Predicting Rain Attenuation for Satellite Signals Based on Publicly Available Rainfall Data

David M Vermillion

6 March 2023

Table of contents

Abstract

Predicting attenuation from hourly rainfall on satellite signals would improve response times and operator workload allocated to boost signal strength when affected by rain, improving user experience. This paper examines a composite dataset from the National Climactic Data Center representing 61 original variables engineered into 92 predictor variables supporting the precipitation parameter, spread across 8,877 observations.

This paper examines 8 regression models: a baseline, 3 classical, and 4 neural network models. The training dataset had a precipitation mean of 0.1485 $\frac{mm}{hr}$. The Mean Squared Error (MSE) for the trivial baseline model was 0.8898 $(\frac{mm}{hr})^2$. The best-performing model was a deep neural network model with 10,177 trainable parameters, one input layer, 3 hidden layers, including a dropout layer, and one output layer. It was also trained with early stopping. This model's MSE was 7.4048 $(\frac{mm}{hr})^2$. It showed the most consistent residual plots out of all examined models yet still failed to provide an operationally valuable model. Future efforts to provide a meaningful model include different data aggregation methods, consulting with a meteorologist, and switching from a regression approach predicting the amount of rainfall to a classification approach predicting whether or not rain is likely.

Business Understanding

Problem Motivation

The Fourth Satellite Operations Squadron operates military satellite communication systems in the Super High Frequency (SHF) and Extremely High Frequency (EHF) bands. Knowing when rainfall is likely to be heavy enough to impact those signals would be helpful to users and satellite operators to predict when extra power may be required for operations.

Mission Understanding

In order to understand the situation, one must understand how radio waves interact with water droplets using formulas found in the International Telecommunications Union (ITU) recommendation P.838-3, supplemented by P.618-10 and P.840-8 [1]–[3].

Required Background

Radio waves are part of the electromagnetic (EM) spectrum. The same EM spectrum that makes up light. Rainbows appear when visible light is diffracted by water. Water also diffuses light enough that far enough down into the ocean, light no longer reaches through. That same diffraction affects radio waves when they encounter water in the atmosphere as clouds or rain. Water will sufficiently diffract those radio waves at appropriate wavelengths to produce a noticeable drop in signal strength. These effects are particularly pronounced with the volatile gasses and solid particles in rocket plumes [4].

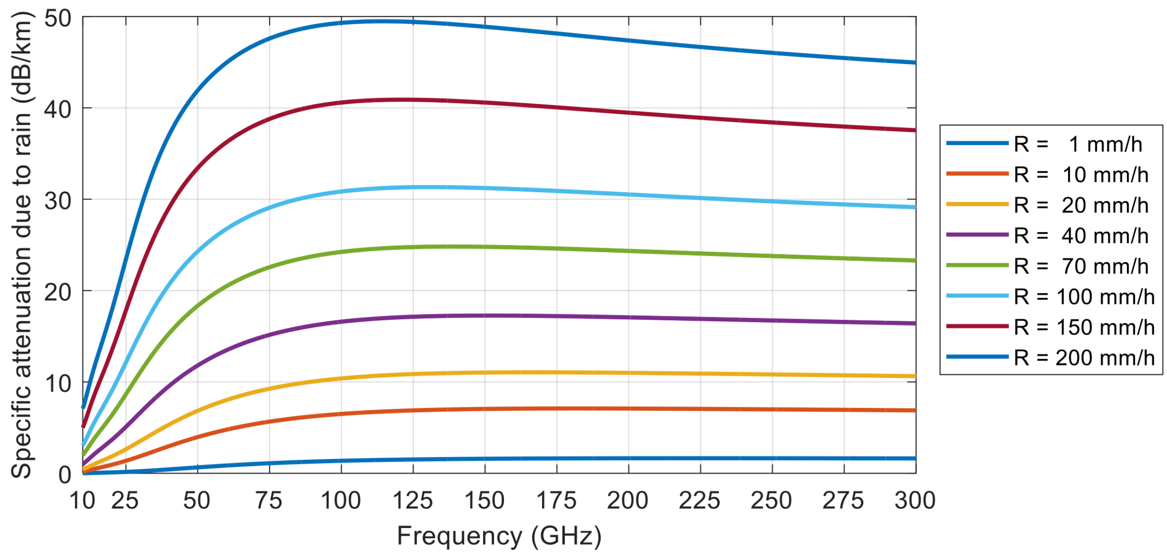


Figure 1: Frequency Attenuation by mm of Rainfall per Hour Shows Proportionally Higher Effects with Increasing Frequency [5]

At a sufficient level, this drop in signal strength makes it harder to receive enough signal for communications to be effective. The exact drop that makes the signal unusable depends on factors such as rain amount, radio frequency, background noise, the size of the antennas, and the power levels those antennas can produce. A recent study by the Politecnico Di Milano, on behalf of the Air Force Research Laboratory (AFRL), exhaustively examined the requirements to account for atmospheric effects on radio communication with satellites, with an eye toward high-speed commercial systems operating above EHF frequencies. However, despite the higher target frequency, their efforts included SHF and EHF calculations, which will be useful for modeling efforts, focusing only on rainfall [5]. In fact, one of the figures in their report, Figure 1, demonstrates how much frequency impacts attenuation by rainfall per hour.

The National Weather Service archives recorded weather at ground stations and from weather satellites. Some ground sites sample every hour. This granularity should provide enough variables and samples to predict when weather systems will likely produce enough precipitation to affect signal strength by an arbitrary amount. Rain is the most apparent form of precipitation. However, sleet, hail, and snow also produce a signal attenuation effect.

Research Question

Given hourly collection samples from ground sites, which variables predict the amount of hourly precipitation? Further, can a more advanced machine learning model more accurately predict when a location will receive sufficient precipitation to cause an arbitrary drop in signal strength, as compared to a conventional regression model? The compared models will predict precipitation amounts. Signal drops based on the precipitation will use the formula found in ITU recommendation P.838-3 [1].

The frequencies plugged into the models for comparisons include the upper and lower bounds of SHF and EHF – specifically, 3, 30 (shared between SHF and EHF), and 300 GHz.

The model will be trained, evaluated, and tested with data from the metropolitan areas of Seattle, Denver, Phoenix, Boston, and Miami to account for geographic differences in weather patterns.

This study will look for drops (signal attenuation) in those frequencies of 25% and 50% at 50 dBm or 100 W between a ground station and a satellite located directly overhead at an altitude of 35,000 km, which is roughly the altitude from Earth's surface to a geostationary orbit (GEO). Besides choosing reasonable arbitrary numbers to start, the model should be robust enough to plug in the distance, transmitting power, frequency, and desired maximum attenuation to determine the likelihood of precipitation reaching that threshold, given the current weather conditions. A future iteration should be made further robust by accounting for additional atmospheric effects.

Success Criteria

Given the five separate regions, a model predicting hourly precipitation up to 12 hours in the future to match the 50% attenuation rate with an accuracy >60% and an accuracy >80% for an attenuation of 25% will be considered adequately successful to be operationally relevant. Further, if the machine learning model has an accuracy improvement of >5% over a classical model, that will be considered more valuable than the classical model.

Data Collection

The National Weather Service has multiple data access methods to obtain archival weather observations, including APIs and download options. Depending on the complexity of accessing the relevant API, that would be the preferred option to deploy a model. Otherwise, downloads will be used for this stage, accessed from the National Climactic Data Center (NCDC) [6].

The first examination will use downloaded CSVs for hourly precipitation and ground weather observations for 2010 in Seattle, Denver, Phoenix, Boston, and Miami. Significant data preparation will be required to align the precipitation measurements with the other measurements because the observation counts do not match. The data documentation for both datasets are also hosted on the NCDC [7], [8].

