Complete Neural Network

William Tholke

June 21, 2021

Foothill College, Intermediate Software Design in Python [CS3B]

1 Sample Runs/Discussion: run_iris()

Sample Run 1: hidden layer of three neurodes, 10001 epochs, 70% training factor, randomly ordered; this is a partial run that includes the RSME for epochs 0-900, 4000-4900, 9000-10001, and the test RSME:

```
Epoch 0 RMSE = 0.6132209806031103
Epoch 100 RMSE = 0.3429896454055927
Epoch 200 RMSE = 0.32417096241542065
Epoch 300 RMSE = 0.3142067944565672
Epoch 400 RMSE = 0.31056125695266157
Epoch 500 RMSE = 0.30541247890481776
Epoch 600 RMSE = 0.30397372104571596
Epoch 700 RMSE = 0.3029599559139647
Epoch 800 RMSE = 0.3048361590180093
Epoch 900 RMSE = 0.30221139582902695
... Epochs 1000-3900 omitted ...
Epoch 4000 RMSE = 0.2914347231295721
Epoch 4100 RMSE = 0.29476105444042183
Epoch 4200 RMSE = 0.2954944894466152
Epoch 4300 RMSE = 0.296216279528089
Epoch 4400 RMSE = 0.29188464485579224
Epoch 4500 RMSE = 0.29644092593477156
Epoch 4600 RMSE = 0.2925294845510375
Epoch 4700 RMSE = 0.2922747320927831
Epoch 4800 RMSE = 0.2961006029670823
Epoch 4900 RMSE = 0.2922030839651154
... Epochs 5000-8900 omitted ...
Epoch 9000 RMSE = 0.29125630617057435
Epoch 9100 RMSE = 0.2896470116020711
Epoch 9200 RMSE = 0.28514388323416284
Epoch 9300 RMSE = 0.2876882230955387
Epoch 9400 RMSE = 0.2892234496724425
Epoch 9500 RMSE = 0.2899735125082878
Epoch 9600 RMSE = 0.2882902624165164
Epoch 9700 RMSE = 0.2882535666138182
Epoch 9800 RMSE = 0.29011198071325034
Epoch 9900 RMSE = 0.28940399187742843
Epoch 10000 RMSE = 0.29097837974132806
Final Epoch RMSE = 0.29097837974132806
Test RMSE = 0.17769261138394016
```

Discussion: The neural network makes signficant progress from epoch 0 (RSME: 0.6132209806031103) to the 500th epoch (RSME: 0.30541247890481776), slows down between the 500th and 4000th epochs, and then stagnates heavily from the 4000th epoch to the final epoch. The test RSME is 0.17769261138394016, as compared to the final training RSME of 0.29097837974132806, indicating that the network is not over-fitted to the training data but is also not classifying all of the testing data.

Sample Run 2: hidden layer of three neurodes, 1001 epochs, 90% training factor, randomly ordered; this is a complete run that includes the RSME for epochs 0-1001 and the test RSME:

```
Epoch 0 RMSE = 0.5719589260654753
Epoch 100 RMSE = 0.3324275923761881
Epoch 200 RMSE = 0.31361423887133005
Epoch 300 RMSE = 0.30535262495276827
Epoch 400 RMSE = 0.30253834558594384
Epoch 500 RMSE = 0.29811941723815466
Epoch 600 RMSE = 0.3002913279502846
Epoch 700 RMSE = 0.29641426154038836
Epoch 800 RMSE = 0.29488499320148537
Epoch 900 RMSE = 0.2957230322295651
Epoch 1000 RMSE = 0.2944339805214937
```

```
Final Epoch RMSE = 0.2944339805214937
Test RMSE = 0.09659712960363329
```

Discussion: The neural network makes significant progress from epoch 0 (RMSE: 0.5719589260654753) to the 200th epoch (RSME: 0.31361423887133005), then seems to converge to a value near 0.29, as shown by the final epoch (RSME: 0.2944339805214937). The test RSME is 0.09659712960363329, as compared to the final training RSME of 0.2944339805214937, indicating that the network successfully classified nearly *all* of the testing data; this is a desired result. This result likely comes from the increase of training factor from 70% in Sample Run 1 to 90% in Sample Run 2. Interestingly, decreasing the number of epochs from Sample Run 1 by 90% (10001 to 1001) does not appear to negatively impact the test RSME.

