Prediction Intervals for Neural Networks: Investigating the Impact of Network Complexity

Richie McNamara

Bowdoin College

Date: May 21, 2023

1 Introduction

Machine learning has proven to be an extremely powerful tool in making predictions based on data. One powerful technique is using a dense neural network (DNN), which has proven to be extremely effective in unpacking data sets and understanding the underlying function. The overall goal of a DNN is to map a feature vector, $\vec{x}$, to a target outcome, y. DNNs are layers of neurons, where each neuron is connected to every neuron in the subsequent layer. Each neuron takes an input, multiplies by a weight, adds a bias, and applies an activation function to produce an output. The activation function (e.g., sigmoid, or ReLU) allows for non-linearity and for the output to be in a desired range. In a DNN, the first layer is the input layer, where the number of neurons is equal to the number of features for the specific data set. The last layer is the output layer, which contains the prediction of the network. All layers in between are considered hidden layers. To train and evaluate a neural network to fit a data set, it is common practice to split the available data into a training set and a test set. The DNN uses the training set to adjust the weights and biases to minimize a loss function (e.g., mean squared error). The test set, which represents unseen data points, is used to evaluate the DNN's performance and assess if it can generalize to new data.

In a regression task, a DNN is attempting to predict a continuous value for the target outcome. Given a trained DNN, a new $\vec{x}_{\text{test}}$ feature that the network has not seen before can be passed into the network, which outputs a predicted $f(\vec{x}_{\text{test}})$ value. In a regression problem, the probability that $f(\vec{x}_{\text{test}}) = y_{\text{test}}$ is zero. While this point prediction may provide insights, it does not convey the uncertainty associated with the prediction. Therefore, it would be valuable to construct a prediction interval (PI) to quantify the range of possible values within which the true target value is likely to lie. It is also necessary to involve some level of confidence in our interval. A $(1 - \alpha)$ PI is one in which the probability that y_{test} is in the PI is $(1 - \alpha)$.

A set of PIs is evaluated on two metrics: width and coverage. The width of a PI is the range between the upper and lower bounds. The coverage of a set of PIs is the percentage of point predictions that fall within the constructed PIs. Due to the construction of the PIs, it is expected to have a coverage that equals the confidence level. Both metrics should be considered in tandem when evaluating PIs. Ideally, PIs should have the smallest average width possible, but not at the expense of a low coverage. A narrow width and high coverage mean the PIs are effectively capturing the true target values.

The complexity of a DNN is simply the number tunable parameters, which is mainly affected by the number of layers and neurons. A well-studied area is how increasing the complexity of a DNN can impact its generalization and predictability. However, the relationship between complexity and the construction of PIs has not been fully explored. This project seeks to research the relationship between the complexity of a DNN and the evaluation of its PIs. This project will be looking at five different data sets, different DNNs of different complexities, and evaluating the PIs that are constructed for each.

2 Constructing A Prediction Interval

There are many approaches to determining a prediction interval, such as Bayesian inference or conformal inference, but using bootstrapping provides a data-driven approach. Bootstrapping is a resampling process that attempts to infer the true value of a statistic or estimator from a sample of the true population. The process requires randomly sampling from the original data set with replacement. Suppose there exists a random data sample of size N, $X = \{x_1, \ldots, x_N\}$, from a population with an unknown distribution. In order to estimate the true value of some statistic $S = s(X)$, take B bootstrapped resamples of X, $\{X_1, \ldots, X_B\}$, where each X_i is a random sample of size N drawn from X with uniform probability. A distribution of S is then $\{s(X_1), \ldots, s(X_b)\}$. The collection of each X_i is meant to capture the underlying distribution of X. It is convention to use a very large number of bootstrapped resamples, with B of at least 1000.

Bootstrapping has been applied to machine learning before in the form of bootstrap aggregation (bagging). The approach is to train B instances of a learning algorithm on different bootstrapped resamples of the training data. The collection of trained models, $F = \{f_1, \ldots, f_B\}$, is called an ensemble. When querying $F(\vec{x}_{\text{test}})$ on the ensemble, the empirical distribution of y_{test} is $\{f_1(\vec{x}_{\text{test}}), \ldots, f_B(\vec{x}_{\text{test}})\}$. The prediction $F(\vec{x}_{\text{test}})$ can then be an aggregate function, such as the mean or mode, of the empirical distribution. The ensemble generally improves predictions since it produces many models that are weakly correlated with one another. Therefore, the aggregation of the empirical distribution will have lower variance.