Data Understanding

Visuals

The dataset has some unusual non-linear patterns in its data distributions. For example, Figure 2 looks similar to a horse head. Examining both axes as q-q plots and histograms show that the data is not normally distributed. Most of the data is non-normal, which could present some challenges for modeling.

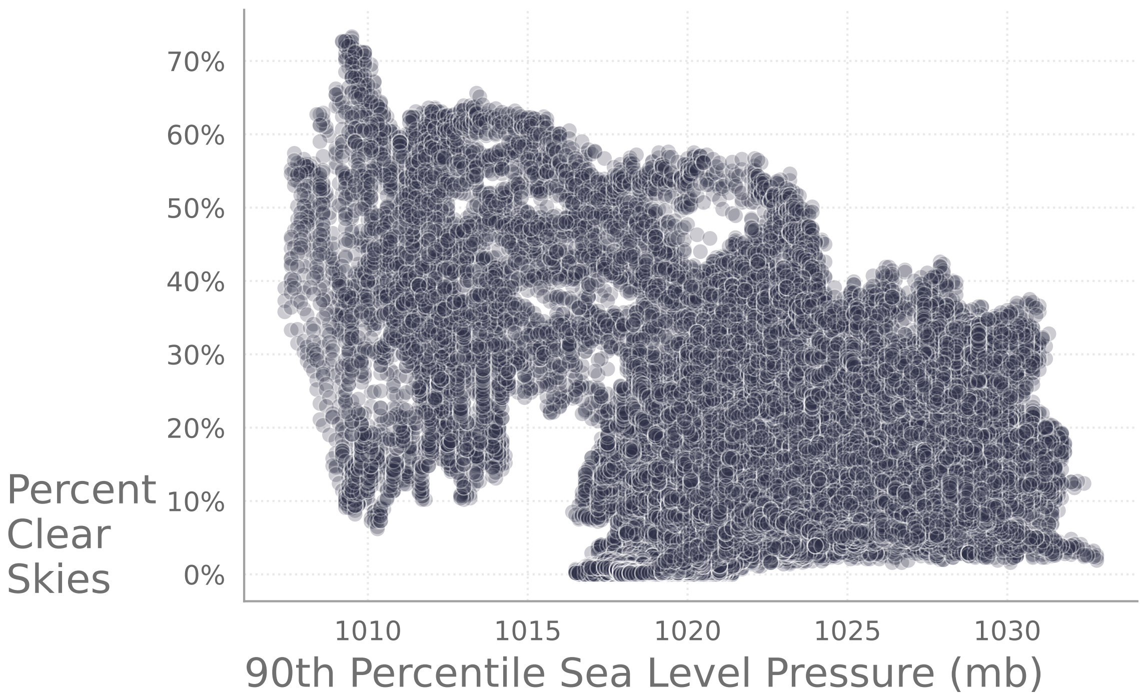


Figure 2: The non-linear relationship between clear skies and sea level pressure present the shape of a horse head.

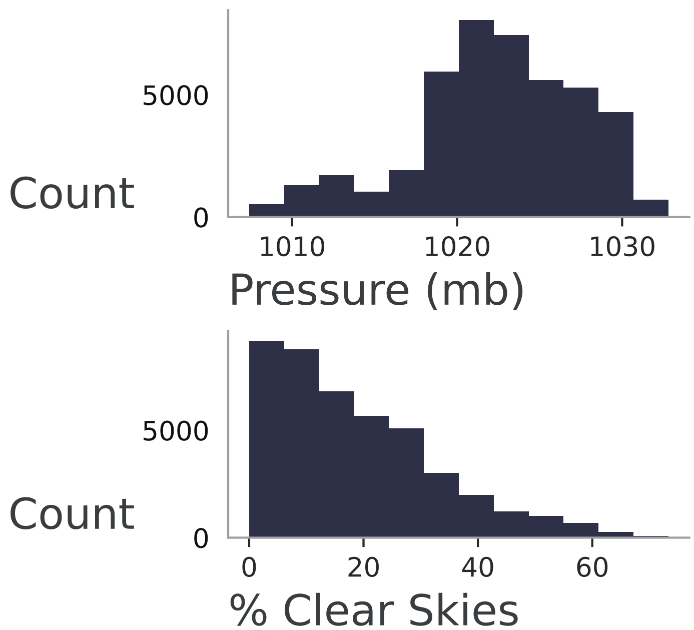


Figure 3: Clear skies percentage is clearly not a normal distribution.

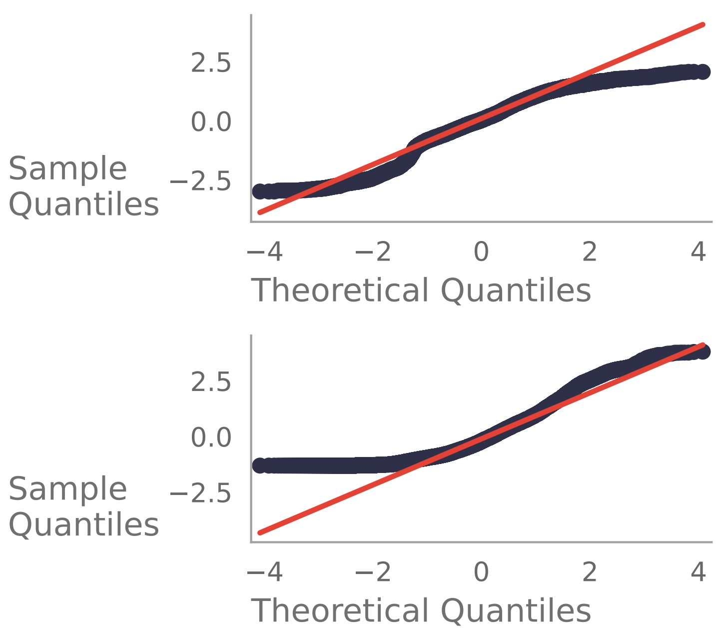


Figure 4: The q-q plots for pressure (top) and clear skies (bottom) show significant departures from a normal distribution.

Tables

The precipitation table (output variable *HPCP*) has the following statistical values for numerical parameters:

Table 1: Hourly Precipitation Data

Stat Value	Elevation	Latitude	Longitude	Rainfall ($\frac{1}{10}$ mm)
count	8,878	8,878	8,878	8,878
mean	265.8608	41.1489	-91.9104	1.4924
std	569.0261	6.7615	22.6294	2.9692
min	3.2000	25.7647	-122.3139	0.0000
25%	8.8000	39.7633	-122.1950	0.0000
50%	112.8000	42.3606	-80.3169	0.5100
75%	190.5000	47.4444	-71.1137	2.5400
max	2170.2000	47.9752	-71.0098	81.2800

The hourly normals (input variables), which is the data used as predictor variables, is much messier. As such, a sample is presented rather than the entire table. All sources are compiled using Zotero.

Table 2: Hourly Normals Data

Parameters	Numerical Parameters	Observations
57	29	43,795

Table 3: Hourly Normals Sample Data Description

Stat Value	Latitude	Longitude	Elevation	Cooling Degree Hours	Clouds Broken Percentage
count	43,795	43,795	43,795	43,795	43,795
mean	37.7480	-98.0787	437.7600	-41.6512	254.9026
std	7.4943	19.3599	655.4319	133.3855	102.9340
min	25.7905	-122.3138	3.7000	-432.0000	61.0000
25%	33.4277	-112.0038	8.8000	0.0000	184.0000
50%	39.7167	-104.7500	112.8000	0.9000	232.0000
75%	42.3606	-80.3163	337.4000	5.8000	308.0000
max	47.4444	-71.0097	1,726.1000	22.6000	620.0000

Data Preparation

Preparing the data for analysis took several steps. After importing the original files with new index columns generated while importing, the first step was to drop invalid values in the *precipitation* dataset. The only invalid value was 25399.75, which made it easy to drop. No other values in *precipitation* or *normals* were invalid.

