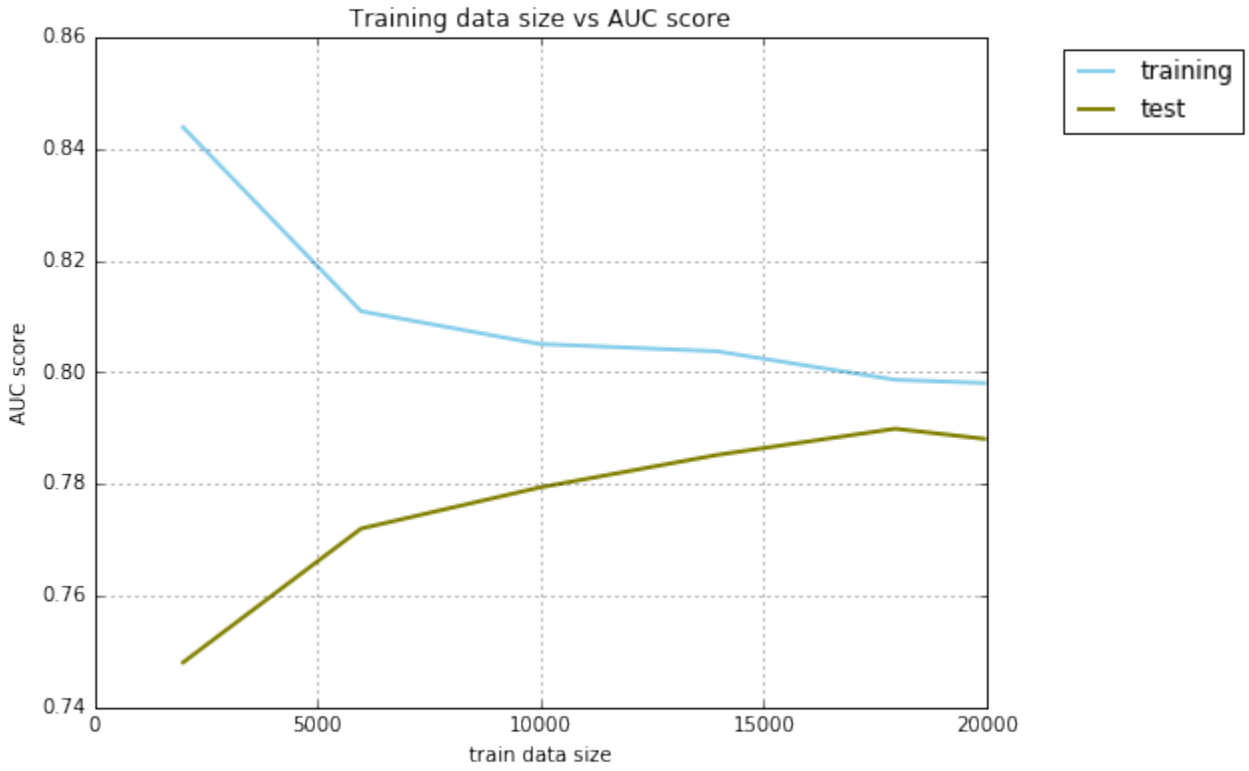
Question 2.2

Logistic Regression :
The figure below shows the learning curve (training data size vs. AUC score) of neural networks.

This is unlike what you have seen in class. With strong regularization, we are limiting the search space. We have fewer bad models because of the statistical fluctuation. It does not mean you are supposed to find a good model. It may be the case that the good model with less bias is out of your search space.

Boosted Decision Tree :

The figure below shows the learning curve (training data size vs. AUC score) of boosted decision tree.



This figure is more like what you have seen in class. As the training data size increases, training AUC score decreases while test AUC score increases. Training accuracy decreases because the model doesn't have the capacity to model more data. Eventually training and test accuracy will meet.

Question 2.3

Let μ_{lr} be the mean AUC score of logistic regression model ,μ_{nn} be the mean AUC score of neural network model and μ_{bd} be the mean AUC score of boosted decision tree.