There are different ways to apply bootstrapping to prediction intervals (PIs), but the method used in this project is the pivot method. The pivot bootstrap method assumes that the distribution of $\{y_{\text{test}}\}$ is approximately normal. The method requires the following inputs: training data $\{(\vec{x}, y)\}$, a new observation $(\vec{x}_{\text{test}}, y_{\text{test}})$, a DNN f, B number of bootstrap resamples, and confidence $1 - \alpha$. First, create an ensemble of B DNNs, $F = \{f_1, \ldots, f_B\}$, trained using $\{(\vec{x}, y)\}$. Using the mean as the aggregate function in the ensemble, calculate the point estimate, $F(\vec{x}_{\text{test}})$. Estimate the component model-fitting error as follows.

$$\sigma^2(\vec{x}_{\text{test}}) = \frac{1}{B-1} \sum_{i=1}^{B} (f_i(\vec{x}_{\text{test}}) - F(\vec{x}_{\text{test}}))^2$$

Then estimate the irreducible error using "out-of-bag" sets as follows. For a given bootstrapped resample, consider all of the data points that are in the original $\{(\vec{x}, y)\}$, but not in the resample, denote this $X_{\text{o}} = \{(\vec{x}_{\text{o}}, y_{\text{o}})\}$. Let N_{o}, be the size of the out-of-bag set. For each bootstrapped resample, X_i, corresponding f_i, and out-of-bag set $X_{\text{o}}^i = \{(\vec{x}_{\text{o}}^i, y_{\text{o}}^i)\}$, the irreducible error is as follows.

$$\sigma^2_{\epsilon(i)} = \frac{1}{|N_{\text{o}}^i - 1|} \sum_{X_{\text{o}}^i} (y_{\text{o}}^i - f_i(\vec{x}_{\text{o}}^i))^2$$

The irreducible error estimate, σ^2_ϵ, is then the average of all the irreducible error for each bootstrapped resample.

$$\sigma_\epsilon^2 = \frac{1}{B}\sum_{i=1}^{B}\sigma_{\epsilon(i)}^2$$

Finally, the pivot bootstrap PI is constructed using the $(1-\alpha)$ percentile of the standard normal distribution.

$$\left[F(\vec{x}_{\text{test}}) - z_{1-\alpha/2}\sqrt{\sigma^2(\vec{x}_{\text{test}}) + \sigma_\epsilon^2}, F(\vec{x}_{\text{test}}) + z_{1-\alpha/2}\sqrt{\sigma^2(\vec{x}_{\text{test}}) + \sigma_\epsilon^2}\right]$$

It is worth noting that the Student's t distribution is typically used instead of the standard normal, however, there has been research regarding the difficulties of determining the degrees of freedom for a neural network. Due to that limitation, the standard normal distribution is used. There has also been research illustrating that the pivot bootstrap method constructs PIs that are more conservative, meaning the widths could be smaller while still maintaining coverage.

3 Experiments

This research project conducts two main experiments using five data sets to investigate two major problems. Can bootstrapping accurately construct PIs for DNNs? Does the level of complexity of a DNN impact the effectiveness of PIs? The first experiment considers synthetic data coming from two different known functions, seeking to answer both questions. The second experiment considers the well-known Boston Housing data set, seeking to apply the second question to real world data.

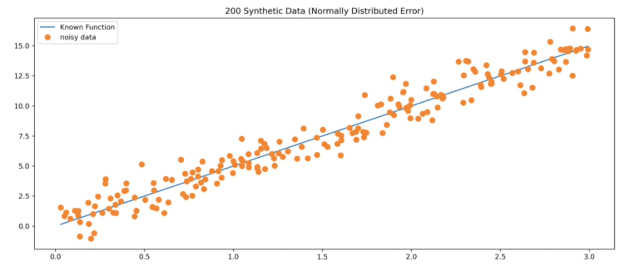
Both of the main experiments will consider three DNNs of varying complexity. PIs will be constructed as shown above and evaluated using the two metrics described. The remaining of this section will be used to describe the experiments in detail.

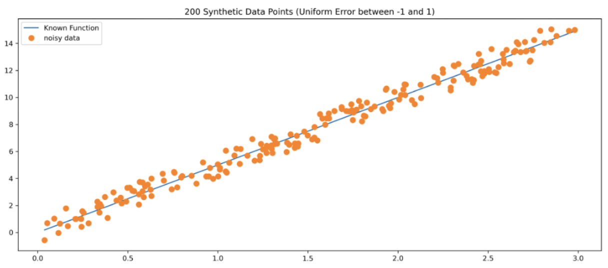
3.1 Synthetic Data

In this experiment, data points will be generated with a known error function, and since the error function is known, there is a known model for PIs. There will also be three different DNNs (simple, medium, and complex) that the data set is used to train and generate prediction intervals using the method above. The predicted interval and the known interval will then be compared. The relationship between the PIs and complexity will also be examined.

3.1.1 Linear Synthetic Data Sets

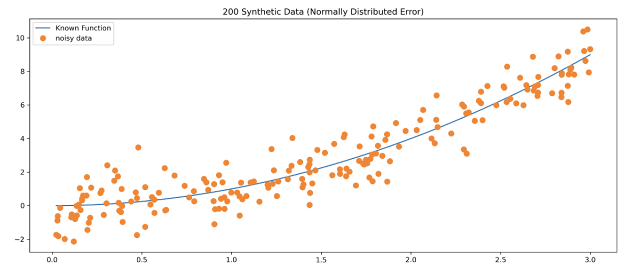
The first batch of synthetic data will come from a simple linear function, $f(x) = 5x$ for 200 random x values between 0 and 3. When generating synthetic data, two different error models will be considered. The first will add error sampled from a standard normal distribution. The second will add error sampled from a uniform distribution between -1 and 1.

