Rayid Ali

Homework 1 Report

HW 1-1: Function Simulation

In this section, the architecture of three distinct neural network models is elaborated upon, each designed with varying complexity to simulate a specific mathematical function. The models incorporate a regularization technique through weight decay, enhancing their generalization by penalizing larger weights.

Model Overview:

Model A: Comprises seven fully connected layers with a total of 571 parameters, utilizing the RMSProp optimizer, MSELoss, and LeakyReLU activation. It features a learning rate of 0.001 and a weight decay of 0.0001.

Model B: Consists of four fully connected layers, holding 572 parameters. It shares the same optimizer, loss function, activation function, learning rate, and weight decay as Model A.

Model C: Simplified to a single dense layer, also with 571 parameters, and follows the same configuration for the optimizer, loss function, activation, learning rate, and weight decay as the previous models.

The models were tasked with approximating the function $sin(5 * pi * x)/5 * pi * x$ with the plot below showcasing the function's graphical representation:

Function Simulation Insights
Each model was trained to convergence, either by fulfilling the maximum epoch condition or exhibiting negligible learning progress. The subsequent analysis reveals Models A and B's rapid convergence in contrast to Model C, which required the full epoch allotment to approximate convergence. This divergence in performance underscores the impact of architectural depth on learning efficacy, with Models A and B's additional layers facilitating more adept function learning.

HW 1-1: Real-world Task Training on MNIST

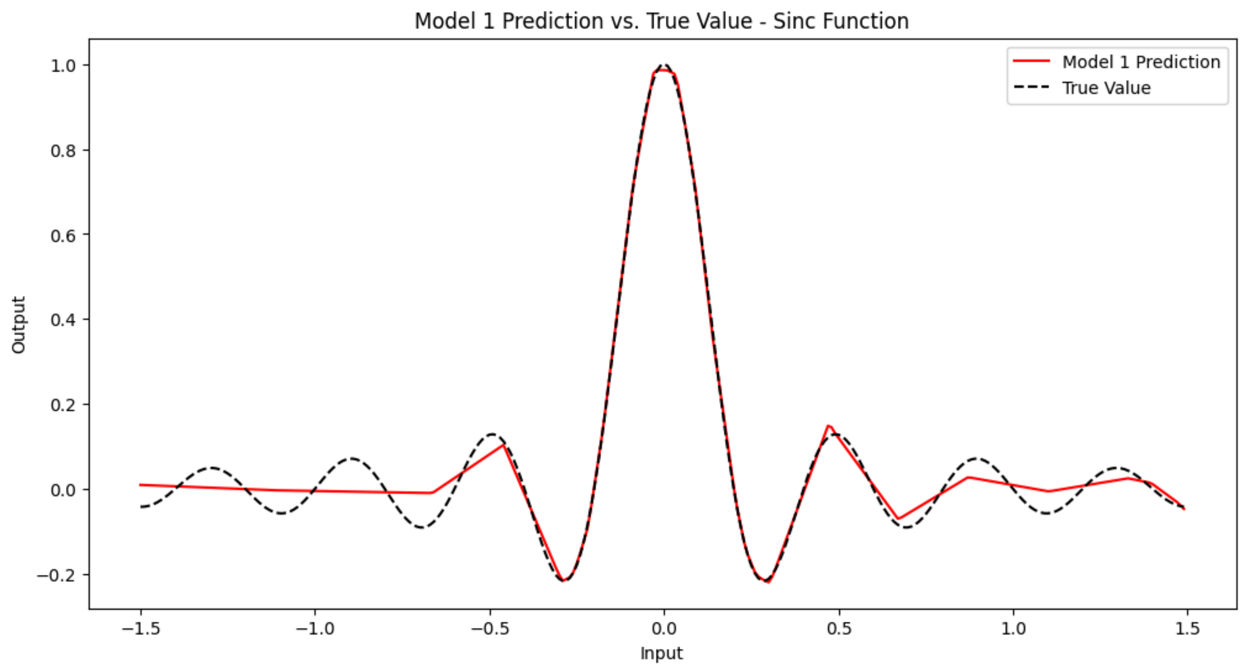
Transitioning to practical applications, the subsequent models are trained on the MNIST dataset, comprising 60,000 training and 10,000 testing images:

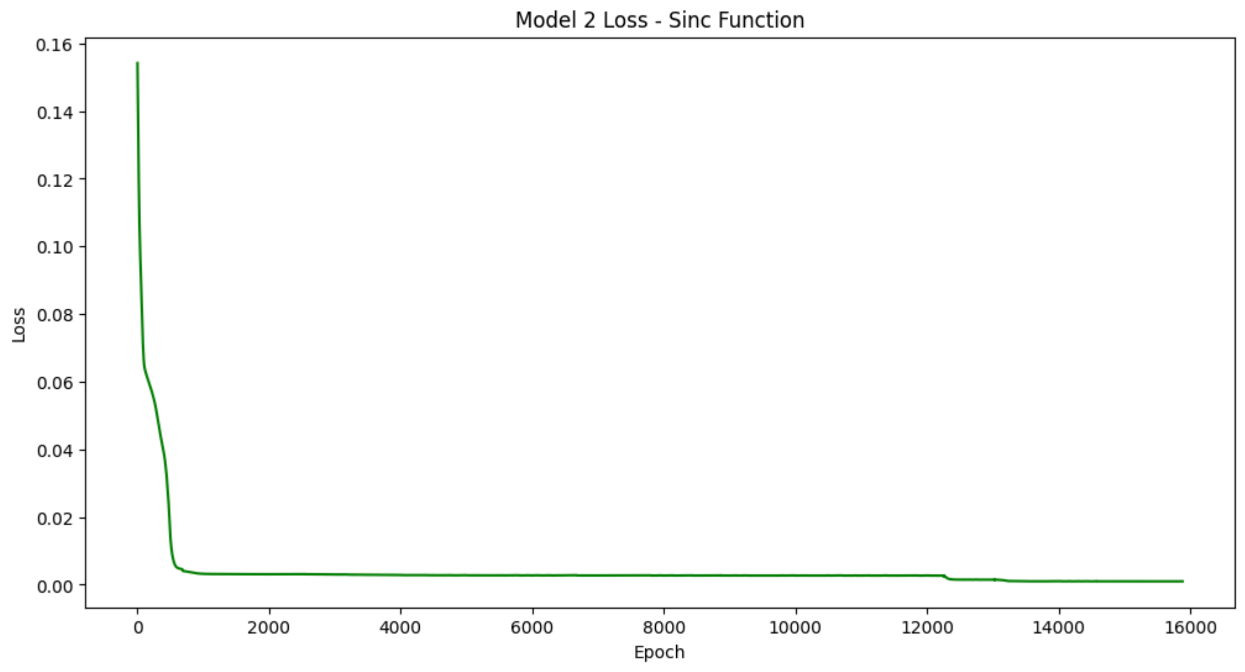
Model 1 (CNN akin to LeNet): Integrates two convolutional layers each followed by ReLU activation and max pooling, concluding with two dense ReLU layers.