When the first column was set as the index from a previous project, the station IDs were the index. Unfortunately, this approach did not work because, with categorical keys, all data from a station reporting an invalid value was wiped out while removing invalid data. This issue took a few hours to debug and was fixed by not importing an index column while reading the CSVs.

The second step was to format dates. Both datasets had different date formats, and neither were standard formats.

After formatting the dates, both datasets were joined on their datetime columns. *Normals* had 43,975 observations, while *precipitation* had 8,878 observations. This discrepancy is because of the reporting differences. Not only were there different sites reporting *normals* and *precipitation* for a given city, but precipitation levels were only reported when observed or starting the day, while *normals* were reported every hour.

In order to properly join the data, *normals* and *precipitation* were split by state. This approach worked because there was only one city per state in the selected data. Then the two sets per state were combined with an inner join. This process matched records and provided a final product as a dataframe of 8877 observations of 93 total variables.

Next, variables were renamed to be descriptive, and dashes were dropped in favor of underscores. This formatting change was because Classical Model 3 could not correctly parse columns with the dash included. Additionally, a constant column was added for the classical models to prevent uncentered estimations.

While normalization was attempted in the data preparation phase, it resulted in minor changes to model performance. In fact, it resulted in abysmal neural network performance. Instead, a normalization layer inside the neural network models performed better.

Finally, the data was split into 80% training, 10% validation, and 10% test data with the random seed of 20191220. This seed value was chosen because it represents a date of significance to the author.

Metrics and Modeling

The primary comparison metric between models was mean squared error (MSE). Residual analyses also played a role in comparing models. Mean squared error was chosen because it simplifies trivial model calculations and holds the same meaning across all the modeling efforts examined.

Table 4: Comparing Regression Model Performances

Regression Model	MSE $(\frac{1}{10}\ mm)^2$
Trivial	8.8979
Ordinary Least Squares with All Parameters	7.9449
Ridge Regression	7.9449
Sparse Ordinary Least Squares	8.0116
Linear Neural Network	11.3267
Linear Neural Network without Normalization	18,309.5313
Deep Neural Network	7.4048
Deep Neural Network without Normalization	9.8263

Trivial Baseline Model

The trivial baseline model was computed to show whether or not any models predicted better than taking an average of the data and comparing mean squared error by datapoint. The mean of the predicted training dataset was 1.4845 $(\frac{1}{10}\ mm)$, with a mean squared error of 8.8979 $(\frac{1}{10}\ mm)^2$. This wide confidence interval indicates that taking the mean of the prediction variable is not a good prediction model.

Classical Model 1

This model was an ordinary least squares regression including all components of the training data, and created with `statsmodels.api.OLS`. It resulted in a mean squared error of 7.9449 $(\frac{1}{10}\ mm)^2$, which is better than the baseline model. However, it only explained 10% of the data and, as demonstrated by the residual plot, failed to meaningfully match the data.

The relevance of input variables varied considerably, with individual p-values ranging from < 0.0001 up to > 0.9000.

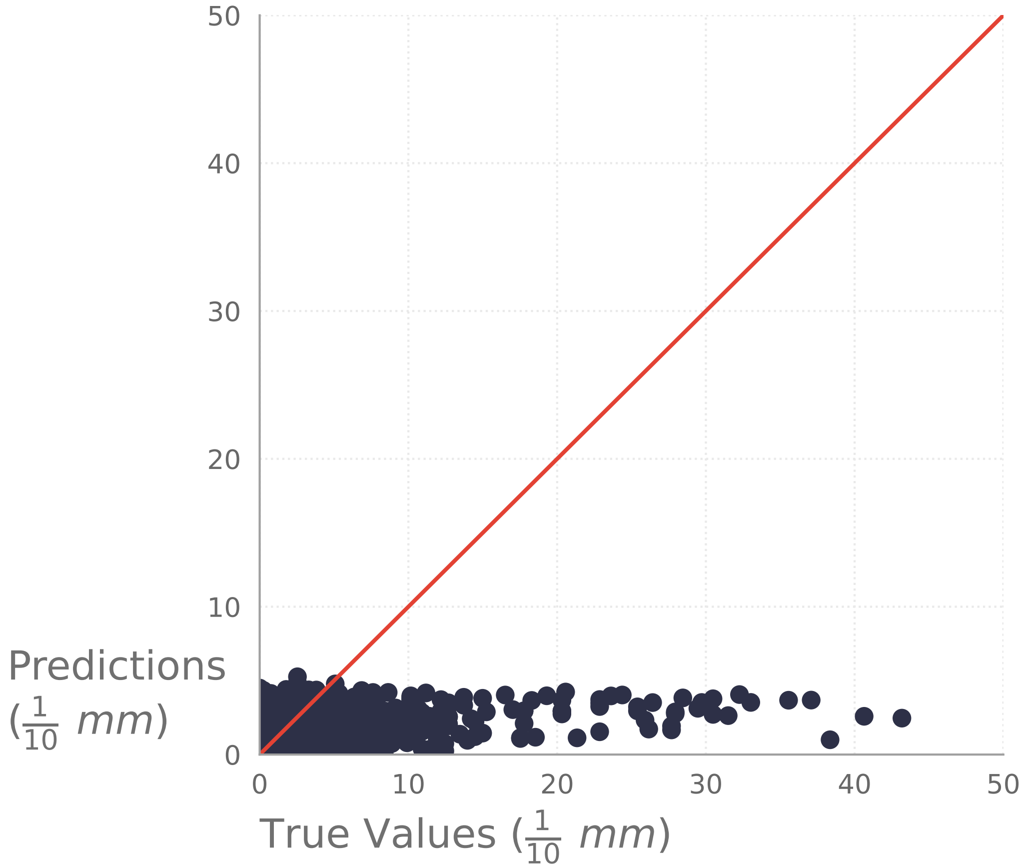


Figure 5: The predictions for the OLS built with all parameters (Model 1) do not fit the observed values for the training data.

Classical Model 2

The second classical regression model was a Ridge Regression model created using `sklearn.Ridge`. While it should be more computationally relevant, the results were almost identical. The mean squared error did not change between the two models until 13 decimal places. This model was more complicated to create with identical results. Once again, it is not a helpful model.

