Deep Learning Homework Report 1

Jayanth Talasila

jtalasi@clemson.edu

Dr.Feng luo

CPSC-8430

GITHUB

HW1-1 Report Questions

Simulate a Function:

- Three neural networks named model0, model1, and model2 with different numbers of parameters were trained on two functions. Model0 has 571 parameters and has seven layers, Model1 has 572 parameters and has four layers, and Model2 has 572 parameters and only one layer. The functions they were trained on were (i) y = (sin(5 * (π * x))) / (5 * (π * x)) and (ii) y = sign(sin(5 * π * x1)). All models were trained using a learning rate of 0.001.

Function plot for for y = (sin(5 * (π * x))) / (5 * (π * x))

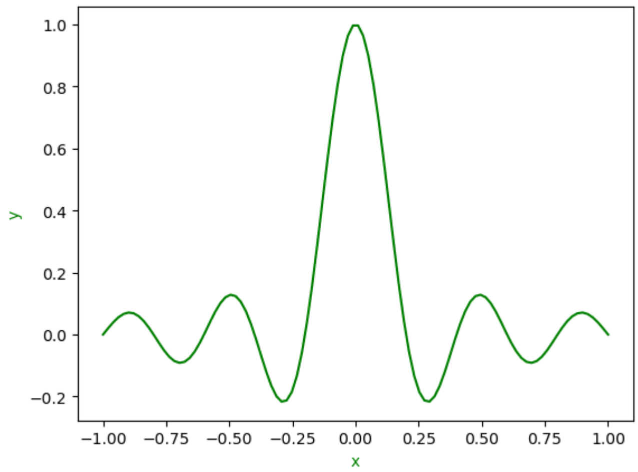


Fig1

Figure 3 presents the comparison between the actual values and the predicted values for the (sin(5*(pix)))/((5(pi*x)) function. This figure provides a visual representation of the performance of the models during the training process, and how well they were able to capture the underlying patterns in the function.

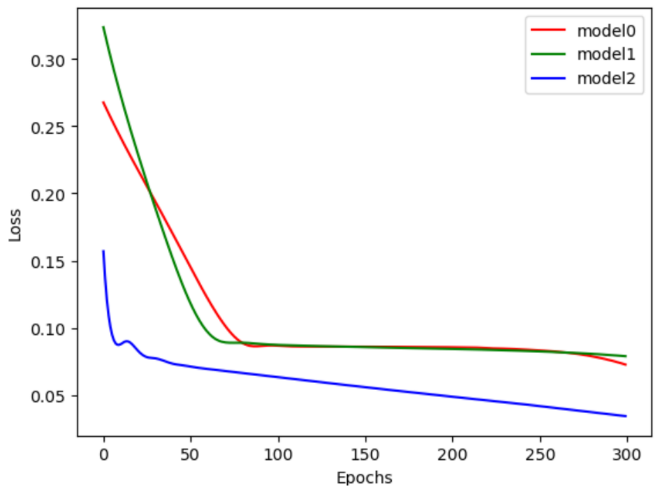


Fig-2

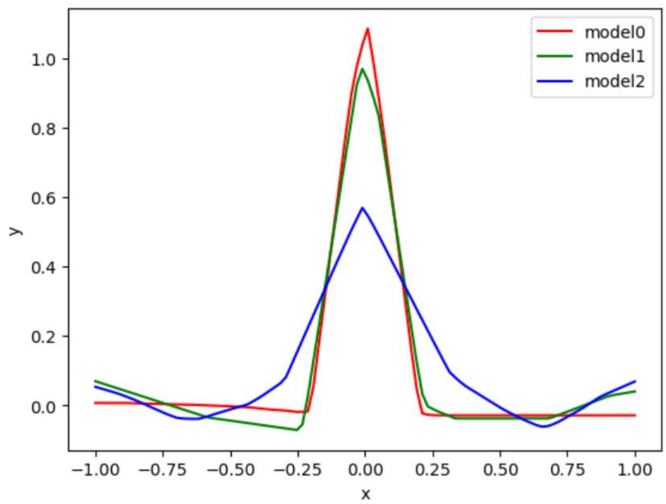


Fig-3

Function plot for function y = sign(sin(5 * π * x1)).