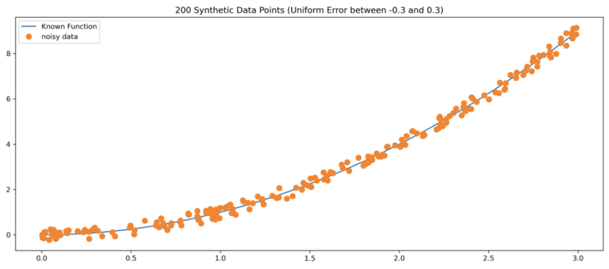




3.1.2 Non-linear Synthetic Data Sets

A second batch of synthetic data will be generated from the quadratic function $f(x) = x^2$ for 200 random x values between 0 and 3. The first error model will be sampled from a standard normal distribution. Whereas, the second will add error sampled from a uniform distribution between -0.3 and 0.3.





3.1.3 Synthetic Data DNNs

Three different DNNs, each of which has a different complexity will be used to learn the underlying function for each of the four synthetic data sets. All of the networks will be trained using stochastic gradient descent for 100 epochs with a batch size of 32 and a mean squared error (MSE) loss function. All of the activation functions will be the ReLU function. The following table summarizes the number of neurons in each layer of the DNNs.

DNN Name	Input Layer	Hidden Layer 1	Hidden Layer 2	Output Layer
Simple Syn	1	2		1
Med Syn	1	10		1
Complex Syn	1	10	5	1

3.1.4 Experiment 1

The first experiment will consider each of the four data sets described so far. Each data set will be split into training and testing sets, with 60% of the data in the training set. For each of the three DNNs, 1000 instances of the DNN will be in an ensemble. The ensemble will be trained and a 95% confidence ($\alpha = 0.05$) PI will be constructed as explained above for each of the testing data points. The average, maximum, minimum, and standard deviation of the PI widths, along with the coverage will be recorded. The mean squared error of the test prediction and the known test labels will be recorded as well.

3.2 Boston Housing Data

In this experiment, PIs will be constructed for the Boston Housing data set for three different DNNs. Since the true interval is unknown, the relationship between the DNN complexity and PI metrics will be studied. The PIs produced from the experiment will be compared to other researchers' intervals.

3.2.1 Data set

This benchmark data set from 1978 provides information on various features related to housing in areas around Boston. There are 506 data points, for different suburbs near Boston. For each suburb, there are 12 features and one label, the median house price.

3.2.2 Boston Housing Data DNNs

Three more DNNs with different complexity levels will be used to make predictions for the Boston housing data. All of the networks will be trained using stochastic gradient descent for 100 epochs with a batch size of 32. All of the activation functions will be the ReLU function. The following table summarizes the number of neurons in each layer of the DNNs.

DNN Name	Input Layer	Hidden Layer 1	Hidden Layer 2	Output Layer
Simple BH	12	16		1
Med BH	12	16	8	1
Complex BH	12	128	64	1

3.2.3 Experiment 2

The second experiment will split the Boston housing data set into a training and test set, where 70% of the data is in the training set. All three of the DNNs will have their own ensembles with a size of 1000 and trained on the training data. A 95% confidence ($\alpha = 0.05$) PI will be constructed as explained above for each of the testing data points. The average, maximum, minimum, and standard deviation of the PI widths, along with the coverage will be recorded. The mean squared error of the test prediction and the known test labels will be recorded as well.

4 Results

This section details the results from the experiments described above. First, the two linear synthetic data sets PIs will be examined and compared to the true error intervals for each DNN. Next, the two non-linear synthetic data sets PIs will explored along with their relation to the true intervals for each DNN. Then, the PIs for each DNN for the Boston Housing data set will be shown and compared to previous research. Finally, the relation between DNN performance and prediction precision will be discussed.