Model 2 (Custom CNN): Employs three convolutional layers, where the latter two are succeeded by max pooling, ReLU activation, and dropout. The architecture finalizes with two dense layers, the last employing log_softmax.

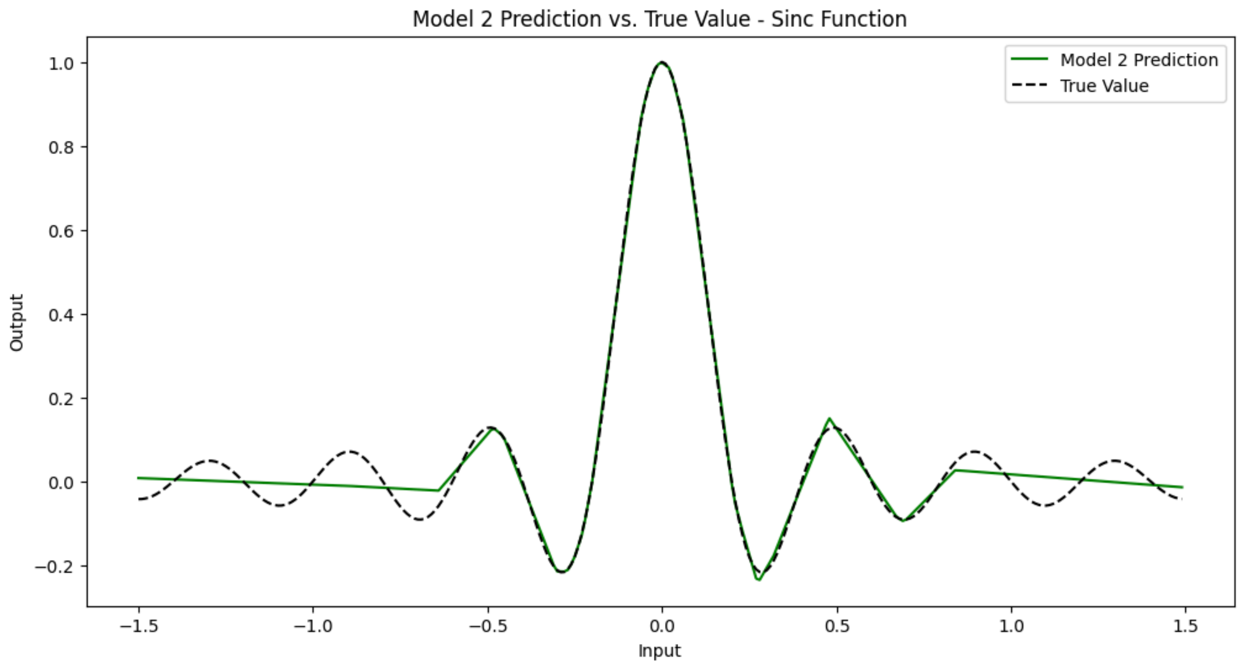
Rayid Ali

Training these models under the parameters of a 0.01 learning rate, 0.5 momentum, and cross-entropy loss for 10 epochs, we observe that Model 1, adhering to the LeNet blueprint, surpasses the custom CNN (Model 2) in learning efficiency, as evidenced by lower loss metrics and superior accuracy.

