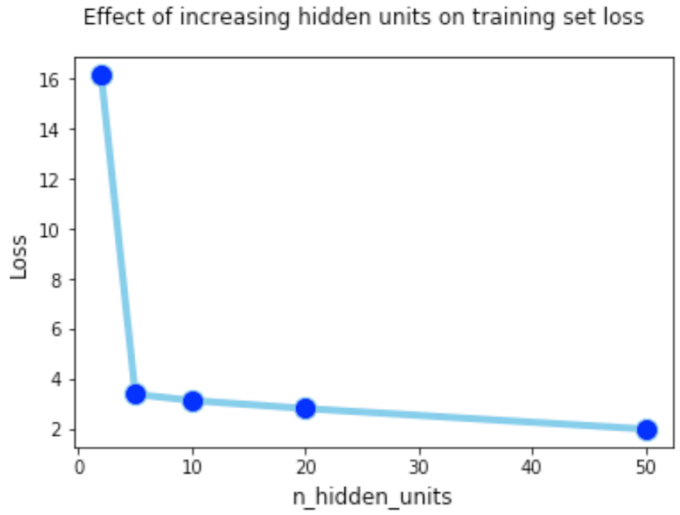
ROB 537 Learning Based Control
HW #1

a) Increasing the number of hidden units increases classification accuracy and decreases the MSE loss, thereby improving confidence in predictions on the training set.

Performed tests on the Train, Test 1, Test 2, Test 3 sets by varying the number of hidden units but keeping the **learning rate to 0.1** and **n_epochs = 10**



Hidden Units = 2	**Loss (0.5 * SE)**	**Accuracy (%)**
Train	16.13	97
Test 1	14.19	98
Test 2	19.45	81
Test 3	26.58	52

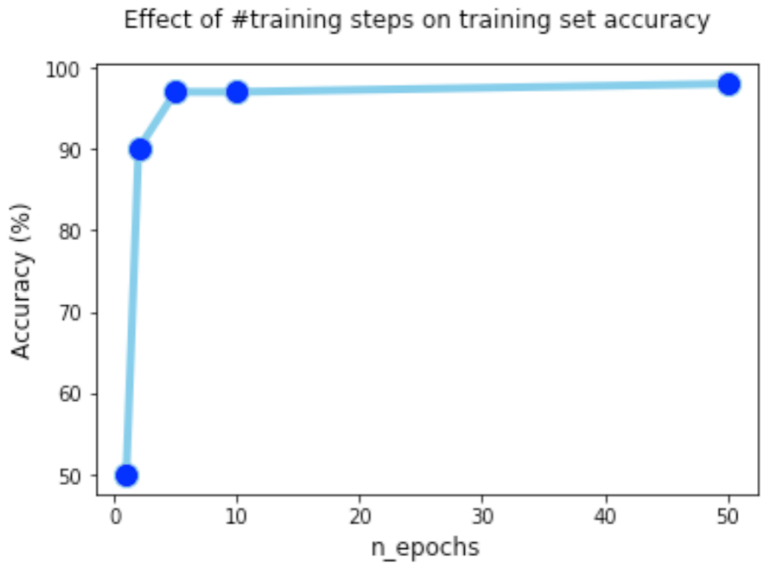
Hidden Units = 5	**Loss**	**Accuracy**
Train	3.38	97
Test 1	2.60	98
Test 2	11.37	86
Test 3	39.25	51

Hidden Units = 10	**Loss**	**Accuracy**
Train	3.13	97
Test 1	2.37	99
Test 2	11.03	86
Test 3	39.23	51

Hidden Units = 20	Loss	Accuracy
Train	2.81	97
Test 1	2.03	98
Test 2	10.54	85
Test 3	41.84	51

Hidden Units = 50	Loss	Accuracy
Train	1.99	98
Test 1	1.58	99
Test 2	10.50	86
Test 3	43.53	50

b) The accuracy improves with more training steps. The trend is it improves by a large amount in the beginning uptill 1 to 2 epochs then accuracy improvements are little but the loss keeps improving even when trained till 50 epochs.
Tests were done with **learning rate = 0.1, hidden_units = 10**



n_epochs = 1	Loss	Accuracy (%)
Train	24.6	50
Test 1	19.1	50
Test 2	22.7	49
Test 3	29.14	28

n_epochs = 2	Loss	Accuracy (%)
Train	9.17	90
Test 1	7.1	99
Test 2	14.7	85
Test 3	30.4	51

n_epochs = 5	Loss	Accuracy (%)
Train	3.8	97
Test 1	3.4	99
Test 2	12.01	87
Test 3	36.12	51

n_epochs = 10	Loss	Accuracy (%)
Train	3	97
Test 1	2.1	98
Test 2	10.7	86
Test 3	40.7	50

n_epochs = 50	Loss	Accuracy (%)
Train	1.6	98
Test 1	1.12	99
Test 2	10.3	87
Test 3	45.01	52

c) A learning rate of 1 to 0.1 worked best, upon decreasing the learning rate further to 0.01 and 0.001 the network was not learning much and the accuracy was not as high as lr = 1 or lr = 0.1.
The tests were performed with no of **hidden units = 10** and network was trained for **10 epochs**

lr = 1	Loss	Accuracy (%)
Train	1.6	98
Test 1	0.9	98
Test 2	10.9	86
Test 3	46.2	51

lr = 0.1	Loss	Accuracy (%)
Train	2.8	98
Test 1	2.3	98
Test 2	11.01	85
Test 3	39.9	52

lr = 0.01	Loss	Accuracy (%)
Train	13.1	97
Test 1	12.04	98
Test 2	18.4	85
Test 3	27.5	50

lr = 0.001	**Loss**	**Accuracy (%)**
Train	26.9	44
Test 1	27.01	43
Test 2	27.9	26
Test 3	33.6	21

d) Other critical parameters like unit **normalizing the data** and **shuffling the data** proved useful. Normalizing the data to a mean of 0 and std equal to 1 helped faster convergence. Shuffling the data was critical as without it the neural net would just learn to output same results irrespective of input data.

Tests were done with **learning rate = 0.1** and **hidden units = 10, training for 2 epochs**

With Noramalization	**Loss**	**Accuracy (%)**
Train	9.2	**95**
Test 1	7.12	97
Test 2	14.9	82
Test 3	30.7	52

Without Normalization	**Loss**	**Accuracy (%)**
Train	10.6	**85**
Test 1	7.9	90
Test 2	18.9	66
Test 3	33.4	51

With Shuffling	**Loss**	**Accuracy (%)**
Train	9.2	**95**
Test 1	7.12	97
Test 2	14.9	82
Test 3	30.7	52

Without Shuffling	**Loss**	**Accuracy (%)**
Train	14.6	**85**
Test 1	10.3	95
Test 2	17.5	76
Test 3	28.5	51

e)
n_epochs = 10, lr = 0.1

Hidden Units = 50	**Loss**	**Accuracy**
Train	1.99	98
Test 1	1.58	99
Test 2	10.50	86
Test 3	43.53	50

f) Yes the performance did differ on different test sets, my hypothesis is that, based on the above table and analysis in previous sections,
Test 1 seems to be a subset of Train set since its loss, accuracy is similar even when network capacity is low.

Test 2 is similar to training set but **has some different examples which are unseen in training set**

Test 3 is a completely **different set with a data distribution different from training set**, this hypothesis can be proved by looking at section-**a** tables that even when network capacity is increased, the loss on Test 1, Test 2 slightly deceases, but for Test 3 it increases.