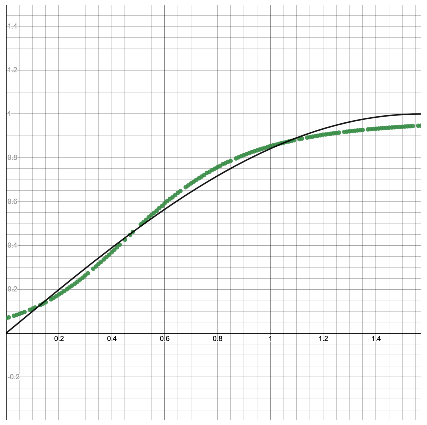
2 Sample Run/Discussion: run_sin()

Sample Run 1: hidden layer of three neurodes, 10001 epochs, 10% training factor, randomly ordered; this is a partial run that includes the RSME for epochs 0-900, 4000-4900, 9000-10001, and the test RSME:

```
Epoch 0 RMSE = 0.3210370009660702
Epoch 100 RMSE = 0.3176358546632305
Epoch 200 RMSE = 0.3150198185790094
Epoch 300 RMSE = 0.31209939186060226
Epoch 400 RMSE = 0.307870874408097
Epoch 500 RMSE = 0.300087840899046
Epoch 600 RMSE = 0.28382378789741713
Epoch 700 RMSE = 0.2559031805100752
Epoch 800 RMSE = 0.2238898990767785
Epoch 900 RMSE = 0.19579019094358918
... Epochs 1000-3900 omitted ...
Epoch 4000 RMSE = 0.045545949632551756
Epoch 4100 RMSE = 0.044755920769309086
Epoch 4200 RMSE = 0.044014574934214915
Epoch 4300 RMSE = 0.043318156373913716
Epoch 4400 RMSE = 0.04266411823264981
Epoch 4500 RMSE = 0.04204847938541597
Epoch 4600 RMSE = 0.04146886643823504
Epoch 4700 RMSE = 0.0409225800066788
Epoch 4800 RMSE = 0.04040753949097107
Epoch 4900 RMSE = 0.039921811586134476
... Epochs 5000-8900 omitted ...
Epoch 9000 RMSE = 0.03167663084187742
Epoch 9100 RMSE = 0.03160128685601634
Epoch 9200 RMSE = 0.03152841051110542
Epoch 9300 RMSE = 0.031457795509788056
Epoch 9400 RMSE = 0.03138929009257179
Epoch 9500 RMSE = 0.03132329067237128
Epoch 9600 RMSE = 0.03125876222157173
Epoch 9700 RMSE = 0.031196849268610664
Epoch 9800 RMSE = 0.031136334898400588
Epoch 9900 RMSE = 0.03107775967350073
Epoch 10000 RMSE = 0.031020938763814374
Final Epoch RMSE = 0.031020938763814374
Test RMSE = 0.10464967241608952
```

Discussion: The neural network makes slow progress from epoch 0 (RSME: 0.3210370009660702) to the 800th epoch (RSME: 0.2238898990767785), speeds up considerably leading into the 4900th epoch (RSME: 0.039921811586134476), then slows down until the final epoch. The test RSME is 0.10464967241608952, as compared to the final training RSME of 0.031020938763814374, indicating that the network is not learning as expected; the network is over-fitted to the training data.

In the visualization below, the green data points that form what looks to be a line, which represents the inputs and outputs associated with testing. The black line represents $\sin x\{0 < x < \frac{\pi}{2}\}$.