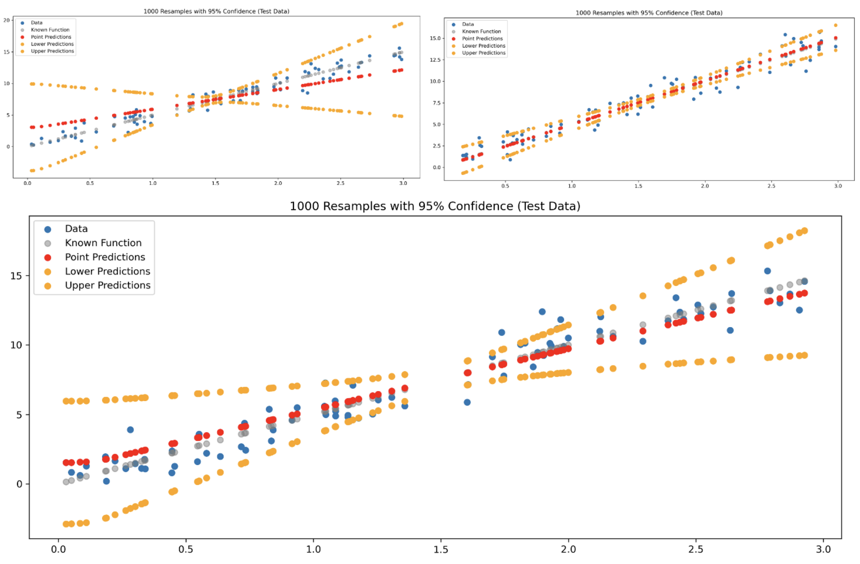
4.1 Linear Synthetic Data

Using the pivot bootstrap method, the PIs for the test data sets evaluated on the three different DNNs are shown in the following table.

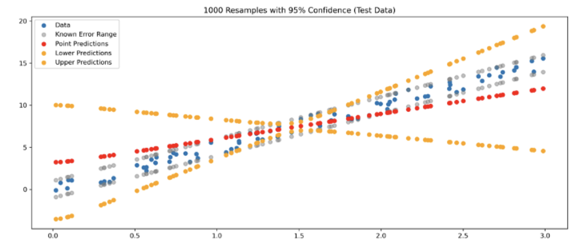
NORMAL ERROR						
Model	**Avg. Width**	**Max. Width**	**Min. Width**	**Std. Width**	**Coverage (%)**	**Test Data MSE**
Simple Syn	7.14	14.66	1.07	4.06	90.00	2.93
Med Syn	1.71	3.09	0.65	0.79	51.25	1.06
Complex Syn	5.09	8.97	1.73	2.30	87.50	1.36

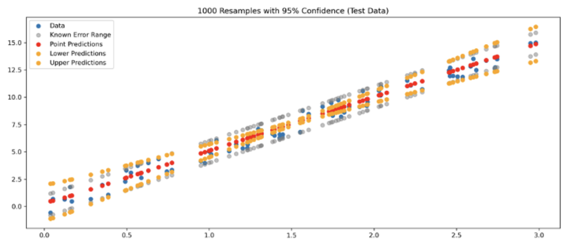
UNIFORM ERROR						
Model	**Avg. Width**	**Max. Width**	**Min. Width**	**Std. Width**	**Coverage (%)**	**Test Data MSE**
Simple Syn	7.39	14.82	0.98	3.98	93.75	3.58
Med Syn	1.47	3.20	0.37	0.87	62.50	0.35
Complex Syn	4.80	8.91	1.20	2.32	96.25	0.73

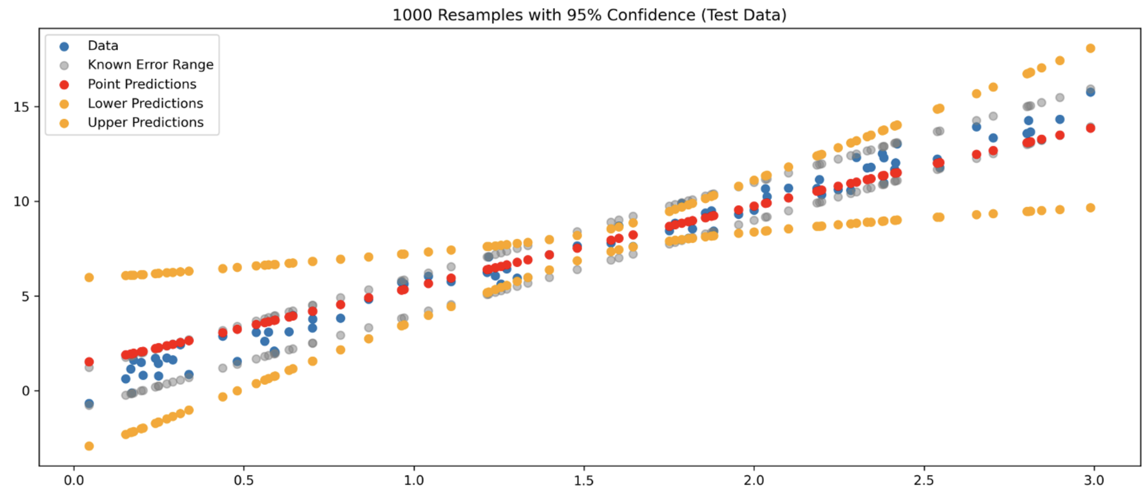
The estimated PIs can be plotted along with the data points and the point estimates. The following plots display the results for the linear synthetic data with normal error. The top two plots from left to right are using Simple Syn and then Med Syn. The bottom plot is using Complex Syn. All three plots use the same legend.



