# VIBRATION-BASED MACHINE LEARNING MODELS FOR CONDITION MONITORING OF RAILROAD ROLLING STOCK

A Thesis

by

SERGIO M. MARTINEZ

Submitted in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE IN ENGINEERING

Major Subject: Mechanical Engineering

The University of Texas Rio Grande Valley

July 2023

# VIBRATION-BASED MACHINE LEARNING MODELS FOR CONDITION MONITORING OF RAILROAD ROLLING STOCK

A Thesis
by
SERGIO M. MARTINEZ

COMMITTEE MEMBERS

Dr. Constantine Tarawneh
Co-Chair of Committee

Dr. Mohamadhossein Noruzoliaee
Co-Chair of Committee

Dr. Fatemeh Nazari
Committee Member

Dr. Heinrich Foltz
Committee Member

August 2023

Copyright 2023 Sergio M. Martinez

All Rights Reserved

# ABSTRACT

Martinez, Sergio M., Vibration-Based Machine Learning Models for Condition Monitoring of Railroad Rolling Stock. Master of Science in Engineering (MSE), August 2023, 79 pp., 13 tables, 28 figures, references, 26 titles.

One of the primary causes of rolling rail stock derailment is attributed to bearing and axle failure. The health of a train bearing is primarily monitored at target locations through wayside and Hot Bearing Detectors. This can lead to bearing failure and potential derailments at points in between them. To remedy this, the University Transportation Center for Railway Safety (UTCRS) has developed an onboard monitoring system that can continuously monitor the vibration response, which directly correlates to the health of bearings. This data is used to train regression-based machine learning algorithms and long-term prediction neural networks to predict bearing health. The models are intended to work in tandem with the onboard monitoring sensors as a means of two-way practical validation. The models tested were the Gradient Boosting Machine architecture for scheduled predictions and the Informer neural network architecture for long-term predictions of ongoing routes. The dataset for these models comes from the expansive experiment record data available at the UTCRS. Ultimately, these machine learning algorithms will enhance railcars' safety and save companies money by allowing for predictable maintenance periods.

# DEDICATION

This thesis is dedicated to my family, friends, and everyone who supported me in obtaining such a milestone. The first person I would like to thank is my mother, Amalia G. Martinez. I would not have been able to accomplish all that I have without the unending support you have given me. To my father, Sergio M. Martinez Sr., I am grateful for the work you put into supporting our family and always supporting me in pursuing higher education.

I thank my sisters, Amy D. Martinez, Sydney M. Martinez, and Arianna D. Martinez, for our fond memories and laughs. To Amy, I am grateful for you being there for me, from helping with my schoolwork to introducing me to many hobbies I still have. To Sydney, your passion for doing things the right way inspires me, and I am thankful for you being there. To Arianna, thank you for allowing me to be a role model and helping you get closer to your goals.

I love you all and am grateful you were always there for me.

# ACKNOWLEDGMENTS

Thank you, Dr. Tarawneh, for allowing me to work in your center and offering anything I could need to succeed in my career. I cannot express how grateful I am for the opportunities you provided and for your passion and dedication to the students working in your center.

I want to thank Dr. Noruzoliaee for his guidance on this project. Thank you for providing me with much-needed knowledge in the field of machine learning and for keeping me on pace. I am grateful for your encouragement and willingness to see me succeed.

To the other students at the University Transportation Center for Railway Safety (UTCRS), thank you for providing me with the help and information I needed to work on my project efficiently. I could not have asked for a better team. You all made it fun to work and helped me stay productive, so genuinely, thank you for being there.

This project was made possible through the funds provided by the University Transportation Center for Railway Safety (UTCRS) at the University of Texas - Rio Grande Valley, the National Science Foundation CREST Center for Multidisciplinary Research Excellence in Cyber-Physical Infrastructure Systems under NSF Award No. 2112650, and the Dwight David Eisenhower Transportation Fellowship Program (DDETFP) under USDOT Award No. 693JJ32345193. The opinions expressed in this thesis are solely those of the authors and do not necessarily represent those of the NSF or the USDOT.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

# CHAPTER I

# INTRODUCTION AND MOTIVATION

## 1.1 Introduction

Over the past decade, there have been 1,671 derailments directly attributed to mechanical and electrical failures, according to the United States Department of Transportation and the Federal Railroad Administration. Of these derailments, the highest percentage, 8.2% (203), were caused by journal-bearing defects and failure. (Federal Railroad Administration, n.d.).

Derailments pose a severe risk to infrastructure and safety, making it imperative to implement preventive measures. To facilitate these preventative measures, this research, conducted as part of the University Transportation Center for Railway Safety, implements machine learning algorithms and deep neural networks to predict when these bearings will fail.

## 1.2 Tapered Roller Bearings Overview

Tapered roller bearings comprise multiple subcomponents, the cup, spacer ring, cage, 23 rollers, a cone, the grease seal, and the wear ring, as seen in Figure 1.

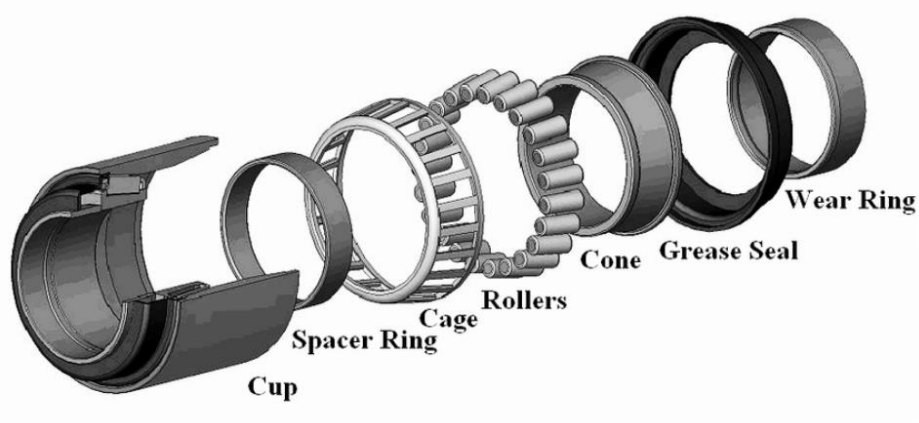



Figure 1: Exploded view of a tapered roller bearing (Hernandez, 2020).

The primary components of a tapered roller bearing are two cone assemblies housed within a cup. A cone assembly refers to a set of rollers on the cone held in place by a cage. The spacer ring acts as a means to prevent the cone assemblies from grinding against one another. The grease seal keeps any outside elements from entering the cone assemblies that would introduce impurities to the grease. The wear ring separates bearings from one another to ensure they rotate independently, not to expose the components to unnecessary friction. These primary members experience the most radial stress and, as such, are the components most in need of monitoring. (Hernandez, 2020).

### 1.2.1 Types of Defects and Failures

There are three major categories of bearing failure: local, distributed, and geometric defects. Local defects only encompass a section of the bearing, including pits, cracks, and spalls. Distributed defects refer to defects that affect the entire length of the bearing, such as water etching. Geometric defects refer to when components are not in tolerance. (Hernandez, 2020).

## 1.3 Wayside Detection Systems

The current technologies to monitor the condition of critical components of rolling rail stock, such as tapered roller bearings, are referred to as wayside technologies. As the name implies, these technologies go mounted on either the side of or directly on the rails as opposed to on the actual train. The temperature, vibration, and load profiles are needed to fully describe the characteristics of a tapered roller bearing. The three most commonly used wayside technologies in North America reflect these profiles being Hot Bearing Detectors (HBDs), Trackside Acoustic Detector Systems (TADS™), and Wheel Impact Load Detectors (WILDs).

### 1.3.1 Hot Bearing Detectors (HBDs)

HBDs measure the temperature profile of critical rolling rail stock components via infrared sensors. Specifically, these sensors check to see if the temperature of any service bearing is 94.4°C or 170°F above ambient. Companies affiliated with the railway industry construct HBDs in areas where the gap between them ranges from 20 to 45 km. There are currently over 6,000 HBDs in service in North America. Figure 2 shows a diagram of an HBD. (Stewart, Flynn, Marquis, & Sharma & Associates, 2020).

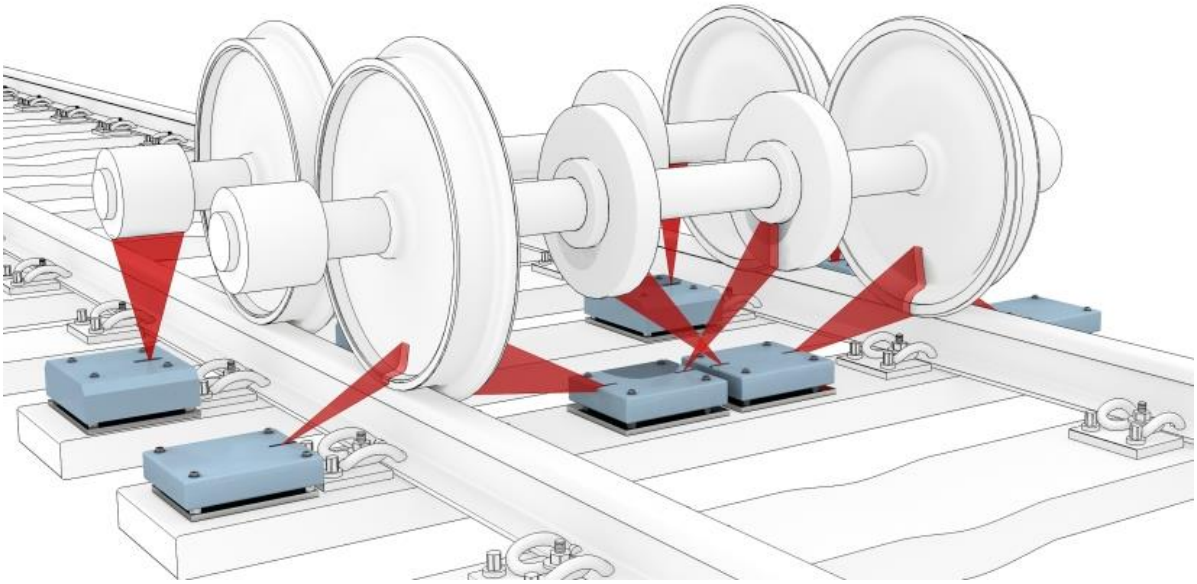


Figure 2: Hot Bearing Detector (HBD) diagram (Stewart, Flynn, Marquis, & Sharma & Associates, 2020).

HBDs have led to fewer derailments and an increased number of non-verified bearings in the railway industry. The biggest shortcoming of HBDs is that temperature is a very reactive property, meaning that once a defect is detected, it may be too late to prevent derailment. A 100-freight car train operating at a speed of 50 miles per hour needs around 3 miles to come to a complete stop. This number goes up to approximately 6 miles for a 150-freight car train operating at the same speed. (Tarawneh, Aranda, Hernandez, Crown, & Montalvo, 2020).

### 1.3.2 Trackside Acoustic Detector Systems (TADS™)

Trackside Acoustic Detector Systems (TADS™) are stationary technologies that monitor the vibration profile of tapered roller bearings. TADS™ are a form of acoustic bearing detector that observe the acoustic response from the tapered roller bearings and classify them as "growlers" if severely spalled. If a TADS™ detects a growler, a signal for an immediate stop of

the train is sent to the operators. Figure 3 depicts an image of a TADS™. (Stewart, Flynn, Marquis, & Sharma & Associates, 2020).

Figure 3: Trackside Acoustic Detector Systems (TADS™) (Stewart, Flynn, Marquis, & Sharma & Associates, 2020).

### 1.3.3 Wheel Impact Load Detectors (WILDs)

Wheel Impact Load Detectors (WILDs) monitor the load profile at fixed points along railway routes. To prevent overloading that can lead to potential derailments, companies like LBFoster implement these detectors. WILDs are track-mounted strain gauges that measure the wheel-to-rail contact force. Figure 4 shows a WILD in service. (LBFoster US, n.d.).

Figure 4: Wheel Impact Load Detectors (WILDs) (LBFoster US, n.d.).

## 1.4 Derailment Case Studies

### 1.4.1 Canadian Pacific Freight Train 220-24

Between Sudbury and Mactier, Canada, on January 26, 2011, Canadian Pacific Railway freight train 220-24 experienced a twenty-car derailment that leaked non-odorized liquified petroleum gas. A spall on the inboard cup raceway led to the seizing of a roller bearing. The seized roller bearing overheated and ultimately burnt off the axle journal. This derailment occurred after eight separate HBDs failed to flag the train to stop. The derailment happened in a location where the gap between the HBDs was more extensive than 25 miles (40 km), as shown in the map in Figure 5.

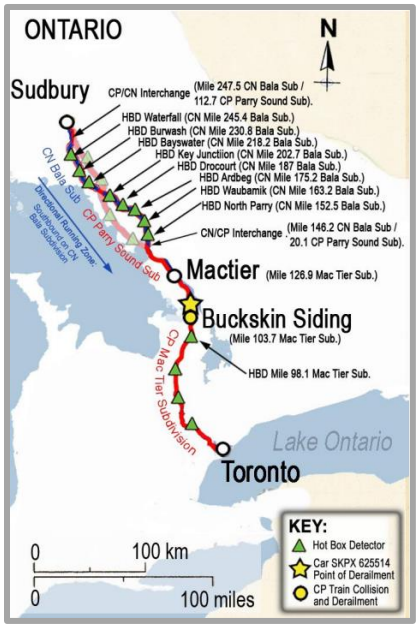



Figure 5: Map of the Canadian Pacific freight train 220-24 derailment (CP Freight, 2011).

As such, in response to this derailment, Canadian Pacific installed more HBDs in these locations where the gaps between detectors were as large or larger. (CP Freight, 2011).

### 1.4.2 Norfolk Southern Train 32N

Norfolk Southern train 32N experienced a 35-car derailment on February 3, 2023, in East Palestine, Ohio. Due to the inadequacy of the current condition monitoring technologies and the nature of what they detect, the railway industry failed to prevent this derailment. Before the train was flagged to stop, the temperature of the bearing that failed read 103°F at an HBD. As mentioned in the Hot Bearing Detector section, HBDs flag trains to stop if the temperature exceeds 170°F. The bearing then began to fail and was not flagged to stop until the following HBD read 253°F. By this point, the defective bearing caught fire for over 20 miles and the train ultimately derailed. The overlying issue with the current reliance on HBDs is that temperature is a reactive indicator, so it may already be too late to save the train once the temperature profile

detects a defect. This derailment also helped further indicate the need for live condition monitoring technologies to become more prevalent in the industry. (National Transportation Board, 2023).

## 1.5 Onboard Wireless Monitoring Systems

The current peak of bearing condition monitoring technology are onboard sensors capable of live monitoring the three characteristic profiles: temperature, vibration, and load. The temperature profile allows for knowledge of imminent failure, while the vibration profiles allow for a more proactive response. Finally, the load profile gives insight into the stresses acting on the components. The onboard sensors being implemented in the field today reflect these parameters.