3 Sample Runs/Discussion: run_XOR()

Sample Run 1: hidden layer of three neurodes, 10001 epochs, 100% training factor, randomly ordered; this is a partial run that includes the RSME for epochs 0-900, 4000-4900, 9000-10001; note that, since the four pieces of data in *load_XOR*() are interdependent, 0% of the data is tested:

```
Epoch 0 RMSE = 0.5100681711791044
Epoch 100 RMSE = 0.501615411125597
Epoch 200 RMSE = 0.5010871311701888
Epoch 300 RMSE = 0.5010292362372026
Epoch 400 RMSE = 0.5010033678398855
Epoch 500 RMSE = 0.5009773144325932
Epoch 600 RMSE = 0.5009519644751111
Epoch 700 RMSE = 0.5009270536752622
Epoch 800 RMSE = 0.500900162474366
Epoch 900 RMSE = 0.5008785096646876
... Epochs 1000-3900 omitted ...
Epoch 4000 RMSE = 0.49755580198414956
Epoch 4100 RMSE = 0.4971279921118139
Epoch 4200 RMSE = 0.4966335413168149
Epoch 4300 RMSE = 0.4960792819331531
Epoch 4400 RMSE = 0.49544425112354135
Epoch 4500 RMSE = 0.49472529626405437
Epoch 4600 RMSE = 0.49391263141236863
Epoch 4700 RMSE = 0.49299642819733897
Epoch 4800 RMSE = 0.4919693885878862
Epoch 4900 RMSE = 0.49081988557667366
... Epochs 5000-8900 omitted ...
Epoch 9000 RMSE = 0.3271383138518327
Epoch 9100 RMSE = 0.3189864277668445
Epoch 9200 RMSE = 0.3108387230755795
Epoch 9300 RMSE = 0.30276177586617414
Epoch 9400 RMSE = 0.2948131281462414
Epoch 9500 RMSE = 0.28704552299747826
Epoch 9600 RMSE = 0.27949332264814825
Epoch 9700 RMSE = 0.2721914787068405
Epoch 9800 RMSE = 0.26515649885877823
Epoch 9900 RMSE = 0.2584089033349249
Epoch 10000 RMSE = 0.2519476090355202
Final Epoch RMSE = 0.2519476090355202
```

Discussion: The neural network makes slow progress from epoch 0 (RSME: 0.5100681711791044) to the 4900th epoch (RMSE: 0.49081988557667366), then speeds up leading into the final epoch (RMSE: 0.2519476090355202). The network trained poorly with 10001 epochs, but might have continued its trajectory towards a lower RSME if more epochs were included. A second run with the nearly training parameters but with 50001 epochs is needed to conclude whether the aforementioned hypothesis is valid.

Sample Run 2: hidden layer of three neurodes, 50001 epochs, 100% training factor, randomly ordered; this is a partial run that includes the RSME for epochs 0-900, 19000-19900, 23000-23900, 35000-35900, 50000; note that, since the four pieces of data in $load\_XOR()$ are interdependent, 0% of the data is tested:

```
Epoch 0 RMSE = 0.5775997163360249
Epoch 100 RMSE = 0.5142212337483002
Epoch 200 RMSE = 0.5021566462013546
Epoch 300 RMSE = 0.5011343532141063
Epoch 400 RMSE = 0.501036732648534
Epoch 500 RMSE = 0.5010145076700274
Epoch 600 RMSE = 0.5009942111564101
Epoch 700 RMSE = 0.5009802798190848
Epoch 800 RMSE = 0.5009629789410812
Epoch 900 RMSE = 0.5009516742405496
```

```
... Epochs 1000-18900 omitted ...
Epoch 19000 RMSE = 0.42223555634332643
Epoch 19100 RMSE = 0.41617649258028133
Epoch 19200 RMSE = 0.40976973347108125
Epoch 19300 RMSE = 0.4030181592414226
Epoch 19400 RMSE = 0.3959419262661664
Epoch 19500 RMSE = 0.38853539327851677
Epoch 19600 RMSE = 0.3808487743641256
Epoch 19700 RMSE = 0.3729029184729653
Epoch 19800 RMSE = 0.3647434555412751
Epoch 19900 RMSE = 0.3564367644488665
... Epochs 20000-22900 omitted ...
Epoch 23000 RMSE = 0.18124260867364342
Epoch 23100 RMSE = 0.17853301008077954
Epoch 23200 RMSE = 0.1759270465214544
Epoch 23300 RMSE = 0.17341871159276756
Epoch 23400 RMSE = 0.17100375308515367
Epoch 23500 RMSE = 0.16867619178194615
Epoch 23600 RMSE = 0.1664315209690606
Epoch 23700 RMSE = 0.1642653880253981
Epoch 23800 RMSE = 0.162173910092067
Epoch 23900 RMSE = 0.16015313377691667
... Epochs 24000-34900 omitted ...
Epoch 35000 RMSE = 0.08066203618941267
Epoch 35100 RMSE = 0.08038478973065409
Epoch 35200 RMSE = 0.08011032543886919
Epoch 35300 RMSE = 0.07983857376602208
Epoch 35400 RMSE = 0.07956950624952377
Epoch 35500 RMSE = 0.07930307081154613
Epoch 35600 RMSE = 0.07903922492088508
Epoch 35700 RMSE = 0.07877792956378017
Epoch 35800 RMSE = 0.07851914633752932
Epoch 35900 RMSE = 0.07826283070747257
... Epochs 36000 to 49000 omitted ...
Final Epoch RMSE = 0.05632031517439128
```