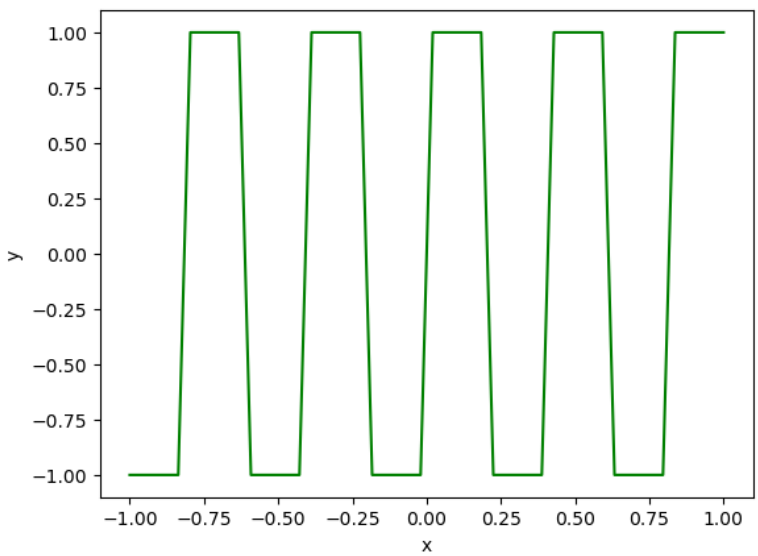


Fig-4

simulation is presented in Figure 5. In this simulation, the aim was to train these models to accurately predict the output for the sign(sin(5piX1)) function. Figure 6 showcases the comparison between the actual values and the predicted values for the sign(sin(5piX1)) function. As can be seen from the graph