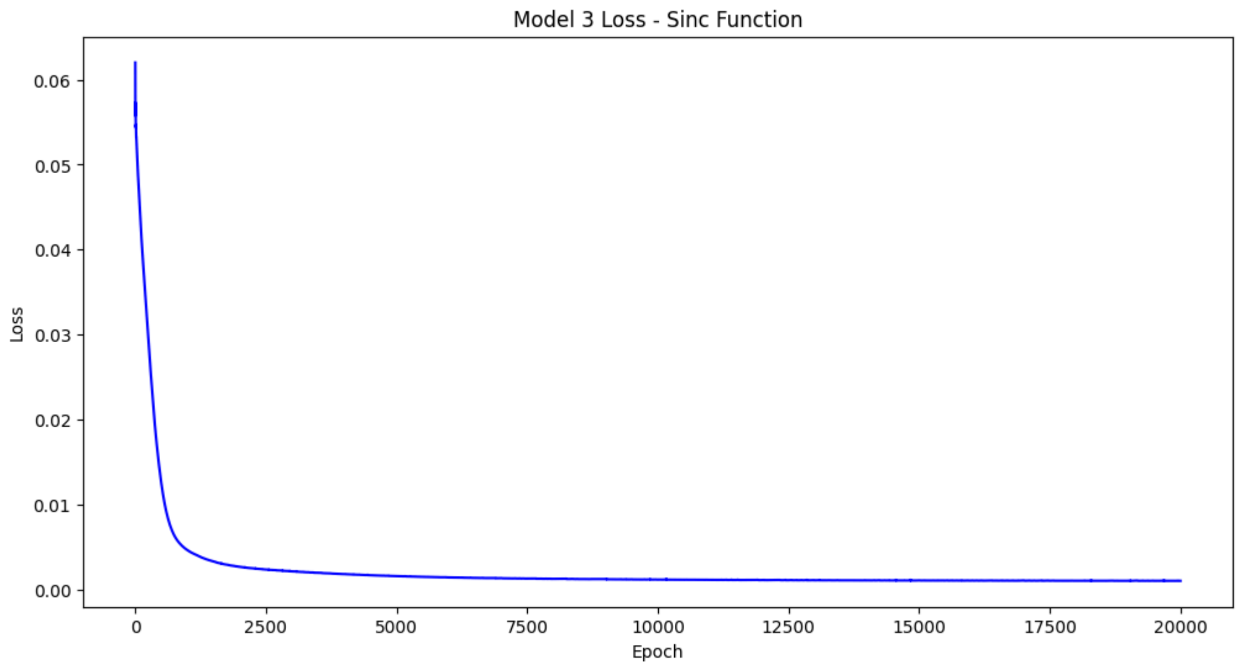




Rayid Ali

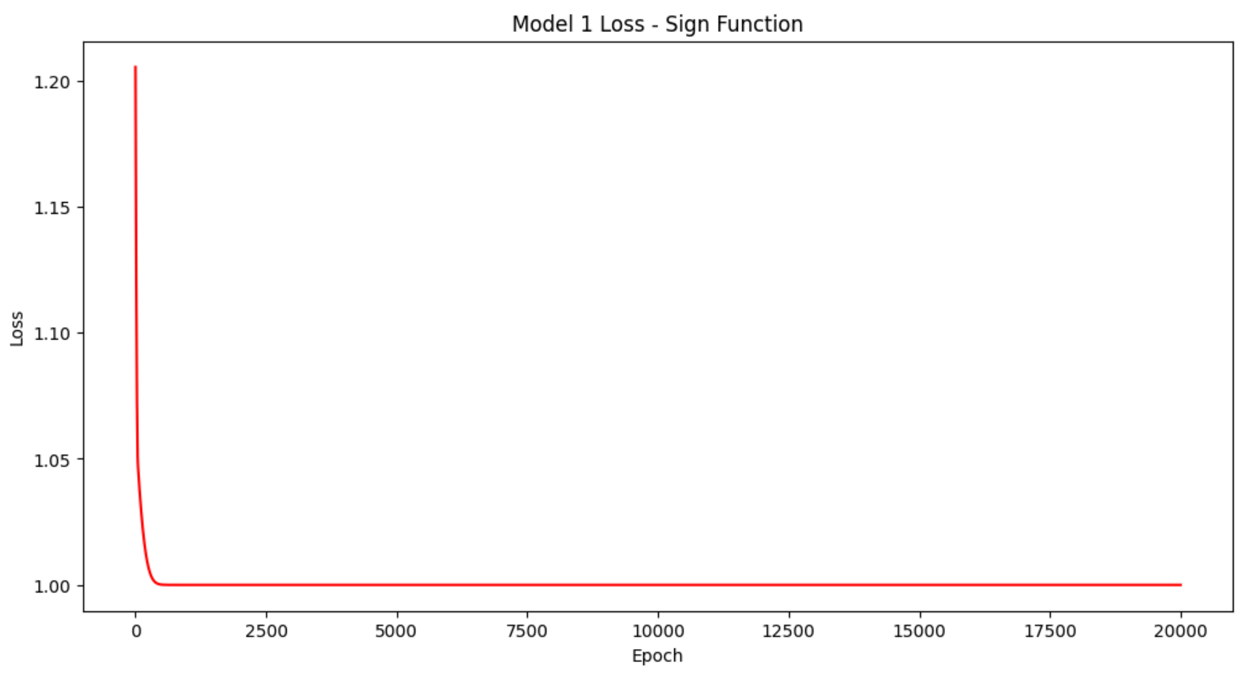


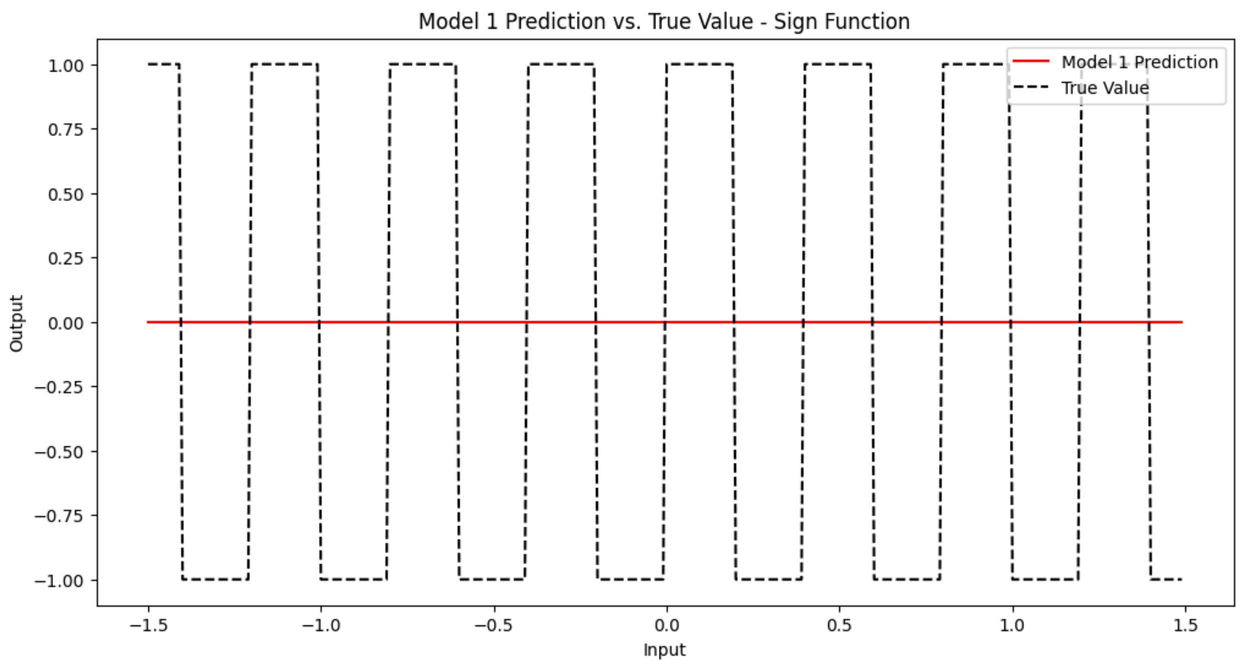
Model 2 Prediction vs. True Value - Sinc Function
Model 2 Prediction
True Value
Output
Input
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5



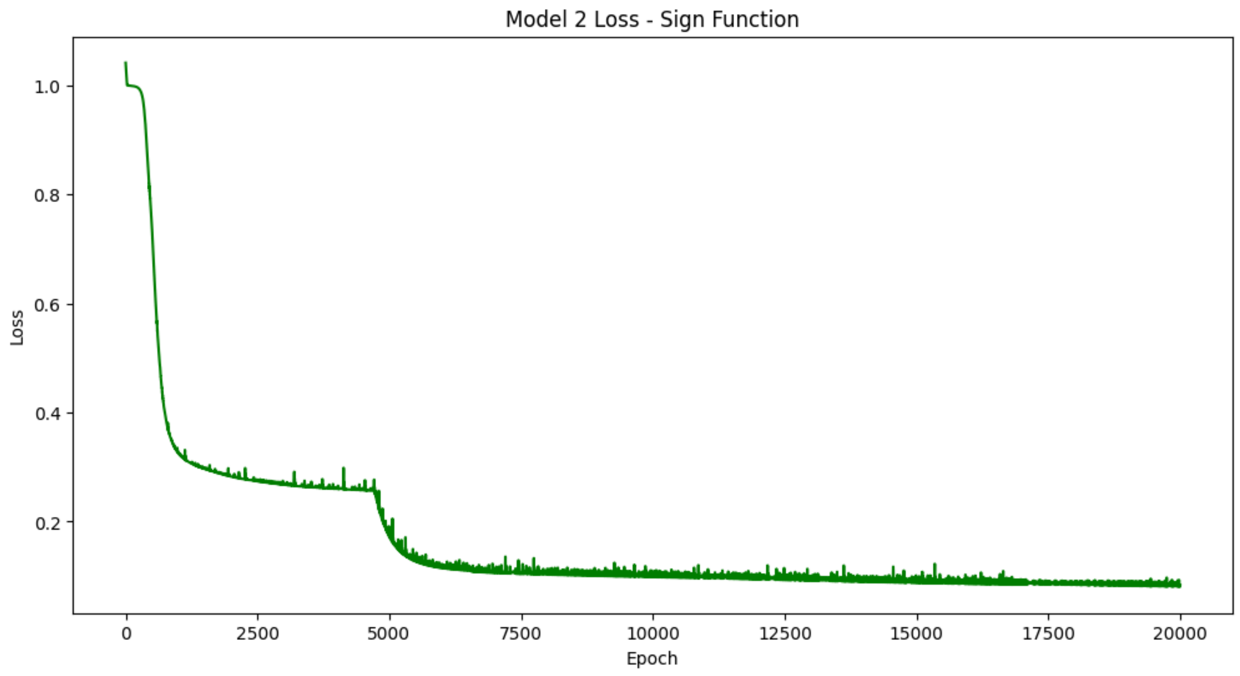
Model 3 Loss - Sinc Function
Loss
Epoch
0.06
0.05
0.04
0.03
0.02
0.01
0.00
0
2500
5000
7500
10000
12500
15000
17500
20000