Discussion: The neural network makes major progress from epoch 0 (0.5775997163360249) to the final epoch (RSME: 0.05632031517439128). The data show that the network will continue its trajectory towards a lower RSME with more epochs, validating the hypothesis from the first sample run. It should be noted that, upon an ad hoc sample training run with the same parameters but with 500,001 epochs, the network requires significantly more epochs to make the same amount of progress it had initially made with less epochs.

4 Summary of Results: run_iris(), run_sin(), and run_XOR()

The two sample runs of $run\_iris()$ illustrate that increasing the number of epochs renders a lower RSME and that a higher RSME from a smaller number of epochs can be negated with a significant increase in the training factor. In the first sample run, the network failed to classify all of the testing data with a learning rate of 70%. In the second sample run, the network successfully classified most of the testing data with a learning rate of 90%.

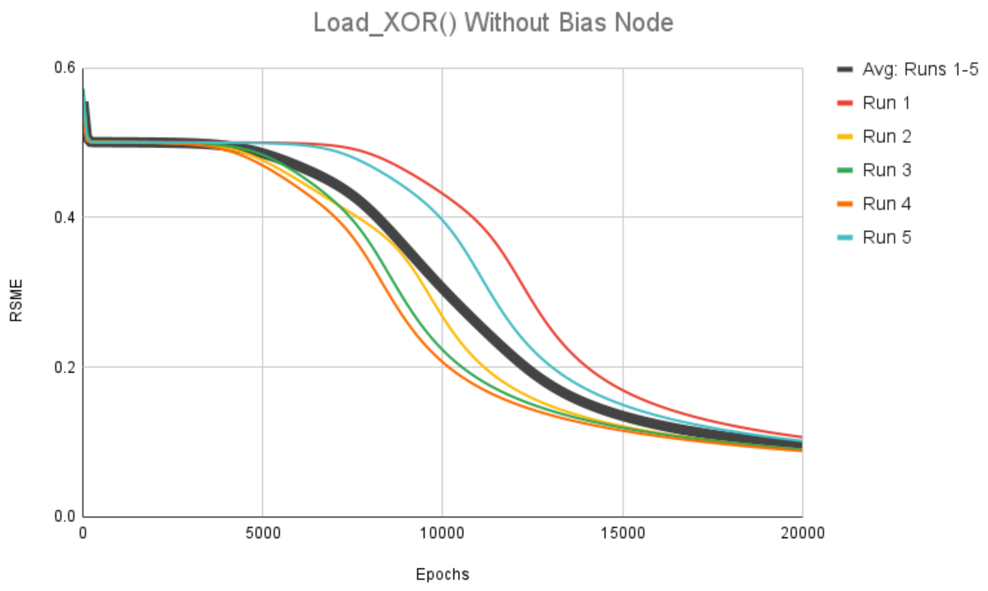
The sample run of $run\_sin()$ shows that the network failed to learn as expected and is over-fitted to the training data, as illustrated by the graph of a restricted sine wave superimposed on the inputs and outputs associated with testing.

The two sample runs of $run\_XOR()$ shows that the network will continue its trajectory towards a lower RSME with more epochs, but slows down significantly between 10001 and 50001 epochs. The network makes major progress when more epochs are included.

