CIS 581 001 – COMPUTATIONAL LEARNING (WINTER-2023)

PROJECT - 2

Handwritten Digits Recognition using Neural Networks

NAME	:	SURYA SUBRAMANI
UMID	:	69499602
DATE OF SUBMISSION	:	18-04-2023

Contents

PREAMBLE

Report Objective

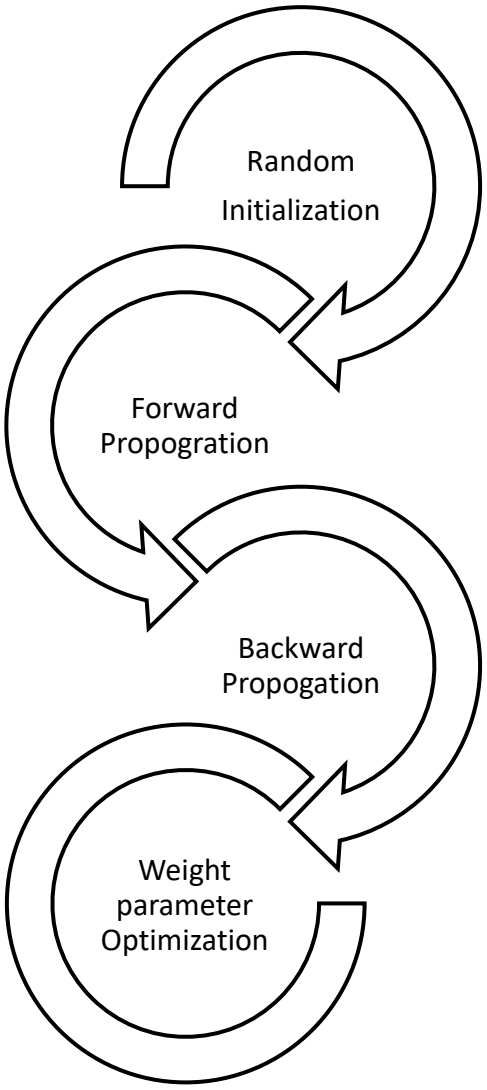
The main objective of this report is to understand how the neural networks works with different architectures like perceptron, Multi-Layer Perceptron and Deep Neural Network with fixed layers. The core concepts like forward and backward propagation, Gradient descent optimization have been experimented in this project.

Contribution and credits

This project's python script was adapted from Professor Luis Ortiz's python package "THE NEURAL NETWORK – BLACK BOX". His python script exposed to the actual programming part of neural networks and its architecture.

EXPERIMENTAL SETUP

1) Algorithm Flow



2) List of Architecture Considered

- A **(multi-output) perceptron** with 1024 input units and 10 output units, with (gradient descent) learning rate $\in \{4\ i\ |i = 0, 1, 2, 3, 4\}$
- A **deep neural network with the same number of units per hidden layer** $\in \{4\ 2\ ,\ 4\ 3\ ,\ 4\ 4\}$, network depth (i.e., number of hidden layers) $\in \{1, 2, 3, 4\}$, and (gradient descent) learning rate $\in \{4\ i\ |i = -2, -1, 0, 1, 2\}$
- **A deep neural network with 2 hidden layers, with the number of units in the first layer** $\in \{4\ 3\ ,\ 4\ 4\}$ (connecting from the input layer), followed by a smaller number of units $\in \{4\ 2\ ,\ 4\ 3\}$ in the next layer (connecting to the output layer), and (gradient descent) learning rate $\in \{4\ i\ |i = -3, -2, -1, 0, 1\}$

3) Objective

The main objective of the experiment to select the best neural network model based on the learning rate, number of hidden layers and its units. At the end of this experiment, we will be having three best neural networks with above mention architectures respectively.

4) Data Description

The data set of examples consists of images of handwritten digits represented as a 32x32 matrix of pixels, each pixel taking value 1 or 0 corresponding to a black or white pixel, respectively. For each image, each pixel corresponds to a binary attribute, with values 0 or 1, to be treated as real-valued. Therefore, for the original data set, each input consists of $32 \times 32 = 1024$ (binary) attributes.

- **optdigits train.dat** contains the training data.
- **optdigits test.dat** contains the test data.
- **optdigits trial.dat** contains an example of each digit from the test data set.

Each file is composed of one data example per line. Each line of the files but now each line contains 1025 integers separated by a single space. The first 1024 correspond to the values of a bitmap of the image, where each consecutive sequence of 32 values, starting with the value in the first column, corresponds to a row of black (0) and white (1) pixel values of the image of the handwritten digit. The value of the digit is given as the last element of the 1025 vector/row. A proper reshaping of the vector composed of the first 1024 binary values in the file-row will produce a 32x32 black-and-white image. The training, test, and trial data sets have 1934, 946 and 10 examples, respectively.

RESULTS

1) Gradient Descent Fitting Curves:

Using all the training data, for the best model class selected,

(a) Record

Perceptron Best Model Records

Learning Rate: 16 **Architecture:** 1024 input units and 10 output units

Iteration no	Train misclassification error	Test misclassification error	Train proxy error	Train proxy error
1	0.8971	0.9101	0.0486	0.0489
2	0.8573	0.8742	0.0852	0.0868
3	0.6877	0.6882	0.0409	0.0419
4	0.8971	0.9101	0.0486	0.0489
.	.	.	.	.
.	.	.	.	.
998	0	0.0391	0	0.0055
999	0	0.0391	0	0.0055
1000	**0**	**0.0391**	**0**	**0.0055**

Click this link for the records 

Deep Neural Network Best Model Records

Learning Rate: 16

Architecture: 1024 input units and 10 output units, 1 hidden layers with 64 units and alpha = 16

Iteration no	Train misclassification error	Test misclassification error	Train proxy error	Train proxy error
1	0.9069	0.9038	0.0684	0.0684
2	0.8992	0.9080	0.0764	0.0776
3	0.7539	0.7209	0.0946	0.0922
4	0.9069	0.9038	0.0684	0.0684
.	.	.	.	.
.	.	.	.	.
998	0.0057	0.0402	0.0010	0.0037
999	0.0057	0.0402	0.0010	0.0038
1000	**0.0057**	**0.0402**	**0.0010**	**0.0038**

Click this link for the records 

Deep Neural Network with Same Number of Layers but Different Units Best Model Records

Learning Rate: 16

Architecture: 1024 input units and 10 output units, layer-1 with 256 units and layer 2 with 64 units and learning rate 16

Iteration no	Train misclassification error	Test misclassification error	Train proxy error	Train proxy error
1	0.8992	0.9027	0.0946	0.0947
2	0.8795	0.8901	0.0493	0.0494
3	0.8454	0.8689	0.0682	0.0694
4	0.8992	0.9027	0.0946	0.0947
.	.	.	.	.
.	.	.	.	.
242	0.0233	0.0455	0.0028	0.0042
243	0.0233	0.0455	0.0028	0.0042
244	**0.0233**	**0.0444**	**0.0028**	**0.0042**

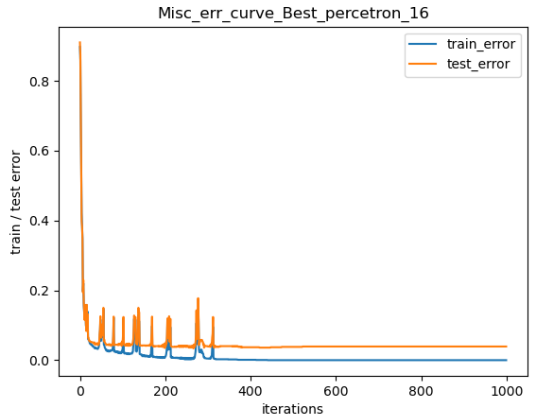
Click this link for the records best_NN1_[[1024, 256, 64, 10], 16].csv

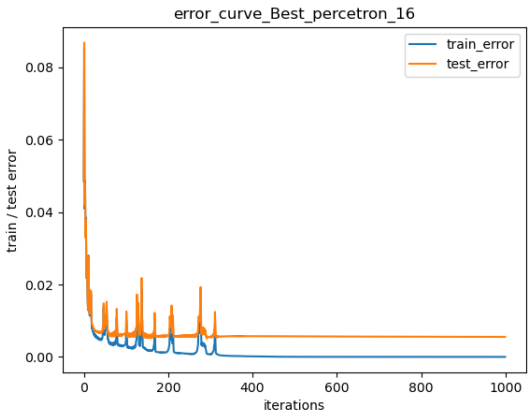
(b) Curves

Perceptron Best Model Records

Learning Rate: 16

Architecture: 1024 input units and 10 output units

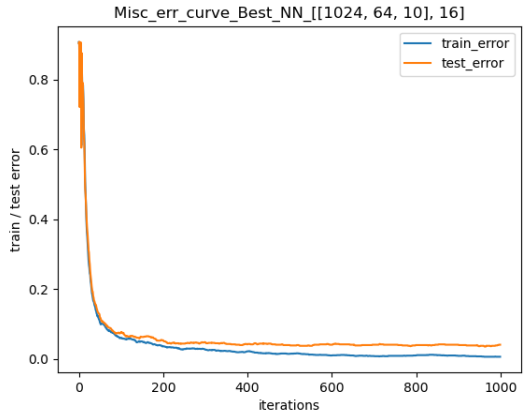


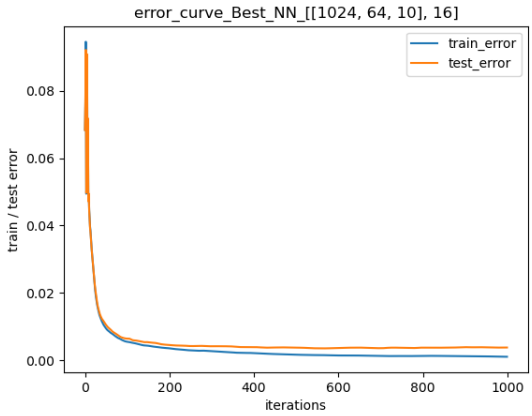


Deep Neural Network Best Model Records

Learning Rate: 16

Architecture: 1024 input units and 10 output units, 1 hidden layers with 64 units and alpha = 16

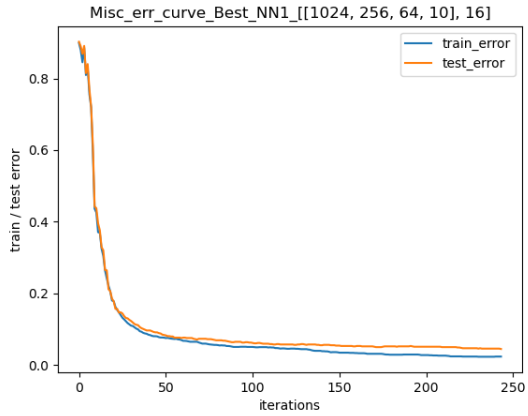


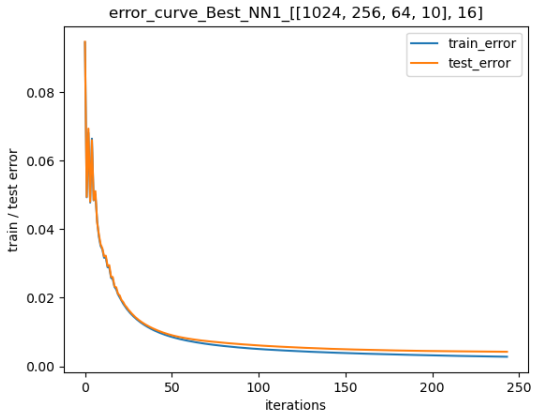


Deep Neural Network with Same Number of Layers Best Model Records

Learning Rate: 16

Architecture: 1024 input units and 10 output units, layer-1 with 256 units and layer 2 with 64 units and learning rate 16





(c) Evaluate each example in the trial data set on the best/final neural network learned and report the corresponding output classification.