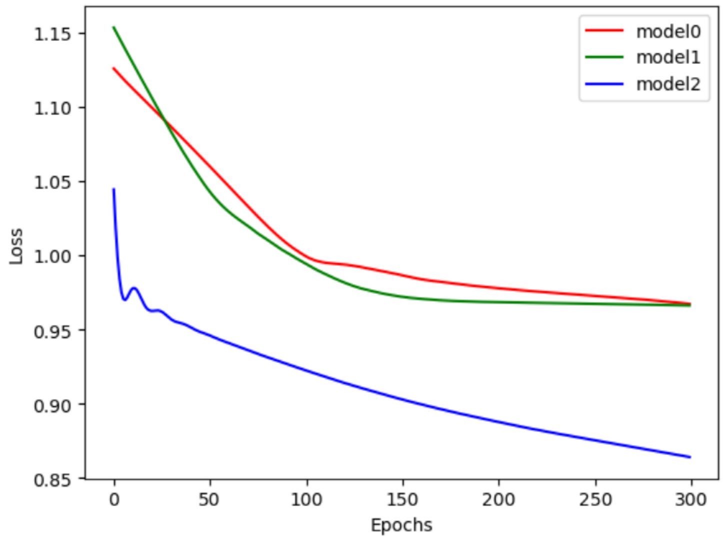


Fig-5

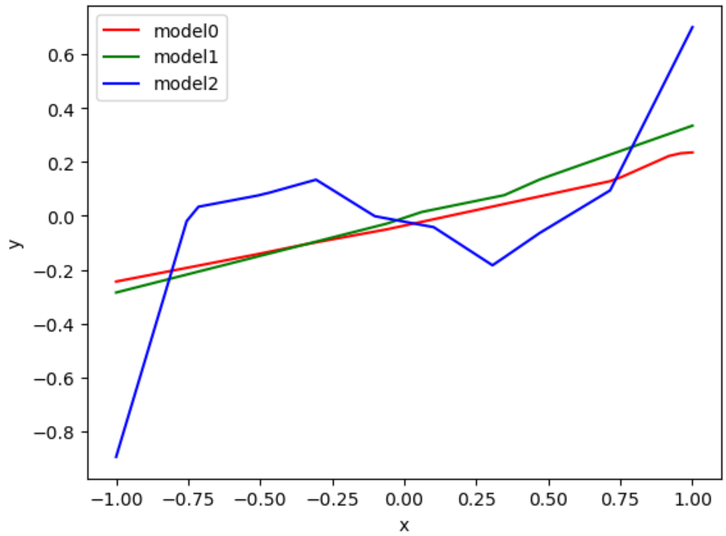


Fig-6

- The loss functions for both functions showed convergence after a reasonable number of training iterations.
- Have use more than 2 models in the previous questions.
- Have used more than one function in the previous question.

Train on Actual Tasks

- For the train on actual tasks section, I have built two CNN models. The data set that I used is downloaded from Kaggle and it contains of 60000 of training and 10000 for testing.

- performing 60 epochs of training and testing. The training process involves looping through batches of training data, applying the model to the batch, and keeping track of the number of correctly predicted samples.

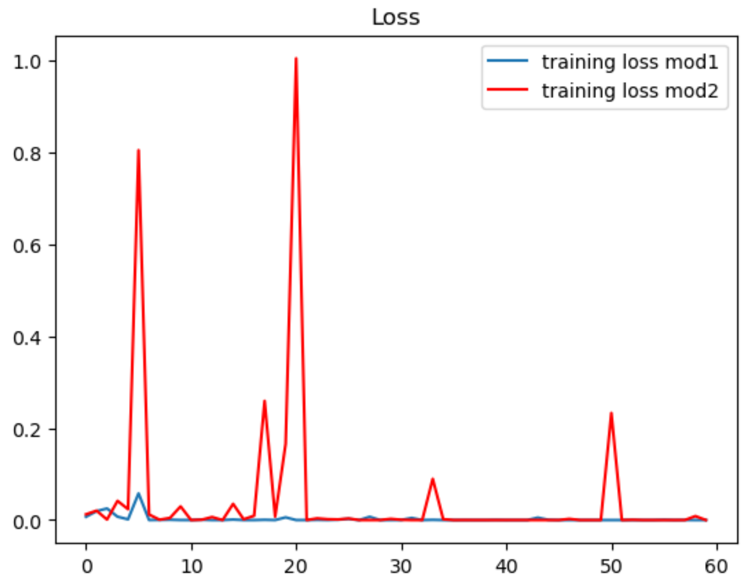


Fig-7

training loss for two models (mod1 and mod2) on a graph. Convergence of the both the models is occurred 55 epochs.

-

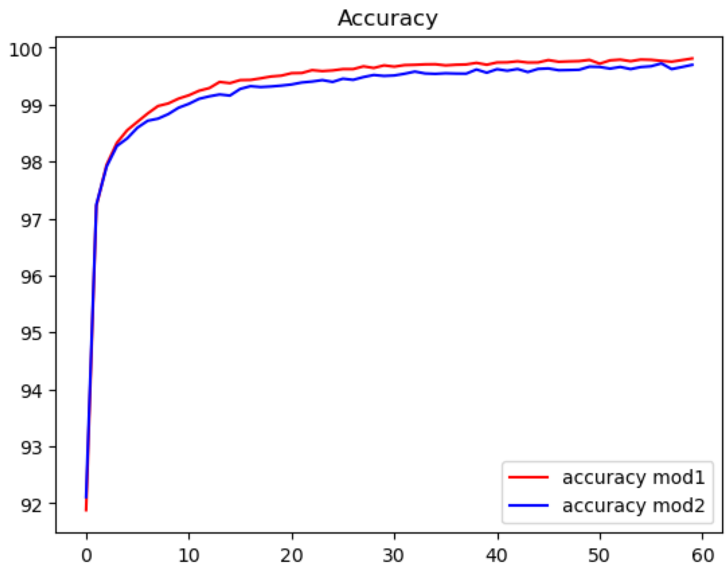


Fig-8

plotting the accuracy of two models (mod1 and mod2) on a graph. The accuracy is calculated by dividing the number of correctly predicted samples by the total number of samples in each batch. In this code, the total number of samples in each batch is assumed to be 600 (10 batches of 60 samples each). It can be seen that both models perform better on the training data compared to the test data, which is a common characteristic of machine learning models.