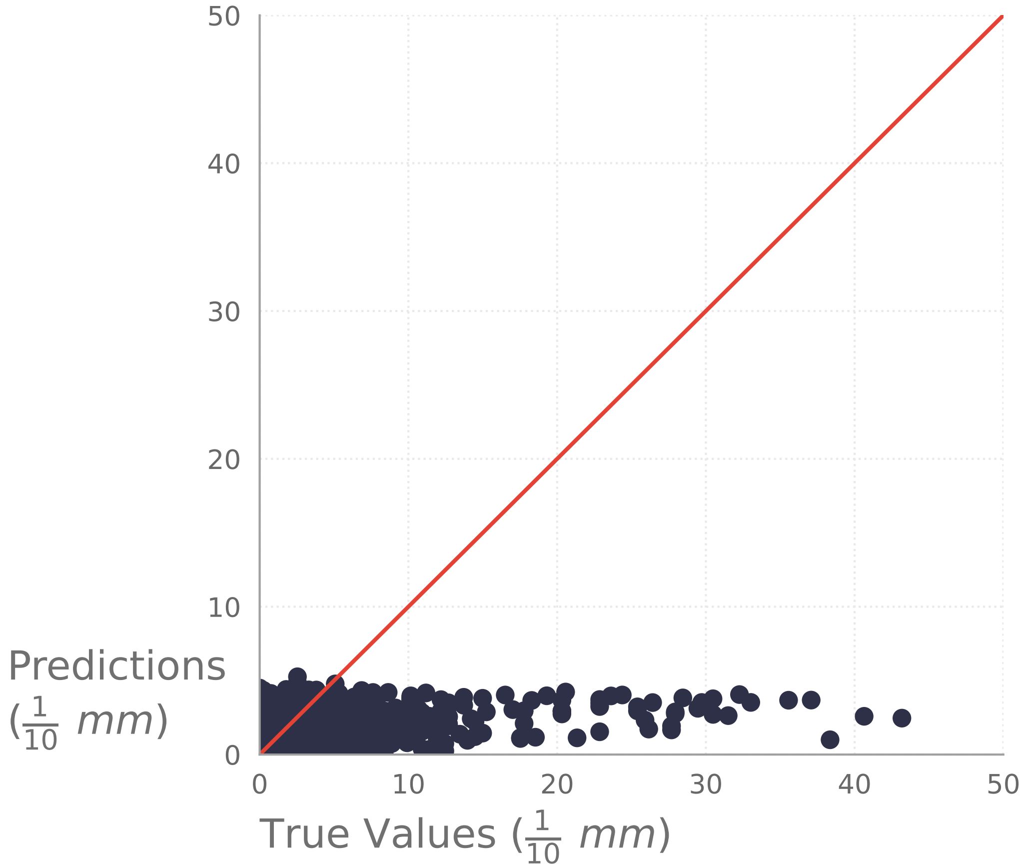


Figure 6: The predictions for the Ridge Regression Model (Model 2) do not fit the observed values for the training data and are nearly identical to the residuals from Model 1.

Classical Model 3

The final classical model was created by manual stepwise linear regression, optimizing for BIC while removing α values until no individual parameter had $\alpha > 0.05$. It was created with `statsmodels.formulas.api.ols`. BIC continued decreasing the entire time. The mean squared error for Model 3 fluctuates with each run relative to the other two, settling at 8.0116 $(\frac{1}{10}\ mm)^2$ at the time of writing. This fluctuation is despite the random seed being set the same during this period. Regardless, the residuals are still in the same state of not matching the predictions with the observations.

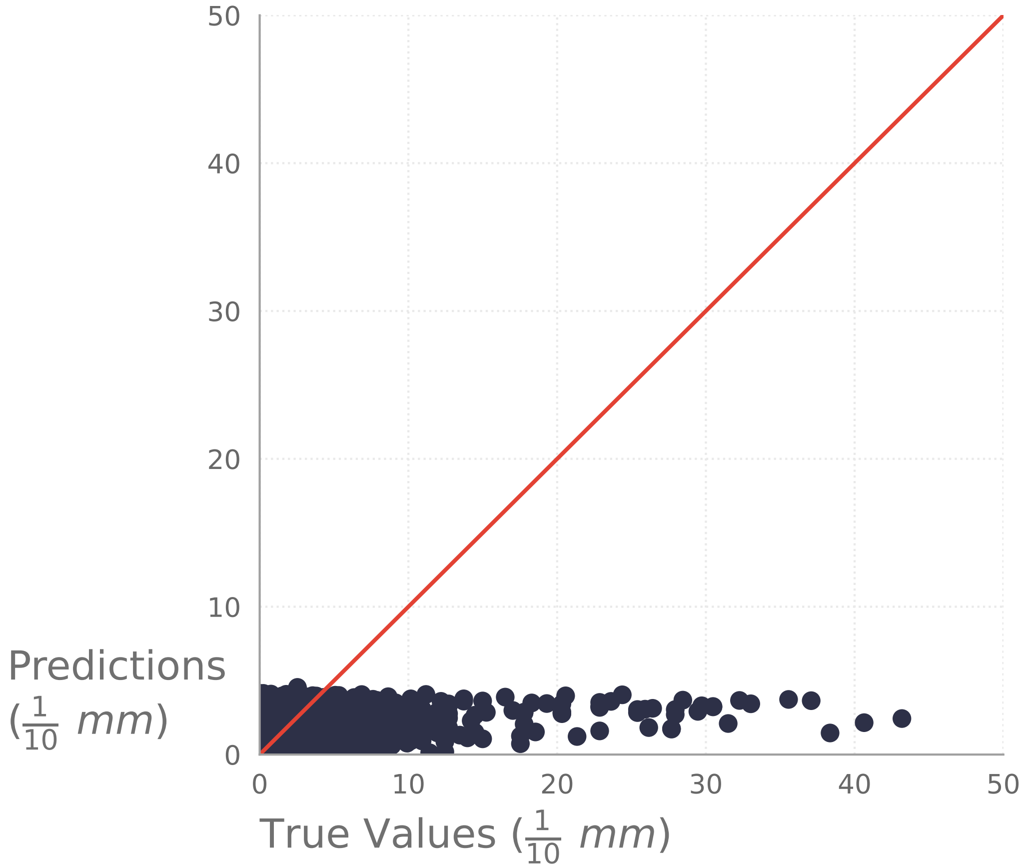


Figure 7: The predictions for the sparse least squares model (Model 3) still do not fit the observed values for the training data.

Classical Models Compared

The OLS regression model with all parameters included was selected because it would serve as a baseline to show what can be modeled with little pre-processing. The ridge regression model was selected next to show what a more intelligent selection method could predict. Finally, the sparse ordinary least squares model shows how little changed in the prediction with very few parameters selected. Residuals for all three models were nearly identical but can be seen in their sections.

Table 5: Comparing Classical Regression Model Performances

Regression Model	R^2	MSE $(\frac{1}{10}\ mm)^2$
Trivial	0.0000	8.8979
Ordinary Least Squares with All Parameters	0.1071	7.9449
Ridge Regression	0.1071	7.9449
Sparse Ordinary Least Squares	0.0996	8.0116

Linear Neural Network Model

The neural networks trained better based on mean absolute error rather than mean squared error as their loss function. In fact, the loss plots were 5x larger with mean squared error. The linear neural network model was created as a baseline to compare against the trivial baseline, classical approaches, and the deep neural network. All neural network models were created with `tf. keras`. The normalization layer was created using the mean and variance of the training data.

The optimizer for the linear model was Adam, with a learning rate of 0.1 and a mean absolute error loss function. This model was trained over 200 epochs and validated against the validation set split in the data preparation phase. This model had a mean squared error of 11.3267 $(\frac{1}{10}\ mm)^2$. As shown in Figure 8, both the training and validation losses are flat with noise for 200 training epochs.

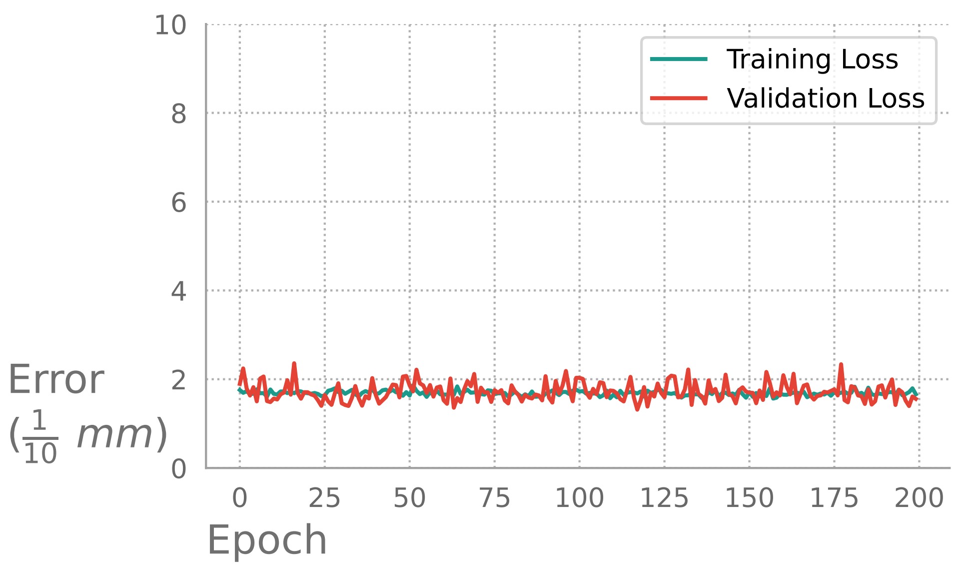


Figure 8: The training and validation losses are flat with noise for 200 training epochs.