5 Extra Credit 1: Bias Nodes

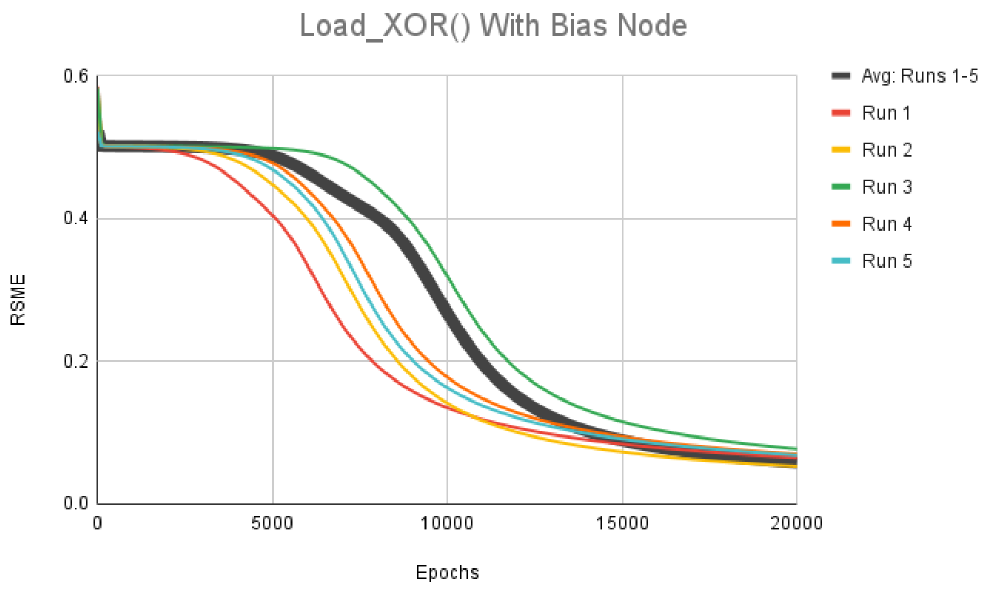
Theorem 1 (Bias Node) *If the values of input neurodes are 0, resulting in a weighted sum of inputs of 0, the first layer of a neural network will remain untrained; let the bias node be a single node that serves as a constant nonzero input for every node in a hidden layer.*

Without Bias Node, Sample Runs 1-5: hidden layer of three neurodes, 20000 epochs, 100% training factor, randomly ordered:



Discussion: Training *load_XOR* in five sample runs, without a bias node and for 20000 epochs, and averaging the RSME associated with each 100th epoch across all sample runs asserts the following: the network seems to learn slowly between its 0 and 4000th epochs, learns quickly between its 5000th and 15000th epochs, and slows back down beyond its 15000th epoch. Note that the average RSME at epoch 0 is 0.5549791965 while the average RSME at epoch 20,000 is 0.09494039341.

With Bias Node, Sample Runs 1-5: hidden layer of three neurodes, 20000 epochs, 100% training factor, randomly ordered;



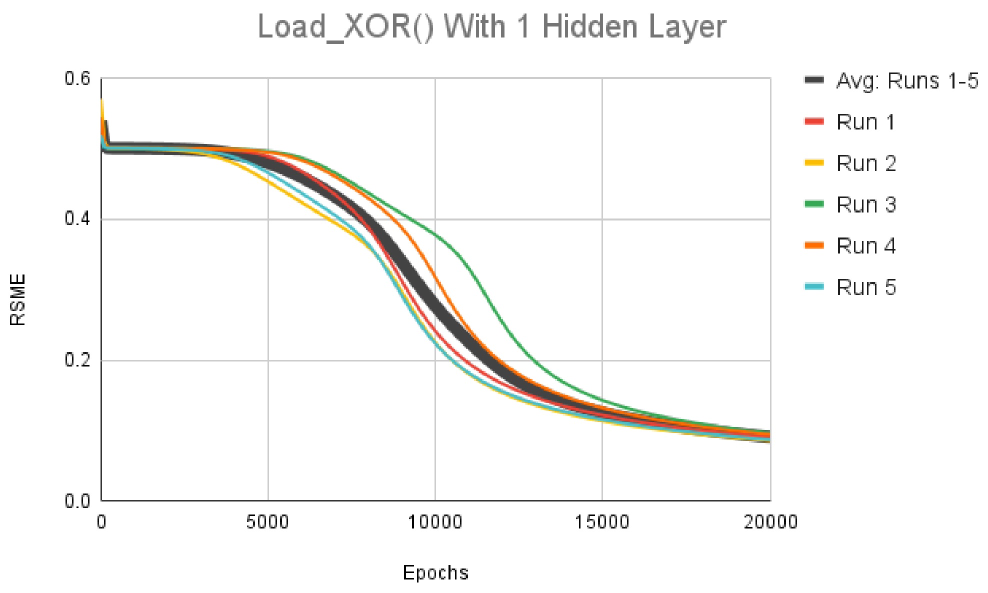
Discussion: The above visualization represents the RSME of *load_XOR*() across each 100th epoch as we train it up to 20,000 epochs; however, as compared to the visualization on page 9, we have added a third input layer (all with *features* values of 1).