HW1-2: Optimization:

Visualize the optimization process:

training of a deep neural network with three fully connected layers and 57 parameters to replicate the function (sin(5*(pix)))/((5(pi*x)). The loss function used is the cross-entropy loss and the optimization algorithm used is Adam. The learning rate for Adam is set to 0.001. Eight different training series, each of 30 training epochs was carried out, during which collected Weights were gathered, and dimension reduction was achieved through PCA.

below picture shows the 2nd layer of the network

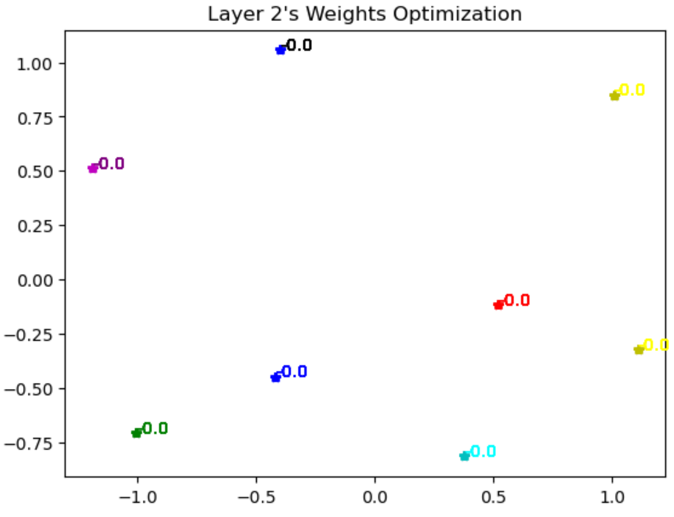


Fig-9

Below picture indicates whole model optimization.

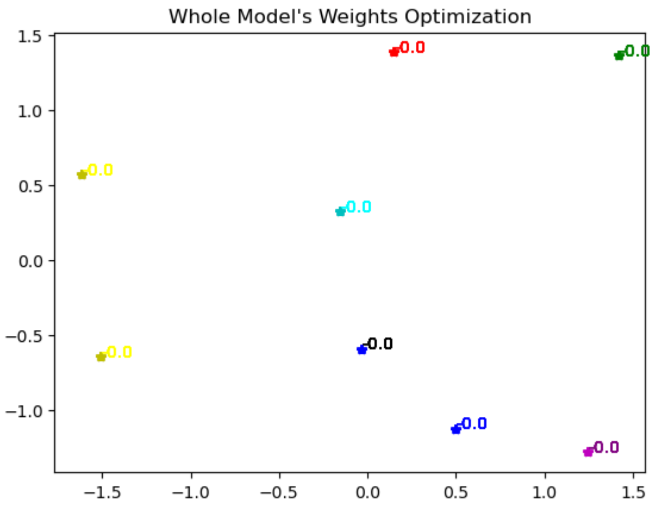


Fig-10

Figures 7 and 8 show the optimization process of a deep neural network, which is accomplished through backpropagation. The figures show the gradual fine-tuning of the weights over time as the network continuously adjusts its parameters to better fit the training data. This highlights the importance of allowing the network enough time to train and the need for patience in the training process.

Observe gradient norm during training:

Figure 9 provides a visual representation of the loss that the model experienced during each epoch of training. The loss function measures the difference between the model's predictions and the actual target values.