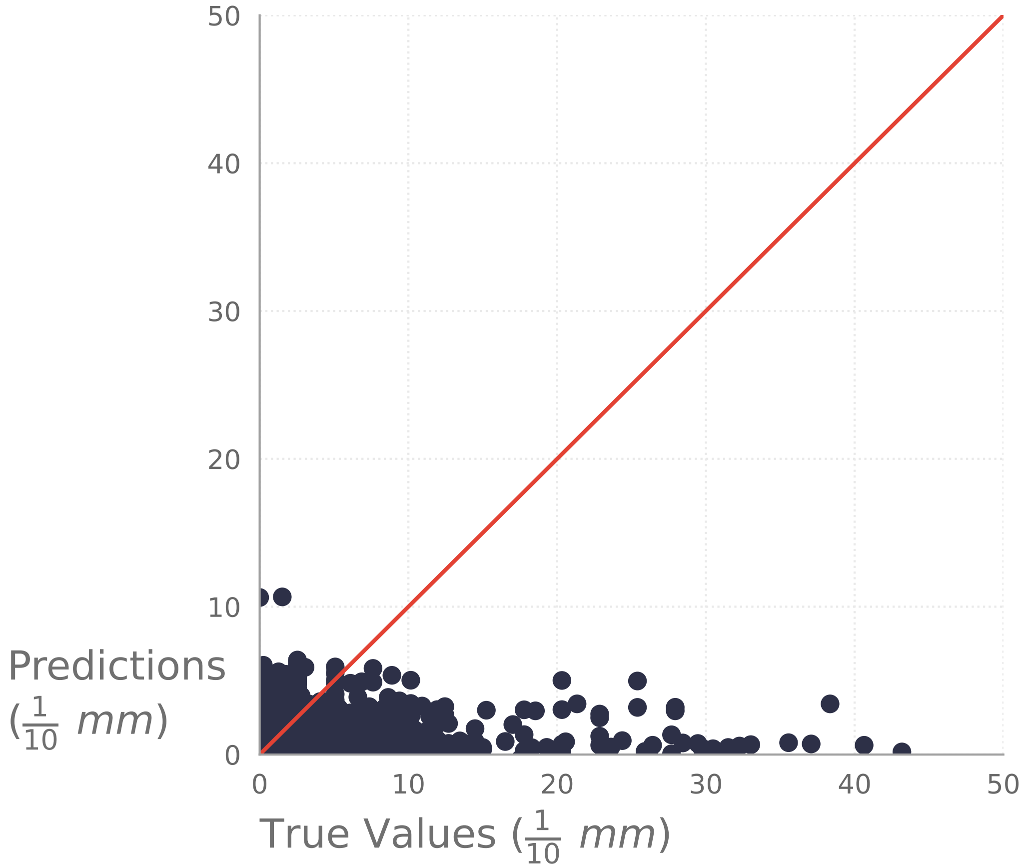


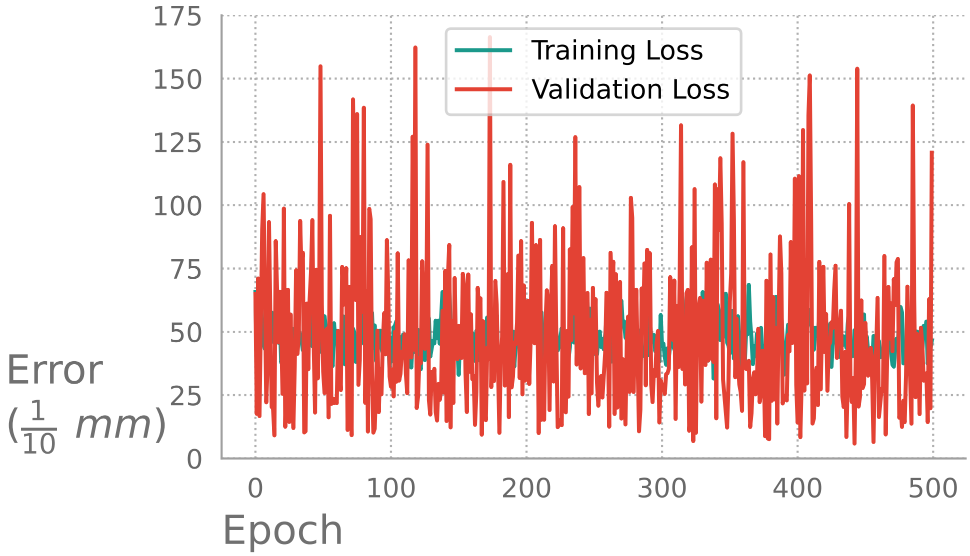
Figure 9: Residuals for the linear neural network are clustered, but do not follow the prediction line.

Linear Neural Network Model Without Normalization

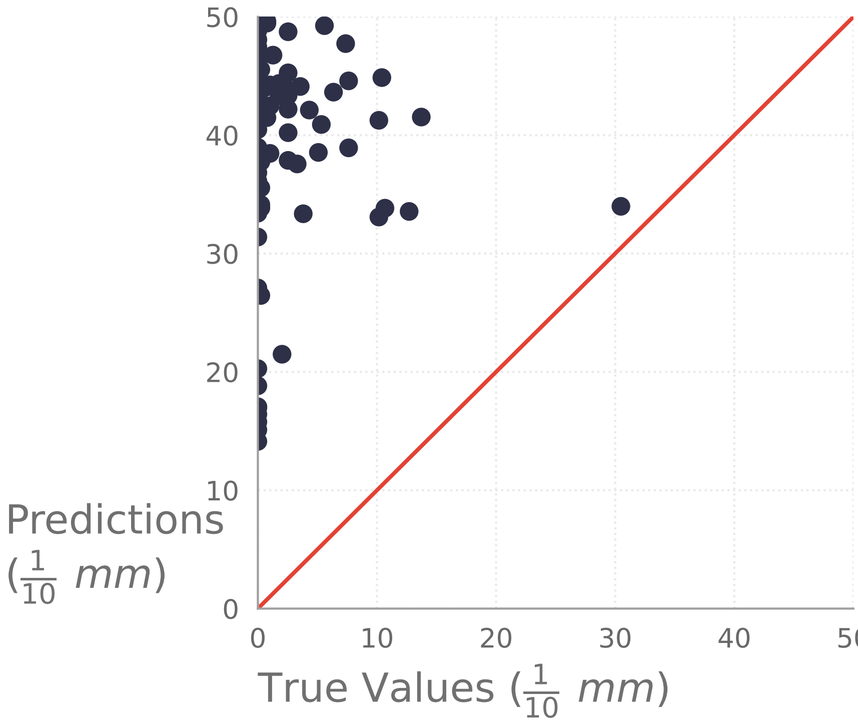
Normalization as a layer within the neural network is required for a good model. When removed, the resultant model is excessively noisy with much higher error rates, as demonstrated in Figure 10. The MSE for the linear neural network without normalization was 18,309.5313 $(\frac{1}{10}\ mm)^2$.

Deep Neural Network Model

Once the network can be expanded beyond linear relationships, it performs better. This model received the most variety of experimentation. The winning relationship was a dropout rate of 20%, using the mean absolute error loss function, optimizing with Adam and a learning



(a) The training and validation losses are extremely noisy when the normalization layer is removed.