The following plots display the results for the linear synthetic data with uniform error. The top two plots from left to right are using Simple Syn and then Med Syn. The bottom plot is using Complex Syn. All three plots use the same legend.







4.1.1 Interpreting Results

The normal and uniform error cases had very similar results both quantitatively and qualitatively. For the remainder of this section, the results will apply to both data sets, unless otherwise noted.

The PIs seem to "pinch" towards the middle of the x value range, as seen in the plots above. This result is expected since each DNN is using mean squared error as a loss function. In order to minimize loss, the model will be more likely to predict a y value close to the mean, instead of taking a big risk and predicting a value on the outer edge. It is also noted that the degree to which the PIs "pinch" is directly related to the test MSE. If the test MSE is low, then the pinching is lower, which can also be seen by the smaller standard deviation of the PI widths.

The Simple Syn DNN produced PIs with a decently high coverage, but still less than the expected value of 95%. However, the average widths were higher than any other DNN. This means that the DNN is not confident in its predictions and is over estimating the error function. The Med Syn DNN produced the smallest average widths, but the coverage was not maintained at all. This DNN is confident in its predictions, but does not capture the overall error function, as a PI should. The Complex Syn DNN's results diverge for the normal and uniform cases. For uniform error, it seems to strike a balance between the other two DNNs and generate more accurate PIs. It is able to have coverage of 96.25%, which is close to 95%. This result also aligns with the prior research noted above claiming that the pivot bootstrap method creates more conservative PIs. It has a smaller average width than Simple Syn, while still maintaining coverage. The normal error saw smaller widths than Simple Syn, but also less coverage. In all cases, the lowest MSE does not correspond to the most complex DNN nor the most effective PIs. The Med Syn DNN has the lowest test MSE, but as noted above does not maintain coverage for its PIs.

Since the normal error is coming from a standard normal distribution, 95% of the error samples will be between -1.96 and 1.96. This means a 95% confidence interval should have a width of 3.92. The Complex Syn DNN had an average width of 5.09, which is decently close to the true interval. The uniform error is coming from a uniform distribution between -1 and 1, so 95% of the samples will fall between -0.95 and 0.95. A 95% confidence interval should have a width of 1.9. The complex Syn DNN had an average width of 4.8, which is over double the interval. It seems that the true intervals were not able to be completely reconstructed, however this could be because the DNNs are not fully optimized to model the functions or because the pivot bootstrap method has its limitations.

4.2 Non-linear Synthetic Data

The same process as above is now applied to the non-linear synthetic data sets. The following table outlines the PI statistics.

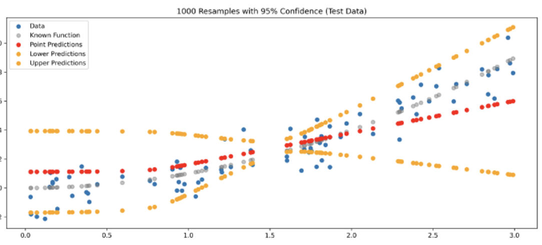
NORMAL ERROR

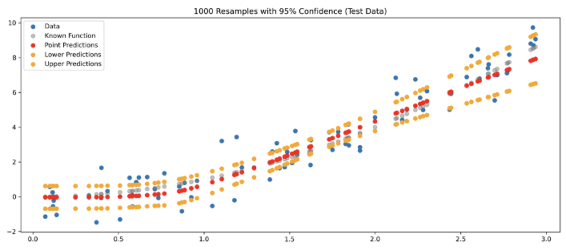
Model	Avg. Width	Max. Width	Min. Width	Std. Width	Coverage (%)	Test Data MSE
Simple Syn	4.66	10.21	0.90	2.44	81.25	2.40
Med Syn	1.42	2.82	0.84	0.54	55.00	0.85
Complex Syn	2.93	6.40	0.98	1.33	85.00	0.99