Model	Correctly Classified/946	Misclassification error
Perceptron	908	0.04016
Neural network with multiple layers	912	0.03594
Neural network with two converging layers	909	0.03911

Click this link for the records

1. Handwritten images that are misclassified by Model 1 - Perceptron

Index	y_trial	Predicted_classification
45	9	7
47	8	9
134	8	6
144	3	9
154	1	9
166	9	1
175	9	1
224	3	5
298	1	2
399	4	0
426	8	1
450	5	8
451	3	9
468	3	2
476	3	8
544	4	1
554	9	0
610	3	9
635	6	8
637	2	8
643	9	8
651	2	7
664	2	8
691	4	8
700	8	5
708	9	5
722	2	8
725	2	8
729	7	4
732	4	8
741	5	9
756	9	2

793	3	9
825	9	3
843	4	9
846	1	7
909	6	4
926	2	9

2. Handwritten images that are misclassified by Model 2 – Neural network with multilayer

Index	y_trial	Predicted_classification
7	4	9
27	7	9
81	2	3
134	8	6
144	3	9
154	1	9
166	9	3
175	9	1
315	4	1
349	8	9
399	4	0
427	3	2
440	5	1
450	5	6
451	3	9
463	5	6
468	3	2
469	7	9
471	3	5
492	3	8
505	7	9
544	4	1
643	9	8
708	9	5
722	2	8
729	7	4
732	4	8
825	9	3
843	4	9
846	1	7
885	8	9
909	6	4
926	2	9
944	1	4

3. Handwritten images that are misclassified by Model 2 – Neural network with Fixed number of Hidden Layer

Index	TRUE	classified_mod3
30	9	4
32	5	9
45	9	7
134	8	6
144	3	9
154	1	9
166	9	2
175	9	1
209	8	2
259	1	9
349	8	1
399	4	0
451	3	9
471	3	5
476	3	8
554	9	0
610	3	9
637	2	8
643	9	8
651	2	8
664	2	8
691	4	8
708	9	5
722	2	8
725	2	8
729	7	4
732	4	8
756	9	2
793	3	9
825	9	7
843	4	8
846	1	7
860	2	6
861	7	1
863	1	7
909	6	4
926	2	9

y_trial	classified_perc	classified_mod2	classified_mod3
8	6	6	6
3	9	9	9
1	9	9	9
9	1	3	2
9	1	1	1
4	0	0	0
3	9	9	9
9	8	8	8
9	5	5	5
2	8	8	8
7	4	4	4
4	8	8	8
9	3	3	7
4	9	9	8
1	7	7	7
6	4	4	4
2	9	9	9

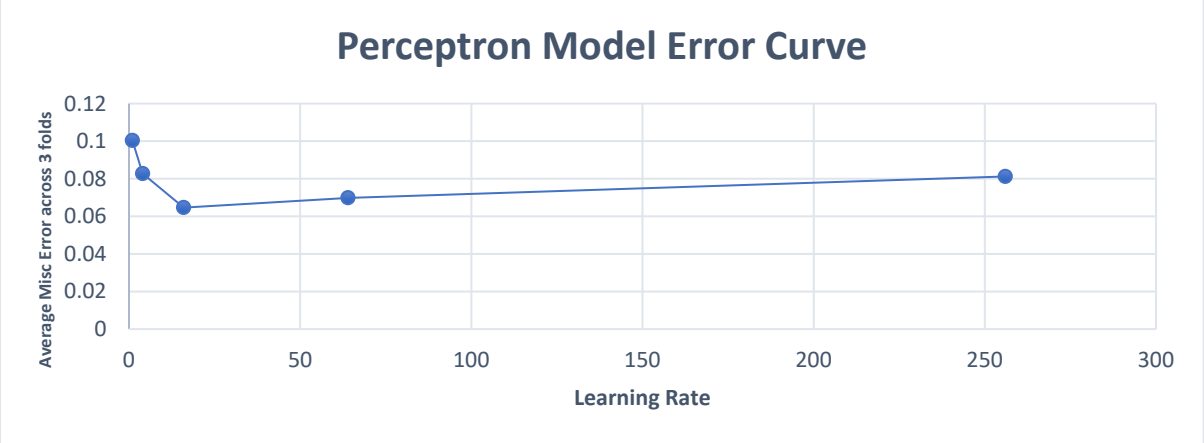
The above figure tabulates the index of the image along with its label (y_trial) in which all models are misclassified them.

2) CV Error Curves:

Google drive link for individual CV curves and its data

I- Model – 1

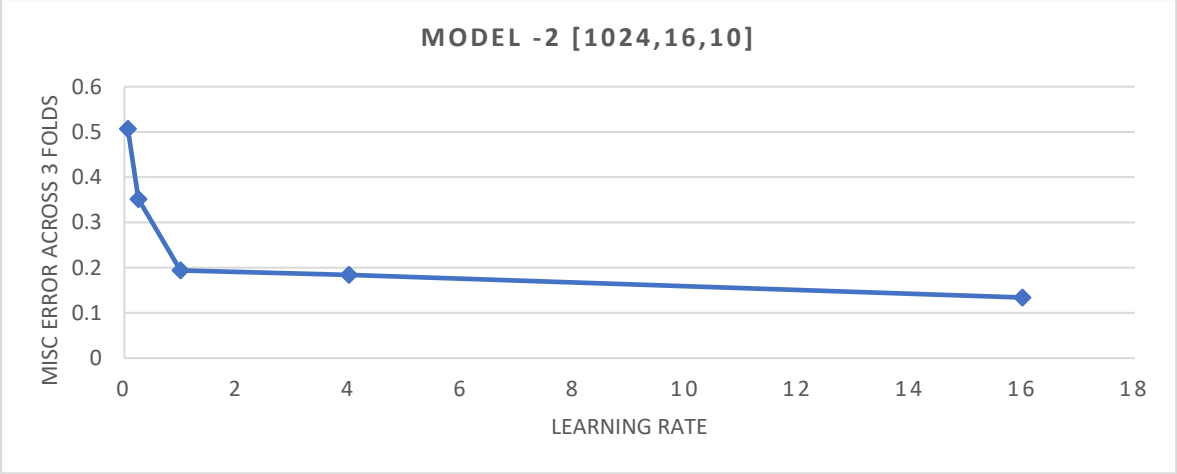
Learning rate	misc_test	misc_train	RMSE_test	RMSE_train
1	0.100317	0.007757	0.011609	0.001672
4	0.082735	0	0.010048	0.000114
16	0.06464	0	0.008854	0.000003
64	0.069813	0	0.009887	0
256	0.08119	0	0.010522	0