(b) When the linear neural network is not normalized, the data presents as a vertical distribution clustered quite far from the prediction line.

Figure 10: Non-normalized linear neural network model.

rate of 0.001, early stopping with a patience of 10 epochs, and building the network with the following layer parameters:

1. Normalization layer
2. 64-neuron layer with a RELU activation function
3. 64-neuron layer with a RELU activation function
4. Dropout layer
5. Output layer

This structure is formally shown in Table 6 and Figure 11. Additionally, early stopping [9, pp. 141–142] and dropout layers [9, pp. 365–368] are also examined as regularization methods for model improvement in [9].

Additional experimented changes that did not improve the model included the following:

- Trying an input layer of 8,464 ($92 * 92$) neurons, 50 neurons, and 25 neurons
- Trying 3 and 1 dense hidden layer
- Using Stochastic Gradient Descent as the optimizer
- Using SELU as the activation function
- Changing the training epochs to 1,000
- Setting the early stopping parameter to 50 epochs waiting period

Addressing the concept map in [10], between the above experiments and the resulting model, searches for optimal parameters were made in the structure by changing layers and activation functions. Optimization branches were examined by changing defaults, loss functions, and batch normalization. Learning rates were copied from [11] and unexamined in favor of experimenting with learning counts, I.e., epochs. Finally, the two regularization efforts used were early stopping and dropout. Early stopping improved modeling time but made minimal differences to the model. Dropout made a noticeable improvement in the test data residuals below 1 mm, indicating better model generalization.

The data split in the data preparation phase (80/10/10) was created to provide the model with the most data for training and equal quantities for validation and testing. The random split ensured that none of the datasets was markedly different in distribution. Given 92 parameters and ~8,000 training points, this split provided enough data points for training and testing.

The mean squared error for the deep neural network was 7.4048, lower than any of the classical regression models — by 0.5. It was also lower than the trivial model by 1.5. Additionally, while validation and training losses slowly improved, the residuals were better than the classical models. Further, the histogram of prediction errors against the holdout dataset shows them reasonably centered on zero, with most errors in the ± 2 region, as shown in Figure 12d.

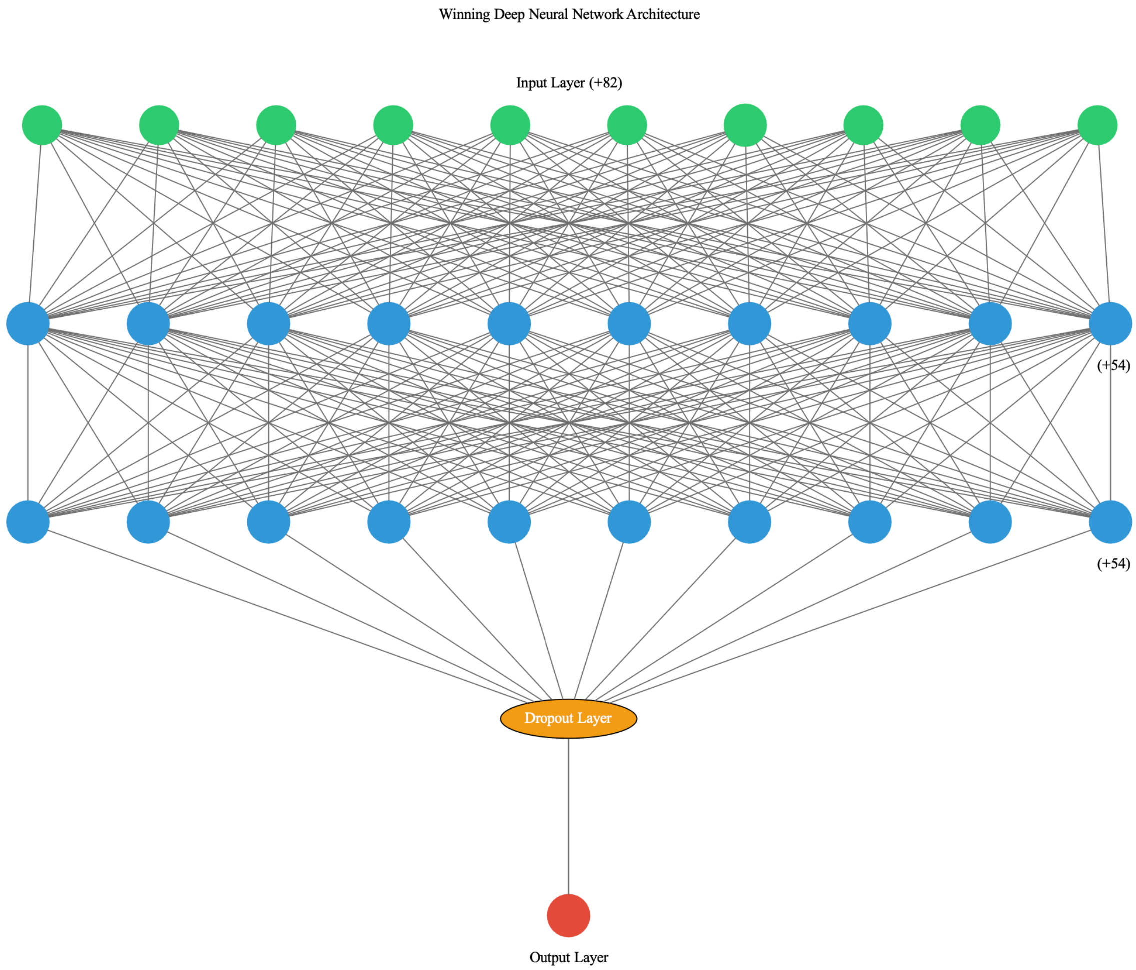
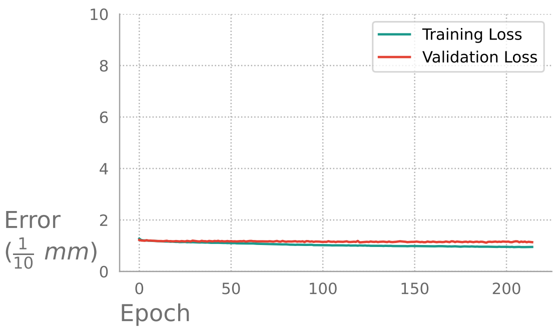


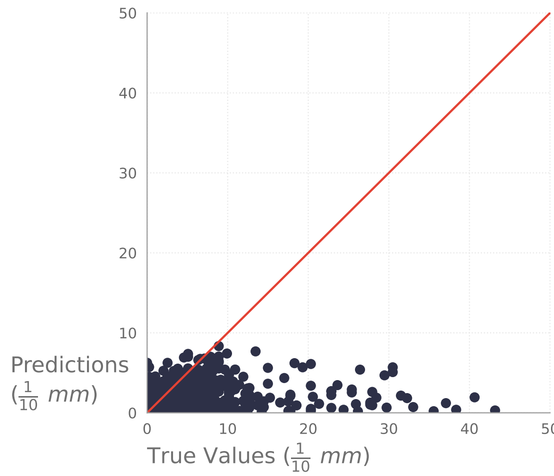
Figure 11: DNN architecture, with normalization layer hidden