Note that the average RSME at eppch 0 is 0.5218718477 while the average RSME at epoch 20,000 is 0.0576872021. Comparing the average final epoch RSME of the sample runs without the bias node, 0.09494039341, to the same metric *with* the bias node, 0.0576872021, it is apparent that adding the bias node lessens the error associated with learning. This likely occurred because the bias node enabled the network to fit the data even when the input features were equal to 0.

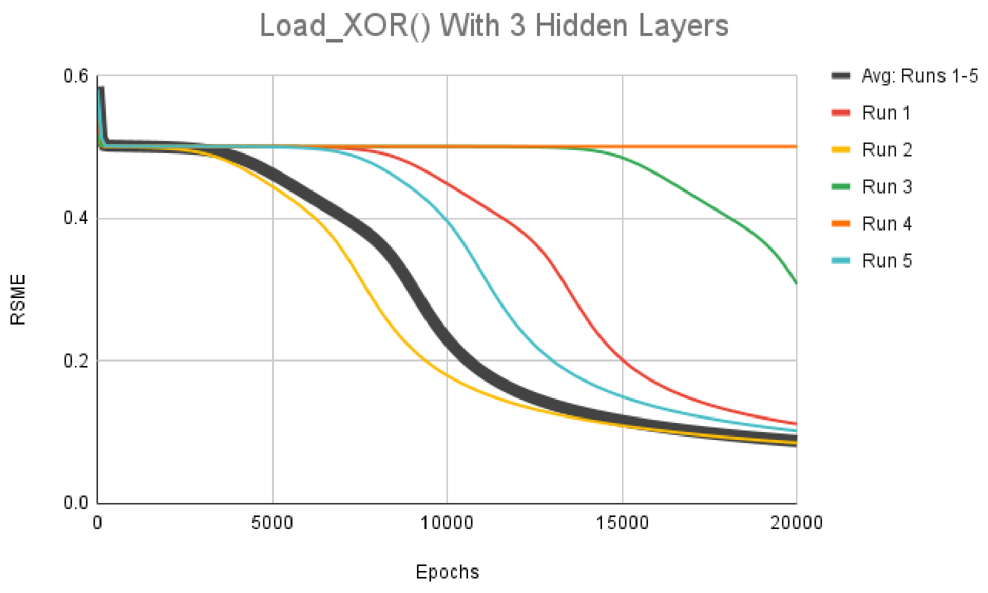
The abrupt declination of the black-colored curve in the above visualization can be attributed to the outlier run, run 3.

6 Extra Credit 2: Vanishing Gradient

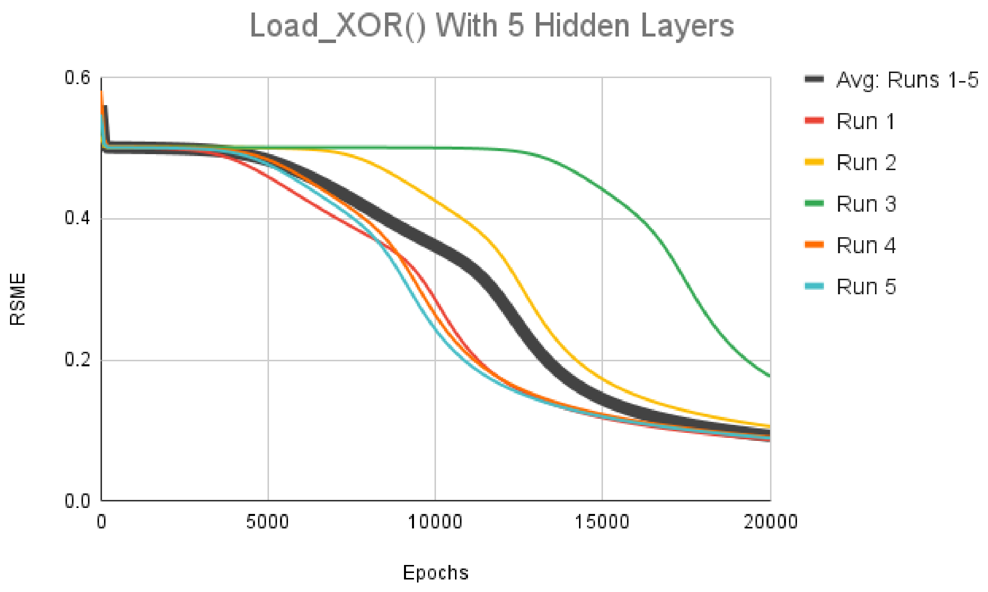
With 1 Hidden Layer, Sample Runs 1-5: hidden layer of three neurodes, 20000 epochs, 100% training factor, randomly ordered:



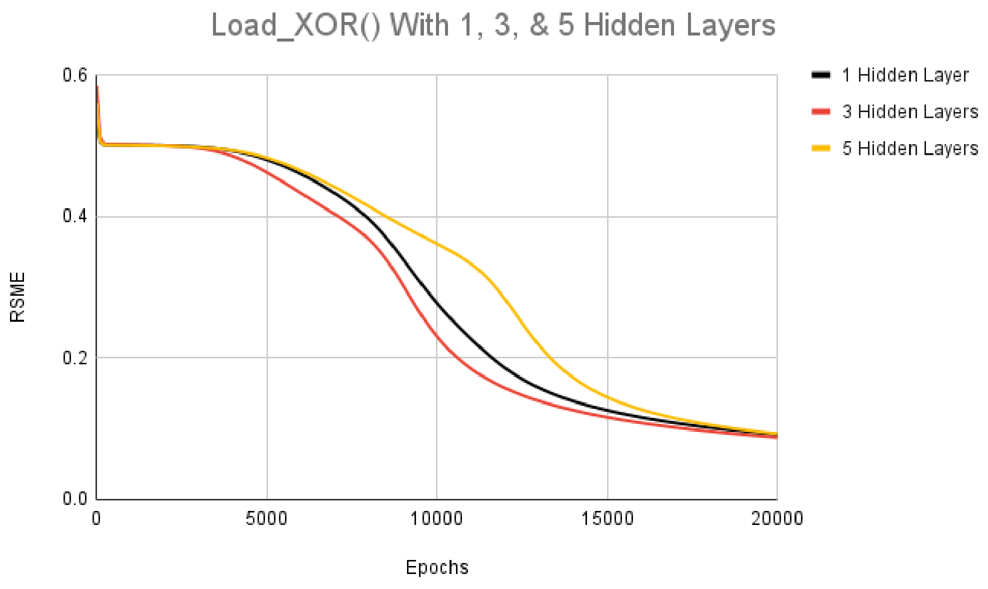
With 3 Hidden Layers, Sample Runs 1-5: 3 hidden layers of three neurodes, 20000 epochs, 100% training factor, randomly ordered:



With 5 Hidden Layers, Sample Runs 1-5: 5 hidden layers of three neurodes, 20000 epochs, 100% training factor, randomly ordered:



All Cases, Sample Runs 1-5: 1, 3, 5 hidden layers of three neurodes, 20000 epochs, 100% training factor, randomly ordered:



Summary of Results: Each of the above three visualizations represent the average RSME of *load_XOR*() across five runs with 20000 total epochs and either 1, 3, or 5 hidden layers of three neurodes.

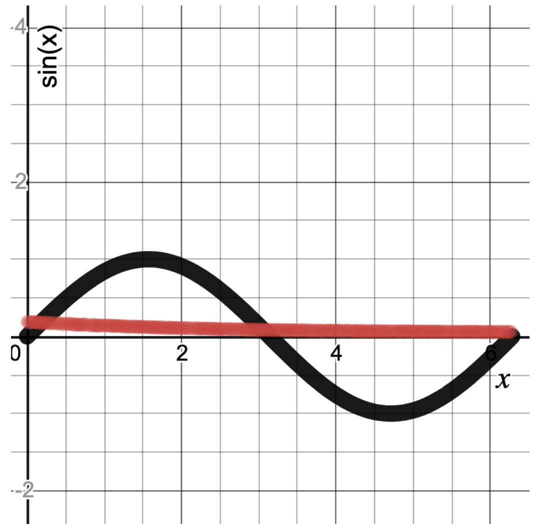
- Initial/Final RSME for 1 hidden layer: 0.539235577 and 0.09139540319.
- Initial/Final RSME for 3 hidden layers: 0.5847723894 and 0.08731580358.
- Initial/Final RSME for 5 hidden layers: 0.560100383 and 0.09247950949.