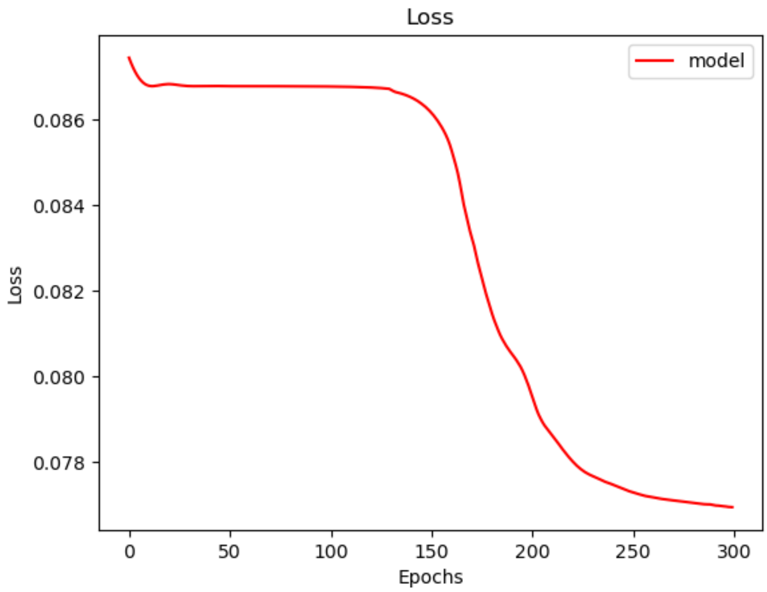


Fig-11

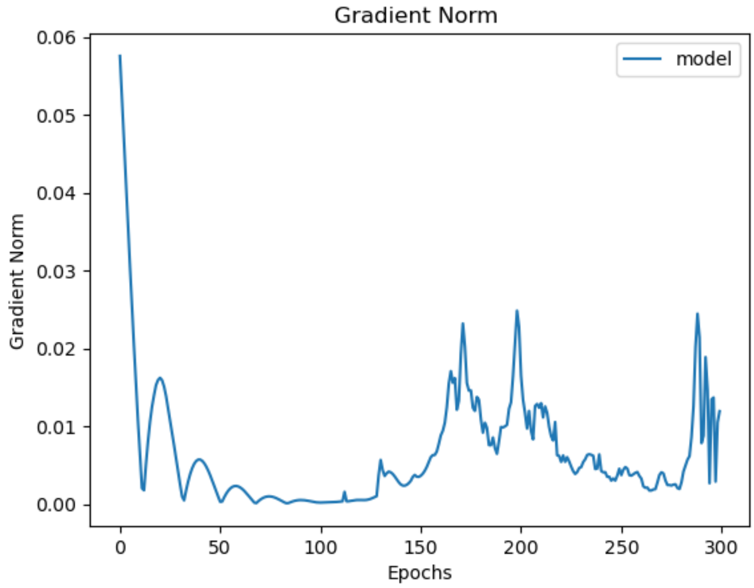


Fig-12

Figure 10, displays the gradient norm for each epoch, which represents the rate at which the model's parameters are changing.

The spikes that can be seen in Figure 10 correspond to the changes in the slope of the loss function, as represented in Figure 9. The spikes in the gradient norm indicate sudden changes in the optimization direction.

What happens when gradient is almost zero?

When the gradient is nearly zero, the model is on the verge of achieving a local minimum in the loss function, which is used to optimize the model's parameters. During the training phase, the gradient is utilized to update the model's parameters to minimize the loss function.

Part-3 Generalization:

Can network fit random labels?

The training data set I used is downloaded from Kaggle. It’s a mnist training set. The class has four fully connected layers. The number of neurons in each layer can be identified through the parameters i, z1, z2, z3, and o. the learning rate is 0.001. The activation function used between the linear layers is ReLU

Graph(fig11) showing the loss for both the training and test sets for each epoch in the model training process.