Table 6: Deep Neural Network Model Layers

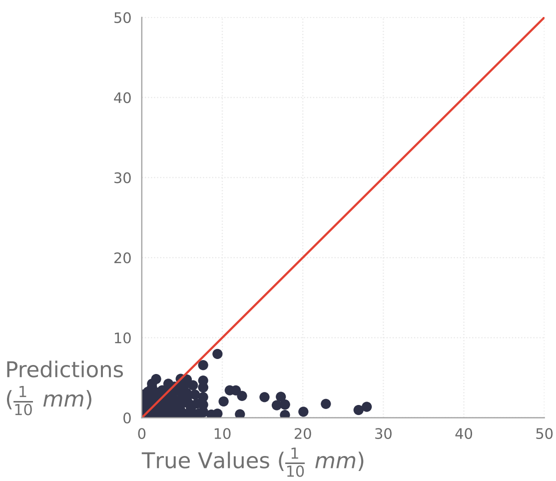
Layer (type)	Output Shape	Parameters
normalization_2 (Normalization)	(None, 92)	185
dense_22 (Dense)	(None, 64)	5,952
dense_23 (Dense)	(None, 64)	4,160
dropout_3 (Normalization)	(None, 64)	0
dense_24 (Dense)	(None, 1)	65
Total Parameters		10,362
Trainable Parameters		10,177
Non-Trainable Parameters		185



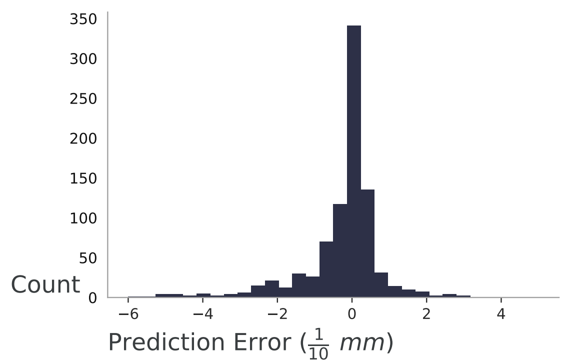
(a) The training and validation losses for the deep neural network slowly improved with each passing training epoch and remained extremely close to each other.



(b) Residuals for the deep neural network are spread slightly closer to the prediction line than the classical models, but still do not follow an accurate fit.



(c) Residuals on the holdout dataset for the deep neural network are clustered near the origin, but don't follow the prediction well. Although the residual distribution does look fairly similar to the training residuals, indicating a low probability of overfitting.



(d) The prediction error is mostly centered around +/- 2 with very few extreme outliers.

Figure 12: Deep neural network model.

Deep Neural Network Model Without Normalization

Using the same structure as the winning deep neural network but removing normalization and early stopping provided more of a learning *curve*, with optimal training around the same epoch count as the primary model. However, there was also a sharp jump followed by a plateau around 150 epochs, as demonstrated in Figure 13a. Regardless, the loss values converged to similar values for normalized and non-normalized variants. Nonetheless, despite similar convergences in the loss functions, the residuals in Figure 13b show that it predicted in a manner similar to the design of the trivial baseline. I.e. it predicted roughly identical values, regardless of the actual values.

Neural Networks Compared

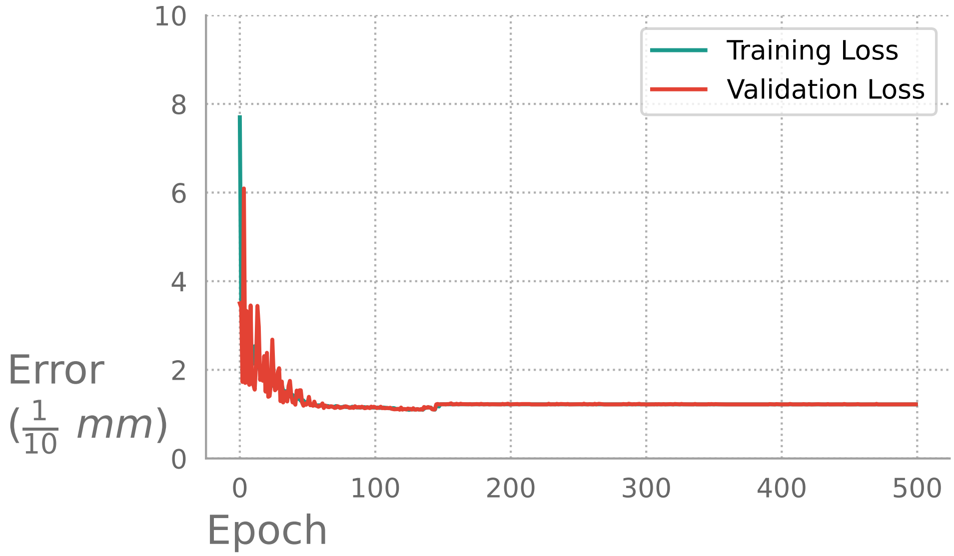
The winning model has the smallest MAE and MSE of any model examined. The residuals were also the cleanest on the deep neural network model than any other model.

Table 7: Comparing Neural Network Regression Model Performances

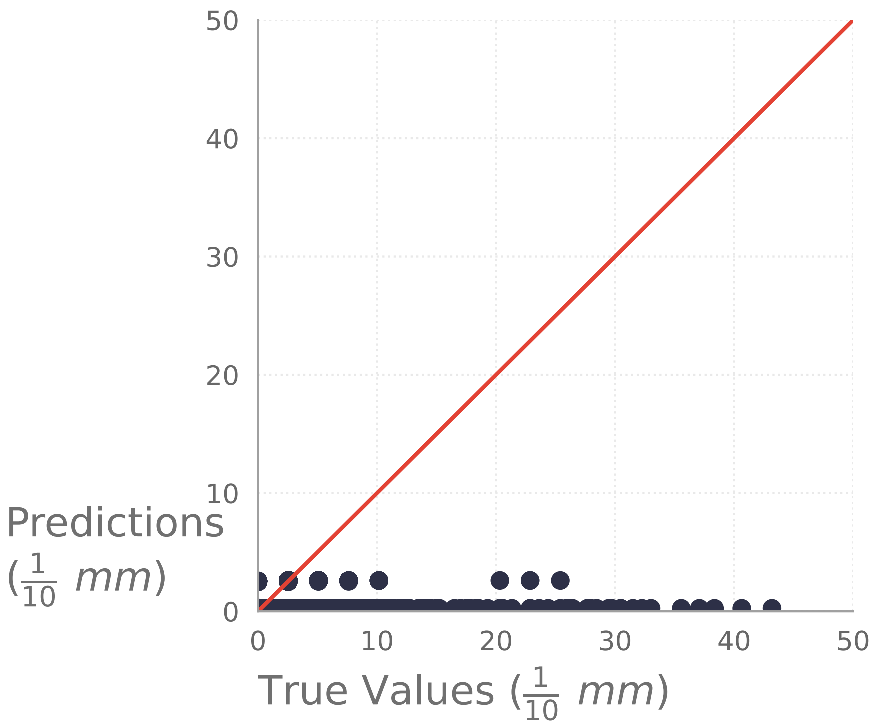
Regression Model	MAE ($\frac{1}{10}$ mm)	MSE ($\frac{1}{10}$ mm)^2
Trivial	1.5957	8.8979
Linear Neural Network	1.6174	11.3267
Linear Neural Network without Normalization	118.2756	18309.5313
Deep Neural Network	0.8963	7.4048
Deep Neural Network without Normalization	1.2206	9.8263

Model Evaluation

None of the examined regression models are operationally relevant. As an example test point, there was an observation where the rainfall was 0.5080 $\frac{mm}{hr}$. The model predicted 0.0825 $\frac{mm}{hr}$, 16% of the actual quantity. This precipitation discrepancy presented a negligible difference in signal attenuation at 3 GHz. However, that difference was not negligible for 30 and 300 GHz, as demonstrated in Table 8. Further, the attenuation gap grows with increasing rainfall because decibels operate on a logarithmic scale. At the maximum recorded sample in the test dataset of 3.8350 $\frac{mm}{hr}$, the difference in predicted and actual attenuation at 300 GHz grows to 34%. A 3-decibel drop indicates 50% less signal strength. As we see by the second sample, the model could not predict it at all (0.2088 $dB \neq 3.7962\ dB$). In fact, it predicted *less* rainfall than the previous sample point. This inferior fit holds in general with the model residuals Figure 12b and Figure 12c. Since the model performs so poorly with data in the training timeline, it cannot be expected to predict future rainfall, either.