### 1.5.1 UTCRS Wireless Sensor

The University Transportation Center for Railway Safety (UTCRS) invented a means to monitor tapered roller bearing temperature and vibration profiles in a non-intrusive manner using the UTCRS wireless sensor, which Figure 6 shows.

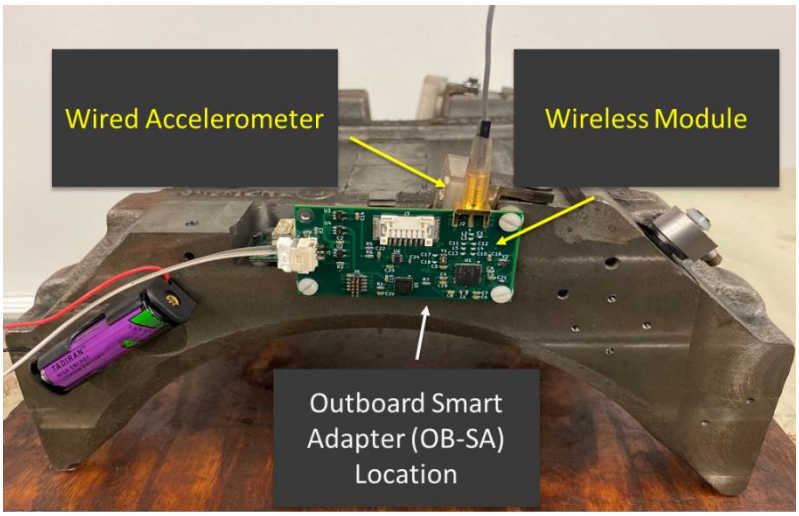



Figure 6: UTCRS Wireless Sensor labeled display (Cuanang, 2020).

This sensor was developed and intended for academic and research purposes, with one of its most highlighted features being that it goes mounted on the bearing adapter, meaning the condition of the actual bearing does not affect the performance of the sensor, making it reusable. The UTCRS wireless sensor collects at a rate of 5,120 Hertz for 16 seconds every 10 minutes. Once the data is collected, the sensor transmits it to a central monitoring unit via Bluetooth. (Cuanang, 2020).

### 1.5.2 HUM Industrial Technology Boomerang

The HUM Industrial Technology Boomerang is a field application version of the UTCRS wireless sensor developed by the UTCRS and put into production by HUM Industrial Technology. Figure 7 depicts a field-tested version of the HUM Boomerang.

Figure 7: Mounted HUM Industrial Technology Boomerang (Cantu, 2021).

This sensor can also monitor the vibration and temperature responses of the tapered roller bearings while being mounted to the adapter. The HUM Boomerang can output bearing operating temperatures with an accuracy of within eight °C and an accuracy of ±1 g regarding the RMS of the vibration profile readings. This sensor collects at a rate of 5,200 Hertz for 1 to 4 seconds every 10 minutes. (Cantu, 2021).

## 1.6 Related Work

In July 2022, the University Transportation Center for Railway Safety published a thesis by Leonel Villafranca covering their first attempt at implementing machine learning for bearing condition monitoring. (Villafranca, 2022). The thesis title is “Predicting the remaining service life of railroad bearings: leveraging machine learning and onboard sensor data.” In this work, Leonel Villafranca implemented three machine learning models to utilize the data from the UTCRS wireless sensors to predict the bearing’s remaining service life. The models implemented were a Gradient Boosting Machine, an eXtreme Gradient Boosting Machine, and a Federated Learning algorithm. The models used generalized features or input data to predict the

vibration response's root mean square (RMS). Specifically, the features utilized were the mileage, the speed in miles per hour, and the loading conditions (as in unloaded, partially loaded, or fully loaded) that a bearing experienced.

The models used in this research are a Gradient Boosting Machine and the Informer neural network. Where this research differs from Leonel's implementations is through some notable aspects. The first difference is that the models in this research use a larger dataset that considers more types of experiments than solely field service tests. These models also are trained using features that characterize the data more accurately by considering the effects of each combination of generalized features. The Gradient Boosting Machine trained in this research was validated by testing on experiments not present in the dataset, proving that the model is practically viable rather than solely having theoretical accuracy. Finally, the most significant and impactful difference is that this research implements deep learning neural networks, a more involved subset of machine learning algorithms.

## 1.7 Motivation

The ability to predict when critical components will fail provides another step toward ensuring safety. As this research is conducted for the University Transportation Center for Railway Safety and its partners, these models will be able to work in tandem with their current onboard sensors, namely the UTCRS wireless sensor and the HUM Boomerang. Despite safety being of the utmost importance, it is hard to invest in it from a business aspect if there is no immediate need. Thankfully, there is a driving economic factor to this research as well. In North America, private companies own the railcars; however, they do not control when they go into maintenance. The railway industry decides the maintenance periods, so private companies must endure potential month-long periods of unexpected downtime. This research will allow for

means to predict when a bearing fails, enabling the creation of scheduled maintenance periods. The implementation of various machine learning models will facilitate this goal. The first machine learning model is a Gradient Boosting Machine that estimates the vibration profile given details on a route's mileage, speed, and load conditions. This model will create scheduled predictions before the route's operation. The second model is, more specifically, a deep neural network capable of reading time-series-based data. This model is the Informer Neural Network, used to make long-term predictions of ongoing routes. These models are discussed more in-depth in Chapter II.

# CHAPTER II

# MACHINE LEARNING BACKGROUND AND RELEVANT IMPLEMENTATIONS

## 2.1 Machine Learning Overview

Artificial Intelligence (AI) refers to anything that allows a computer to mimic the thought processes of human intelligence. A popular subset of AI is Machine Learning (ML) which involves algorithms that can improve at a task given a training method and experience. Machine learning algorithms utilize defined iteration-based methods to minimize a loss function that describes the difference between predicted and experimental values. Machine learning models take input data, referred to as features, that describe the predicted variable. There are two major tasks for machine learning models: classification and regression. Classification models utilize string-based input and output data, while regression models utilize strictly numeric values for a form of interpolation to occur. This research utilizes a machine learning model for a regressive task to predict the Root Mean Square (RMS) values of the vibration profile of tapered roller bearings. The primary requisite for a machine learning algorithm is a dataset to be trained on and make a foundation on which predictions can be made. Machine learning algorithms split their datasets into three sub-categories: training, validation, and testing. The algorithm operator decides the split for the ratio of data; however, it is standard practice to allot the largest portion of data to training. (Dhande, 2020).

### 2.1.1 Bias and Variance

A lack of balance between the bias and variance of a model's prediction will lead to doubt on the validity of said prediction. Bias refers to the systematic errors made during a model's learning. A high bias means that the model is oversimplified or underfitting. If a model is underfitting, it will fail to correctly highlight the data's complexity. Underfitting occurs when the features implemented fail to describe the target variable, or the model is too simple to capture the complexity needed to make a proper prediction. Variance refers to the model's ability to capture and replicate the fluctuations while training. If the variance is too high, the model will fully mimic the exact response from training, resulting in overfitting. Overfitted models will likely fail to predict new or unseen data adequately. (Banoula, 2023). Machine learning models are ideally optimized to be neither underfit nor overfit. This research balances the bias and variance through five-fold cross-validation, an ensemble method built into the Gradient Boosting Machine, and iterative testing methods scored on several factors. Cross-validation is an iterative process that separates the data into equally sized groups, or folds, to train and validate. The details on the ensemble method are discussed in the Gradient Boosting Machine section in this chapter.

### 2.1.2 Decision Trees

Decision trees are another type of machine learning algorithm suitable for regression tasks. The first step that the algorithm takes is to choose a feature and define a threshold to split the data into two subsets. The data is then split recursively and continuously applied to the subsets created in the previous step. This recursive split leads to forming "tree-like" structures, hence the name. Figure 8 depicts an example of a decision tree.

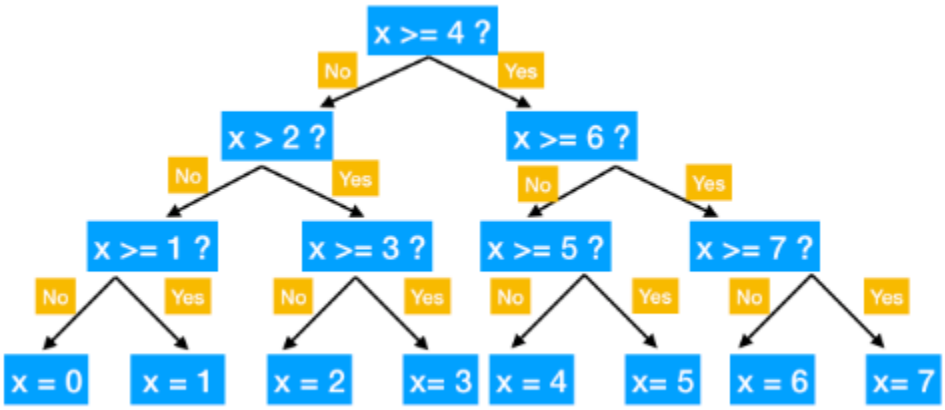



Figure 8: Decision tree (Lin & Li, 2023).

The top of a decision tree is referred to as the root node. The root node poses the initial condition before passing it into the following condition. The root node points to the internal nodes and each pair of arrows represents a true or false condition. Internal nodes can be distinguished by having arrows pointing into and out of them. The internal nodes serve as intermediary conditions, or comparisons against the defined threshold, for the current data point the algorithm is testing. Continuing down the hierarchical structure, nodes that only have arrows pointing to them are termed leaf nodes. These leaf nodes are the final condition on which a data point gets tested before the algorithm predicts.

### 2.1.3 The I.I.D. Assumption

The I.I.D. assumption is a generalized assumption that applies to machine learning algorithms and statistic-based models. The primary definition of the assumption states that the data should be independent and identically distributed. Independent means that the value of a data point is not influenced by, nor influences, another data point. This part of the assumption allows machine learning models to focus on a single data point at a time rather than on relations

between them. If this part of the assumption is invalidated, the predictions could have unnecessary correlations. The second half of the assumption, the identically distributed portion, states that the data points should be sourced from the same probability distribution. If this section of the assumption is invalidated, the model may need help to predict unseen or new data. The I.I.D. assumption specifies that each data point must be independent of one another and output identical responses if the same features are input. The nature of this research invalidates the I.I.D. assumption. The dataset more closely resembles a time series, given that the vibration response is unique to each bearing. It is unlikely for the RMS of two separate bearings to be equal even if the same values for the features are input. Despite this, this research formats the model so that the predictions are more of a practical generalization.

## 2.2 Scoring Systems

A self-imposed scoring mechanism dictates the performance of a machine learning model. The most common forms of scoring systems involve the coefficient of determination and evaluation based on the error between the predicted and actual values.

### 2.2.1 Coefficient of Determination ($R^2$)

The coefficient of determination, $R^2$, value is a parameter that describes the accuracy of a line of fit for regressive models. The coefficient of determination measures the amount of variation that the model captures. Equation 1 shows the formula for the coefficient of determination.

$$R^2 = 1 - \frac{RSS}{TSS} = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y}_i)^2} \quad (1)$$

RSS refers to the sum of the squares of residuals, and TSS to the total sum of squares. The $y_i$ in the equation above denotes the observed value at iteration $i$, while $\hat{y}_i$ is the predicted value at iteration $i$, and $\bar{y}_i$ is the mean of the observed values. (Chicco, Warrens, & Jurman, 2021).

It is common practice to plot the actual values versus the predicted values with the ideal response to be relatively linear. The values for the coefficient of determination typically range between zero and one, with a value of one being able to describe the variance in the model entirely. A coefficient of determination ranging from 0.6 to 0.8 is appropriate for a regressive task.

### 2.2.2 Error Evaluation Methods

The three most commonly used error evaluation methods are the mean absolute error (MAE), the root mean square error (RMSE), and the mean squared error (MSE). Each method utilizes a comparison that measures the difference between the predicted and observed values. The formulas for each of these methods are formatted based on their namesake. The equations for the mean absolute error, root mean square error, and the mean squared error are shown in Equations 2, 3, and 4, respectively. (Chicco, Warrens, & Jurman, 2021)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2}$$

$$RMSE = \sqrt{\sum_{i=1}^{n}\frac{(y_i - \hat{y}_i)^2}{n}} \tag{3}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{4}$$

The $n$ in the equations above denotes the number of elements, the $y_i$ is the observed value at iteration $i$, and the $\hat{y}_i$ is the predicted value at iteration $i$.

These models are formatted similarly; however, each has its specialties and excels at their intended application. Mean absolute error is used in situations when each error benefits from being weighed equally. MAE penalizes large and small errors equally, and as such, this error method is less sensitive to outliers in the data. Root mean square error is the most commonly used evaluation metric in situations where a normal distribution is expected. In contrast to MAE, RMSE is sensitive to outliers in data, giving more weight to larger errors. Mean squared error is used in similar situations to root mean squared error with the benefit of being differentiable. This error method can be applied to models that iteratively utilize derivatives to minimize their loss function. For example, the Gradient Boosting Machine utilizes an iterative Gradient Descent. The most notable drawback of using MSE is that it changes the scale of the original inputs, making it more difficult to interpret than the other two metrics. (Chicco, Warrens, & Jurman, 2021).

### 2.2.3 Loss Functions

This loss function is comprised of a known output value based on the current iteration, $y_i$, and a guessed value, ρ. Traditionally for a regression task, the loss function is either the squared error or absolute error. These loss functions can be seen in Equations 5 and 6, respectively.

$$L(y, F) = (y - F)^2 \quad (5)$$

$$L(y, F) = |y - F| \quad (6)$$

These models fall short in certain aspects, however. The absolute error loss function cannot differentiate values at zero, and the squared error loss function is sensitive to outliers in

the data. The Huber loss function can alleviate both of these shortcomings, and as such, is the loss function implemented in this research. The loss function can be seen in Equation 7.

$$L_\delta\big(y, f(x)\big) = \begin{cases} \frac{1}{2}\big(y - f(x)\big)^2, for\ |y - f(x)| \le \delta \\ \delta \cdot \left(|y - f(x)| - \frac{1}{2}\delta\right), otherwise \end{cases} \quad (7)$$

The Huber loss function utilizes a piece-wise approach that implements the squared error and an alternate approach to the absolute error loss functions where they are best applicable. The only introduced variable in Equation 7 is $\delta$ which denotes the difference between the predicted and actual values. Therefore, the Huber loss function is denoted by $L_\delta$. (Huber, 1965).

## 2.3 Gradient Boosting Machine

Gradient Boosting Machine is an ensemble method that utilizes decision trees for classification or regression tasks. The procedure of the algorithm begins with the formula in Equation 8.

$$F_0(\boldsymbol{x}) = argmin_\rho \sum_{i=1}^{N} L(y_i, \rho) \quad (8)$$

Equation 8 specifies that the initial function estimation, an initial predicted value, $F_0(\boldsymbol{x})$, is composed of a function that finds a predicted value that minimizes the sum of a derivable loss function, $L$. A constant value initializes the model. The loss function is a function of both observed, $y_i$, and predicted, $\rho$, values. (Friedman, Oct., 2001).