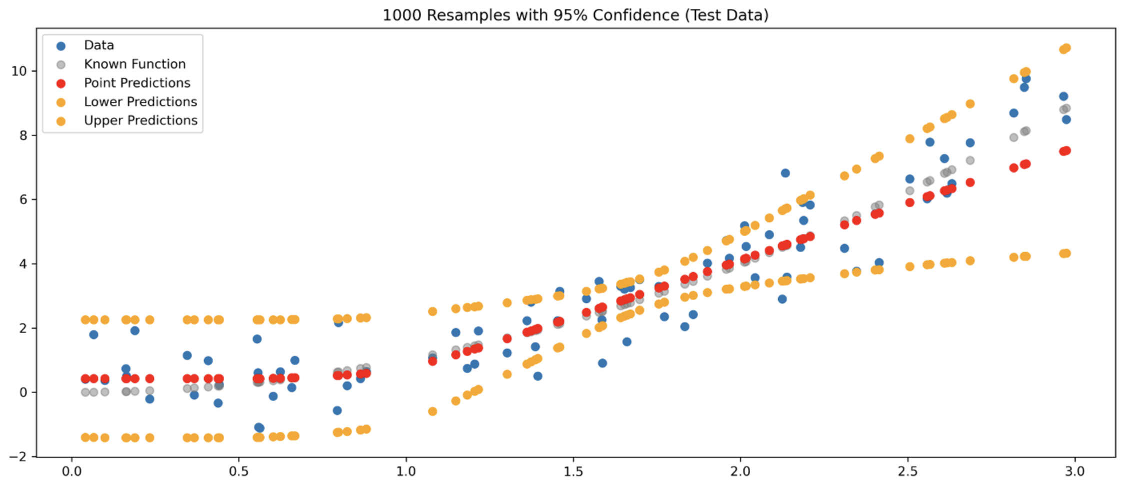
UNIFORM ERROR

Model	Avg. Width	Max. Width	Min. Width	Std. Width	Coverage (%)	Test Data MSE
Simple Syn	4.78	9.37	0.66	2.35	91.25	1.33
Med Syn	0.98	1.93	0.21	0.45	67.50	0.21
Complex Syn	3.07	7.01	0.60	1.68	95.00	0.24

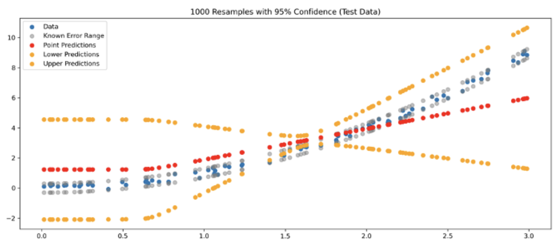
The following plots of the PIs are for the non-linear synthetic data with normal error. The top two plots from left to right are using Simple Syn and then Med Syn. The bottom plot is using Complex Syn. All three plots use the same legend.

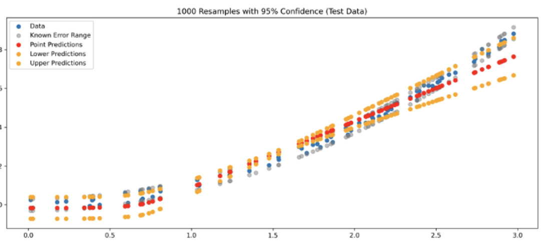


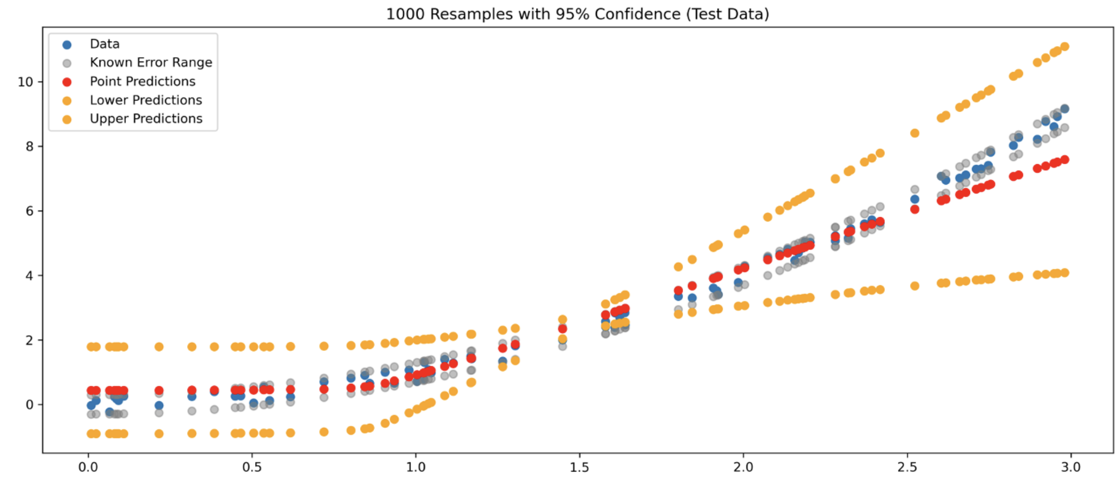




The following plots of the PIs are for the non-linear synthetic data with uniform error. The plots are in the same ordering as before.







4.2.1 Interpreting Results

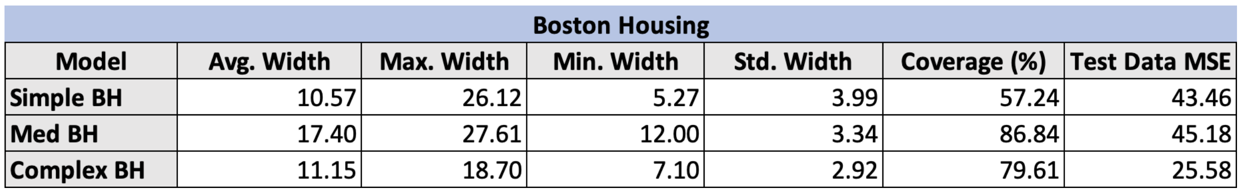
For the non-linear data sets, the normal and uniform error cases had very similar results both quantitatively and qualitatively. For the remainder of this section, the results will apply to both data sets, unless otherwise noted.