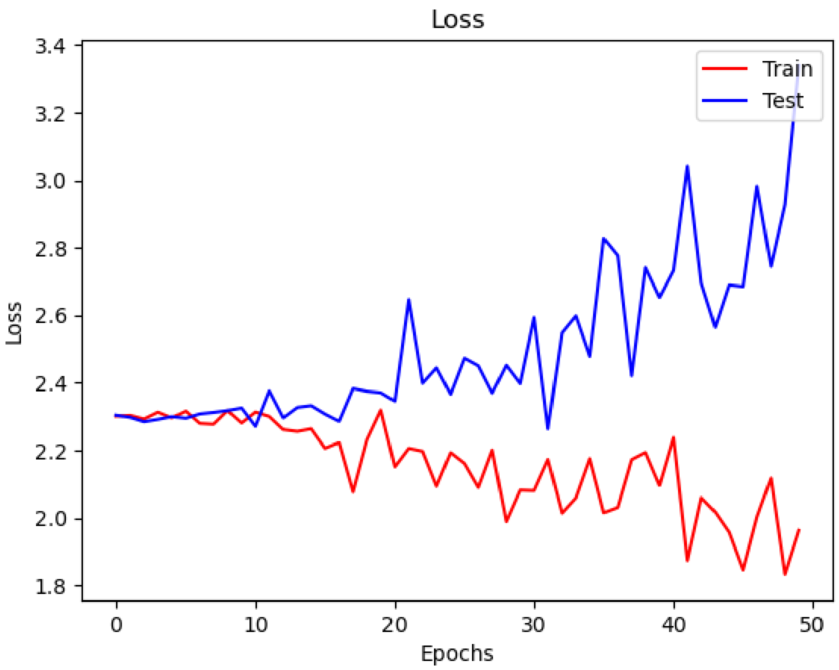


Fig13

In fig11, we can observe that the test data has an increase in loss while the training data has the loss decreasing. So it cant fit random variables.

Number of parameters v.s. Generalization:

In this part of HW we train 10 CNN models with mnist data set, while all models having the same structure but changing the parameters. All the models learning rate is 0.001.

Loss vs number of parameters

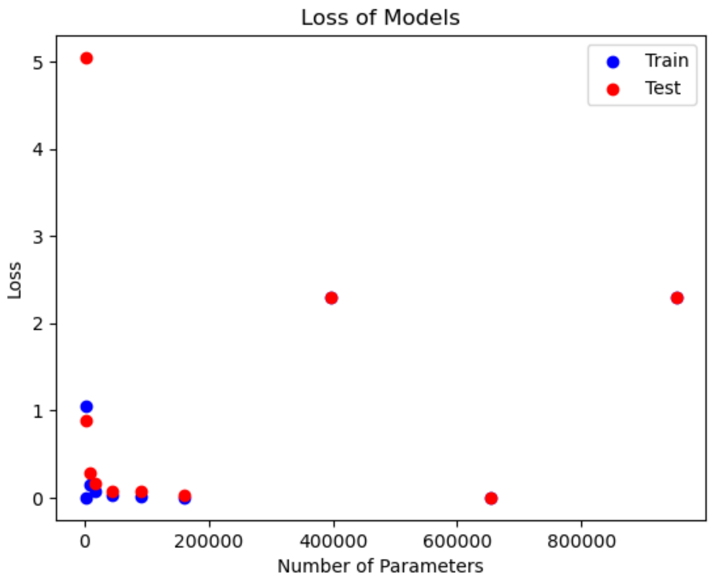


Fig-14

Accuracy vs number of parameters.