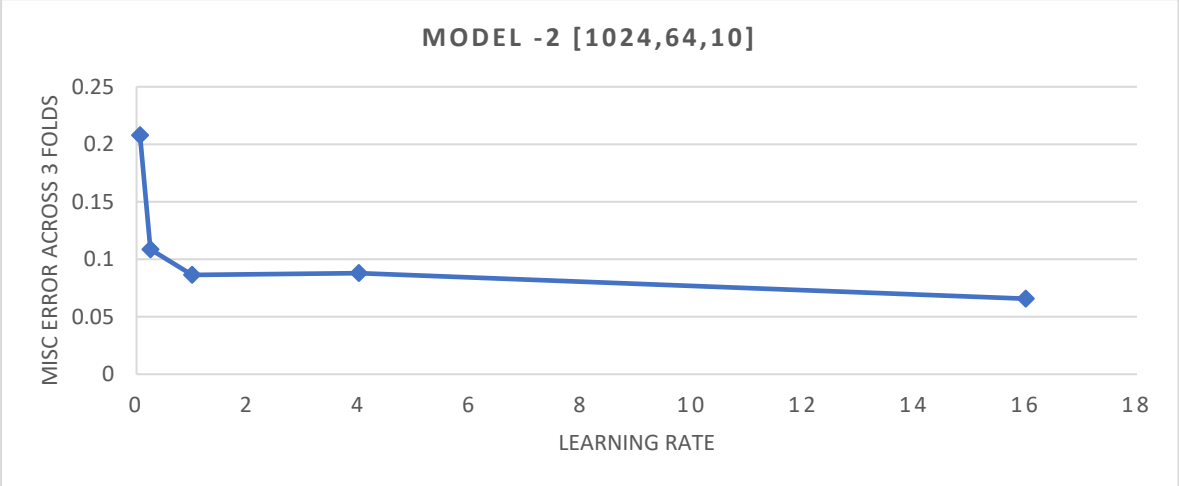


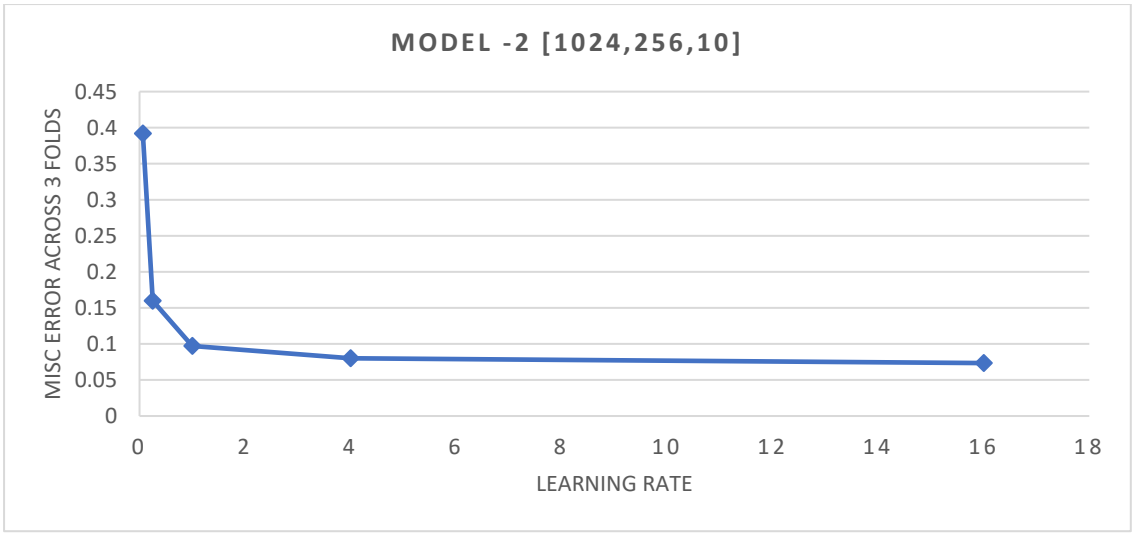
II- Model – 2

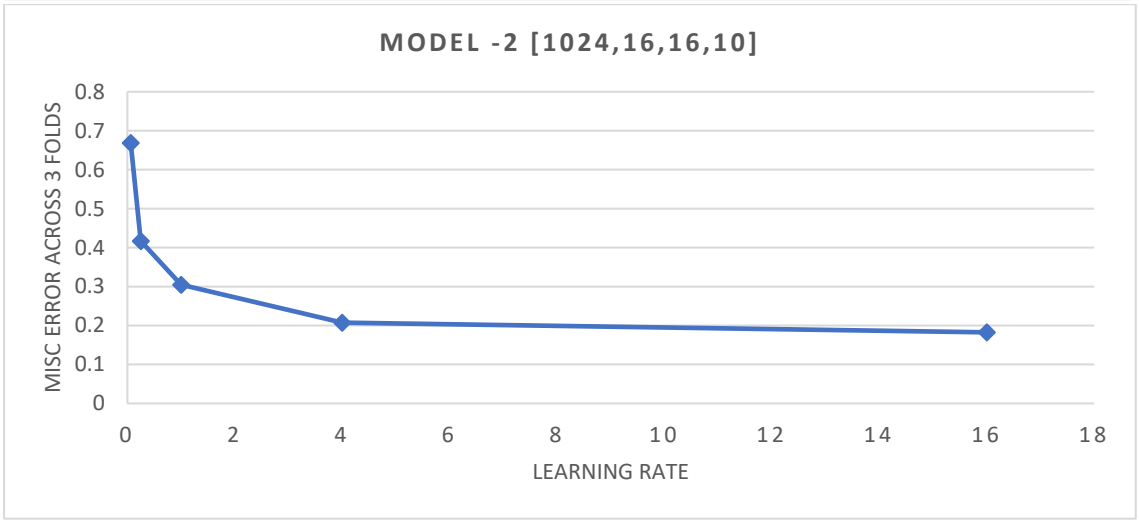
Number of layers	Layer units	Learning rate	misc_test	misc_train	RMSE_test	RMSE_train
1	16	0.0625	0.506766	0.458373	0.037605	0.036715
1	16	0.25	0.351623	0.289547	0.028289	0.026593
1	16	1	0.193899	0.137805	0.016657	0.013562
1	16	4	0.184074	0.107298	0.013209	0.008839
1	16	16	0.133937	0.047565	0.010205	0.004094
1	64	0.0625	0.207867	0.173477	0.020524	0.018928
1	64	0.25	0.108592	0.067479	0.012431	0.009252
1	64	1	0.086358	0.034904	0.008329	0.004309
1	64	4	0.087915	0.016286	0.007173	0.00223
1	64	16	0.065678	0	0.00595	0.000227
1	256	0.0625	0.391947	0.335838	0.031552	0.027933
1	256	0.25	0.159789	0.100053	0.015856	0.011297
1	256	1	0.09721	0.027147	0.009988	0.004109
1	256	4	0.08015	0.006981	0.007233	0.001703
1	256	16	0.073436	0.004137	0.00613	0.00093
2	16	0.0625	0.668543	0.640911	0.042439	0.042174
2	16	0.25	0.416764	0.379267	0.033939	0.033228
2	16	1	0.304573	0.25672	0.022929	0.020771
2	16	4	0.207357	0.154597	0.015478	0.011935
2	16	16	0.182516	0.102124	0.013131	0.008006
2	64	0.0625	0.252853	0.178909	0.023938	0.02137
2	64	0.25	0.116865	0.073681	0.012923	0.009964
2	64	1	0.097225	0.034124	0.008478	0.004551
2	64	4	0.094623	0.009049	0.007457	0.001647
2	64	16	0.102382	0.009826	0.008452	0.001049
2	256	0.0625	0.313375	0.284644	0.025785	0.023612
2	256	0.25	0.14272	0.078595	0.013639	0.009345
2	256	1	0.094104	0.020426	0.009793	0.003402
2	256	4	0.079633	0.013186	0.006556	0.001961
2	256	16	0.164436	0.102888	0.012886	0.008685
3	16	0.0625	0.728577	0.69388	0.044172	0.04416
3	16	0.25	0.548134	0.518076	0.03956	0.039223
3	16	1	0.364015	0.315666	0.026081	0.023771
3	16	4	0.251275	0.187959	0.018083	0.014056
3	16	16	0.290121	0.237045	0.021177	0.016773
3	64	0.0625	0.319562	0.26603	0.030818	0.02924
3	64	0.25	0.126692	0.086606	0.013316	0.011137
3	64	1	0.093584	0.037491	0.008084	0.004016
3	64	4	0.085844	0.004137	0.006876	0.000833
3	64	16	0.231111	0.155905	0.017645	0.011593
3	256	0.0625	0.281289	0.248713	0.024017	0.021552
3	256	0.25	0.136517	0.073682	0.013166	0.008931
3	256	1	0.084803	0.022236	0.00779	0.00304

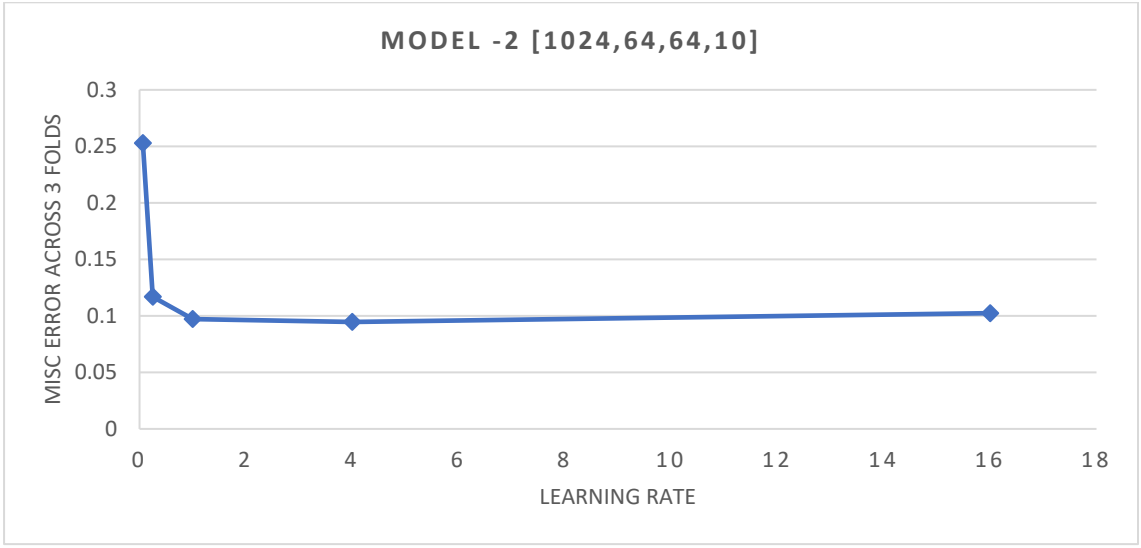
3	256	4	0.097227	0.025076	0.007881	0.002777
3	256	16	0.724265	0.697595	0.039147	0.037904
4	16	0.0625	0.845393	0.837903	0.044835	0.044769
4	16	0.25	0.740478	0.718704	0.04382	0.043743
4	16	1	0.510801	0.466167	0.032678	0.031173
4	16	4	0.35264	0.338928	0.02556	0.024365
4	16	16	0.583764	0.530241	0.034416	0.032656
4	64	0.0625	0.384687	0.35575	0.034037	0.033529
4	64	0.25	0.170125	0.110908	0.015815	0.012805
4	64	1	0.106509	0.036455	0.00855	0.003887
4	64	4	0.104452	0.012669	0.008103	0.001125
4	64	16	0.692877	0.683292	0.039986	0.038746
4	256	0.0625	0.280253	0.223373	0.023165	0.020614
4	256	0.25	0.10032	0.068252	0.01073	0.007872
4	256	1	0.085832	0.032578	0.007288	0.003478
4	256	4	0.401863	0.35208	0.026055	0.02292
4	256	16	0.802974	0.797317	0.043575	0.042995