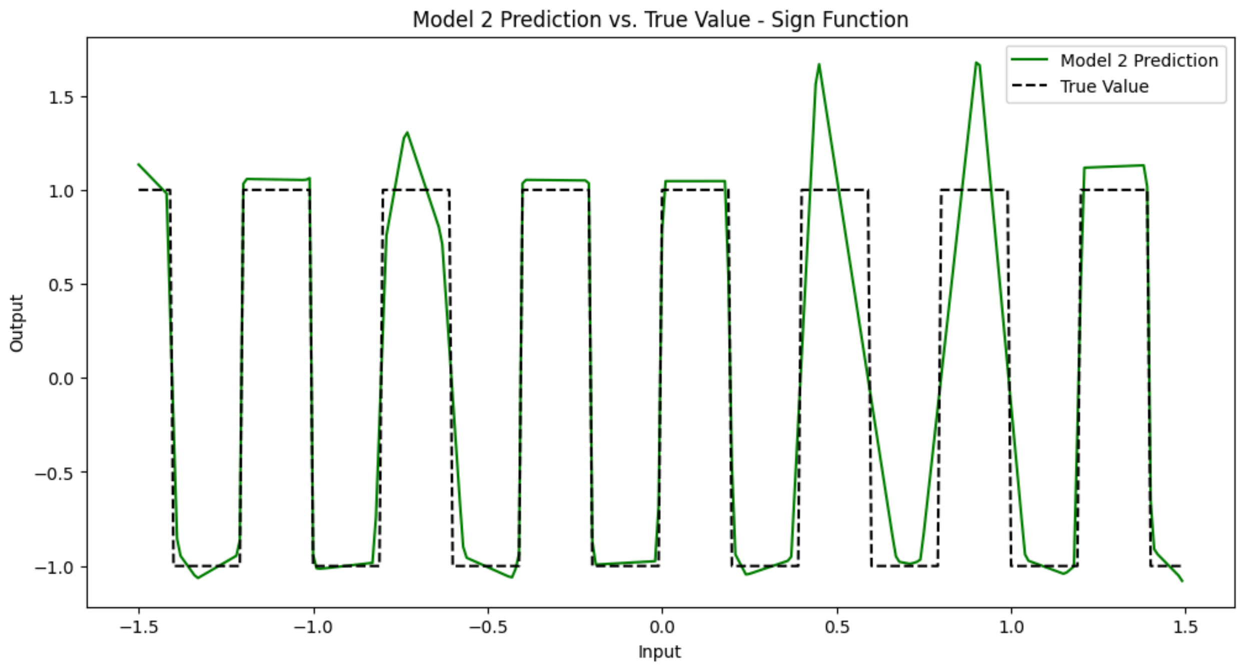
Rayid Ali


Model 1 Loss - Sign Function
Loss
1.20
1.15
1.10
1.05
1.00
0
2500
5000
7500
10000
12500
15000
17500
20000
Epoch

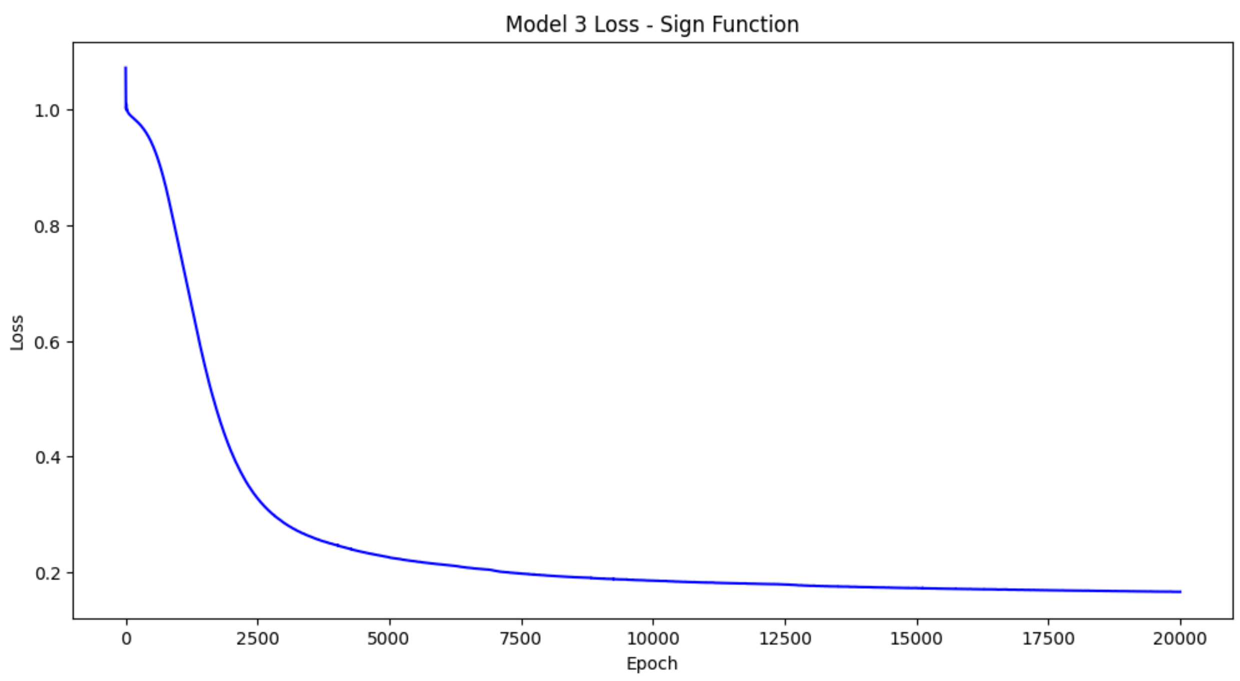

Model 1 Prediction vs. True Value - Sign Function
Model 1 Prediction
True Value
Output
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
Input