The non-linear case had the same “pinching” effect as the linear case. This is expected due to the same MSE loss function for the DNNs. It also had the same qualitative results for the relationship between complexity and the effectiveness of the PIs. The Simple Syn DNN had the largest widths and decent coverage. The Med Syn DNN had the smallest widths but was lacking coverage severely. The Complex Syn DNN was able to lower the widths from the Simple Syn and also gain coverage. Overall, the more complex DNN created more effective PIs. It is important to note that for the uniform error case, the Complex Syn DNN had exactly 95% coverge, which aligns with the 95% confidence interval. It would be interesting to see if a more complex DNN could be constructed to lower the width more, and maintain the 95% coverage.

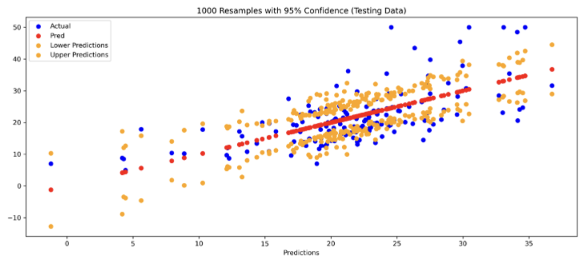
The actual PI for the normal case is also between -1.96 and 1.96, with a width of 3.92. The Complex Syn DNN had a average width of 2.93, which is smaller and makes sense since the coverage is lower than 95%. The uniform error is coming from a distribution between -0.3 and 0.3, so 95% of the samples will be between -0.285 and 0.285. The true width is 0.57 and the Complex Syn DNN had an average width of 3.07, which is very large compared to the true interval. Again, it seems that the true intervals were not able to be created and this could be because the DNNs are not complex enough to capture the non-linearity of the function or the pivot bootstrap method is not suitable.

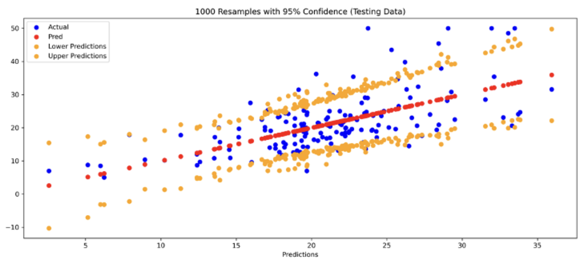
4.3 Boston Housing Data

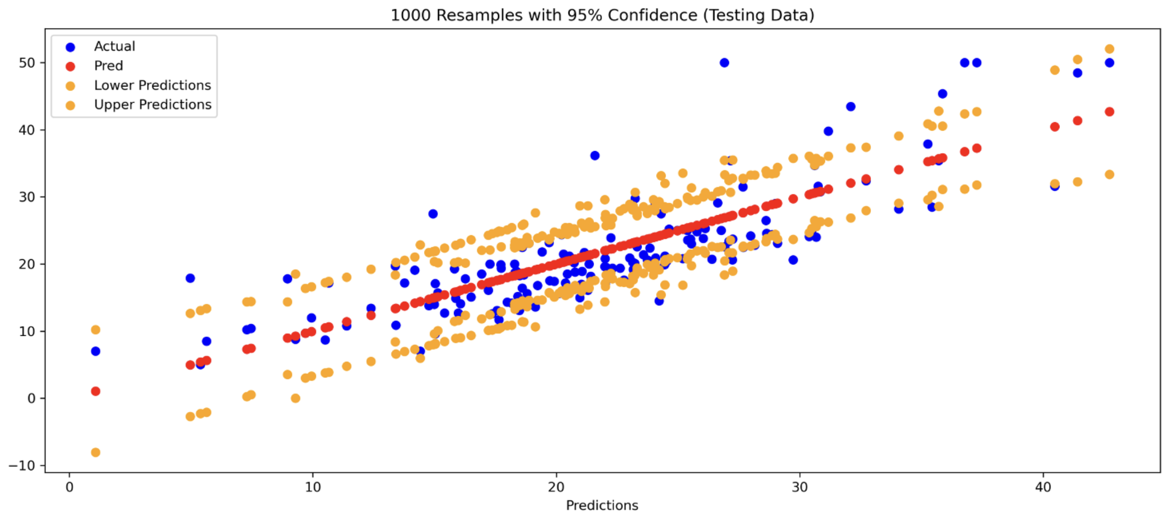
Finally, the Boston Housing experiment results are in the table and plots below. The top two plots from left to right are using Simple BH and then Med BH. The bottom plot is using Complex BH. All three plots examine the predicted y label compared to the point estimate, actual y_{test} point, the upper bound of the PI, and the lower bound of the PI.