As shown by the visualization of the five sample runs with a single hidden layer, nearly all runs follow a similar learning pattern and converge to the average final RSME of 0.09139540319. In the case of 3 hidden layers, runs 2 and 3 diverge from the average of all five runs; similarly, in the case of 5 hidden layers, run 3 diverges from the average. Despite this divergence, the average final RSME for the cases with both 3 and 5 hidden layers are within 0.0011 of the same metric for the case with 1 hidden layer. That being said, it can be concluded that adding more hidden layers does not lead to performance benefits in terms of RSME; on the contrary, adding more hidden layers increases the probability of highly inaccurate sample runs. In the context of the dataset for *load_XOR*(), there is no case wherein the implementation of more than 1 hidden layer renders significant performance benefits.

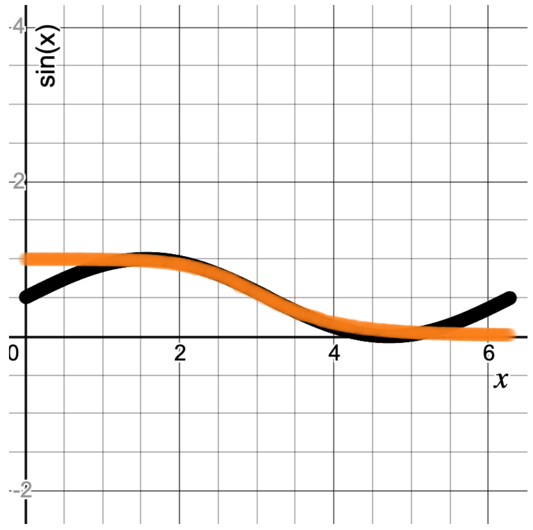
7 Extra Credit 3: Preprocessing Data

Theorem 2 (Preprocessing Data) *In the context of machine learning, the preprocessing of data can be defined as the method or mechanism (in our case, a function) by which raw data is converted into a clean data set.*

Without Preprocessing of Data, Sample Run: 1 hidden layer of three neurodes, 10001 epochs, 10% training factor, randomly ordered; *the black line represents the unprocessed sin(x) data and the unoptimized output data is represented by the red line*:



With Preprocessing of Data, Sample Run: 1 hidden layer of three neurodes, 10001 epochs, 10% training factor, randomly ordered; *the black line represents the preprocessed sin(x) data and the optimized output data is represented by the orange line*:



Summary of Results: The first visualization illustrates the testing outcome after training a network with unprocessed input data (the x and y values of a sine wave from 0 to 2π). The unoptimized output data produced by testing the network renders a result that is completely unusable; the network failed testing. There is no point at which the network seems to have learned from the input data, which is likely because of the fact that the y values of a sin wave from *pi* to 2π are negative. The network eventually passes the the negative input values through the $\_sigmoid()$ function shown below:

```
def _sigmoid(value):
    """ Keep input values bound to a known range & return
        the
    result of the sigmoid function at value.
    """
    return 1 / (1 + np.exp(-value))
```

Since $\_sigmoid()$ is bounded from 0 to 1, negative input values are not capable of being handled. It might look like the aforementioned problem can be handled by utilizing a preprocessing function that returns the absolute value of all input data, but said function would be training and testing against non-valid input data (not of a sine wave).

The function below, which returns a nested list of y values within the range of 0 to 0.5, acts as a more effective mechanism for preprocessing input data:

```
def pre_process(sin_X):
    """ Preprocess the input data from run_sin_2pi()."""
    ret_val = []
    for i in sin_X:
        ret_val.append([(0.5 * math.sin(i[0])) + 0.5])
    return ret_val
```

With the inclusion of the above function, as illustrated by the second visualization shown above, the input data is effectively preprocessed such that the sine wave maintains its shape, albeit within a different range; rather than being incompatible with *_sigmoid()*, the network tests more effectively. It should also be noted that, in spite of the relative improvements from data preprocessing, the network fails to learn near the initial and final x values.

Since the reason for these failures cannot be concluded, further testing should be done to see whether the network will encounter the same failures on a graph of the average of many runs with varying epochs and learning rates.