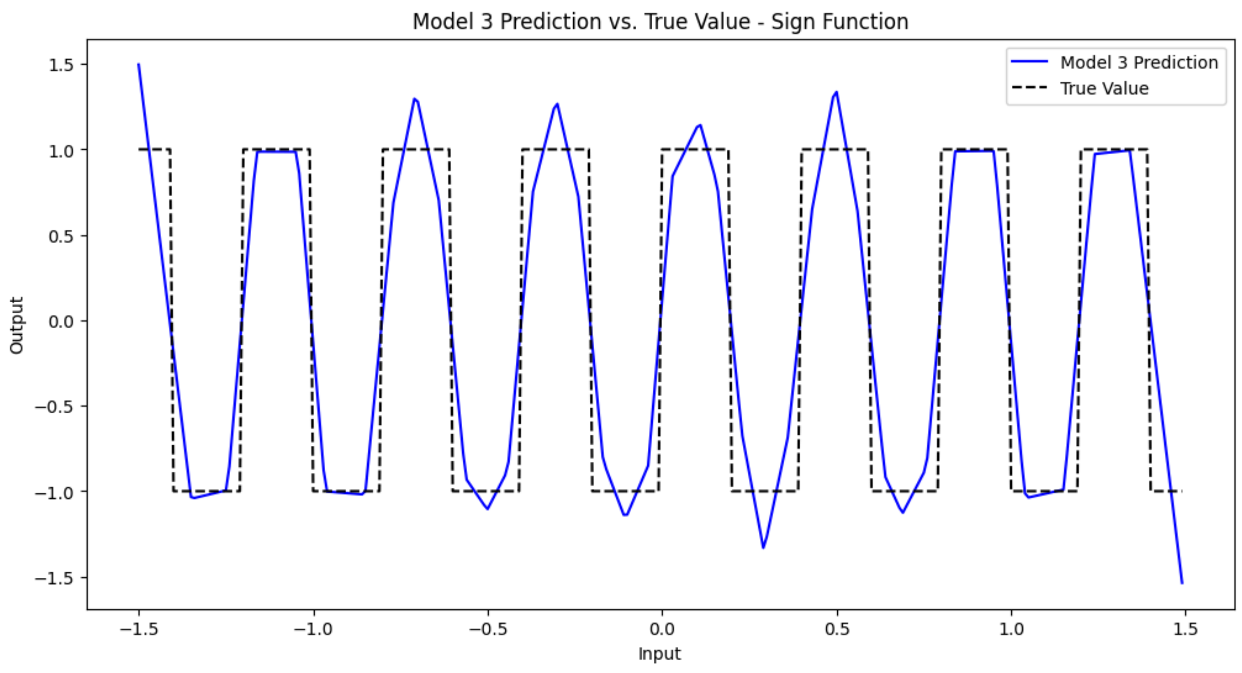
Rayid Ali


Model 2 Loss - Sign Function
Loss
Epoch
1.0
0.8
0.6
0.4
0.2
0
2500
5000
7500
10000
12500
15000
17500
20000

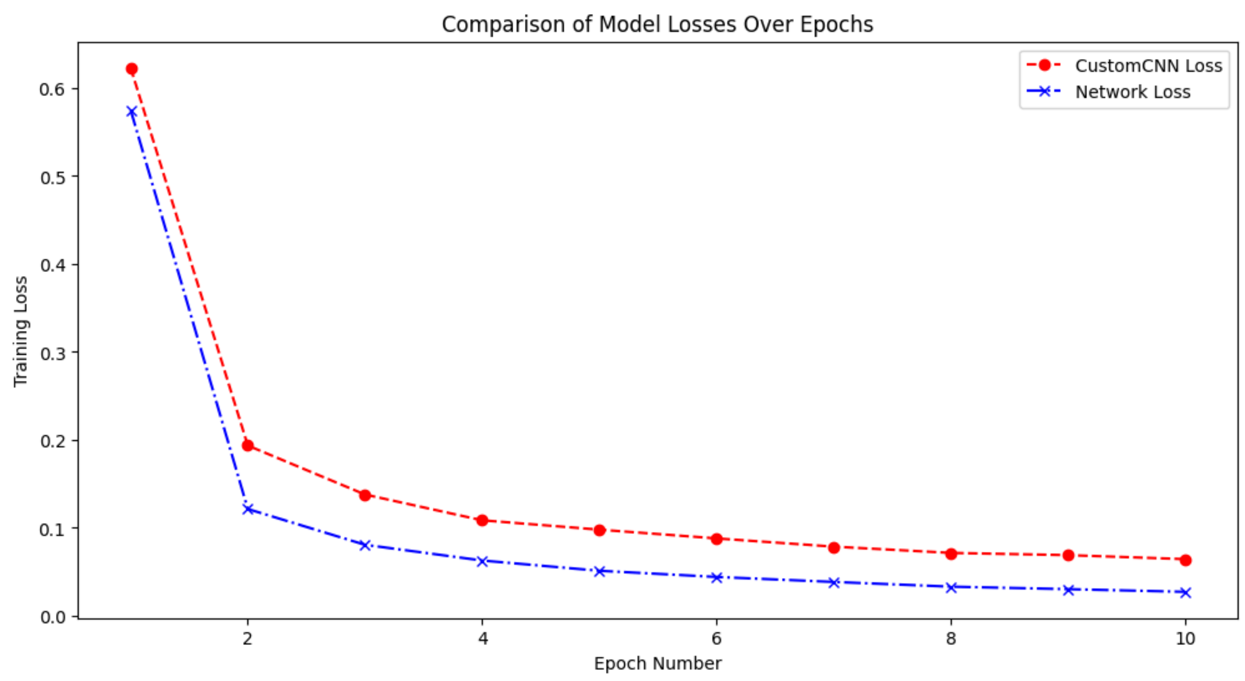

Model 2 Prediction vs. True Value - Sign Function
Model 2 Prediction
True Value
Output
Input
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5