### 2.3.1 Gradient Boosting Machine For Loop

The remainder of the Gradient Boosting Machine algorithm occurs within a for-loop. A for-loop specifies how many iterations a process repeats itself. Several procedures and equations run in each iteration before outputting a result. In this algorithm, the output is analyzed and

compared against the previous iteration's output. The for-loop portion of the algorithm can be seen below.

$$For\ m = 1\ to\ \boldsymbol{M}\ do:$$

$$\tilde{y}_i = -\left[\frac{\partial L\left(y_i, F(\boldsymbol{x}_i)\right)}{\partial F(\boldsymbol{x}_i)}\right]_{F(\boldsymbol{x})=F_{m-1}(\boldsymbol{x})}, i = 1, N \tag{9}$$

$$\boldsymbol{a}_m = argmin_{\boldsymbol{a},\beta} \sum_{i=1}^{N} [\tilde{y}_i - \beta h(\boldsymbol{x}_i; \boldsymbol{a})]^2 \tag{10}$$

$$\rho_m = argmin_{\rho} \sum_{i=1}^{N} L\left(y_i, F_{m-1}(\boldsymbol{x}_i) + \rho h(\boldsymbol{x}_i; \boldsymbol{a}_m)\right) \tag{11}$$

$$F_m(\boldsymbol{x}) = F_{m-1}(\boldsymbol{x}) + \rho_m h(\boldsymbol{x}; \boldsymbol{a}_m) \tag{12}$$

endFor

end Algorithm

The first line defines the range of the for-loop and, more precisely, defines the number of decision trees made. It specifies from an individual tree, $m$, to $\boldsymbol{M}$ total number of trees. Equation 9 computes a residual, $\tilde{y}_i$,or the observed values minus the predicted values, by taking a derivative of the loss function with respect to the predicted value. This derivative is called a gradient, from which the gradient boosting machine gets its namesake. The previous predicted value for the gradient is evaluated before comparing it against the samples, $i$, being tested. (Friedman, Oct., 2001).

The algorithm then creates a regression tree with splitting variables, $\boldsymbol{a}_m$, fitted to the residual values, as seen in Equation 10. The other new variables introduced here are the step direction, $h$, and the $\beta$, when in conjunction with the step direction, describe the best greedy step. (Friedman, Oct., 2001).

After splitting the residuals into regression trees, the algorithm makes an output value based on the current tree, $\rho_m$. The output value is determined by finding the argument about the predicted value that minimizes the loss function while considering the previous prediction. This process is shown in Equation 11. The algorithm then makes a new prediction, $F_m(\boldsymbol{x})$, using the previous prediction, output value, and step direction. (Friedman, Oct., 2001).

## 2.4 Deep Learning Overview

A further subset of machine learning is Deep Learning (DL). The main difference between machine learning and deep learning is that deep learning utilizes a series of hidden layers alongside weights and biases to make predictions. Another critical aspect is that deep learning introduces nonlinearity into the algorithm. This nonlinearity means that the models referred to as Neural Networks (NNs) can exceed the limitations of solely linear models, grasping hierarchical representations and capturing interactions between features. The training procedure in machine learning models needs to be explicitly defined, as seen in the Gradient Boosting Machine; however, the nonlinearity in deep learning neural networks allows for the ability to create its own unique training procedure. (Dhande, 2020).

### 2.4.1 Multi-layered Perceptrons

The most basic unit of a deep learning algorithm is called a perceptron. A perceptron comprises of inputs, weights, a weighted sum or input function, and a nonlinear function sequentially processed before generating an output. Chaining perceptrons creates a model denoted as a Multi-Layered Perceptron (MLP). The most basic of MLPs is a feed-forward system in which information is never backpropagated, meaning there is no feedback or improvement from the information obtained by the previous iteration. As mentioned previously,

the Gradient Boosting Machine invalidates the I.I.D. assumption. Sequence-based neural networks needed to be developed to make predictions that previous data can influence or reiterate using time-series-based data. (Bento, 2021).

### 2.4.2 Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNNs) are the first significant time-series-based neural network. Recurrent neural networks build upon multi-layered perceptrons by utilizing the same architecture while introducing recurrence. This recurrence means that one perceptron's result directly influences the next's result in the chain. A recurrent neural network is characterized by the formula seen in Equation 13. (Rumelhart, Hinton, & Williams, 1986).

$$\hat{y}_t = f(x_t, h_{t-1}) \qquad (13)$$

This equation states that the prediction, $\hat{y}_t$, is a function of the current input, $x_t$, and the previous step's, $h_{t-1}$, information. Recurrent neural networks are weak to long-term dependent datasets due to a recurring problem in deep learning known as the exploding and vanishing gradient. Exploding is when too large of a value is passed into the gradient, which heavily skews the prediction to output a significantly larger value. Vanishing refers to the opposite scenario where too small of a value is passed, resulting in a prediction heavily skewed by outputs with too small of a value. (Rumelhart, Hinton, & Williams, 1986). Figure 9 shows the basic framework for a recurrent neural network.

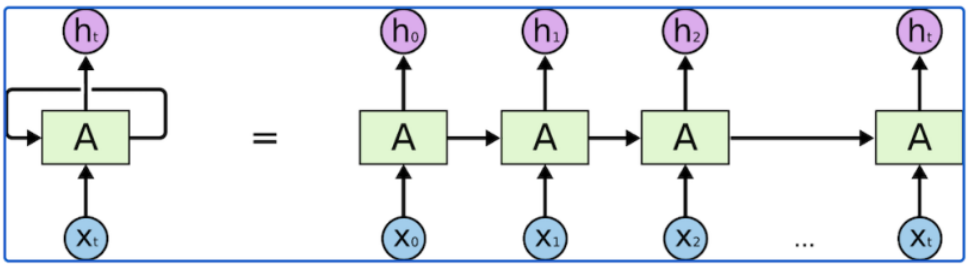



Figure 9: Recurrent neural network framework (Rumelhart, Hinton, & Williams, 1986).

### 2.4.3 Long-Short Term Memory (LSTM)

A neural network specially designed to target some of the shortcomings of a recurrent neural network is the long-short term memory NN. The critical difference between the two models is the introduction of a memory cell in place of recurrence. This memory cell is capable of retaining information and updating an internal state. The internal state is regulated via three gates, an input, output, and forget gate. The input gate determines how many datapoints get stored in the memory cell. It utilizes a sigmoid activation function along with the current input and previous hidden state to output a binary response for each element. As the name implies, the forget gate determines how much previous information gets forgotten before the memory is updated. The output gate utilizes the same parameters as the input gate but is used to generate outputs. (Hochreiter & Schmidhuber, 1997). Figure 10 depicts the framework for the LSTM model.

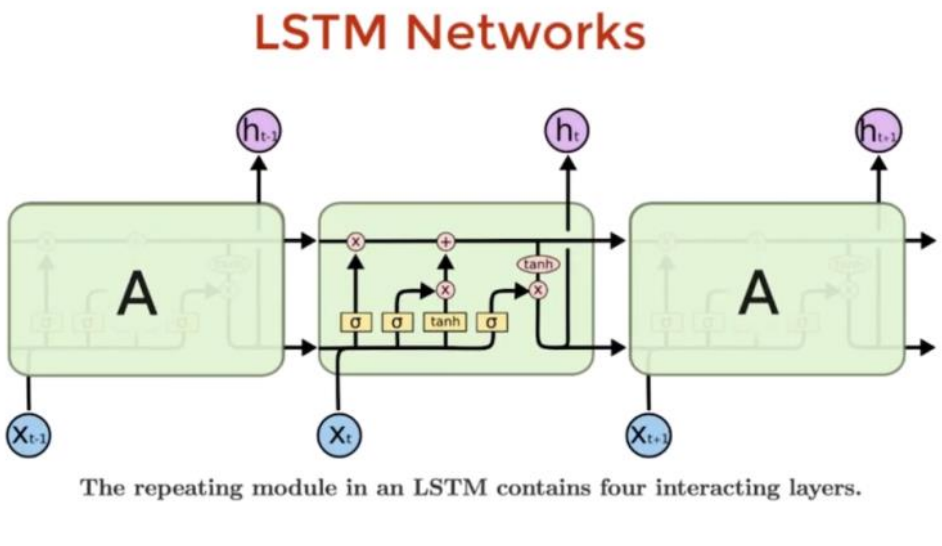



Figure 10: Long-short term memory neural network (Hochreiter & Schmidhuber, 1997).

While the long-short term memory model can store data, it has shortcomings. The LSTM model requires a higher training time and memory usage than recurrent neural networks. The other major drawback is that LSTM models are sensitive to initial conditions (Hochreiter & Schmidhuber, 1997).

### 2.4.4 Encoder-Decoder

The next significant evolution in sequence-based neural networks sought to accommodate larger datasets by re-contextualizing them. The Encoder-Decoder neural network is the model created capable of these feats. The encoder refers to a procedure that can reshape the data to create a context matrix, sometimes referred to as a concatenated feature map, using a different sequence-based neural network. The most commonly implemented neural networks are recurrent neural networks and long short-term memory models, both discussed in this chapter. This context matrix gets inputted to the decoder, which utilizes the information to generate an output sequence. The decoder usually takes the form of a recurrent neural network. One of the

highlighting aspects of decoders is that they utilize an attention mechanism to differentiate sections of the input sequence and determine what data gets prioritized. The significant drawbacks of the encoder-decoder neural network are that it can struggle with chains of dependencies, tend to oversimplify the inputs, and has a relatively long training time (Sutskever, Vinyals, & Le, 2014).

### 2.4.5 Transformer Architecture

Following the encoder-decoder model, the Transformer neural network was developed in 2017 by Vaswani et. Al. The model adapts the encoder-decoder model, emphasizing the self-attention mechanism. The first step that the Transformer neural network takes is input embedding. Once the inputs are embedded, the model then positionally encodes them. Post-encoding, a multi-head attention mechanism is activated. This attention mechanism decides the most relevant data from the given set of inputs before passing it into a feed-forward network. The data is then normalized and passed through an output layer that re-contextualizes it. The losses are then calculated. (Vaswani, et al., 2017). Figure 11 shows the framework for the Transformer neural network.

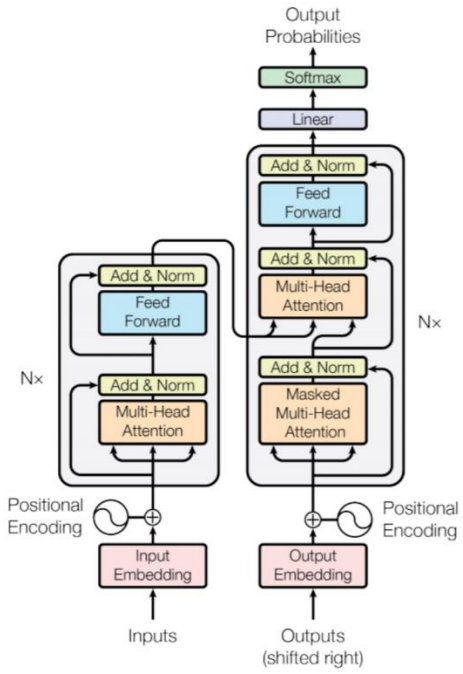



Figure 11: Transformer neural network architecture (Vaswani, et al., 2017).

The Transformer neural network has notable limitations, however. Due to its quadratically computing iteration method, the self-attention mechanism maximizes the memory usage per layer and the overall time complexity. The model also struggles with longer inputs due to this significant memory cost, and outputs are relatively slow when predicting longer sequence lengths (Vaswani, et al., 2017).

## 2.5 Informer Architecture

The Informer neural network was developed to mitigate the limitations of the Transformer neural network. The Informer neural network was created in 2020 by Zhou et al. and has three notable primary characteristics. These are the: ProbSparse self-attention mechanism, self-attention distilling, and a generative style decoder (Zhou, et al., 2021). Figure 12 depicts the framework for the Informer neural network.

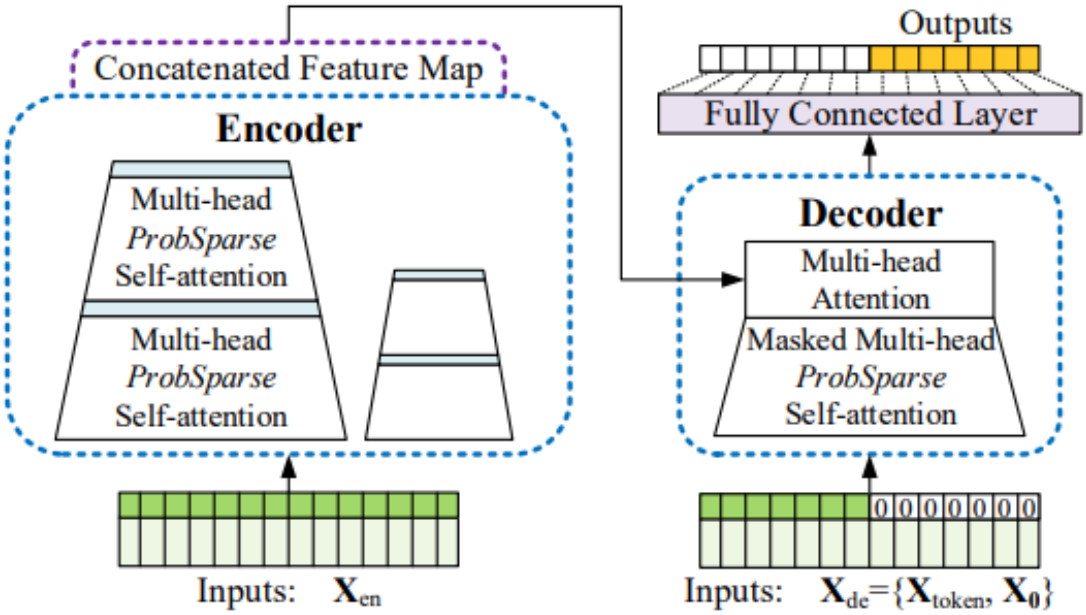



Figure 12: Informer neural network framework (Zhou, et al., 2021).

The model feeds the input data into an encoder where multi-head ProbSparse self-attention mechanisms get iteratively run before a concatenated feature map gets created. This feature map then gets passed into a decoder where more multi-head attention mechanisms are utilized. The data is then re-contextualized, and a prediction and calculated losses are output. After loss calculation, some data gets backpropagated to train the subsequent iteration. The remaining portions are set aside for validation and testing of the trained model (Zhou, et al., 2021).

### 2.5.1 Informer Neural Network: ProbSparse Self-Attention Mechanism

The ProbSparse self-attention mechanism is dependent on the i-th query's attention, which is described in Equation 14 (Zhou, et al., 2021).

$$A(\boldsymbol{q_i}, \boldsymbol{K}, \boldsymbol{V}) = \sum_j \frac{k(\boldsymbol{q}_i, \boldsymbol{k}_j)}{\sum_l k(\boldsymbol{q}_i, \boldsymbol{k}_l)} \boldsymbol{v}_j \quad (14)$$

In this equation, $\boldsymbol{q}_i, \boldsymbol{k}_i, \boldsymbol{v}_i$ refers to the i-th row in $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}$, where $\boldsymbol{Q} \in \mathbb{R}^{L_Q \times d}, \mathbf{K} \in \mathbb{R}^{L_K \times d}, \mathbf{Q} \in \mathbb{R}^{L_V \times d}$. The $d$ variable refers to the input dimension. The actual ProbSparse is represented by Equation 15 (Zhou, et al., 2021).

$$A(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = Softmax\left(\frac{\bar{\boldsymbol{Q}}\boldsymbol{K}^T}{\sqrt{d}}\right)V \quad (15)$$

In this equation, the only variable introduced is the $\bar{Q}$which denotes the sparse matrix.