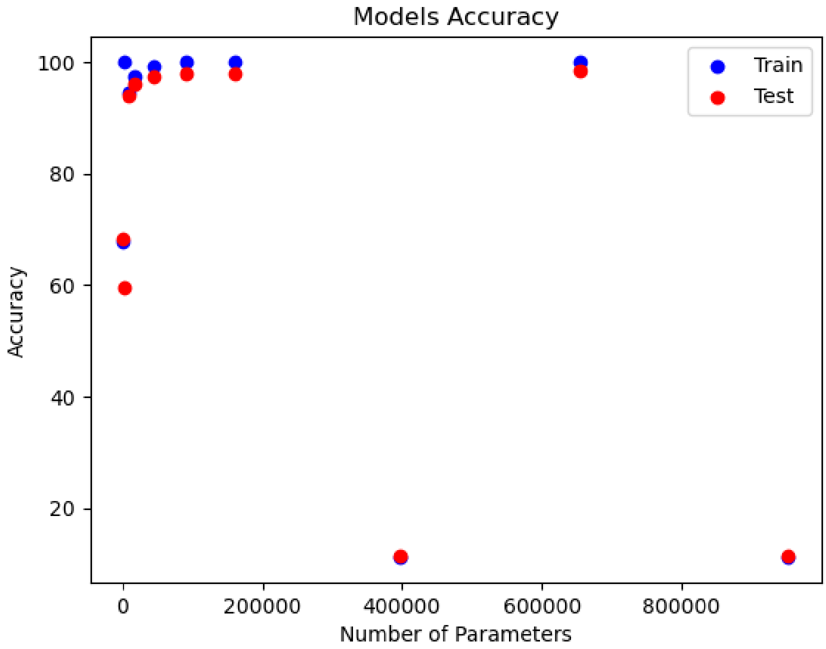


Fig-15

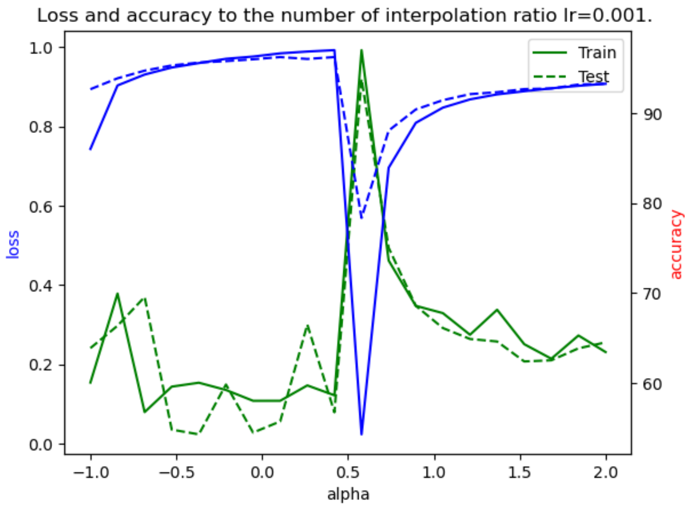
From the Figure presented, it is evident that when the number of parameters within a model structure is increased, the performance of the model improves. increasing the number of parameters results in a better performance of the model. The reason behind this can be said as, with more parameters, the model is able to learn and capture more complex relationships in the data, thus leading to a better fit and higher accuracy.

Flatness v.s. Generalization:

Part-1:

I have chosen mnist data set for training and testing the data. Two DNN models were created and implemented with different batch sizes 64 and 1024 and I have used adam optimizer. The learning rate for both the models was 0.001.

Below fig 16 shoes us the loss and accuracy of the 2 models that we have developed when the learning rate is 0.001



Below fig-17 shows us the loss and accuracy of the 2 models that we have developed when the learning rate is 0.01

Fig-16

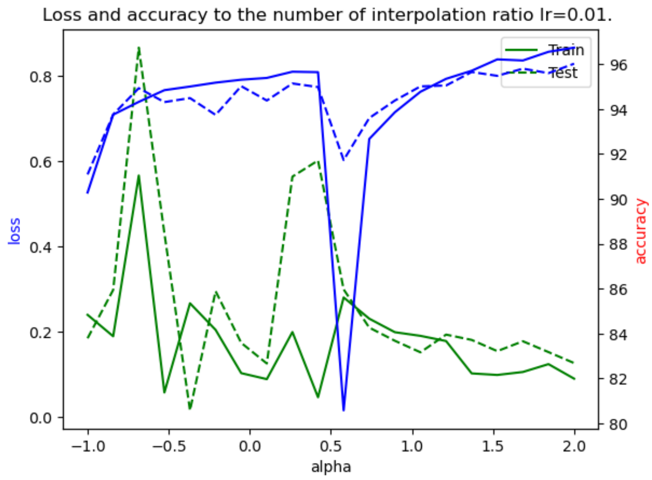


Fig-17

Part 2 :

Five identical Deep Neural Networks with two hidden layers and a total of 16630 parameters were generated. These networks were trained on batches ranging in size from 5 to 1000.