Boston Housing						
Model	**Avg. Width**	**Max. Width**	**Min. Width**	**Std. Width**	**Coverage (%)**	**Test Data MSE**
Simple BH	10.57	26.12	5.27	3.99	57.24	43.46
Med BH	17.40	27.61	12.00	3.34	86.84	45.18
Complex BH	11.15	18.70	7.10	2.92	79.61	25.58







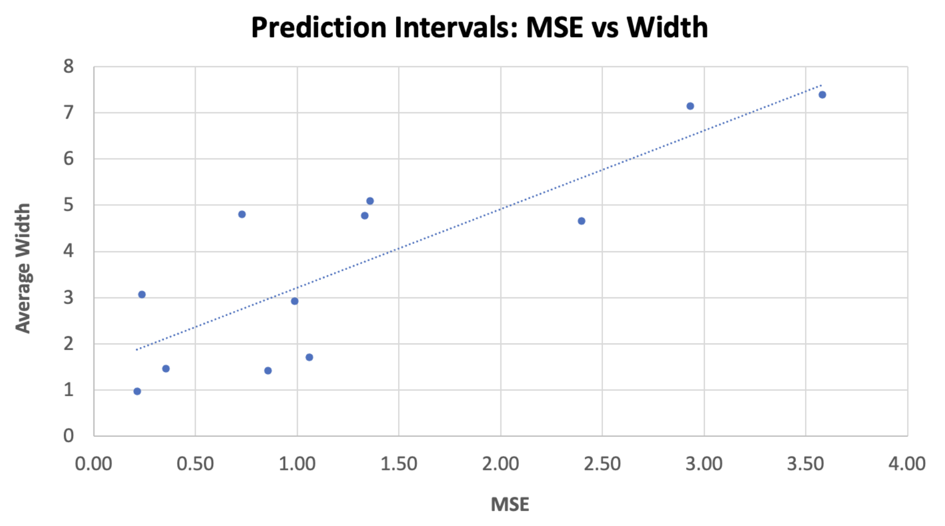
4.3.1 Interpreting Results

The plots show that as the complexity of the DNN increases, the variation of the PIs decreases. This can also be seen by examining the standard deviation of the widths. The Simple BH DNN had the lowest average width, but also the lowest coverage, signalling that the interval is too narrow. The Med BH DNN had the highest average width but also the highest coverage. The Complex BH DNN had slightly wider intervals compared to the Simple BH DNN, but improved its coverage by 20 percentage points. None of the DNNs were able to create an interval with 95% coverage, which means either the pivot bootstrap method is not suitable for this data set, or the DNNs were not complex enough.

Prior research (Contarino et al., 2022) has been done with constructing PIs using DNNs for the Boston Housing data set. Their research used the pivot bootstrap method with 1000 resamples and resulted in an average interval with a width of 16.2 and coverage of 98%. The Med BH DNN had a similar width, but resulted in a coverage of 86.84%. They did use different neural networks in their research, which might cause the PIs to improve.

4.4 More Results

The following is a plot of all the synthetic DNNs mean squared error when trained and tested on all the synthetic data sets against the corresponding average PI width.



There is a clear relationship between MSE of a model and the average width of the PIs produced. As the MSE decreases, the PIs average widths also decrease, indicating that the model is more confident in its point prediction. However, as we have seen in the results before it is also important to make sure PIs maintain coverage. It is not the case that decreasing MSE causes coverage to increase, as seen with the Med Syn DNN. This result implies that when designing a DNN, it might not always be the best case to minimize MSE if the goal is to create intervals for the predictions.

Another interesting result is how the pivot bootstrap method was able to create intervals with around 95% coverage for the synthetic data with uniform noise, but not for the data with normal noise. One of the assumptions for the pivot bootstrap method is that the distribution of the predictions is approximately normal. It would be expected then that the PIs for the data with normal error would be better than the data with uniform error. In both the linear and non-linear cases, Complex Syn DNN constructed PIs with higher coverage for the uniform error. The PIs also had very similar average widths.

5 Conclusions

This project set out to understand if bootstrapping can accurately construct PIs and whether the level of complexity of a DNN impacts the effectiveness of a PI. The experiment with synthetic data shows that the exact known error intervals were not able to be reconstructed, however, increasing complexity did result in smaller intervals that maintained coverage. It is clear that the more complex a DNN is, the better the PIs become. It is important to always consider both width and coverage at the same time, because increasing complexity may decrease the width, but the coverage may be significantly lower. It is important to create a deep-enough DNN with enough neurons to accurately understand the error interval. The results in this paper coincide with the results from other research (Contarino et al., 2022) as they also concluded that complexity impacts width and coverage.

Future work in this area could focus on creating a DNN with optimal hyperparmeters to learn a known function from synthetic data and use other bootstrapping methods to see if the known interval can be reconstructed in that case. With limited time and resources, not all DNNs could be examined in this project. It would also be valuable to see if different levels of confidence can be reconstructed as well, as only 95% were examined in this project.