### 2.5.2 Informer Neural Network: Encoder

The encoder of the Informer neural network utilizes a sequence input that is formatted into the formula seen in Equation 16 (Zhou, et al., 2021).

$$\boldsymbol{X}_{en}^t \in \mathbb{R}^{L_x \times d_{model}} \quad (16)$$

Once the sequence has been input, the self-attention distilling is performed. This procedure is represented in Equation 17 (Zhou, et al., 2021).

$$\boldsymbol{X}_{j+1}^t = MaxPool(ELU(Conv1d([\boldsymbol{X}_j^t]_{AB}))) \quad (17)$$

This process is accomplished by using a one-dimensional convolution passed through the ELU activation function. This data is then MaxPool-ed (Zhou, et al., 2021).

### 2.5.3 Informer Neural Network: Decoder

The decoder for the Informer neural network has its feed contextualized by a start token and a placeholder for the target sequence (Zhou, et al., 2021). The equations defining the start tone, placeholder and decoder feed are listed in equations 18, 19, and 20 respectively.

$$\boldsymbol{X}_{token}^{t} \in \mathbb{R}^{L_{token} \times d_{model}} \tag{18}$$

$$X_{\mathbf{0}}^{t} \in \mathbb{R}^{L_{token} \times d_{model}} \tag{19}$$

$$\boldsymbol{X}_{de}^{t} = Concat(\boldsymbol{X}_{token}^{t}, X_{\mathbf{0}}^{t}) \in \mathbb{R}^{(L_{token}+L_{y}) \times d_{model}} \tag{20}$$

The final notable piece of information pertaining to the decoder of the Informer neural network is that it utilizes generative inference. This inference means that it can predict the output through a simple forward process rather than a more dynamic decoding method (Zhou, et al., 2021).

# CHAPTER III

# EXPERIMENTAL SETUP AND PROCEDURES

## 3.1 Laboratory Overview

The University Transportation Center for Railway Safety (UTCRS) has worked to replicate freight operating conditions to monitor tapered roller bearings' condition accurately. This replication is facilitated by dynamic bearing test rigs fitted with temperature, vibration, and load sensors designed and manufactured by the UTCRS. These sensors go mounted on or built into the bearing adapters, named Smart Adapters, that act as a medium to transfer the load to the tapered roller bearings. The operating ranges tested reflect service conditions. The two most common load conditions are unloaded and fully loaded, where each bearing is subjected to 17% (5,848 lbs.) and 100% (34,400 lbs.) of the total load, respectively. The operating speeds range from 25 to 85 mph.

### 3.1.1 Class and Specifications of Bearings

The University Transportation Center for Railway Safety (UTCRS) primarily tests the two most commonly used classes of bearing for freight rail tracks in North America, class F, and K. However, the dynamic bearing test rigs at the UTCRS can test classes K, F, E, and G. The dimensions and load capabilities according to the Association of American Railroads (AAR) standards of these bearings are listed in Table 1.

Table 1: AAR standards: bearing classes.

| Class | Size [mm] | Size [in] | Load [kN] | Load [lbf.] |
|---|---|---|---|---|
| **Class F** | 165x305 | 6.5x12 | 153 | 34,400 |
| **Class K** | 165x229 | 6.5x9 | 153 | 34,400 |

### 3.1.2 Bearing-Axle Assembly

The bearing assembly process first begins with component selection. Depending on the parameters of the test, the components are selected based on whether the parts meet the AAR standard tolerances. The one parameter that stays consistent amongst every type of experiment conducted is that the diameter of the inner ring of the bearing assembly must be ±6.1875 inches for class K and F bearings. This tolerance ensures that the bearings can be properly press-fit due to the diameter of both correlating axles being 6.1915 inches. Inner rings out of tolerance can result in the bearings spinning independently of the axle during the testing process. The assembly process remains the same between classes of bearings; however, the height of the components differs, and the amount of packed grease gets scaled to the appropriate volume. Once an inner ring with an acceptable inner diameter has been selected, the rollers and raceway get inspected for defects. Post inspection, the rollers are inserted into the cage, ensuring that the tapered end falls into the appropriate location, as shown in Figure 13.

Figure 13: Tapered rollers in a post-experiment cage assembly.

The inner ring is then manually placed into the cage assembly. This results in what is termed as a cone assembly. The roller spacing then gets measured via a feeler gauge with the tolerance being 0.060 thousandths of an inch. This process gets repeated to make a second cone assembly. An appropriate cup is then selected to suit the testing parameters. It is common for spalled cups to be chosen for experiments to determine their effects. The cup is then placed into a 20-ton professional hydraulic jack, where a pre-assembled cone assembly gets inserted tapered side down. The placement in such a manner ensures contact between the inner ring raceway, the tapered rollers, and the outer ring raceway. Once the cone assembly is seated, the grease is packed based on a grease squirt chart shown in Table 2.

Table 2: Bearing grease squirt chart.

| **Bearing Grease Squirt Chart** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bearing Class | Top | | Center | | Bottom | | Total | |
| | OZ | Squirts | OZ | Squirts | OZ | Squirts | OZ | Squirts |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| D | | | | | | | | |
| E | 7 | 9/Slot+10 | 0 | 0 | 7 | 9/Slot+10 | 14 | 434 |
| F | 6.5 | 8/Slot+18 | 9 | 279 | 6.5 | 8/Slot+18 | 22 | 682 |
| G | 6 | 8/Slot+2 | 0 | 0 | 6 | 8/Slot+2 | 12 | 372 |
| K | 6.5 | 8/Slot+17 | 0 | 0 | 6.5 | 8/Slot+17 | 13 | 403 |
| *Assuming: 31 squirts = 1 oz & 23 slots/cone assembly | | | | | | | | |

The bearing is then sealed via the hydraulic jack using a pressure of 1,000 psi. The bearing is then re-inspected, checking for a firm seal to ensure no leakages. Finally, a wear ring gets inserted into the sealed portion with the tapered side facing outwards. The entire assembly then gets flipped, and the process gets repeated for the second cone assembly, with a spacer ring placed within the two cone assemblies. This process is repeated per bearing tested.

Once the bearings are assembled, they are pressed onto the axle using a Tinius Olsen Super L 300-ton hydraulic press. The bearings are press fit to the axle with forces ranging from 150,000 to 250,000 lbs. across the four bearings. The hydraulic press is controlled via software called MTEST QUATTRO. MTEST QUATTRO can dictate the bearing's step size, or displacement per minute and records the force output throughout the process. The maximum force is logged for comparison to the AAR standards to ensure that the bearings are correctly press fit. The order in which the bearings get pressed onto the axle is determined by whether they are control bearings or the focus of the particular experiment. The control bearings are placed in the outermost sections of the axle, labeled positions 1 and 4. The bearings to be observed throughout the experiment are placed in the two center positions labeled 2 and 3 to simulate top loading conditions experienced in the field. Figure 14 shows the positions of the bearings.

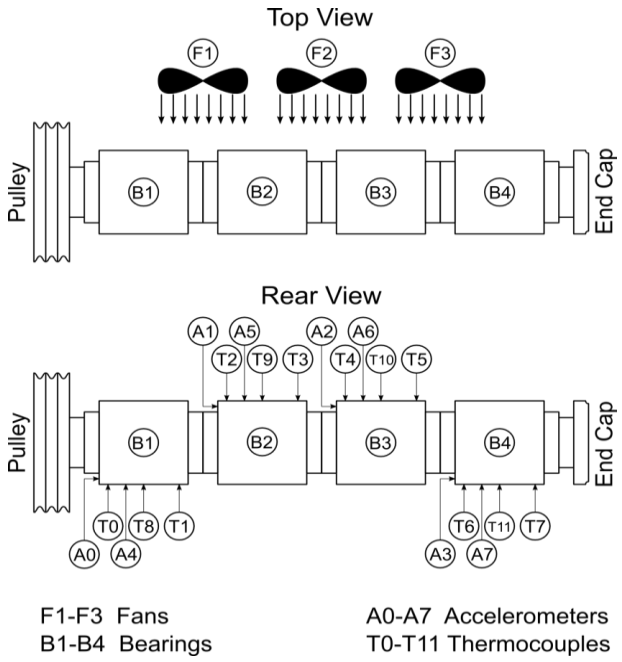



Figure 14: Bearing placement and instrumentation diagram.

### 3.1.3 Bearing Instrumentation

The vibration profile is recorded using two separate accelerometers labeled ADXL and PCB. The ADXL accelerometers go mounted on both the Smart Adapter™ (SA) and Mote (M) locations, with the PCB on the adapter flange. These accelerometers and their mounting locations can be seen in Figure 15.

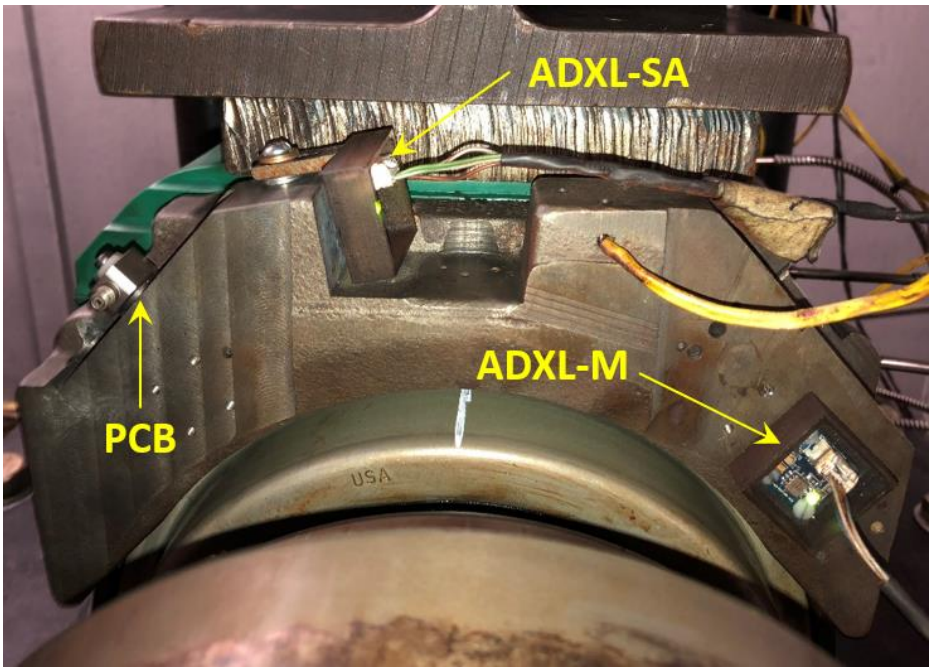



Figure 15: Mounted accelerometers.

The accelerometers record data at a frequency of 5,120 Hertz for 16 seconds every 10 minutes throughout the experiment.

The temperature profiles of the bearings are recorded using two spring-loaded bayonet-type thermocouples per bearing and eight K-type thermocouples on the dynamic test rig. The bayonet-type thermocouples are inserted into the bearing to touch the inboard and outboard raceways. The K-type thermocouples get placed between these two bayonets on the B2 and B3 adapters, and two are designated to measure the ambient temperature before and after the wind has reached the bearings. These locations can be seen in Figure 14. The thermocouples record data every twenty seconds at a sampling rate of 128 Hertz.

### 3.1.4 Dynamic Bearing Test Rigs

Two main configurations for the dynamic bearing test rigs at the UTCRS are Single Bearing Tester (SBT) and Four Bearing Testers (4BTs). The primary difference between these testers is, as their names imply, the number of bearings they can test. The SBT most accurately replicates a field wheelset because it acts as a cantilevered beam. It can also replicate impact data, which mimics the wheel-to-rail contact of a wheelset experiencing a flat. As the name suggests, a flat is when a portion of a wheel flattens due to sudden braking or rail obstructions. The two outer bearings on the 4BT are subject to bottom loading and compared to the top loaded conditions on the two center bearings. One of the 4BT is located within an environmentally controlled chamber, where the temperature can be manipulated between -40°F to 120°F. The dynamic bearing test rigs utilize a variable frequency driver (VFD) for regulating the output of a ten-hp motor that drives the bearing-axle assembly. The speed of the axle ranges from 234 to 796 rpm, which correlates to track speeds of 25 to 85 mph. Table 3 shows the axle speed to equivalent track speed conversion.

Table 3: Axle speed to equivalent track speed.

| **Axle Speed [rpm]** | **Equivalent Track Speed [mph]** | **Equivalent Track Speed [km/h]** |
|---|---|---|
| **234** | 25 | 40 |
| **280** | 30 | 48 |
| **327** | 35 | 56 |
| **374** | 40 | 64 |
| **420** | 45 | 72 |

| | | |
|---|---|---|
| **467** | 50 | 80 |
| **498** | 53 | 85 |
| **514** | 55 | 89 |
| **560** | 60 | 97 |
| **618** | 66 | 106 |
| **700** | 75 | 121 |
| **796** | 85 | 137 |

The dynamic bearing test rigs can apply up to 150% of the AAR standard operating load of any bearing class they test. A load controller handles this operation by regulating the pressure entering a hydraulic cylinder. The load controller is capable of accuracies within 1% of the desired load. It collects 52 samples every second and corrects for deviations. As mentioned, the bearings are most commonly tested at either unloaded (17% full load) or fully loaded (100% full load) conditions. Figure 16 depicts the dynamic bearing test rigs.