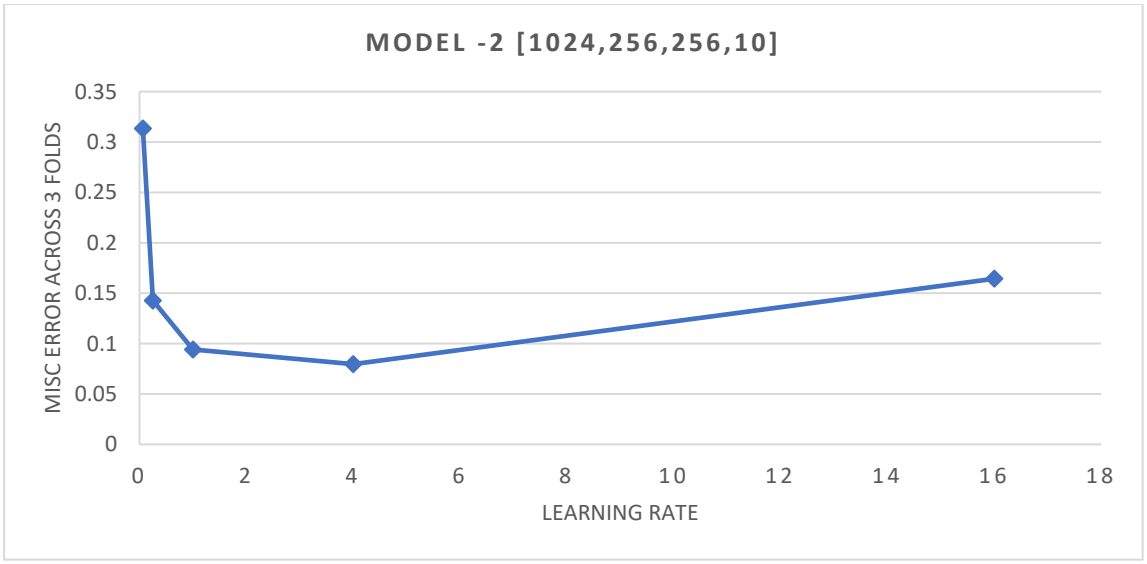


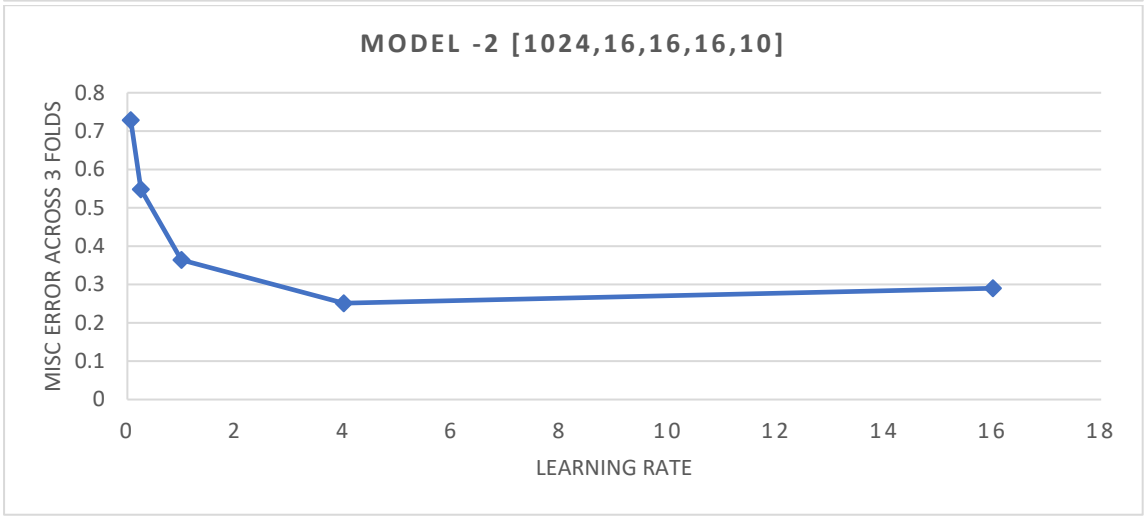


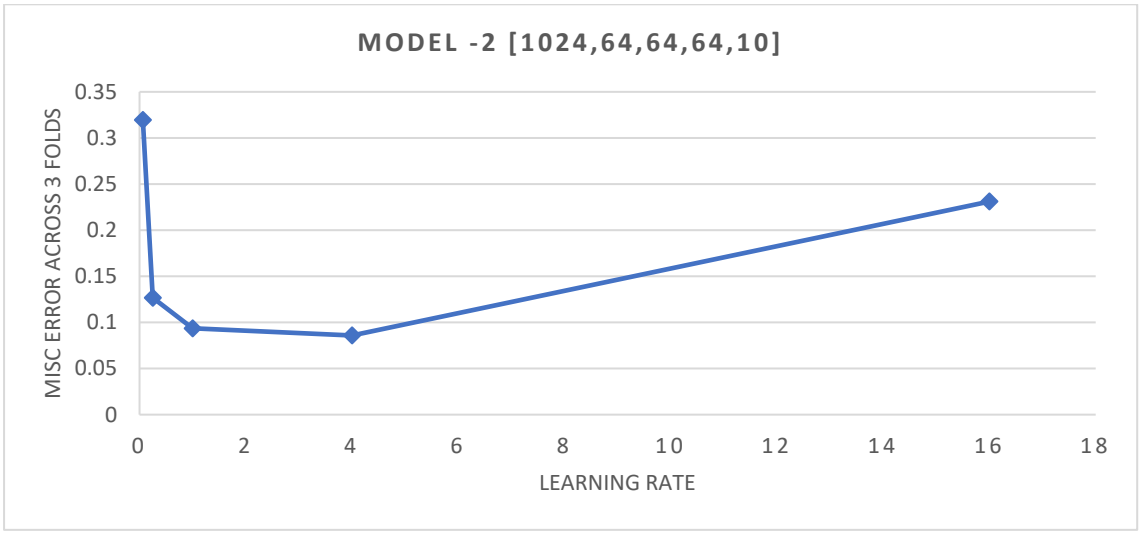

MODEL -2 [1024,256,10]
MISC ERROR ACROSS 3 FOLDS
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0 2 4 6 8 10 12 14 16 18
LEARNING RATE

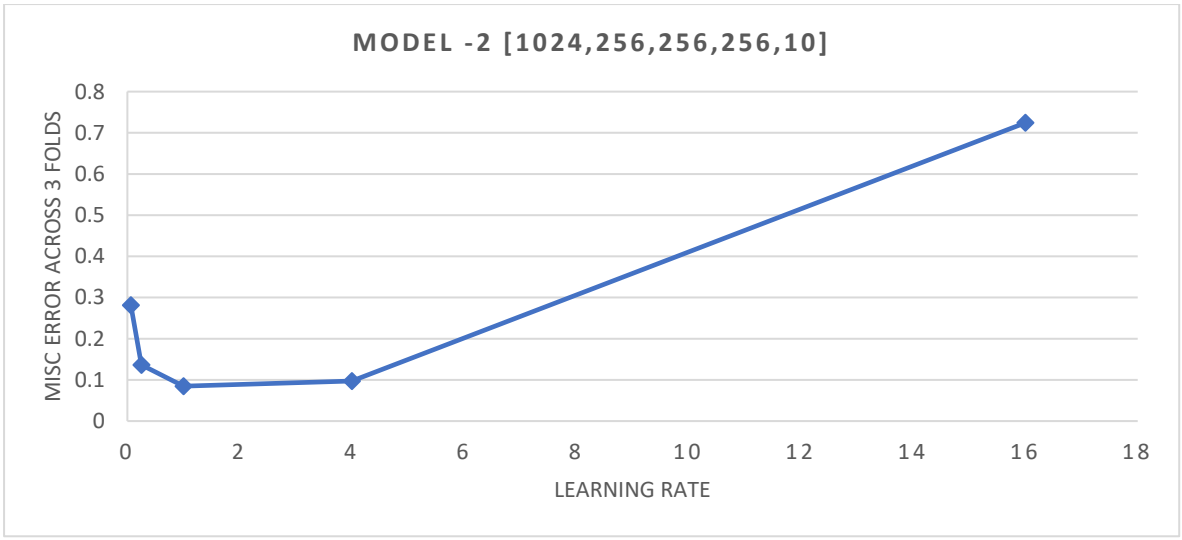

MODEL -2 [1024,16,16,10]
MISC ERROR ACROSS 3 FOLDS
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0 2 4 6 8 10 12 14 16 18
LEARNING RATE

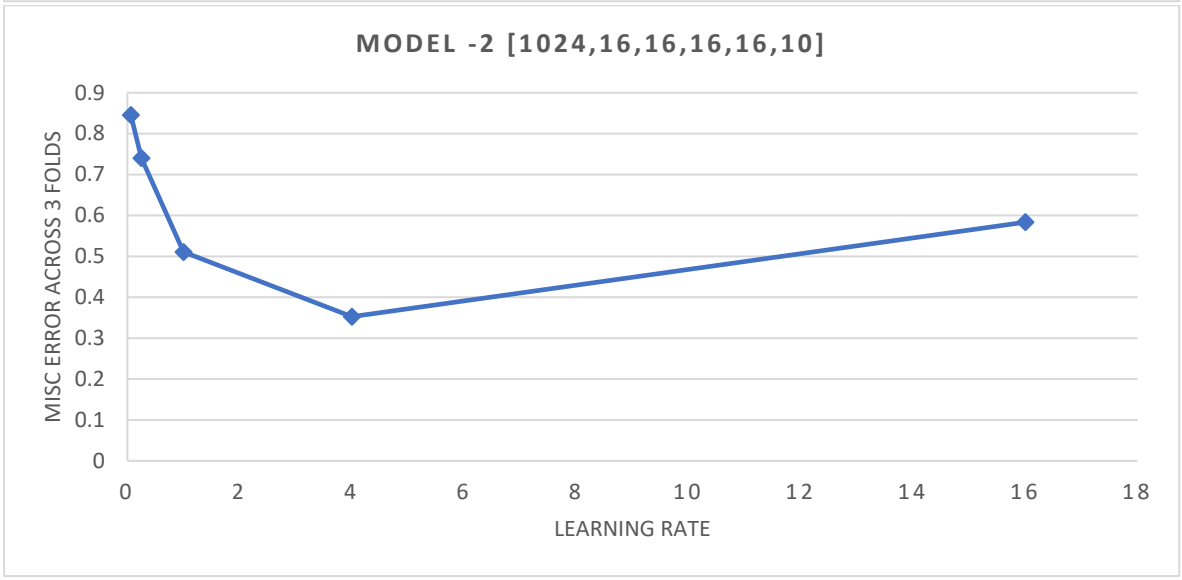

MODEL -2 [1024,64,64,10]
MISC ERROR ACROSS 3 FOLDS
0.3
0.25
0.2
0.15
0.1
0.05
0
0 2 4 6 8 10 12 14 16 18
LEARNING RATE

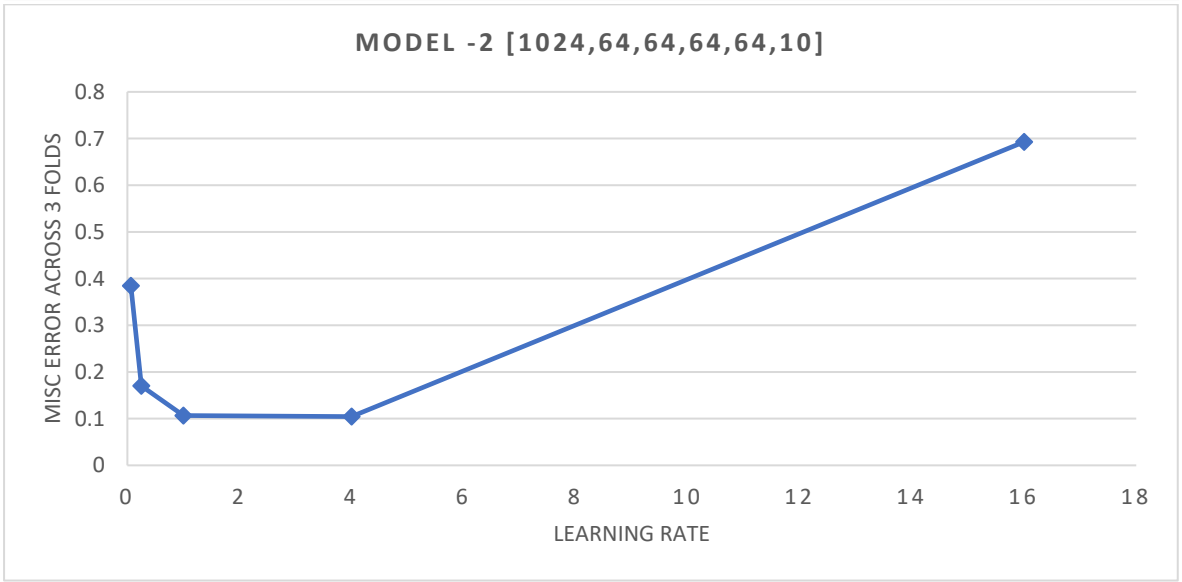

MODEL -2 [1024,256,256,10]
MISC ERROR ACROSS 3 FOLDS
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0 2 4 6 8 10 12 14 16 18
LEARNING RATE