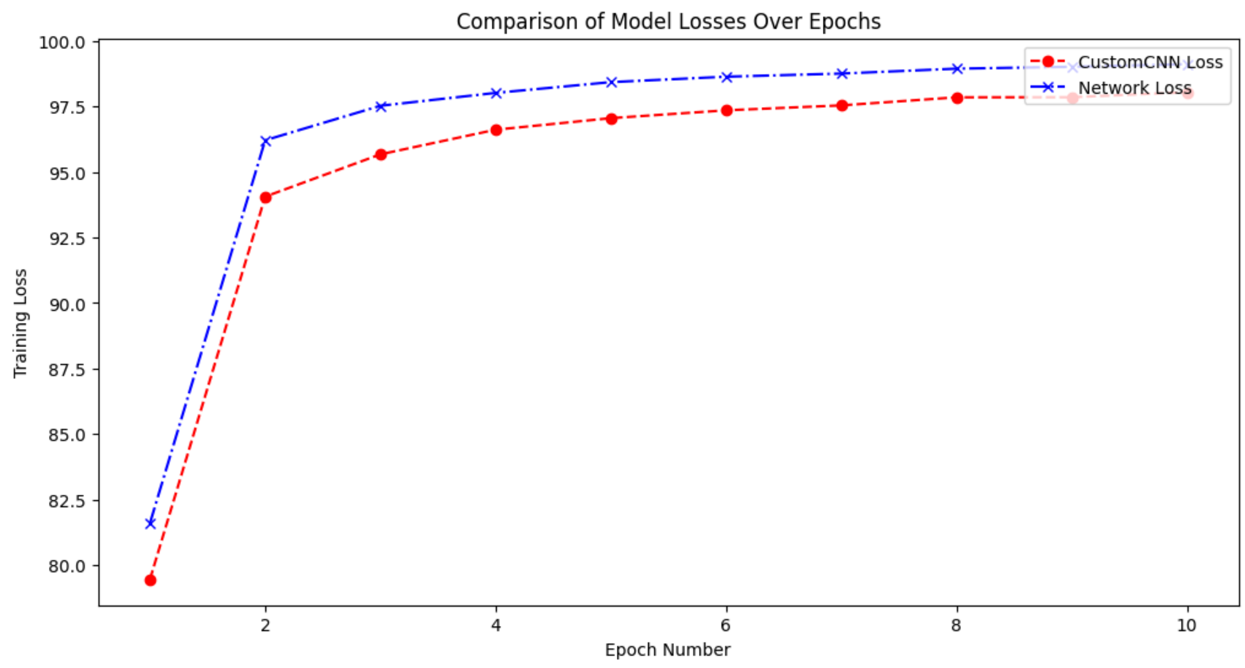
Rayid Ali


Model 3 Loss - Sign Function
Loss
Epoch
1.0
0.8
0.6
0.4
0.2
0
2500
5000
7500
10000
12500
15000
17500
20000


Model 3 Prediction vs. True Value - Sign Function
Model 3 Prediction
True Value
Output
Input
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5

Rayid Ali

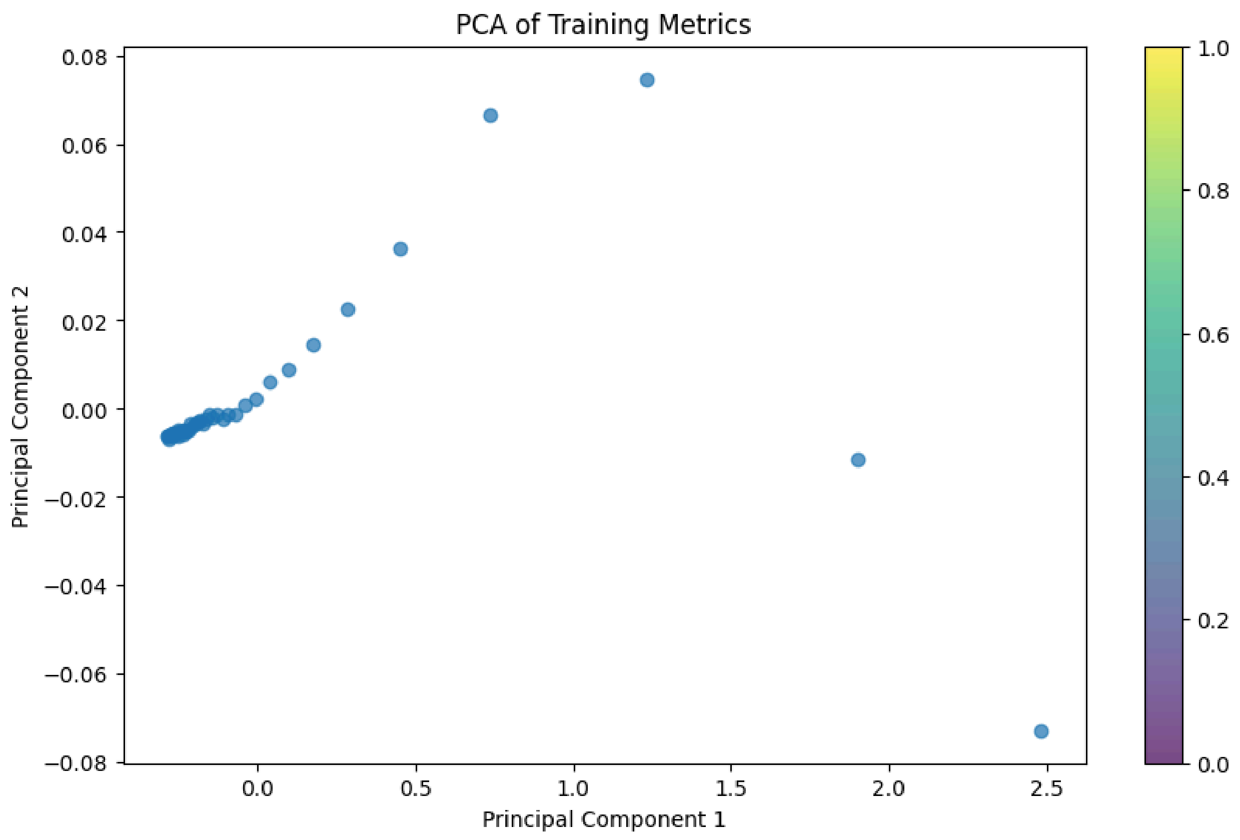




HW 1-2: Optimization Visualization via PCA

This experiment involves training a model on MNIST, capturing weights every three epochs across eight training cycles, and employing PCA for dimensionality reduction. The models, equipped with three dense layers, were optimized with Adam and trained with a 0.0004 learning rate, batch size of 1000, and ReLU activations.