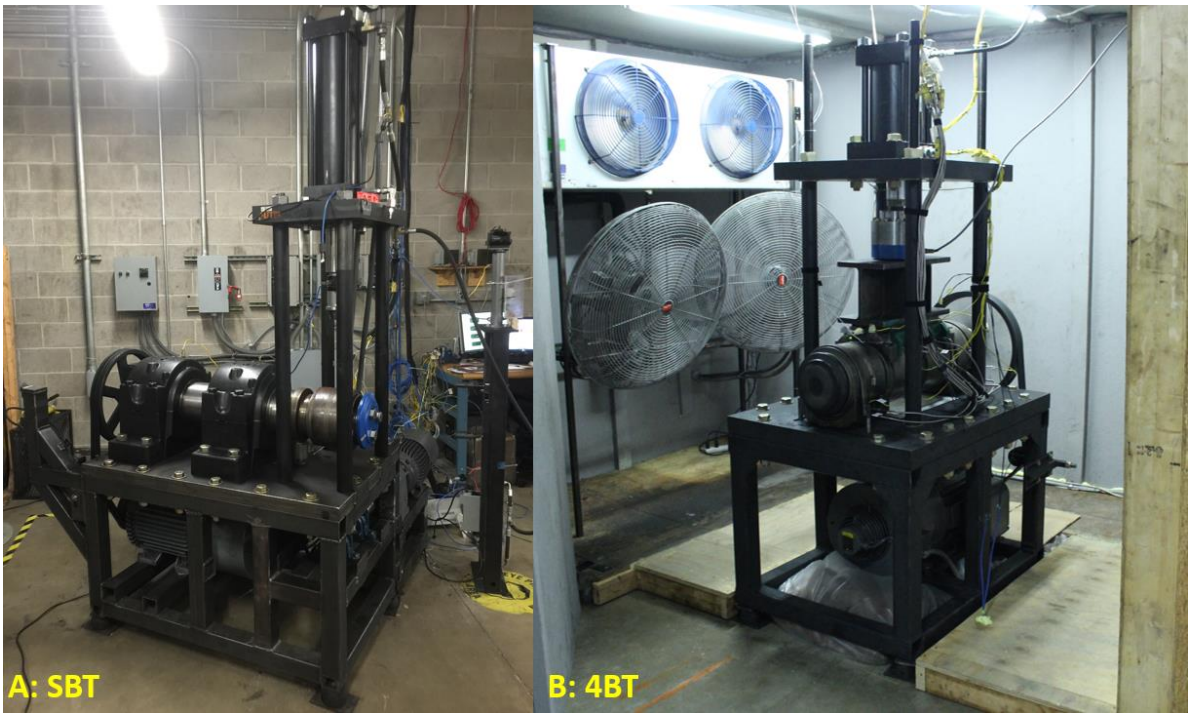



Figure 16: A. Single bearing test rig., B. 4-bearing test rig.

### 3.2 Data Overview

The University Transportation Center for Railway Safety has collected and analyzed various forms of data through various instruments. The data obtained by the sensors is collected by a data acquisition board and then analyzed after being sent to a central processing unit. As mentioned, a combination of vibration, temperature, and load profiles best characterizes tapered roller bearing condition monitoring. As such, the data primarily acquired by the previously mentioned sensors reflect these profiles. More specifically, the UTCRS collects and processes: accelerometer data measured in g, harvesting voltage, PCB data measured in g, temperature data measured in Celsius, VFD voltage, and load cell data in lbs.

### 3.2.1 Data Acquisition

The data is passed through and recorded by a National Instruments (NI) PXIe-1062Q data acquisition system (DAQ). The data gets written to text-based files using a software known as LabVIEW™. Specifically, the temperature and vibration data are recorded on a NI TB-2627 and an 8-channel NI PXI-4472B card, respectively. As mentioned in their respective sections, the vibration and temperature responses are obtained at a sampling rate of 5.12 kHz for 16 seconds every 10 minutes and 128 Hz every 20 seconds, respectively. The load cell data is recorded straight from the DAQ at the same rate as the temperature profile. Once the data is collected, it gets transmitted to a central processing unit for analysis.

### 3.2.2 Data Processing

The analysis of the data is conducted through a software known as MATLAB. Three different levels of analysis are conducted at the UTCRS for condition monitoring of bearings. The first are plots that compare the vibration and temperature profiles to statistically determined thresholds and correlations. Level 1 analysis intends to detect a defect within the bearing. Figure 17 shows an example of these plots.

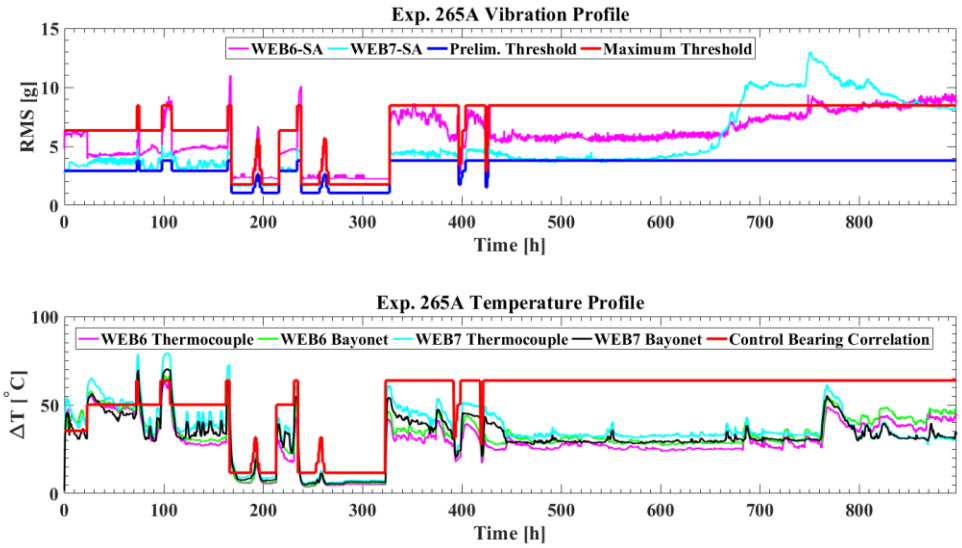



Figure 17: UTCRS level 1 analysis.

The vibration plot depicts the root mean square (RMS) of the vibration profile plotted against a preliminary and maximum threshold. The thresholds get defined by equations dictated by a first-order line of fit with a 95% confidence preliminary interval and a 45% confidence maximum interval. When the vibration responses are under the preliminary threshold, the bearing is statistically healthy and, therefore, in ideal operating conditions. When the response is between the two thresholds, the bearing is no longer considered ideal; however, it is still within operable conditions with the potential for defects to develop. A response above the maximum threshold for a prolonged period (at least five hours) heavily implies the existence of a defect. It should be further analyzed with the next level of analysis. The temperature plot is read similarly by comparing the current temperature of the bearings to a control bearing correlation rather than a statistically generated threshold. The experiment depicted in Figure 17 exemplifies the primary reason for this research's focus on vibration. The vibration profile detected a bearing defect, while the temperature profile had yet to.

Once a defect gets detected using level 1 analysis, level 2 analysis can be conducted to determine what component within the bearing has developed a defect. Level 2 analysis consists of an algorithm that uses a power spectral density (PSD) plot. The plot uses six rotation frequencies that describe an overall square of the frequency domain's magnitudes and the tapered roller bearing's subcomponents. Interpreting these plots allows for the detailing of which component is defective. Figure 18 shows PSD plots depicting the four different signal patterns representing the different defect locations.

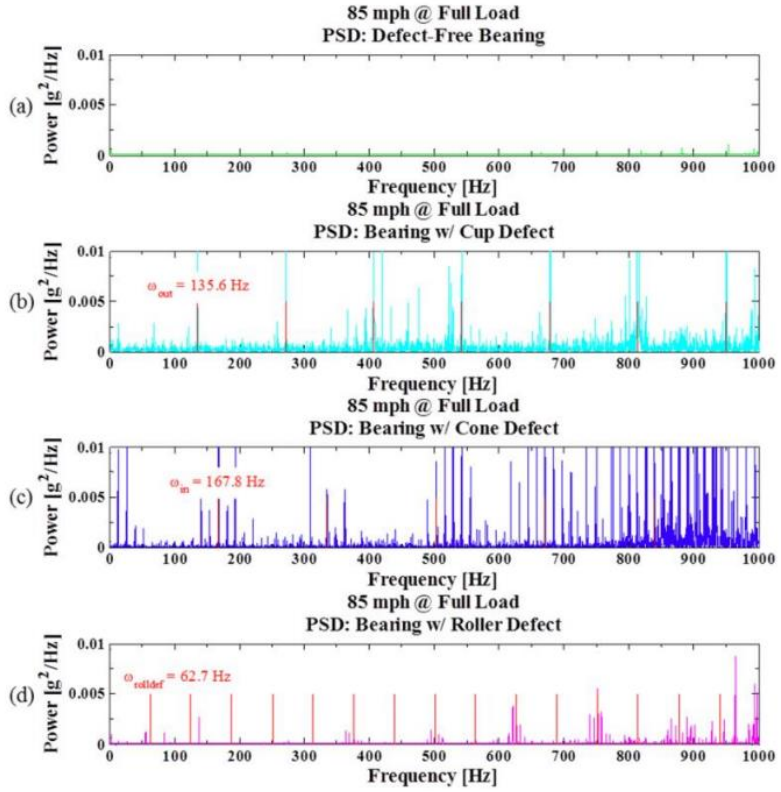



Figure 18: PSD plot detailing: a. defect-free bearing, b. cup defect, c. cone defect, and d. roller defect.

The final level of analysis conducted for the condition monitoring of a bearing at the UTCRS is level 3, where the size of the defect gets estimated without the need to seize operation.

This process is done by estimating the correlation between the RMS of the vibration profile to statistically generated curves. These statistically generated curves exist for the cup and the tapered roller bearing assembly components and can be visualized in Figure 19 and Figure 20.

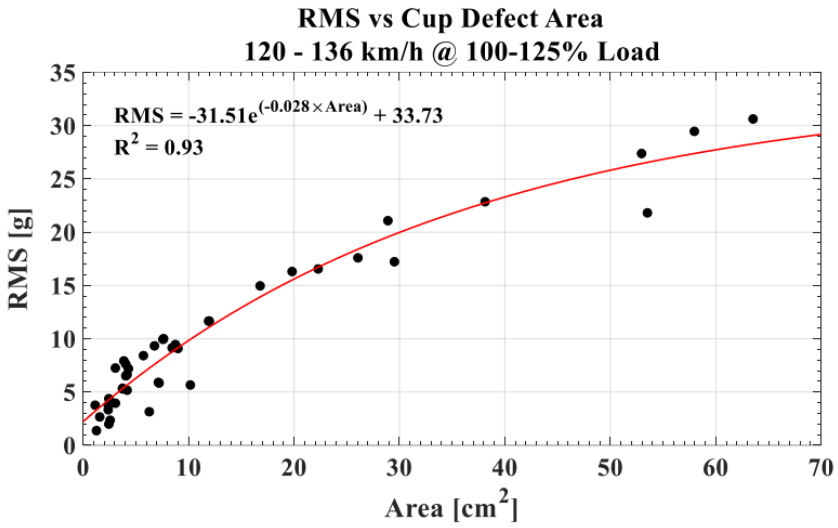



Figure 19: Level 3 analysis for a cup.

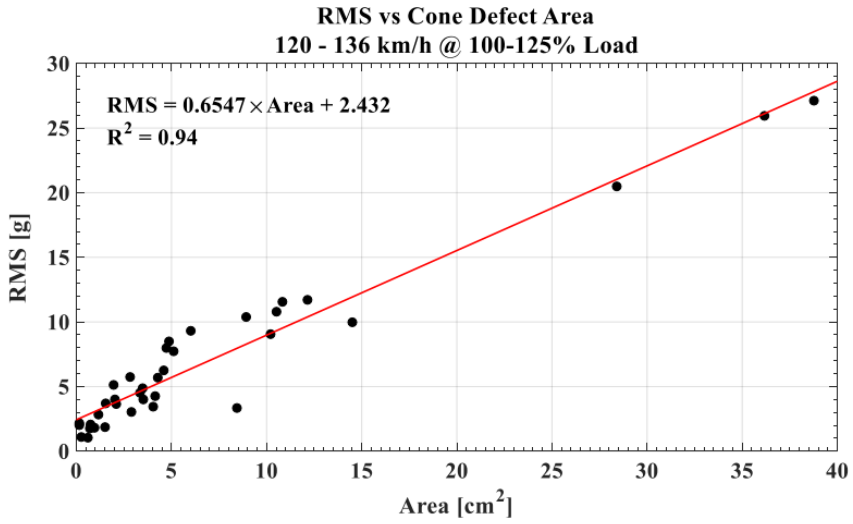



Figure 20: Level 3 analysis for a cone.

# CHAPTER IV

# METHODOLOGY AND MODEL SPECIFICATIONS

## 4.1 Gradient Boosting Machine Specifications

This implementation of the Gradient Boosting Machine aims to make a scheduled prediction of the RMS of the vibration profile. To accomplish this, an expansive dataset was created, input features were defined, and hyperparameters were optimized.

### 4.1.1 Gradient Boosting Machine Dataset

The University Transportation Center for Railway Safety (UTCRS) sorted the data for their experiments into folders labeled by their chronological experiment number and potentially a letter defining the experiment's iteration. For example, if it is the second iteration of experiment 250, the folder holding its data would be labeled 250B. Experiments are split into iterations when the same components are utilized but split into different experiments based on either passing a mileage threshold or implementing varying testing conditions. The data recorded in these folders are the following: accelerometer, temperature, motor power, harvesting voltage, VFD voltage, and load cell data that the instrumentation detailed in Chapter 3 records.

The current dataset utilizes 74 of the most recent experiments that originally span 211,430 data points, with the dataset utilizing 196,537 of those data points. As mentioned, the

dataset was refined to remove outliers that could skew the prediction. Table 4 details the experiments utilized for the dataset.

Table 4: Dataset detailing experiments used.

| **Experiment ID** | | | | |
|---|---|---|---|---|
| 264A | 250C | 229C | 222 | 214C |
| 262D | 250D | 228 | 221 | 214D |
| 261A | 249 | 227A | 220 | 214E |
| 259A | 236 | 227B | 219 | 213 |
| 259B | 235 | 227C | 219B | 213B |
| 259C | 234 | 227D | 218 | 212 |
| 258A | 232 | 227E | 217 | 212B |
| 256A | 231 | 227F | 217B | 211 |
| 252 | 230A | 226A | 216 | 211B |
| 251A | 230B | 226B | 216B | 211C |
| 251B | 230C | 226C | 216C | 210 |
| 251C | 230D | 226D | 215 | 209 |
| 251E | 230E | 225 | 215B | 209B |
| 250A | 229A | 224 | 214 | 209C |
| 250B | 229B | 223 | 214B | |
| **Current Dataset** | | | | |

These predictions aim to have a generalized output capable of predicting when defects are likely to occur and not necessarily replicate every oscillation that an actual vibration profile would

experience. To this end, the dataset was refined to filter out any notable spikes and outliers in the vibration RMS profile that could skew the predictions. This dataset was utilized for every model; however, generalizations and features varied depending on the model.

### 4.1.2 Gradient Boosting Machine Feature Set

The features, or set of inputs, for the Gradient Boosting Machine implemented for this research focus on improving from the project's predecessor. As mentioned, the previous attempt to implement machine learning to predict condition monitoring of bearings conducted by Leonel Villafranca utilized only the mileage, speed, and load condition as input features. (Villafranca, 2022). This research implemented a more comprehensive set of input features that considers the interplay between the individual features. Table 5 details the features utilized for the Gradient Boosting Machine in this research.