Between boosted decision tree and logistic regression,
Null hypothesis $H_0 : \mu_{bd} - \mu_{lr} = 0$
Alternative hypothesis $H_1 : \mu_{bd} - \mu_{lr} > 0$

Between boosted decision tree and neural network,
Null hypothesis $H_0 : \mu_{bd} - \mu_{nn} = 0$
Alternative hypothesis $H_1 : \mu_{bd} - \mu_{nn} > 0$

Significance level will be 0.05, but since we are testing two hypothesis, we need to apply multiple hypothesis test correction. By using Bonferoni's correction, corrected significance level = 0.05/2. Then we perform t-test:

1. Get the difference between the two arrays X and Y, denote it as $d = X - Y$. For example, X is the AUC scores of boosted decision tree and Y is the AUC scores of neural network.

2. Calculate the mean $\bar{d}$ and the standard deviation $s = std(d)$.

3. Get the statistics as
$$t = \frac{\bar{d} - 0}{\frac{s}{\sqrt{n}}},$$
where n is number of samples.

4. Obtain the p-value $p = P(T > t)$.

Note that we are doing one-tailed t-test. More specifically, upper-tailed t-test. However, `ttest_rel` or `ttest_ind` method from `Scipy` package are implemented as two-tailed t-test. Therefore, directly using those methods will not give the correct p-value.

p-value for bd and lr: $1.881e-06 < 0.025$
p-value for bd and nn: $1.748e-05 < 0.025$
Hence, we reject both the null hypotheses. The boosted decision tree is the best model among the three.

Question 2.4

The best performing hyperparameters is reported in the table below:

Fold ID	Learning rate	Batch size	AUC score over validation
0	0.01	100	0.679
1	0.01	100	0.704
2	0.01	100	0.714
3	0.01	1000	0.745
4	0.01	100	0.762
5	0.01	100	0.756
6	0.001	1000	0.750
7	0.01	100	0.721
8	0.01	1000	0.764
9	0.01	1000	0.741

Between neural network and logistic regression,
Null hypothesis $H_0 : \mu_{nn} - \mu_{lr} = 0$
Alternative hypothesis $H_1 : \mu_{nn} - \mu_{lr} > 0$

Between neural network and boosted decision tree,
Null hypothesis $H_0 : \mu_{nn} - \mu_{db} = 0$

Alternative hypothesis $H_1 : \mu_{nn} - \mu_{db} > 0$

Significance level will be 0.05, but since we are testing two hypothesis, we need to apply multiple hypothesis test correction. By using Bonferoni's correction, corrected significance level = 0.05/2. The t-test procedure is the same as in question 2.3.

p-value for nn and lr: $0.9998 > 0.025$
p-value for nn and db: $0.9117 > 0.025$
Therefore,we failed to reject the null hypothesis between neural network and logistic regression and the null hypothesis between neural network and boosted decision tree. Hence, we cannot say that neural network wins.

Question 2.5

Let μ_{lr} be the mean AUC score of logistic regression model and μ_{bd} be the mean AUC score of boosted decision tree.

Since there are four different neural network models lets suppose the μ_{nn1}, μ_{nn2}, μ_{nn3}, μ_{nn4} are the corresponding mean AUC scores for these 4 models. Let's define a random variable:

μ_{NN}=max{μ_{nn1},μ_{nn2},μ_{nn3},μ_{nn4}}

Now our hypothesis tests will be different from Q2.4
Between boosted decision tree and logistic regression,
Null hypothesis $H_0 : \mu_{bd} - \mu_{lr} = 0$
Alternative hypothesis $H_1 : \mu_{bd} - \mu_{lr} > 0$

Between boosted decision tree and neural network,
Null hypothesis $H_0 : \mu_{bd} - \mu_{NN} = 0$
Alternative hypothesis $H_1 : \mu_{bd} - \mu_{NN} > 0$

Using basics of statistics we know that if there are iid random variables X_i, i=1,2,3,4, the distribution of random variable $Y = \max\{X_1, X_2, X_3, X_4\}$ will be different from the X_i's.
So, clearly the testing procedure we did in Q2.4 cannot be used directly for this part. The null hypothesis is defined over the max accuracy of four models, and that needs to be accounted for in the statistical test.