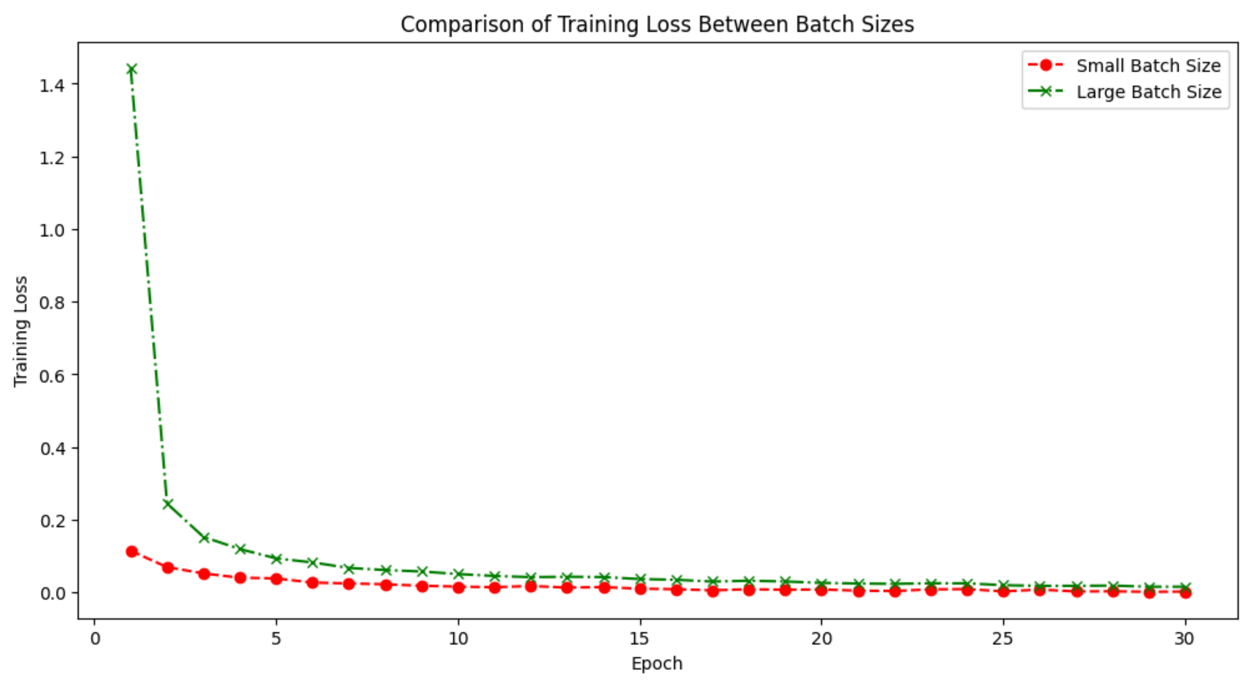
Rayid Ali

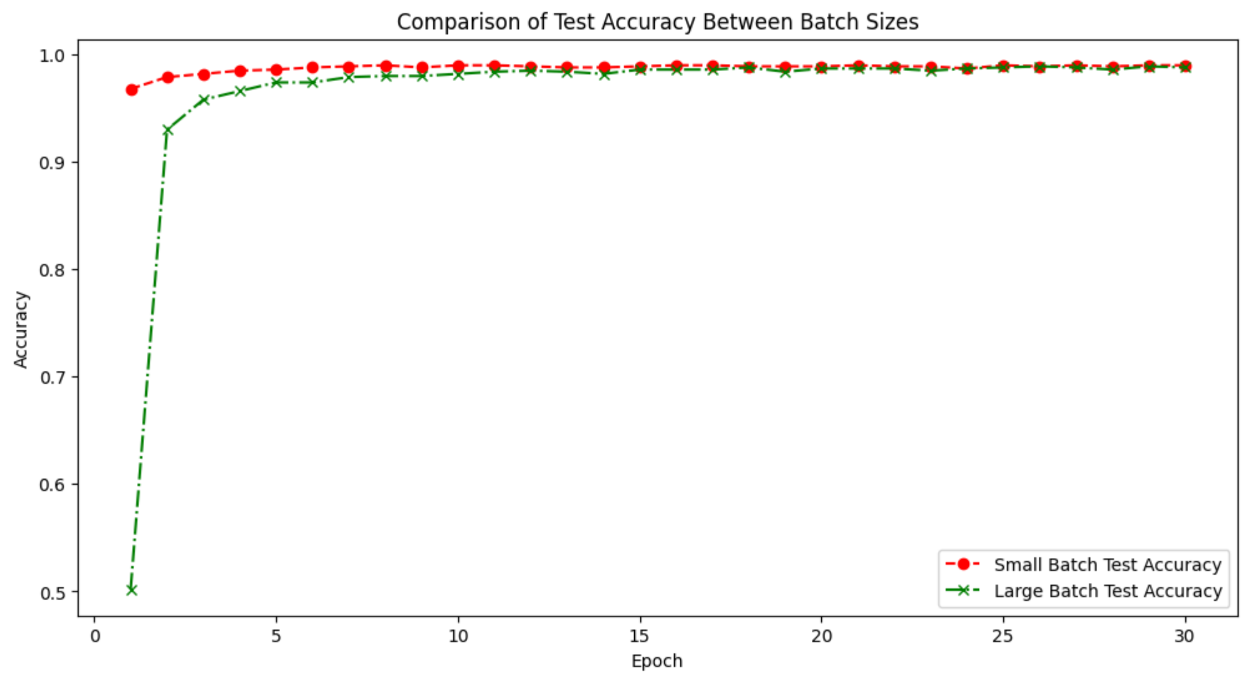


HW 1-3: Exploring Flatness and Generalization

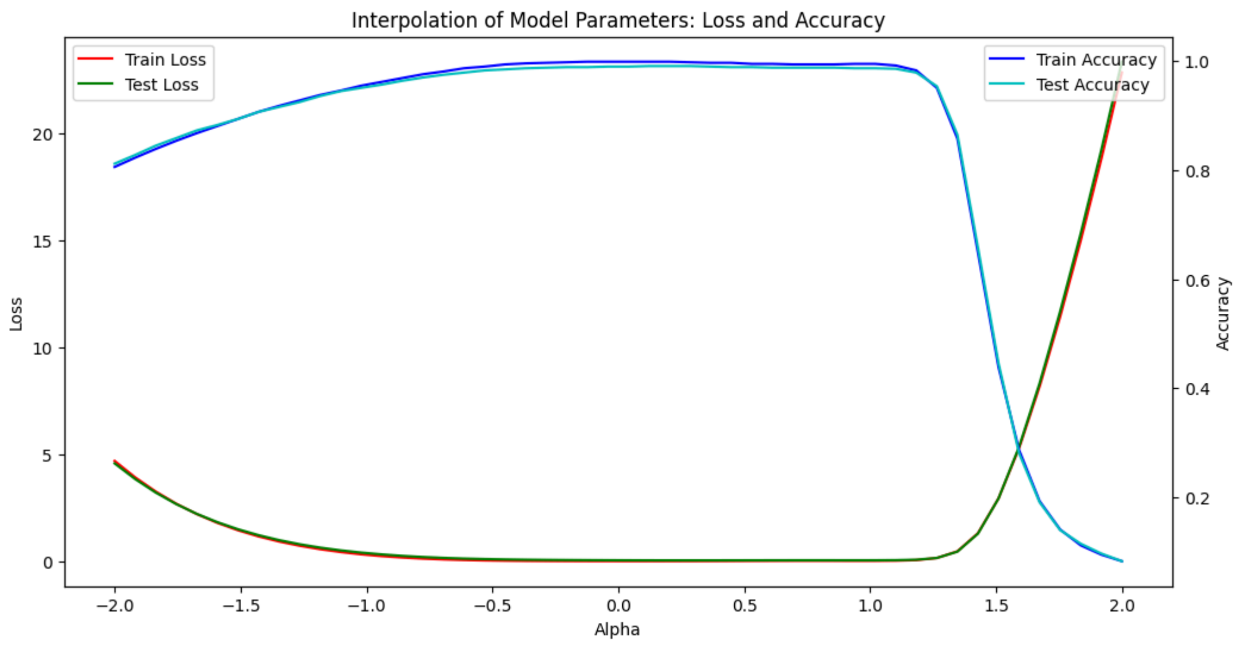
The investigation extends to models featuring three dense and two convolutional layers, examining their behavior under varying batch sizes and learning rates through weight interpolation. This analysis highlights a critical intersection at an alpha value of 1.5, where both test and train accuracies exhibit notable deviations, illustrating the nuanced balance between model complexity, data memorization, and generalization capabilities.