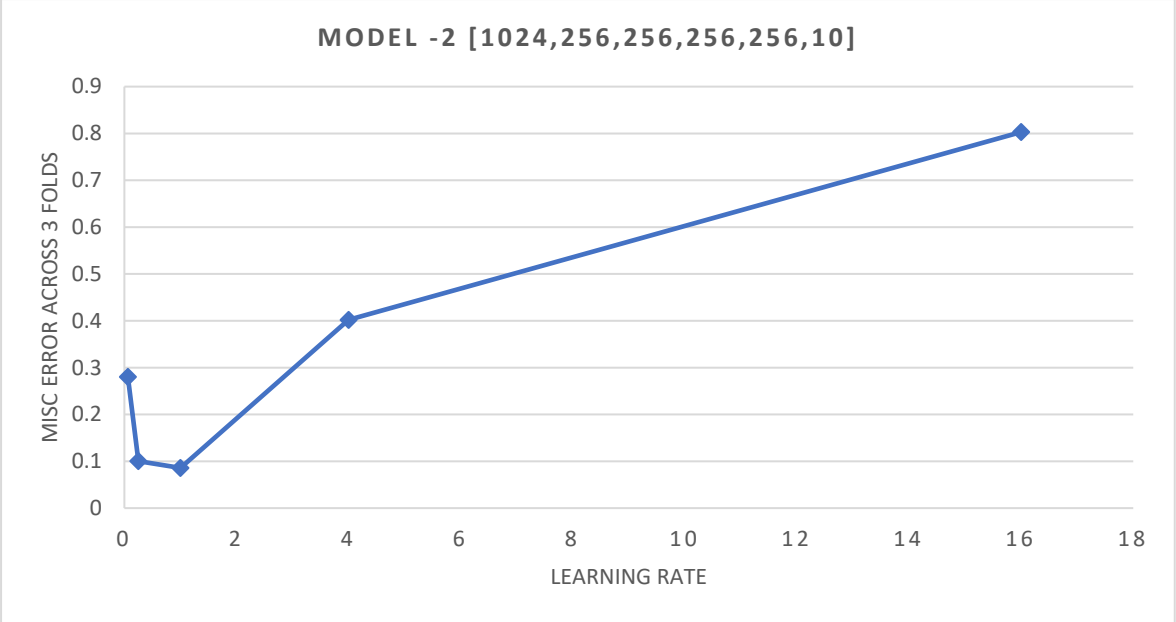

MODEL -2 [1024,16,16,16,10]
MISC ERROR ACROSS 3 FOLDS
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0 2 4 6 8 10 12 14 16 18
LEARNING RATE


MODEL -2 [1024,64,64,64,10]
MISC ERROR ACROSS 3 FOLDS
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0 2 4 6 8 10 12 14 16 18
LEARNING RATE


MODEL -2 [1024,256,256,256,10]
MISC ERROR ACROSS 3 FOLDS
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
2
4
6
8
10
12
14
16
18
LEARNING RATE

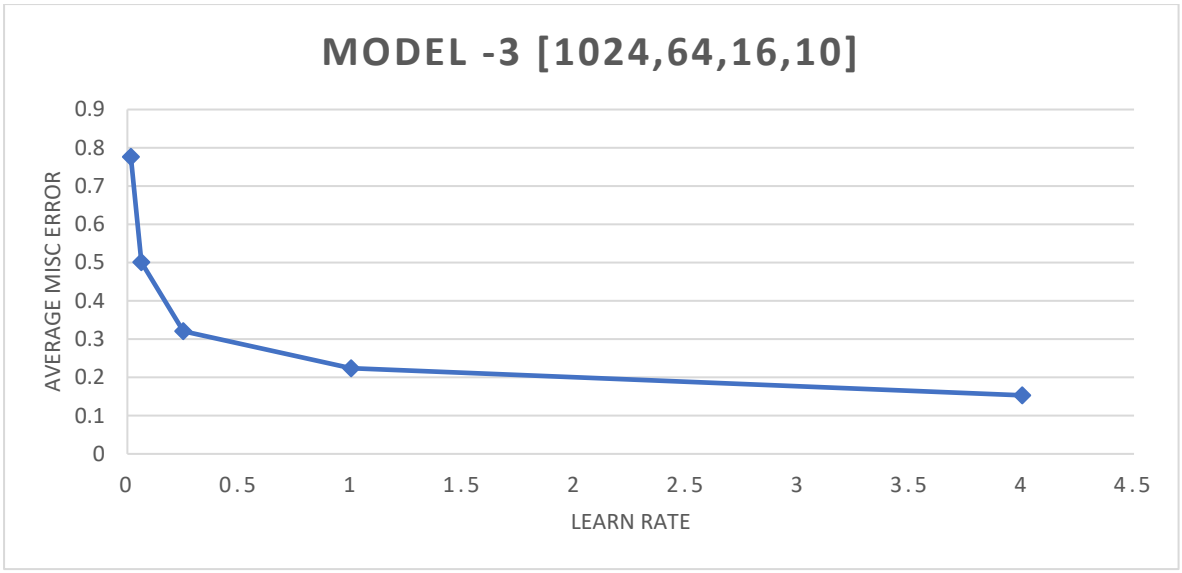

MODEL -2 [1024,16,16,16,16,10]
MISC ERROR ACROSS 3 FOLDS
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
2
4
6
8
10
12
14
16
18
LEARNING RATE

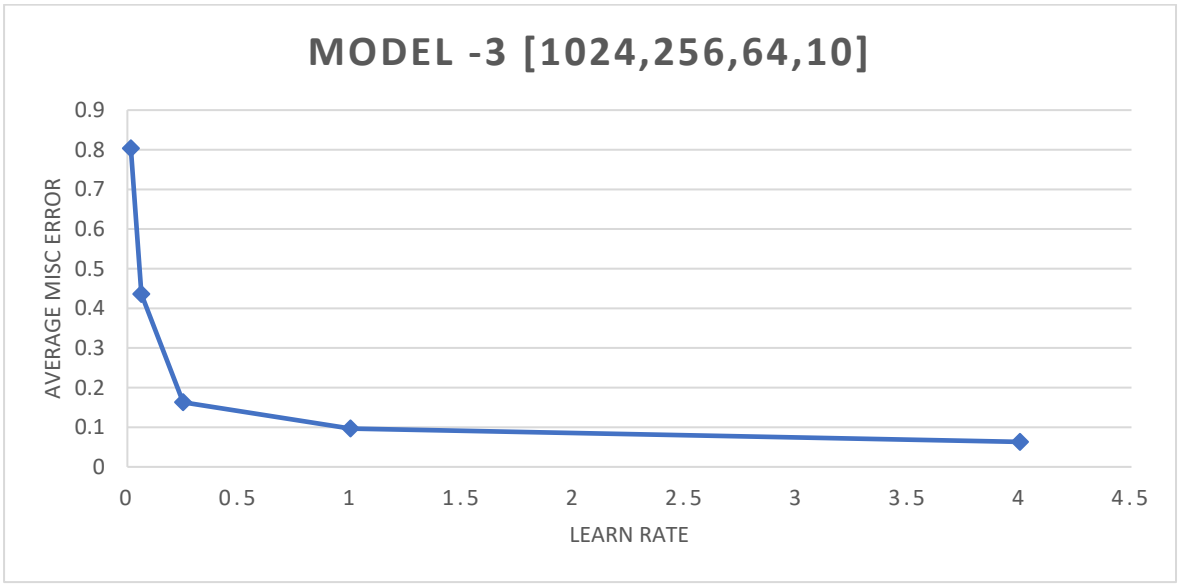

MODEL -2 [1024,64,64,64,64,10]
MISC ERROR ACROSS 3 FOLDS
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
2
4
6
8
10
12
14
16
18
LEARNING RATE



III- Model – 3

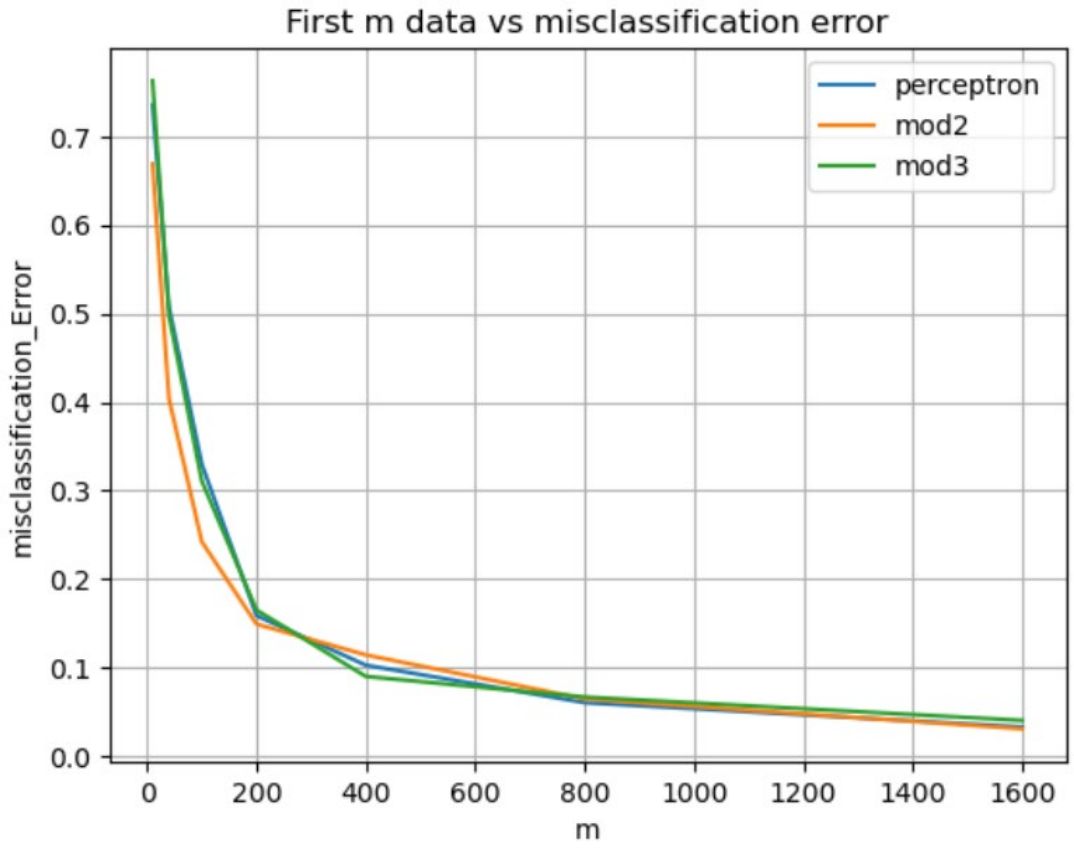
Nunitslayer1	nunitslayer2	Learning rate	misc_test	misc_train	RMSE_test	RMSE_train
64	16	0.015625	0.776163	0.752047	0.043792	0.043471
64	16	0.0625	0.501046	0.487076	0.040009	0.039869
64	16	0.25	0.320572	0.258538	0.028885	0.027165
64	16	1	0.223877	0.168054	0.017835	0.015034
64	16	4	0.153053	0.089719	0.011305	0.007171
256	64	0.015625	0.803532	0.807382	0.044977	0.045118
256	64	0.0625	0.435873	0.412623	0.032743	0.031868
256	64	0.25	0.162898	0.123059	0.017723	0.015739
256	64	1	0.096699	0.06334	0.010782	0.008492
256	64	4	0.063092	0.020425	0.00595	0.002717