Table 5: Gradient Boosting Machine feature set.

| **Feature Set** |
|---|
| Miles ran Unloaded at Low Speeds (MULS) |
| Miles ran Unloaded at Common Speeds (MUCS) |
| Miles ran Unloaded at High Speeds (MUHS) |
| Miles ran Fully Loaded at Low Speeds (MFLS) |
| Miles ran Fully Loaded at Common Speeds (MFCS) |
| Miles ran Fully Loaded at High Speeds (MFHS) |
| Mile ran Overloaded at Low Speeds (MOLS) |
| Miles ran Overloaded at Common Speeds (MOCS) |
| Miles ran Overloaded at High Speeds (MOHS) |

| Gradient Boosting Machine |
| --- |

Unloaded conditions refer to when 5,848 lbs. or 17% of the weight of a fully loaded rail car gets applied to the tester. A fully loaded rail car experiences 34,400 lbs. is called 100% load. Overloaded conditions occur when the force exceeds the 100% load condition. The range for low speed in this feature set is below 40 miles per hour. Common speeds range from 40 to 60 miles per hour, and high speeds are anything exceeding 60 miles per hour.

### 4.1.3 Gradient Boosting Machine Hyperparameters and Optimization

Hyperparameters are the variables used to train machine learning algorithms. Common hyperparameters for machine learning models include learning rate, gamma, maximum depth, number of estimators, regularization lambda, and minimum samples per leaf. The learning rate dictates the rate at which the model learns; the higher the variable, the faster it learns. High learning rates run the risk of skipping past minima that can outperform the one it selects. The gamma acts as a cut-off variable. If an internally defined scoring system passes the gamma value, the data will not contribute to training the model. The maximum depth variable determines the number of branches, or level of specificity in a way, that the model will have. The number of estimators is, as the name implies, and the regularization lambda is a parameter that intentionally lowers the accuracy of the training dataset to obtain a higher accuracy in the testing dataset. Regularization is a means to prevent overfitting, in other words.

The hyperparameters used in this research are the: number of estimators, learning rate, maximum depth, minimum samples per leaf, minimum samples split, and subsample. The values for these hyperparameters were identified and optimized via the use of RandomSearchCV. (Buitinck, et al., 2013). RandomSearchCV is a means to iteratively test a random set of parameters in a defined range to output the combination that generates the highest score. The

parameter distribution is shown in Table 6. The RandomSearchCV in this research was run with fifty iterations to further ensure the efficacy of the combination output.

Table 6: Parameter distribution for RandomSearchCV.

| Number of Estimators | randint(100, 1000) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Learning Rate | 0.001 | 0.01 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| Maximum Depth | randint(3, 35) | | | | | | | | | | |
| Minimum Samples per Leaf | randint(1, 35) | | | | | | | | | | |
| Minimum Samples Split | randint(2, 35) | | | | | | | | | | |
| Subsample | 0.6 | | 0.7 | | 0.8 | | | 0.9 | | 1.0 | |
| Parameter Distribution | | | | | | | | | | | |

The scoring method used in this research was the coefficient of determination. This was done to ensure the model lowers the variation between the predicted and actual values. Table 7 shows the optimized hyperparameters for the Gradient Boosting Machine.

Table 7: Gradient Boosting Machine optimized hyperparameters.

| **Optimized Hyperparameters** | |
|---|---|
| Number of Estimators | 694 |
| Learning Rate | 0.1 |
| Maximum Depth | 6 |
| Minimum Samples per Leaf | 10 |
| Minimum Samples Split | 3 |
| Subsample | 1.0 |

## 4.2 Informer Neural Network Specifications

The target variable for this implementation of the Informer neural network is the same as the Gradient Boosting Machine implementation, being the RMS of the vibration profile of tapered-roller bearings. Similarly, to facilitate this prediction, a dataset, and feature set were prepared, and hyperparameters were optimized.

### 4.2.1 Informer NN Dataset

The Informer neural network can support training on an expansive dataset and on an individual case-by-case basis. The purpose of this model implementation is to predict ongoing routes, and as such, this research utilizes a case-by-case dataset. Specifically, the model is trained using the data from the most recent experiment at the time of writing this, experiment 265A. This dataset consists of 5,434 data points. This is over 37 days' worth of data as a frame of reference. The Informer neural network takes time series data as its inputs, so the time history of the experiment per data point had to be included, unlike the Gradient Boosting Machine.

The data was preprocessed before being input into the model. The data preprocessing was done by normalizing the target variable and all features individually and Winsorization for just the target variable, the RMS of the vibration profile. Equation 21 details the formula used for normalizing the data. In that equation, the $Z$ refers to the normalized data, $x$ refers to the current data point, $\mu$ is the mean of the data, and $\sigma$ is the standard deviation. Winsorization refers to setting values below the $5^{th}$ percentile to the $5^{th}$ percentile and above the $95^{th}$ percentile to the $95^{th}$ percentile. Winsorization benefits the model's performance, given that it addresses the

nature of the vibration profile to spike in data whenever it generates a defect. The overall trend of the data is more critical to the prediction than the amplitude of the response. The standard deviation for experiment 265A was 2.01653, and the mean was 5.83447.

$$Z = \frac{x - \mu}{\sigma} \tag{21}$$

### 4.2.2 Informer NN Feature Set

The Informer neural network utilizes the basic forms of the feature set that the Gradient Boosting Machine uses. This is because the model struggles to handle trailing zero values, which occurs frequently due to the comprehensive description of each possible scenario when using the previous feature set. The Informer neural network is also capable of utilizing the target variable as one of the inputs for making the prediction. The actual features being utilized for this implementation of the Informer neural network can be seen in Table 8.

Table 8: Informer NN feature set.

| **Feature Set** |
|---|
| RMS of the Vibration Profile |
| Mileage |
| Speed (MPH) |
| Load (%) |
| **Informer Neural Network** |

### 4.2.3 Informer NN Hyperparameters and Optimization

The hyperparameters utilized in time series based neural networks vary depending on the complexity of the model. The hyperparameters utilized by the Informer neural network are the following: input sequence length, label length, prediction sequence length, encoder input size, decoder input size, output size, ProbSparse attention factor, dimension of the model, number of encoder heads, number of encoder layers, number of decoder layers, number of stacked layers, dimension of the feed-forward network, attention dropout, attention type, time feature embed type, activation function, whether to use distilling, whether to use output attention, mixing, padding, argument frequency, batch size, learning rate, loss, lradj, whether to use automatic mixed precision training, number of workers, number of iterations, whether to inverse the data, whether to scale the data, number of training epochs, number of patience epochs, and a description of the model's function. The values for these hyperparameters were manually optimized, the process of which can be seen in the Appendix. Table 9 shows the final configuration for these hyperparameters.

The input sequence length, label length, and prediction sequence length are the hyperparameters that describe the scenario for what is being predicted. The input sequence length dictates how much data is utilized to train the model. This research sets the argument frequency to 10T, meaning a ten-minute interval separates each data point. Therefore, the prediction sequence length values are interpreted with this frequency considered. For example, one of the input sequence lengths tested is 288, which refers to 2,880 minutes (48 hours or two days) worth of data used to train the model. The label length dictates the duration for which the predictions should be accurate. This research has a value set to 144, meaning that there are 1,440 minutes (24 hours or one day) worth of data used to compare the loss of the predicted values.

The prediction sequence length defines how many time steps into the future the model predicts. This parameter is also set to 144, meaning that the model predicts 1,440 minutes (24 hours or one day) into the future. This research keeps the sequence length to 288 and varies the label and prediction sequence lengths to 36, 72, and 144. These lengths correspond to six, twelve, and twenty-four hours.

The number of features dictates the encoder and decoder input sizes this implementation of the neural network utilizes to make a prediction. This research has it set to 4 as it uses the RMS of the vibration profile, the mileage, the speed, and the load condition to make its prediction. The output size refers to how many target variables there are. In this implementation, there is only one, the RMS of the vibration profile. The ProbSparse attention factor refers to the width of the model's attention. The dimension of the model refers to the shape of the tensors that represent the learnable parameters within the model. The number of heads refers to the number of parallel attention mechanisms the model runs. The number of encoder and decoder layers describe the complexity of the model through the number of layers the data gets passed through and then out of, respectively. The dimensions of the feed-forward network are similar to the dimensions of the model; however, they describe solely the portion that the feed-forward network makes up. The dropout variable is a means to regularize the data by establishing a probability for the data to be dropped out a unit or set to zero. The attention type dictates whether the ProbSparse attention mechanism is utilized. If it is set to full attention, the model acts as the Transformer neural network and considers the entire range of data in its attention mechanism.

The feature embed type refers to how the Informer neural network should interpret the time history of the data. It is set to fixed in this research, meaning that the model will not look into the times specified to try to generate a trend from that history and will only consider that

there is a fixed interval of time between data points. As mentioned, the frequency is set to 10T, meaning the fixed interval is set to 10 minutes between each data point. The GELU activation function is utilized in this model. This activation function incorporates portions of the sigmoid function and the Gaussian cumulative distribution function. The following three parameters take true or false inputs to dictate whether they are enabled. Distilling refers to a method the informer neural network implements to pass the data from a complex learning algorithm to a simpler one if applicable. Output attention refers to applying an attention mechanism in the decoding process. This research has it set to false as it would add complexity to the model, increasing the processing time and computational demand. The mixing parameter allows the model to learn from information between different parallel learning operations. Padding would add elements to preserve the model's dimension during the encoding process; this research leaves it as zero padding as it interprets it one-to-one. The batch size determines how many data points the model interprets at any given moment. The learning rate is as the name implies. The loss specified represents the evaluation method. In this research, the loss selected to be minimized was the mean absolute error as the effects of doing so influenced the predictions more significantly than mean squared error. This model also considers the mean absolute error despite not needing to be explicitly stated in the hyperparameters.

The documentation states that lradj refers to adjusting the learning rate, with the default being type1. Automatic mixed precision (AMP) is a technique that accelerates training and reduces memory requirements by combining different numerical precisions. The number of workers refers to the number of central processing unit (CPU) cores utilized to perform parallel learning. Iterations refer to the number of times the model conducts the training, testing, and predicting process. In this research, multiple iterations are run and evaluated manually; as such,

the iterations are set to one to get to the interpretation of the results quicker. The scale and inverse parameters change the data structure before and after the training, testing, and predicting procedure. The number of training epochs denotes the maximum number of sets of training and testing iterations the model will run before making a final prediction. The patience specified is the reason for the number of training epochs to denote the maximum number as it states the number of epochs without significant change needed before being able to stop the process early. Finally, the model's description is set to exp, or experiment, in this research. This setting allows for training, testing, and predictions to be conducted and validated.

Table 9: Informer NN optimized hyperparameters.

| **Optimized Hyperparameters** | |
|---|---|
| Input Sequence Length | 288 |
| Label Length | 288 |
| Prediction Sequence Length | 36, 72, 144 |
| Encoder Input Size | 4 |
| Decoder Input Size | 4 |
| Output Size | 1 |
| ProbSparse Attention Factor | 5 |
| Dimension of Model | 1024 |
| Number of Heads | 16 |
| Number of Encoder Layers | 8 |
| Number of Decoder Layers | 6 |
| Dimension of the Feed-Forward Network | 2048 |

| | |
|---|---|
| Dropout | 0 |
| Attention Type | Prob |
| Feature Embed Type | Fixed |
| Activation Function | GELU |
| Distilling | True |
| Output Attention | False |
| Mixing | False |
| Padding | 0 |
| Frequency | 10T |
| Batch Size | 80 |
| Learning Rate | 0.0001 |
| Loss | MAE |
| lradj | Type1 |
| Automatic Mixed Precision | False |
| Number of Workers | 0 |
| Iterations | 1 |
| Inverse | False |
| Scale | False |
| Number of Training Epochs | 10 |
| Epoch Patience | 3 |
| Description | Exp |
| **Informer Neural Network** | |

# CHAPTER V

# RESULTS AND DISCUSSION

## 5.1 Processing Times and Computer Specifications

The Gradient Boosting Machine takes two minutes and twenty-five seconds to train the dataset with the hyperparameters used in this research. RandomSearchCV was iteratively run with the processing time varying from twenty-five to fifty minutes, depending on the complexity of the hyperparameters selected from the parameter distribution. The Informer neural network trains and outputs predictions in thirty-five minutes when using the data from experiment 265A and the hyperparameters specified in Chapter IV. These predictions were made using a Dell Precision 7920. The Dell Precision 7920 has an Intel® Xeon® Gold 5217 central processing unit (CPU) @ 3.00 GHz processor and a second 2.99 GHz processor. This computer has 192 GB of random allocated memory (RAM). The computer uses the Windows 10 Enterprise operating system (OS). The bulk of the processing was conducted on the graphic processing unit (GPU), of which this computer has a dedicated 48 GB of video RAM on its NVIDIA Quadro RTX 8000.

## 5.2 Gradient Boosting Machine Results

### 5.2.1 Gradient Boosting Machine Dataset Prediction

The Gradient Boosting Machine gets optimized to predict against the entire dataset rather than the individual experiments. An appropriate scoring method, the coefficient of determination, was implemented to ensure the model predicted correctly when using the RandomSearchCV to evaluate the hyperparameters. Figure 21 shows the prediction against the dataset alongside the values outputted for the coefficient of determination and the root mean squared error. The values in Figure 21 are the model's outputs for the testing split of the dataset. The results for the training, validation, and testing splits are shown in Figure 22. The coefficient of determination output by the prediction of the dataset, which serves as the model's highest level of accuracy, was 0.95. As mentioned, appropriate values for the coefficient of determination range from 0.6 to 0.8. The root mean squared error was output to be 0.46. This error is due to the nature of tapered roller bearings to have an impulse in the vibration profile when a defect generates or propagates. These impulses were mostly filtered out when refining the data as the amplitude of these impulses is less pivotal to predicting when a bearing will become defective than the overall trend. Other than impulses due to defect propagation, sudden changes in loading or speed conditions leave the model needing to catch up to the appropriate response. The data collected by the UTCRS does not currently depict a gradual change in the loading or speed scenario tested. As such, the model is fed information that implies an immediate change.

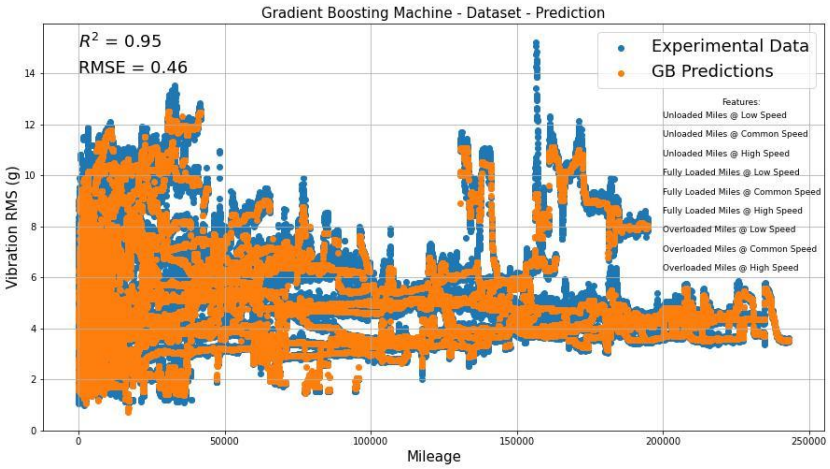



Figure 21: Gradient Boosting Machine dataset prediction.