The network consists of three fully connected (fc) layers.

For each of the four lists, the values are appended in the order of the five models (mod1, mod2, mod3, mod4, mod5).

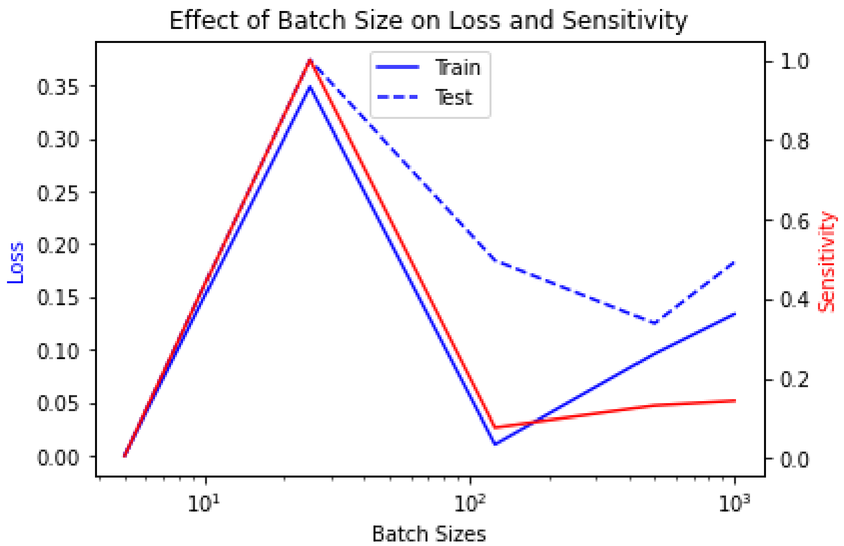


Fig-18

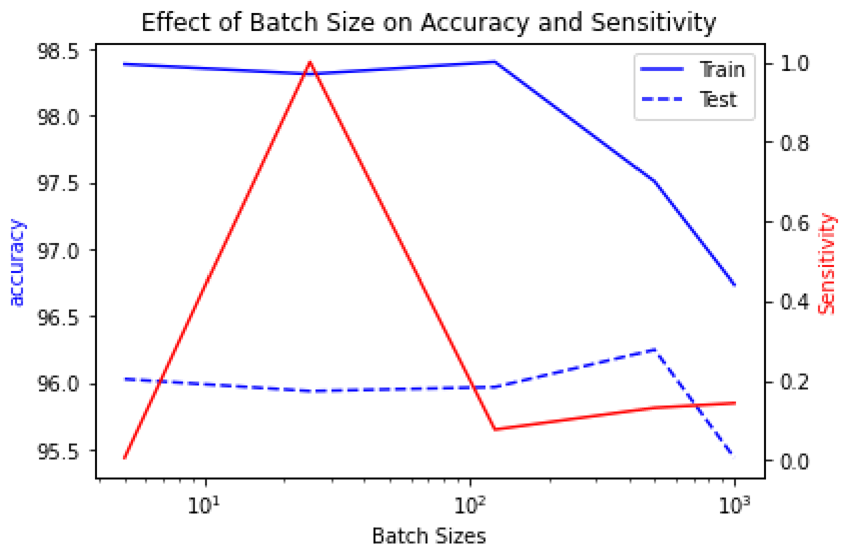


Fig19

After the training phase, the accuracy and loss of each of the five models for both the training and test datasets were computed. The sensitivity of the models was then determined by computing the gradients Frobenius norm.