MODEL -3 [1024,64,16,10]
AVERAGE MISC ERROR
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
LEARN RATE


MODEL -3 [1024,256,64,10]
AVERAGE MISC ERROR
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
LEARN RATE

3) Learning Curves:

(a) for first m data in 1934 images



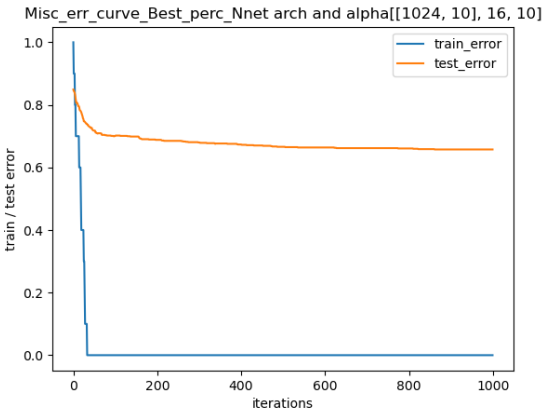
The above graph clearly states the more the data the more precise the classification. The best model is with 1934 images as training data. It might get better if we are able train with 3000 images.

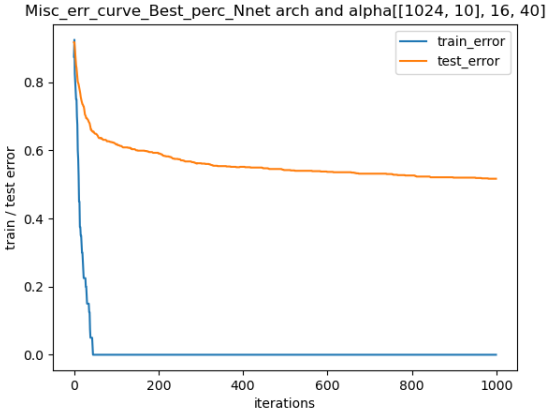
Training a model with more data can often improve its performance and accuracy. This is because having more data provides the model with a larger and more diverse sample of the underlying patterns and relationships in the data.

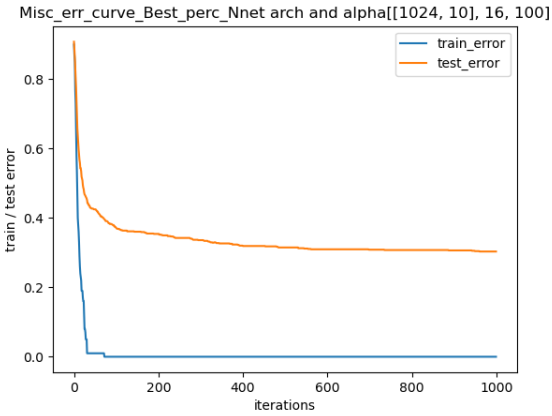
With more data, the model has a greater opportunity to learn a more representative set of features and to better generalize its predictions to new, unseen data. Additionally, more data can help the model identify and correct biases that may have been present in a smaller training set.

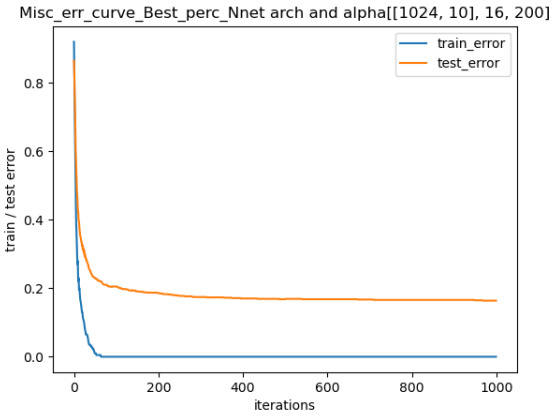
However, it's important to note that simply adding more data to a model doesn't necessarily guarantee better performance. The quality of the data is also important, and there may be diminishing returns as the amount of data increases beyond a certain point. Additionally, training a model with more data can require more computational resources and time, so it's important to balance the benefits of additional data against the costs of training a larger model.

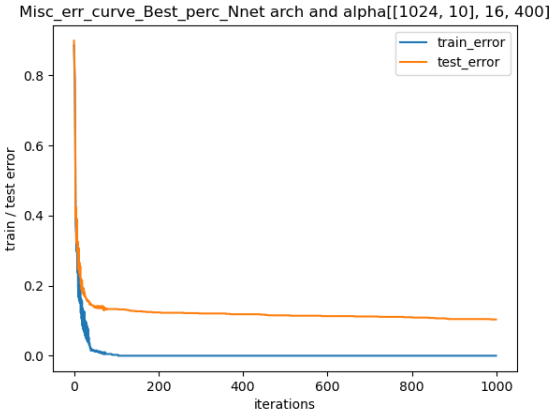
MODEL -1, Learning Curves for m = {10, 20, 40, 100, 200, 400, 800, 1600}

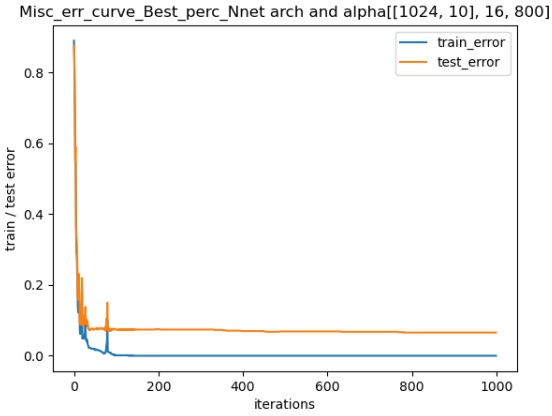


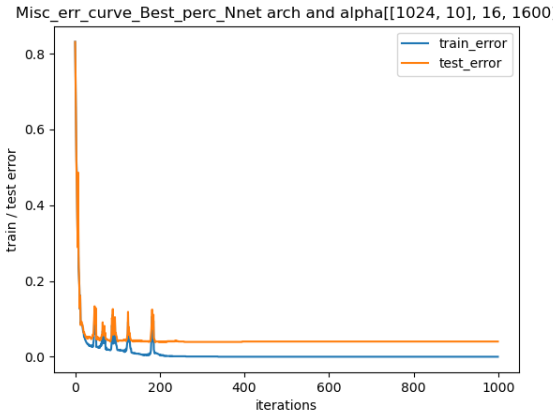




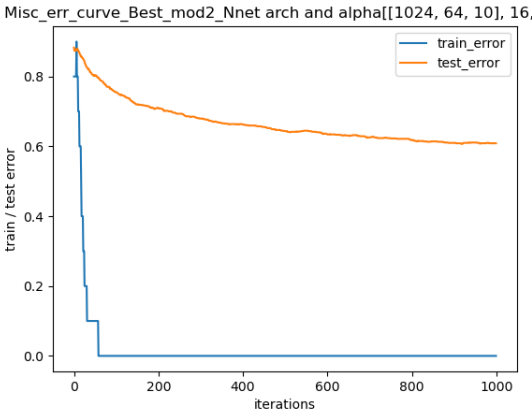


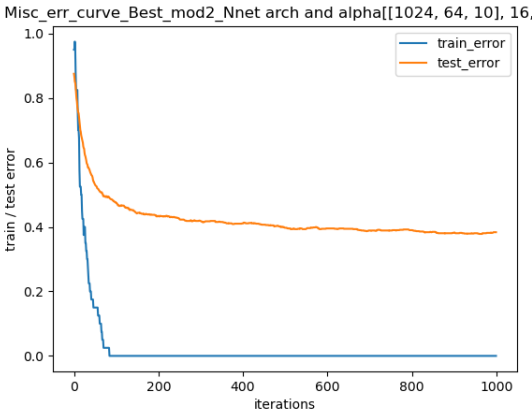


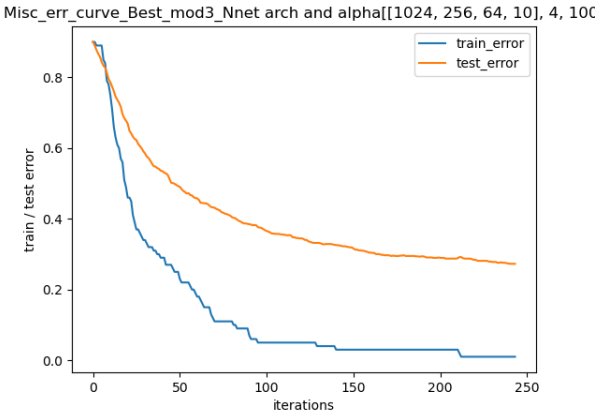


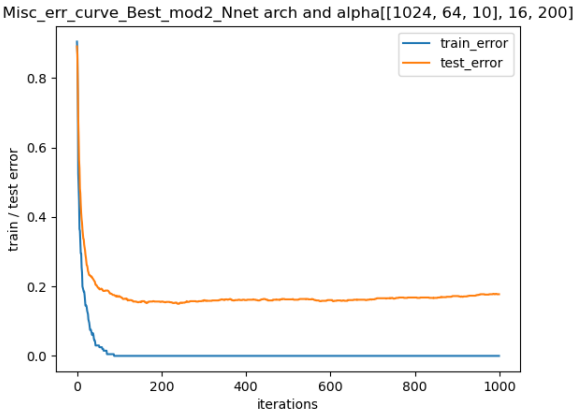


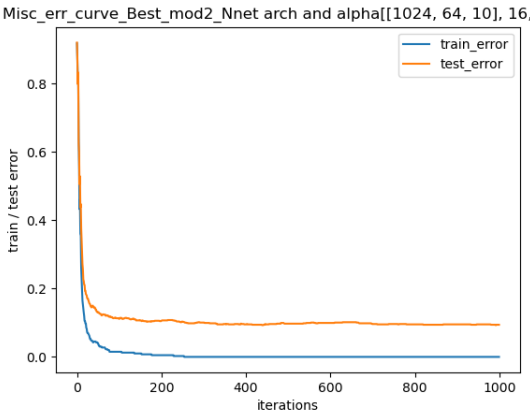
MODEL -2, Learning Curves for m = {10, 20, 40, 100, 200, 400, 800, 1600}