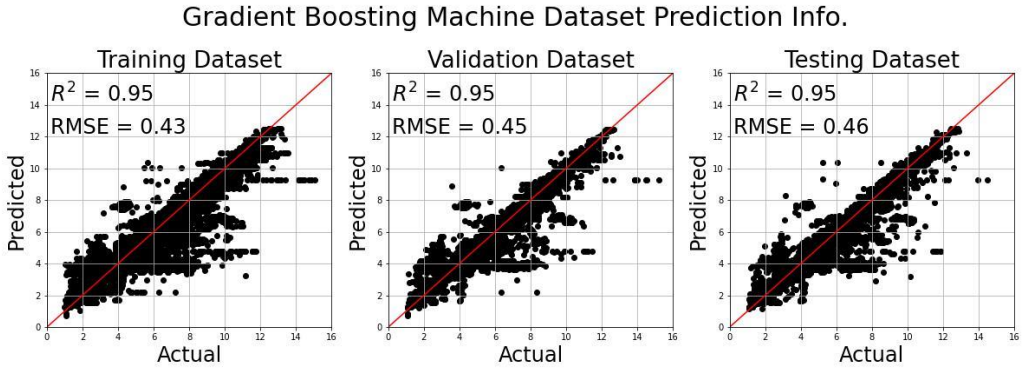



Figure 22: Gradient Boosting Machine dataset information.

### 5.2.2 Gradient Boosting Machine Blind Experiment Prediction

After the Gradient Boosting Machine has been trained and has a prediction against the dataset output, predictions against individual experiments can be made. It is crucial to predict against a blind experiment to ensure that the model's predictions are practically viable. As mentioned in Chapter II, predicting against an experiment within the dataset will result in a prediction that shows the model's theoretical highest level of accuracy but does not exemplify

how the model will perform with unseen data. Figure 23 shows the prediction made by the Gradient Boosting Machine on an experiment not present in the dataset. The coefficient of determination was output to be 0.61, and the root mean squared error was 1.84. The coefficient of determination lies within the acceptable range; however, there is a relatively high level of error. This is likely due to the model predicting the wrong trend between the 10,000 and 20,000-mile mark. A graph depicting the actual and predicted values plotted against each other and a linear line of fit is presented in Figure 24.

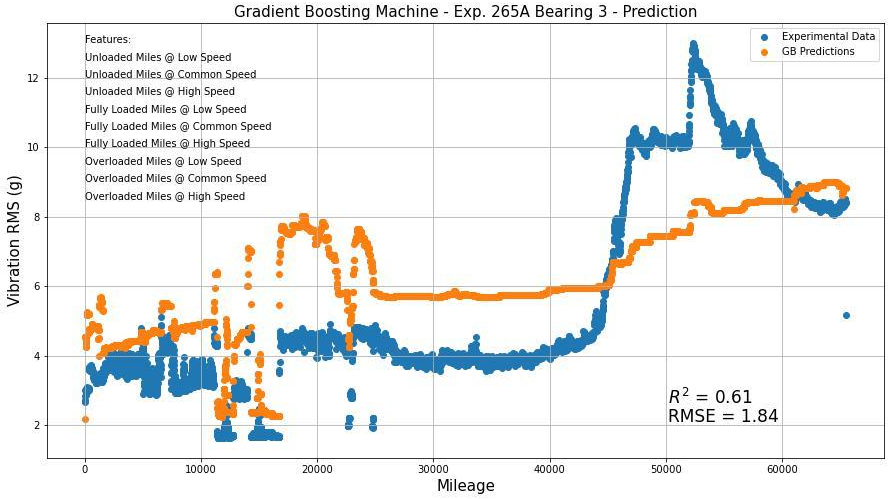



Figure 23: Gradient Boosting Machine experiment 265A prediction.

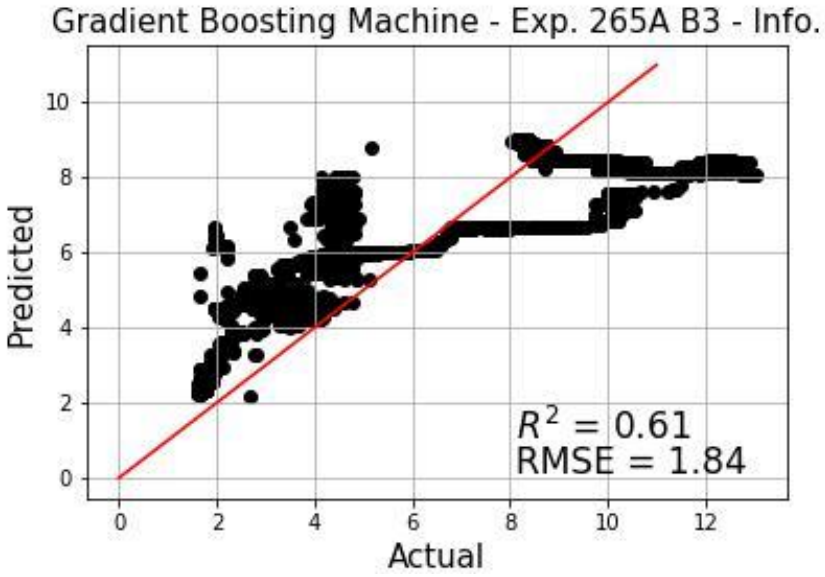



Figure 24: Gradient Boosting Machine experiment 265A information.

### 5.2.3 Gradient Boosting Machine Practical Application via Threshold Comparison

The practical application of the prediction generated by the Gradient Boosting Machine is a comparison to the statistically defined thresholds introduced in Chapter III. The response

shown in Figure 25 displays how the model predicts relative to these thresholds. The model generalizes the response of the vibration of the bearing while correctly matching the trend of staying within the preliminary and maximum thresholds aside from when the load or speed conditions suddenly drop. As mentioned, there currently is no data showing a gradual decrease in either of these features, and as such, the model interprets it as occurring instantaneously. This leaves the model needing to readjust drastically.

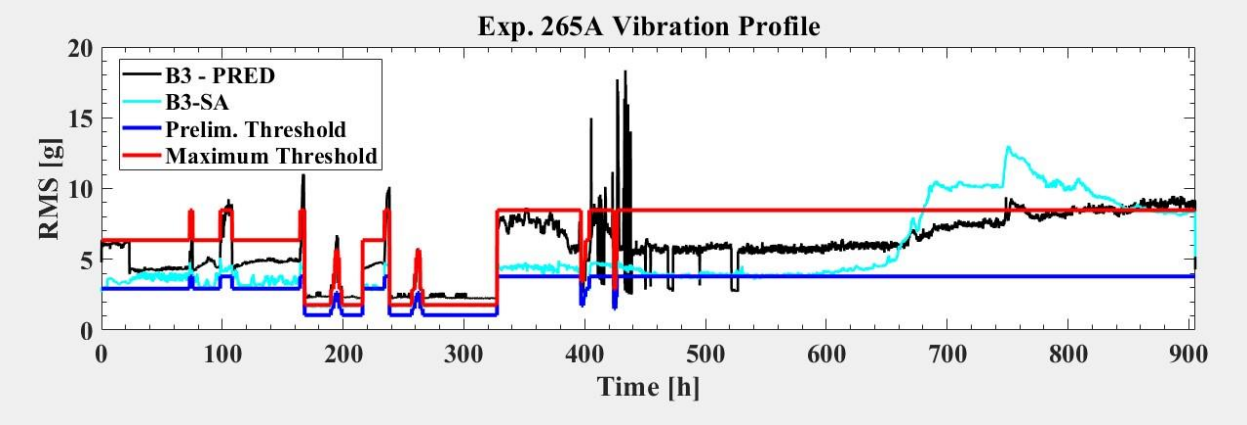


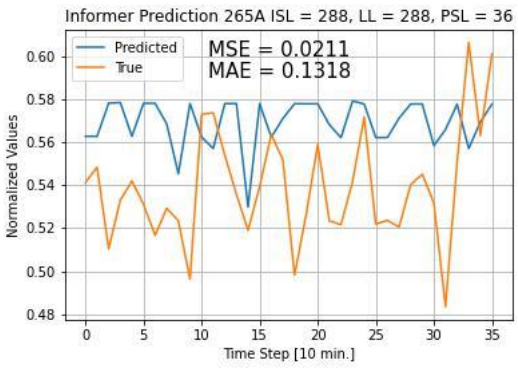

Figure 25: Gradient Boosting Machine prediction plotted against the thresholds.

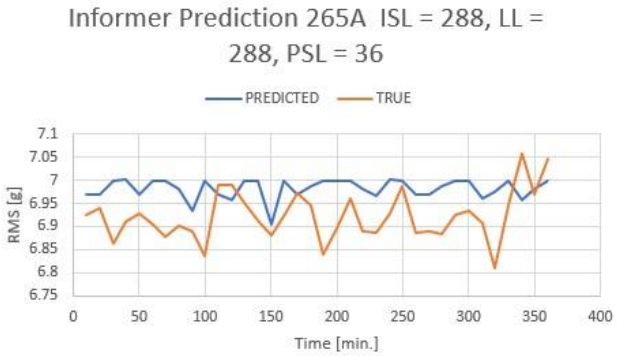
## 5.3 Informer Architecture Results

### 5.3.1 Informer Neural Network Predictions

The first combination of input sequence length, label length, and prediction sequence length tested on this implementation of the Informer neural network was 288, 288, and 36, respectively. This can be interpreted as utilizing two days' worth of data to predict six hours of data, with there being a validation of two days' worth of data being interpreted by the decoder. Figure 26 (a) displays the prediction generated and the associated errors output directly from the Informer neural network. It should be noted that the y-axis consists of normalized values and the x-axis is in time steps. This output from the Informer neural network helps determine the

model's efficacy. In this scenario, the model output a mean square error of 0.0210710 and a mean absolute error of 0.1317983. Figure 26 (b) reverts the normalization done on the target variable and interprets what the time steps represent, making this figure more suitable for practical application. The two graphs represent the same prediction, phrased in a context more suitable for the neural network or the user. The predicted values deviate by no more than 0.1660857 RMS.



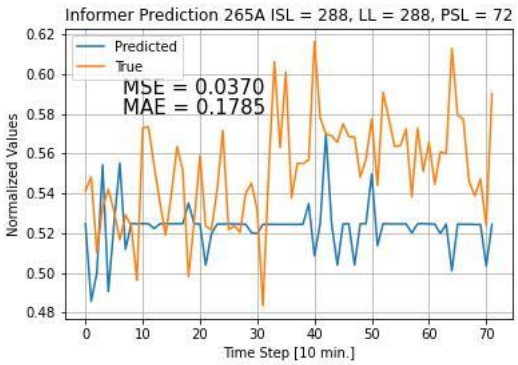



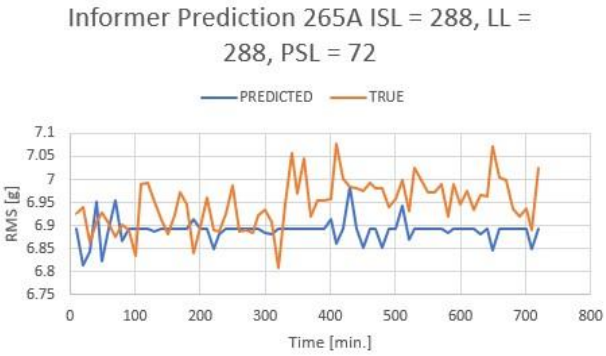

(a) Six-hour prediction normalized values (b) Six-hour prediction RMS values

Figure 26: Informer NN experiment 265A prediction: input sequence length = 288, label length = 36, & prediction sequence length = 36

The second tested combination of input, label, and prediction sequence length was 288, 288, and 72, respectively. The input sequence length and label lengths are the same throughout each scenario tested, with solely the prediction sequence length being altered. This prediction sequence length denotes a prediction twelve hours into the future. Figure 27 shows the prediction made for this prediction sequence length. Once more, the graph on the left represents the form directly output by the Informer neural network, which is suitable for highlighting the level of error produced. In contrast, the other graph displays a practically applicable version of the same prediction with context given to what the normalization means relative to the inputted data. The

mean square error output by the model was 0.0369867, and the mean absolute error was 0.1784802. The maximum deviation for the RMS in this prediction was 0.2257067 RMS.

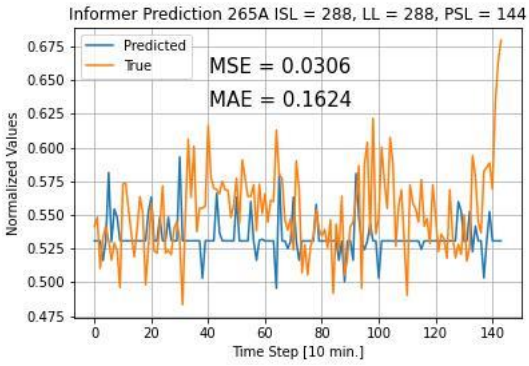


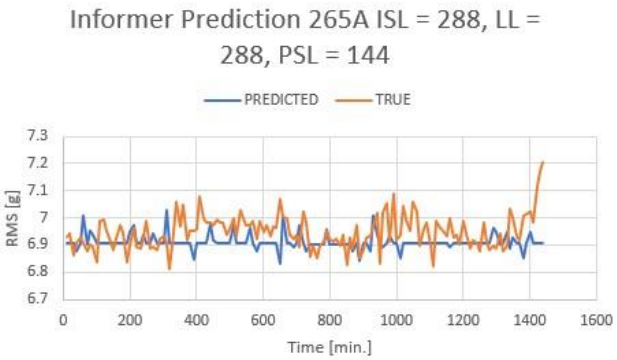




(a) Twelve-hour prediction normalized values (b) Twelve-hour prediction RMS values

Figure 27: Informer NN experiment 265A prediction: input sequence length = 288, label length = 72, & prediction sequence length = 72

The final prediction sequence length tested in this research was 144. This corresponds to a prediction made twenty-four hours into the future. The input sequence length and label length both represent two days' worth of data, the same as the previous two prediction sequence lengths tested. Figure 28 shows the prediction output by the Informer neural network for a one-day prediction. The mean square error generated by the model was 0.0306342, and the mean absolute error was 0.1624286. The maximum deviation between the actual and predicted values of the vibration profile experienced in this scenario was 0.2999828 RMS.




(a) Twenty-four-hour prediction normalized values (b) Twenty-four-hour prediction RMS values

Figure 28: Informer NN experiment 265A prediction: input sequence length = 288, label length = 144, & prediction sequence length = 144

### 5.3.2 Informer Neural Network Prediction Comparison

Table 10 displays the results of the different prediction sequence lengths for easy comparison. The value of the prediction sequence length doubled in each iteration tested. Despite this, the performance of the model remained consistent. The error and loss values output show that the model struggled to predict the twelve-hour prediction more so than the twenty-four hour one. The deviation of the target variable increased as expected, with the value increasing by 0.1338971 RMS over the eighteen-hour increase in prediction time.