Rayid Ali

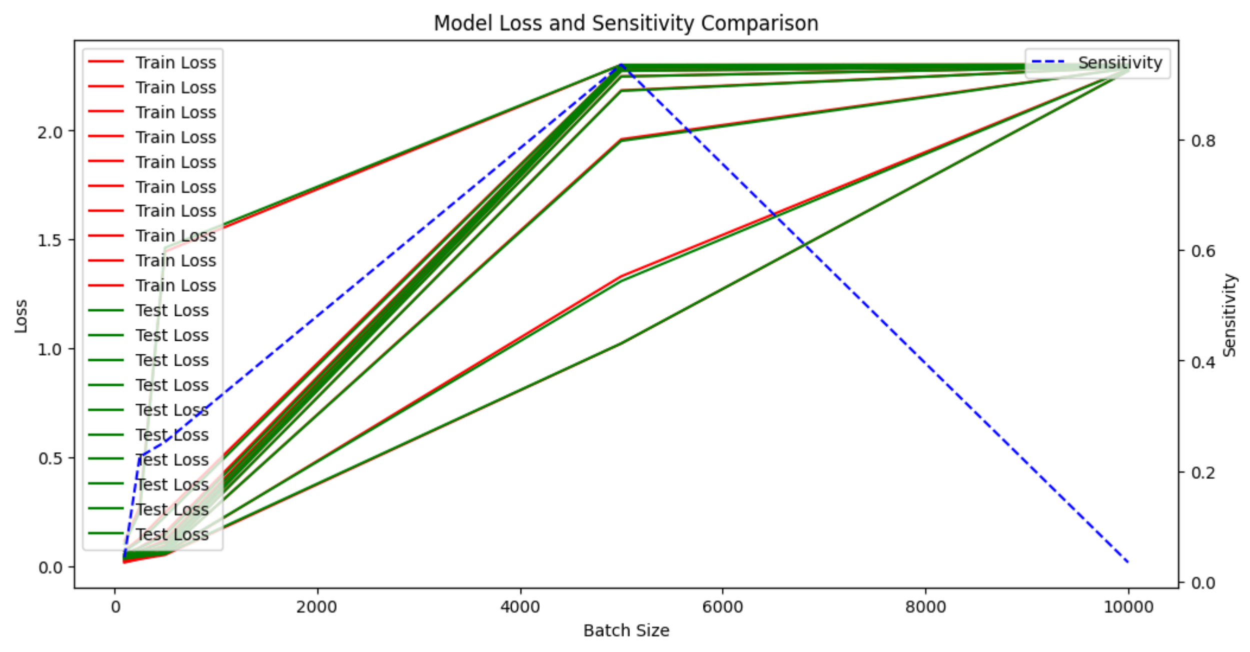

Comparison of Training Loss Between Batch Sizes
Small Batch Size
Large Batch Size
Training Loss
Epoch
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25
30


Comparison of Test Accuracy Between Batch Sizes
Small Batch Test Accuracy
Large Batch Test Accuracy
Accuracy
Epoch
1.0
0.9
0.8
0.7
0.6
0.5
0
5
10
15
20
25
30

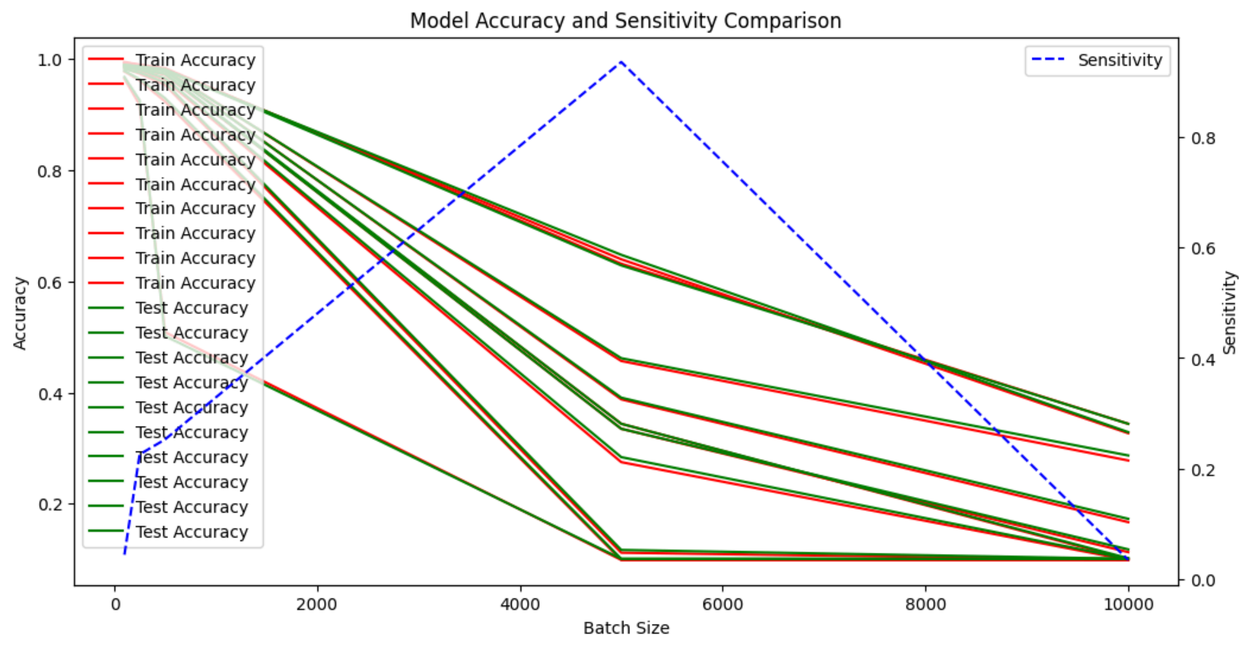
Rayid Ali


Interpolation of Model Parameters: Loss and Accuracy
Train Loss
Test Loss
Train Accuracy
Test Accuracy
Loss
Accuracy
Alpha
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
0
5
10
15
20
0.2
0.4
0.6
0.8
1.0

Rayid Ali



Model Loss and Sensitivity Comparison
Train Loss
Test Loss
Sensitivity
Loss
Sensitivity
Batch Size
0
2000
4000
6000
8000
10000
0.0
0.5
1.0
1.5
2.0
0.2
0.4
0.6
0.8



Model Accuracy and Sensitivity Comparison
Train Accuracy
Test Accuracy
Sensitivity
Accuracy
Sensitivity
Batch Size
0
2000
4000
6000
8000
10000
0.2
0.4
0.6
0.8
1.0
0.0