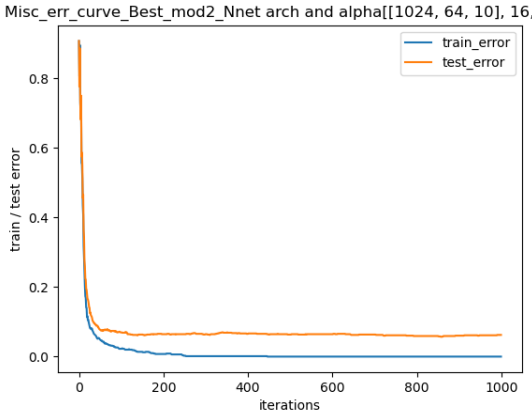


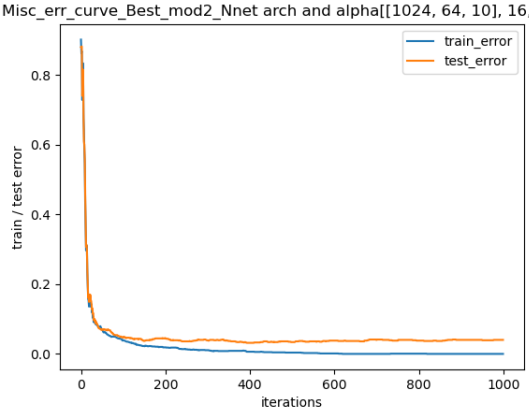




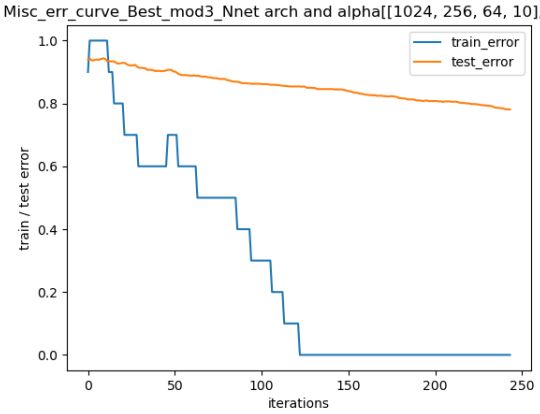


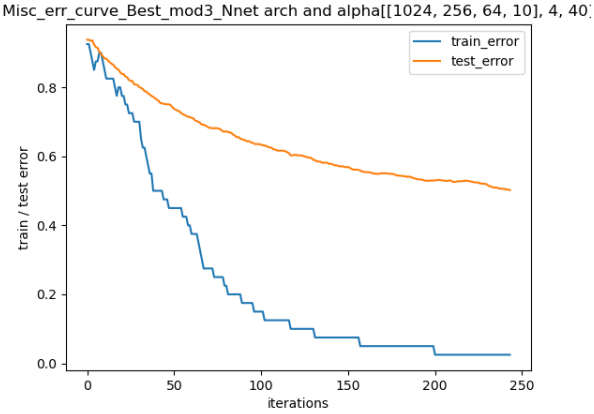


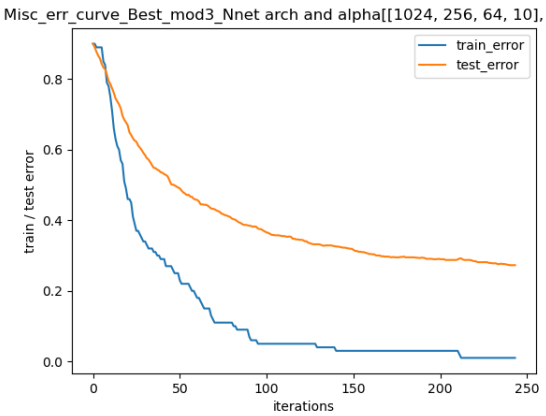


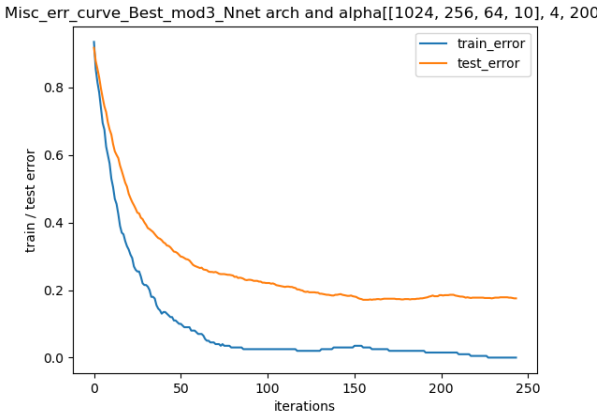


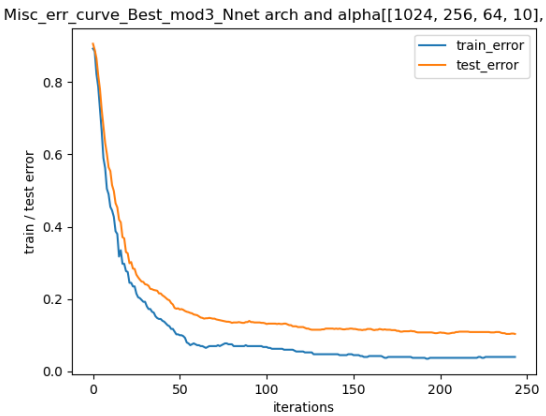
MODEL -3, Learning Curves for m = {10, 20, 40, 100, 200, 400, 800, 1600}