(a) The training and validation losses for the unnormalized deep neural network improved rapidly, spiked, and then plateaued.



(b) Residuals for the deep neural network without normalization follow a horizontal distribution and indicate that the model is predicting nearly identical values for all datapoints.

Figure 13: Non-normalized deep neural network model.

The best model was the deep neural network model. This is determined by it having the lowest error scores and most consistent residuals, as shown in Figure 12b and Figure 12c. Even so, as demonstrated by these two samples, it is inaccurate and therefore not operationally useful. As a result, no inferences can be drawn from the model. Furthermore, the data is highly irregularly shaped, as demonstrated in Figure 2. This non-linear distribution suggests that to make a useful regression model requires much more data, a different modeling approach, domain-suggested feature selection, or a combination of all three.

Table 8: Model Performance on Sample Points

Frequency	Actual Rain ($\frac{mm}{hr}$)	Predicted Rain ($\frac{mm}{hr}$)	Actual Attenuation ($\frac{dB}{km}$)	Predicted Attenuation ($\frac{dB}{km}$)	Attenuation Error Difference (Amplitude)
3 GHz	0.5080	0.0825	0.0001	0.0000	$< 1\%$
30 GHz	0.5080	0.0825	0.1264	0.0225	1%
300 GHz	0.5080	0.0825	1.0632	0.3384	8%
3 GHz	3.8350	0.0383	0.0007	0.0000	$< 1\%$
30 GHz	3.8350	0.0383	0.8599	0.0109	22%
300 GHz	3.8350	0.0383	3.7962	0.2088	34%

Model Application

All the examined models performed poorly. From examining the data and performance metrics, using the existing modeling approach would require considerably more data to perform better than guessing or looking at commercial weather predictions. As of when the data was pulled, only data for 2010 was available. No data was available earlier or later than 2010. Given the volatility and distributions of the data, these are the possible non-exclusive routes to collect more data that could result in better model performances.

- Collect hourly data for all major US cities as the dataset, rather than only five cities.
- Include or focus on daily data rather than hourly data.
- Include radar observations rather than only ground station data.
- Join data for unbalanced observations, including all observations reporting no precipitation.
- Examine data available through the API rather than pulling individual CSVs.
- Synthesize missing data to ensure the timeseries remains intact for time-based predictions.
- Choose parameters based on consultation with a meteorologist to minimize weights on less essential variables.

Another option that already shows promising results is switching from regression to classification. While this would not predict how much extra power might be required to compensate for rainfall, it could indicate which hours most likely require closer attention than others. While rain/not-rain could work, the categories could also be split to say that rain less than X counts as 0 rather than only reported zeros. With that processing, a threshold less than 0.5 $\frac{mm}{hr}$ would ensure that no more than 1 dB/km of attenuation occurs from rain in the no-rain class. A preliminary classification neural network showed precision and recall figures greater than 62%. With more data and further model tuning, that could be significantly improved.

This effort examined only rainfall. However, clouds and other atmospheric effects also impact signal attenuation. Therefore, once the rain modeling is complete, building additional models to account for other atmospheric effects would improve the operational utility of these prediction efforts.

User Interface

As built in this phase, the model outputs a predicted float value of rain in tenths of millimeters. It also requires the input of 92 parameters. As currently built, a working user interface would require the user to download the current observations, format the observations into a dataframe with the appropriate data cleaning steps taken, and specify the frequency of interest before outputting an expected amount of rain in the next hour.

An ideal user interface would be a fillable website plugged into a National Weather Service API. From there, the user would input their location and frequency of interest. Then the regression model would generate the expected amount of rain hourly for the next twelve hours. Alternatively, a classification model would indicate the likelihood of rain during that same timeframe.

Conclusion

While predicting hourly rainfall would be operationally valuable to determine necessary power levels for satellite communication, the models examined based on the data used resulted in predictions worse than chance guessing. The data was collected from the National Climactic Data Center as CSVs from Seattle, Denver, Phoenix, Boston, and Miami for 2010 in two separate sets: one for precipitation and one for predictor variables. These were then inner-joined on datetime by city and spliced into the primary dataset of 8,877 observations, split 80/10/10 into training, validation, and test sets. Examining the data indicated that most variables were not normally distributed. Model comparisons were made based on Mean Squared Error. The trivial baseline had an MSE of 8.8979 $(\frac{1}{10}\ mm)^2$, while the best-performing deep neural network model had an MSE of 7.4048 $(\frac{1}{10}\ mm)^2$. Eight total models were examined for performance: three classical regression models and four neural network models. Residuals for all models were highly skewed. The winning deep neural network model had the best residual distribution and did not overfit the training data. However, the prediction value was still extremely poor. Normalization layers improved linear and deep neural networks but did not provide enough improvement to perform well.

Future efforts should focus on collecting more data, plugging into an API for real-time operational usability, consulting with a meteorologist, and considering switching to classification rather than regression.

Bibliography

[1] International Telecommunications Union, "Specific Attenuation Model for Rain for Use in Prediction Methods," Geneva, Switzerland, 2005. Available: https://www.itu.int/dms_pubrec/itu-r/rec/p/R-REC-P.838-3-200503-I!!PDF-E.pdf

[2] International Telecommunications Union, "Attenuation Due to Clouds and Fog," Geneva, Switzerland, Aug. 2019. Available: https://www.itu.int/dms_pubrec/itu-r/rec/p/R-REC-P.840-8-201908-I!!PDF-E.pdf

[3] International Telecommunications Union, "Propagation data and prediction methods required for the design of earth-space telecommunication systems," Geneva, Switzerland, Oct. 2009. Available: https://www.itu.int/dms_pubrec/itu-r/rec/p/R-REC-P.618-10-200910-S!!PDF-E.pdf

[4] É. Dieudonné, A. Kameni, L. Pichon, and D. Monchaux, "Radio Frequency Attenuation by a Rocket Plume using Diffraction Theory and Finite Element Modeling," *Acta Astronautica*, vol. 158, pp. 334–341, May 2019, doi: 10.1016/j.actaastro.2019.03.032.

[5] L. Lorenzo, "Theoretical and Experimental Investigation of Second-Order Atmospheric Effects Impairing W-Band SatCom Systems," *DTIC, Politecnico Di Milano*, no. AFRL–AFOSR–UK–TR–2022–33, Mar. 2022.

[6] N. Oceanic and Atmospheric Administration, "Datasets | climate data online (CDO) | national climatic data center (NCDC)." Available: https://www.ncei.noaa.gov/cdo-web/datasets#PRECIP_HLY

[7] N. Oceanic and Atmospheric Administration, "Hourly precipitation data documentation." Feb. 2016. Available: https://www1.ncdc.noaa.gov/pub/data/cdo/documentation/PRECIP_HLY_documentation.pdf

[8] N. Oceanic and Atmospheric Administration, "Hourly normals data documentation." Available: https://www.ncei.noaa.gov/pub/data/cdo/documentation/NORMAL_HLY_documentation.pdf

[9] A. Géron, *Hands-on machine learning with scikit-learn, keras, & tensorflow: Concepts, tools, and techniques to build intelligent systems*, Second edition. Sebastopol, CA: O'Reilly Media, Inc, 2019.

[10] Torrey Wagner, "Neural network foundations & processes," Dec. 2022.

[11] François Chollet, "Basic regression: Predict fuel efficiency | TensorFlow Core." Available: https://www.tensorflow.org/tutorials/keras/regression