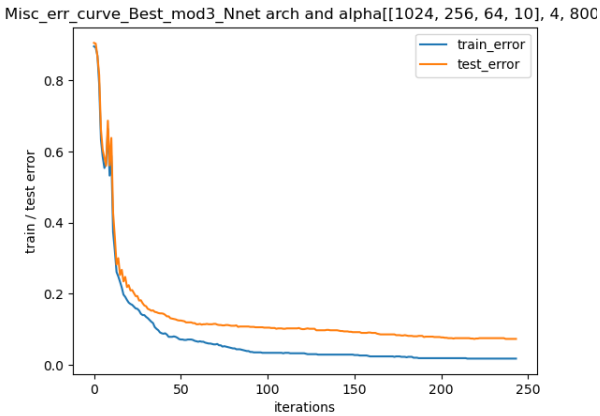


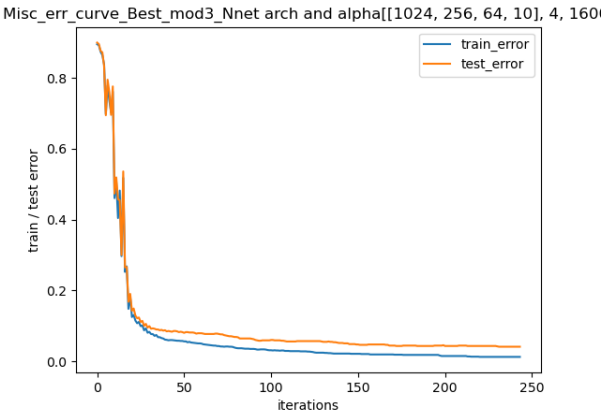






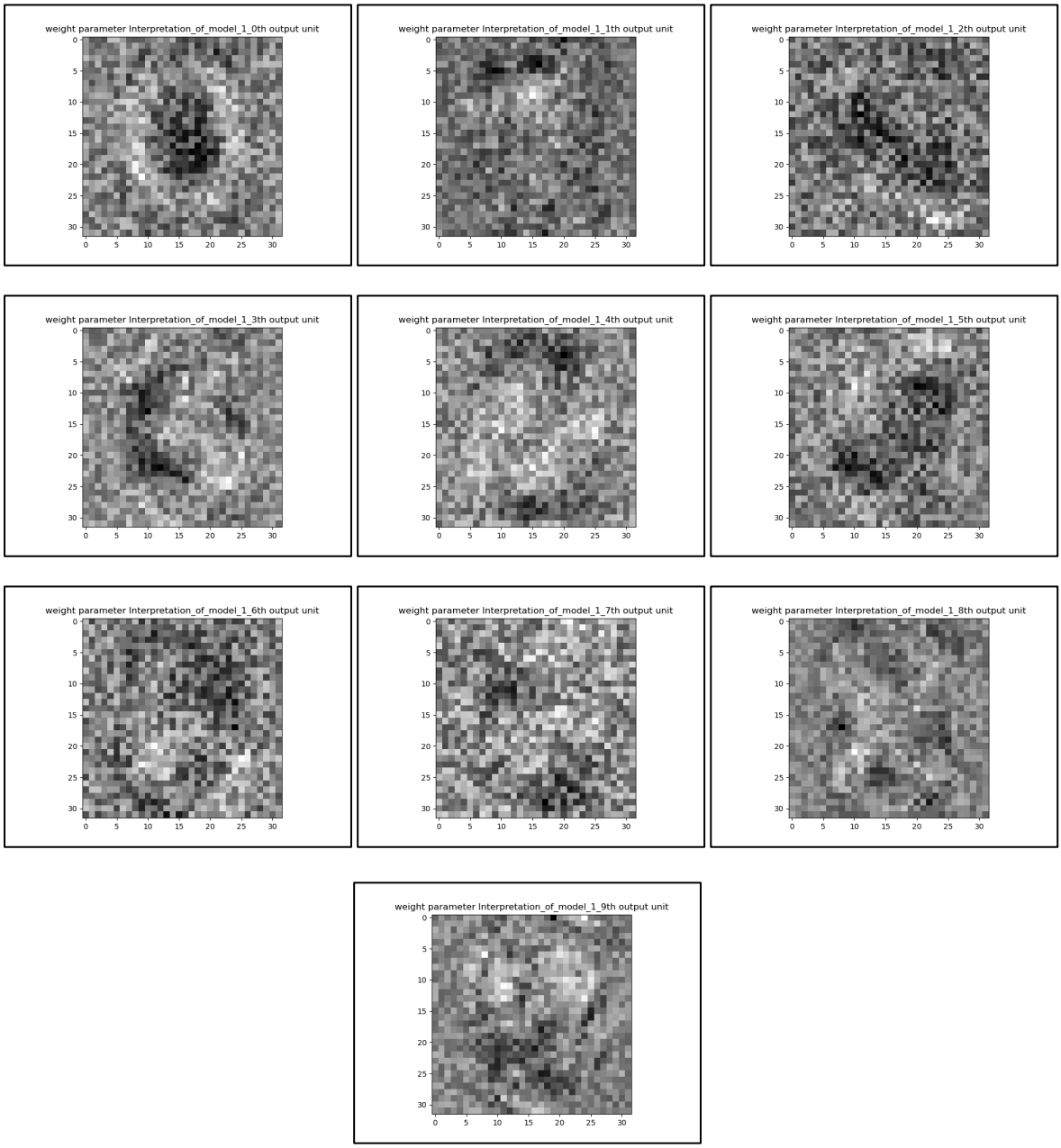






4) Weight Parameter Interpretation

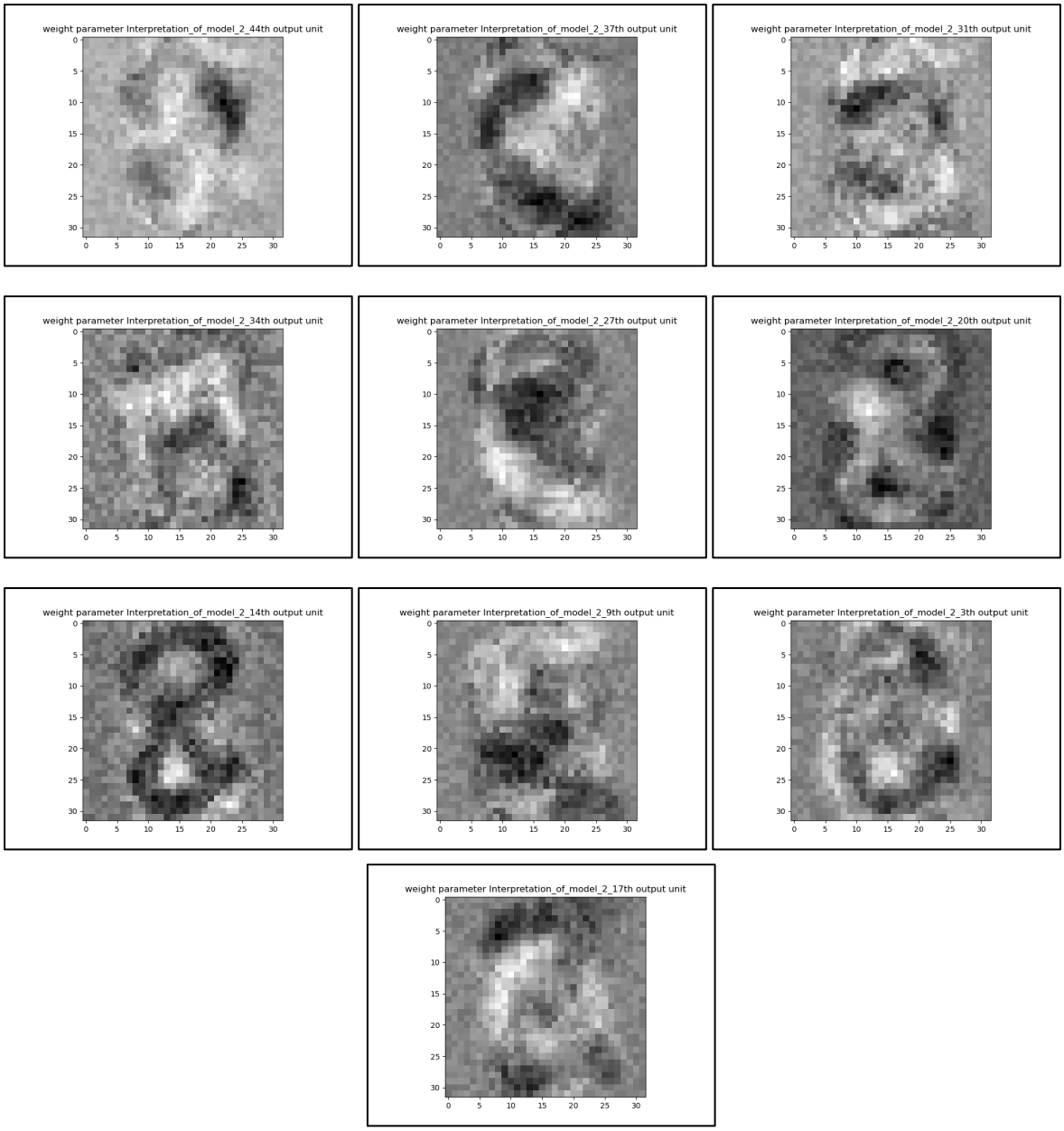
a) The Final Perceptron



A basic linear classifier called a perceptron learns a decision boundary to divide the input into several categories. It simply needs one run through the data to converge and changes the weights based on the misclassification error. It can, however, be sensitive to noisy data and can only learn linearly separable data.

A more potent and adaptable class of models that can learn non-linear decision limits are neural networks. We employed two alternative neural network designs for this project: a shallow network with two hidden layers that each include a separate hidden unit and a deep network with many hidden layers that all contain the same hidden unit. Although the deep network is more expressive and has the capacity to learn more intricate data representations, it is also more prone to overfitting.

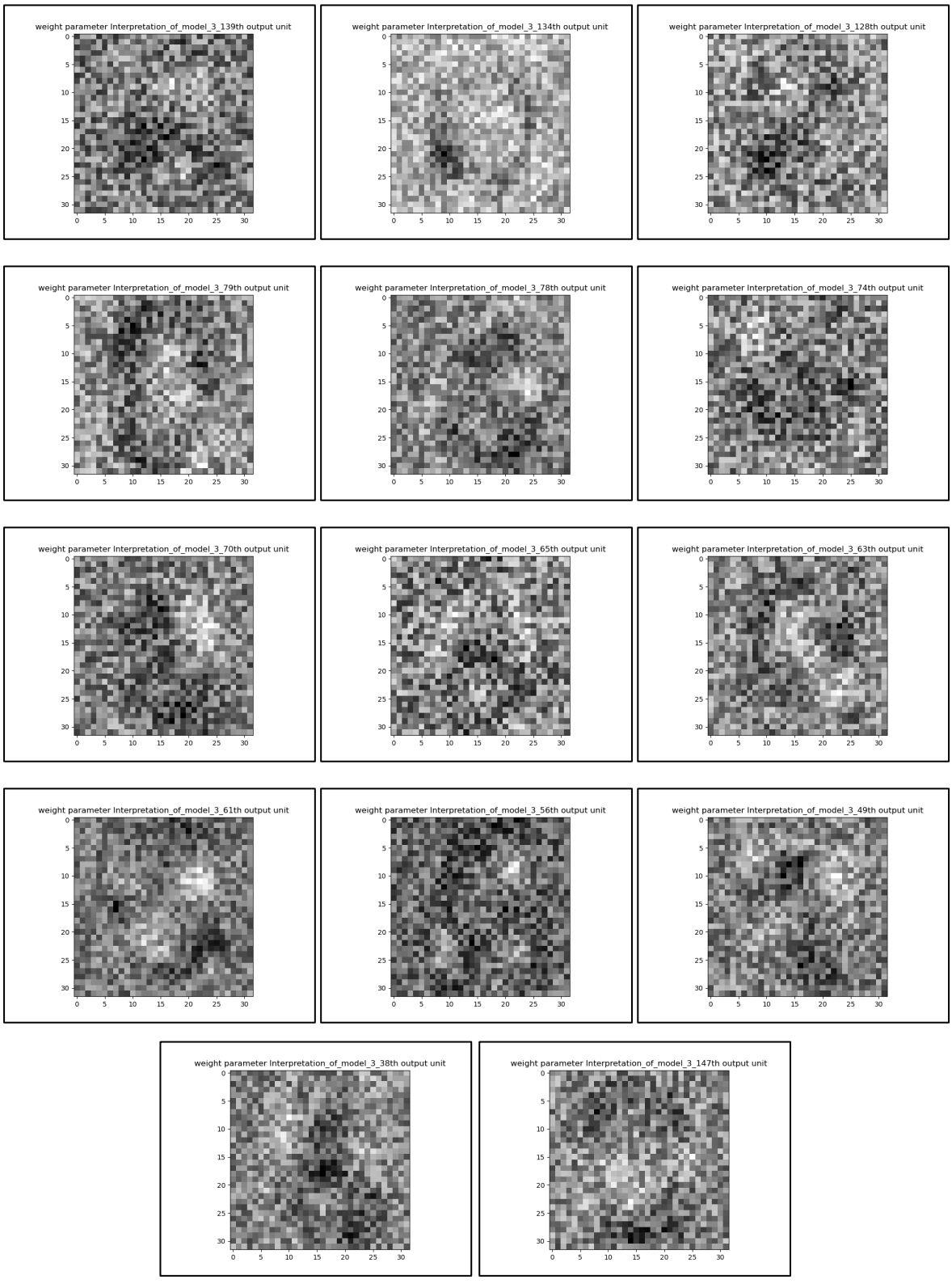
b) Model-2-first hidden layer



Throughout the cross validations, Model 2 with a single hidden layer and 64 hidden units appeared to have the lowest average misclassification error. After considering the misclassification Error, Models 1 and 3 produced findings that were comparable to those of Model 2.

This was also evident in the Trial Dataset, where Model 2 incorrectly identified 34 photos and Models 1 and 3 incorrectly classified 39 and 36 images, respectively.

c) Model-3 – first hidden layer



DISCUSSION AND CONCLUSION:

Model 1 and 2 ran well with learning rate of 4*2 however with Model 3 it was seen that it performed better with 41 as the learning rate. However it has to be noted that the misclassification error varied between 42 and 4*3 for Model 1 on running the CV multiple times.

As obvious as it should be, with increase in the number of the data size it was seen that the models performed better. The errors were seen to generalize near 1000 records of data, this helps in reducing the resources and efficiently training models upto the efficient number of data records.

The weight parameters were well defined for the perceptron model rather than that of the model 2 and model 3. This is due to the fact that the perceptron has just an input and an output layer, whereas the weight parameters for the other networks is derived from the first layer of the networks rather than the last layer.

So the final results with random initialization were tabulated

Model	Misclassification error
Perceptron	0.04016
Neural network with multiple layers	0.03594
Neural network with two converging layers	0.03911