Table 10: Informer NN prediction comparison.

| | **Error** | | **Losses** | | | **Deviation** |
|---|---|---|---|---|---|---|
| **ISL_LL_PSL** | **MSE** | **MAE** | **Training** | **Validation** | **Testing** | |
| **288_288_36** | 0.0353973 | 0.1766557 | 0.0287500 | 0.0621888 | 0.0062551 | 0.1660857 |
| **288_288_72** | 0.0369867 | 0.1784802 | 0.0287544 | 0.0566322 | 0.0165787 | 0.2257067 |
| **288_288_144** | 0.0306342 | 0.1624286 | 0.0382671 | 0.0667856 | 0.0123429 | 0.2999828 |

Informer NN Prediction Comparison

# CHAPTER VI

# CONCLUSIONS AND FUTURE WORK

## 6.1 Conclusions and Future Work

A Gradient Boosting Machine was implemented capable of approximating when a defect can occur, given solely information about the route ahead of time. The prediction mimics a generalized profile of the RMS of the vibration profile of a tapered roller bearing. This prediction is then compared to statistically determined thresholds to determine whether a defect has propagated. The model implemented in this research was capable of outputting a prediction with a coefficient of determination of 0.61 on a blind experiment that generally followed the trend of the actual data relative to the thresholds.

A time-series sequence-based deep neural network in the form of the Informer neural network was implemented to predict the condition of tapered roller bearings on ongoing routes. This deep neural network was capable of outputting significantly more accurate data at the expense of a much higher computational demand. The implementation in this research was capable of predicting a day, or 144-time steps, into the future with a mean square error and mean absolute error as low as 0.0306 and 0.1624, respectively. Once the normalized data is reverted, the condition of the tapered roller bearing can be determined by comparing the RMS value to the defined thresholds as they were for the Gradient Boosting Machine. The model would likely continue to perform adequately for more extensive prediction sequence lengths; however, this was the extent that the computer this research was conducted on could handle.

While time-series sequence-based neural networks are suitable for predicting the condition of a tapered roller bearing, other neural network types should be explored and have their efficacies compared against the results present in this research. In particular, supervised reinforcement and inverse reinforcement learning neural networks, in theory, should be suitable for making similar predictions. Reinforcement learning utilizes an agent interacting with the defined environment to maximize a reward function. Supervised refers to the model being supplied with data for the input and output it will train on and eventually predict. Supervised reinforcement learning means that the labeled data gets tied to state-action pairs and associated with rewards. This way, the agent in the model uses this data to learn a policy that will output a maximized reward. Utilizing a reinforcement learning neural network is difficult because the reward function must be explicitly defined. Inverse reinforcement learning remedies this, however, given that that type of neural network generates an underlying reward function given data from an expert. A reinforcement learning neural network should be implemented if a reward function is adequately defined. Inverse reinforcement learning seems to be a more appropriate starting point, however.

Aside from delving into other neural network types, this project can continue to improve by further refining the dataset and testing parameters. The current dataset for the Gradient Boosting Machine is extensive, spanning 74 experiments; however, data on new types of bearings and experiments targeting specific defect types are continuously being obtained by the University Transportation Center for Railway Safety. It is critical to update these models to maintain relevance and be practically applicable in the ever-changing railway industry. The Gradient Boosting Machine's performance can be enhanced by implementing external ensemble methods such as bagging or nested regressors. Computational demand rises when adding these to

the model. However, it should output a model more capable of capturing the underlying trends presented without necessarily divulging into neural networks. One other area of potential improvement lies in the processing capabilities of the computers used to train these models. The hyperparameters specified for the Informer neural network pushed the computer to its computational limits. The performance of these models is limited by what the computer can process. The computer used was computationally impressive; however, this model can further improve if there was access to a supercomputer.

# REFERENCES


Banoula, M. (2023, February 21). *The Best Guide to Regularization in Machine Learning*. Retrieved from Simplilearn: https://www.simplilearn.com/tutorials/machine-learning-tutorial/regularization-in-machine-learning

Bento, C. (2021, September 21). *Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis*. Retrieved from Towards Data Science: https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141#:~:text=A%20Multilayer%20Perceptron%20has%20input,use%20any%20arbitrary%20activation%20function.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., . . . Varoquaux, G. (2013). API design for machine learning software: experiences from the scikit-learn project. *arXiv:1309.0238v1 https://doi.org/10.48550/arXiv.1309.0238*, 1-15.

Cantu, L. (2021). *Assessing the Effectiveness and Efficacy of Wireless Onboard Condition Monitoring Modules in Identifying Defects in Railroad Rolling Stock.* Edinburg, TX, United States: Master's Thesis, The University of Texas Rio Grande Valley.

Chen, T., & Guestrin, C. (August 2016). Xgboost: A scalable tree boosting system. *In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785-794.

Chicco, D., Warrens, M. J., & Jurman, G. (2021). The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. *PeerJ Comput. Sci.*, 7:e623 DOI 10.7717/peerj-cs.623.

CP Freight. (2011). Railway Investigation Report R11T0016 - Transportation Safety Board of Canada. Tsb.gc.ca.

Cuanang, J. R. (2020). *Optimizing a Railroad Bearing Condition-Monitoring Algorithm for Use with an Onboard Wireless Low-Power Sensor Module.* Edinburg, TX, United States: Master's Thesis, The University of Texas Rio Grande Valley.

Dhande, M. (2020, July 3). *What is the Difference between AI, Machine Learning and Deep Learning?* Retrieved from Geospatial World: https://www.geospatialworld.net/blogs/difference-between-ai%EF%BB%BF-machine-learning-and-deep-learning/.

Federal Railroad Administration. (n.d.). *Federal Railroad Administration, Office of Safety Analysis*. Retrieved from 3.10 - Accident Causes: https://safetydata.fra.dot.gov/OfficeofSafety/default.aspx

Friedman, J. H. (Oct., 2001). Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics, Vol. 29, No. 5*, 1189-1232.

Hernandez, V. V. (2020). *Assessing the Performance of Reconditioned Railroad Tapered-Roller Bearings Used in Freight Rail Service.* Edinburg, TX, United States: Master's Thesis, The University of Texas Rio Grande Valley.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation, 9 (8)*, 1735-1780.

Huber, P. J. (1965). A Robust Version of the Probability Ratio Test 36 (6) https://doi.org/10.1214/aoms/1177699803. *Ann. Math. Statist.*, 1753-1758.

LBFoster US. (n.d.). *WILD: Wheel Impact Load Detector*. Retrieved from LBFoster US: https://lbfoster.com/en/market-segments/rail-technologies/solutions/rail-monitoring/track-monitoring-systems

Lin, H., & Li, M. (2023). 11.4 Regression and Decision Tree Basic. In H. Lin, & M. Li, *Practitioner's Guide to Data Science* (pp. 231-240). Boca Raton, FL: CRC Press.

Molnar, C. (2022). *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable (Second Edition).* Independently published.

Montalvo, J. M. (2019). *Defect Detection Algorithm Optimization for Use in Freight Railcar Service.* Edinburg, TX, United States: Master's Thesis, The University of Texas Rio Grande Valley.

National Transportation Board. (2023, February 3). *Norfolk Southern Railway Train Derailment with Subsequent Hazardous Material Release and Fires*. Retrieved from NTSB: https://www.ntsb.gov/investigations/Pages/RRD23MR005.aspx

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature, 323*, 533-536.

Stewart, M. F., Flynn, E., Marquis, B., & Sharma & Associates, S. (2020, October). Department of Transportation. Federal Railroad Administration. An Implementation Guide for Wayside Detector Systems. Washington, DC, United States: GPO, 2019.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. *arXiv:1409.3215v3 https://doi.org/10.48550/arXiv.1409.3215*, 1-9.

Tarawneh, C., Aranda, J., Hernandez, V., Crown, S., & Montalvo, J. (2020). An Investigation into Wayside Hotbox Detector Efficacy and Optimization. *International Journal of Rail Transportation, 8:3*, 264-284. DOI: 10.1080/23248378.2019.1636721.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., . . . Polosukhin, I. (2017). Attention Is All You Need. *arXiv:1706.03762v5 https://doi.org/10.48550/arXiv.1706.03762*, 1-15.

Villafranca, L. (2022). *Predicting the Remaining Service Life of Railroad Bearings: Leveraging Machine Learning and Onboard Sensor Data.* Edinburg, TX, United States: Master's Thesis, The University of Texas Rio Grande Valley.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., & Zhang, W. (2021). Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *arXiv:2012.07436v3 https://doi.org/10.48550/arXiv.2012.07436*, 1-15.

# APPENDIX

# APPENDIX

## INFORMER NEURAL NETWORK OPTIMIZATION

Table 11: Prediction Sequence Length = 36 Hyperparameter Optimization

| | A | B | C |
|---|---|---|---|
| **seq_len** | 288 | 288 | 288 |
| **label_len** | 288 | 288 | 288 |
| **pred_len** | 36 | 36 | 36 |
| **enc_in** | 4 | 4 | 4 |
| **dec_in** | 4 | 4 | 4 |
| **c_out** | 1 | 1 | 1 |
| **factor** | 5 | 5 | 5 |
| **d_model** | 2048 | 2048 | 1024 |
| **n_heads** | 16 | 16 | 16 |
| **e_layers** | 8 | 8 | 8 |
| **d_layers** | 6 | 6 | 6 |
| **s_layers** | 3,2,1 | 3,2,1 | 3,2,1 |
| **d_ff** | 2048 | 2048 | 2048 |
| **dropout** | 0 | 0 | 0 |
| **attn** | prob | prob | prob |
| **embed** | fixed | fixed | fixed |
| **activation** | gelu | gelu | gelu |
| **distil** | TRUE | TRUE | TRUE |
| **output_attention** | TRUE | TRUE | FALSE |
| **mix** | TRUE | TRUE | FALSE |
| **padding** | 0 | 0 | 0 |
| **freq** | 10T | 10T | 10T |
| **batch_size** | 32 | 64 | 80 |
| **learning rate** | 0.0001 | 0.0001 | 0.0001 |
| **loss** | mae | mse | mse |
| **lradj** | type1 | type1 | type1 |

| | | | |
|---|---|---|---|
| **use_amp** | FALSE | FALSE | FALSE |
| **num_workers** | 0 | 0 | 0 |
| **itr** | 1 | 1 | 1 |
| **inverse** | FALSE | FALSE | FALSE |
| **scale** | FALSE | FALSE | FALSE |
| **train_epochs** | 10 | 10 | 10 |
| **patience** | 3 | 3 | 3 |
| **des** | exp | exp | exp |

Table 12: Prediction Sequence Length = 36 Results

| | **mse** | **mae** | **Train** | **Vali** | **Test** |
|---|---|---|---|---|---|
| **A** | 0.02820841 | 0.155339 | 0.023539 | 0.0689807 | 0.0192879 |
| **B** | 0.057836667 | 0.227328226 | 0.031426 | 0.0540676 | 0.0266259 |
| **C** | 0.03539734 | 0.17665574 | 0.02875 | 0.0621888 | 0.0062551 |
| **288_288_36 Results** | | | | | |

Table 13: Iterations ran to understand the trend prior to honing in.

| Train | Vali | Test | MSE | MAE | drop out | e_layers | d_layers | factor | learning rate | model capacity | d_ff | batch size |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.436072 | 0.021721 | 1.5481095 | 1.448077 | 1.17306888 | 0.1 | 2 | 2 | 4 | 0.0001 | 256 | 512 | 16 |
| 0.476064 | 0.024016 | 2.0839612 | 2.125759 | 1.41129017 | 0.3 | 2 | 2 | 4 | 0.0001 | 256 | 512 | 16 |
| 0.494274 | 0.04057 | 1.9779085 | 2.033492 | 1.3776716 | 0.5 | 2 | 2 | 4 | 0.0001 | 256 | 512 | 16 |
| 0.414301 | 0.036724 | 1.7752101 | 1.243763 | 1.0842278 | 0.1 | 4 | 6 | 4 | 0.0001 | 256 | 512 | 16 |
| 0.448239 | 0.040362 | 1.4853896 | 1.320443 | 1.11205733 | 0.3 | 4 | 6 | 4 | 0.0001 | 256 | 512 | 16 |
| 0.478364 | 0.043693 | 0.8932604 | 1.238729 | 1.07634258 | 0.5 | 4 | 6 | 4 | 0.0001 | 256 | 512 | 16 |
| 0.437918 | 0.029643 | 1.6135515 | 1.517182 | 1.19003713 | 0.1 | 2 | 2 | 8 | 0.0001 | 256 | 512 | 16 |
| 0.475307 | 0.022974 | 1.432793 | 1.463351 | 1.17308962 | 0.3 | 2 | 2 | 8 | 0.0001 | 256 | 512 | 16 |
| 0.49478 | 0.103645 | 1.0336585 | 1.504414 | 1.18253529 | 0.5 | 2 | 2 | 8 | 0.0001 | 256 | 512 | 16 |
| 0.392868 | 0.057543 | 1.9652369 | 1.714868 | 1.26011574 | 0.1 | 4 | 6 | 8 | 0.0001 | 256 | 512 | 16 |
| 0.44751 | 0.023007 | 1.3063574 | 1.315471 | 1.1115402 | 0.3 | 4 | 6 | 8 | 0.0001 | 256 | 512 | 16 |
| 0.496458 | 0.025545 | 1.2568649 | 1.294044 | 1.10922289 | 0.5 | 4 | 6 | 8 | 0.0001 | 256 | 512 | 16 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.431942 | 0.02848 | 1.8497318 | 1.713091 | 1.27887464 | 0.1 | 2 | 2 | 10 | 0.0001 | 256 | 512 | 16 |
| 0.466273 | 0.025325 | 1.5162288 | 1.511773 | 1.2003758 | 0.3 | 2 | 2 | 10 | 0.0001 | 256 | 512 | 16 |
| 0.505608 | 0.030525 | 1.3577464 | 1.428963 | 1.16352165 | 0.5 | 2 | 2 | 10 | 0.0001 | 256 | 512 | 16 |
| 0.374213 | 0.07397 | 1.9903061 | 1.368665 | 1.1319797 | 0.1 | 4 | 6 | 10 | 0.0001 | 256 | 512 | 16 |
| 0.432284 | 0.03321 | 1.5179077 | 1.536725 | 1.20079243 | 0.3 | 4 | 6 | 10 | 0.0001 | 256 | 512 | 16 |
| 0.49588 | 0.029855 | 1.3352138 | 1.303712 | 1.10613453 | 0.5 | 4 | 6 | 10 | 0.0001 | 256 | 512 | 16 |
| 0.43645 | 0.112946 | 2.1561859 | 1.292292 | 1.10486972 | 0.1 | 2 | 2 | 4 | 0.001 | 256 | 512 | 16 |
| 0.502707 | 0.101832 | 1.9053359 | 0.848181 | 0.88319731 | 0.3 | 2 | 2 | 4 | 0.001 | 256 | 512 | 16 |
| 0.510314 | 0.108322 | 2.2000051 | 1.285636 | 1.10246801 | 0.5 | 2 | 2 | 4 | 0.001 | 256 | 512 | 16 |
| 0.610902 | 0.024015 | 1.1673616 | 1.181127 | 1.05277908 | 0.1 | 4 | 6 | 4 | 0.001 | 256 | 512 | 16 |
| 0.545454 | 0.06749 | 1.671152 | 0.693238 | 0.79157627 | 0.3 | 4 | 6 | 4 | 0.001 | 256 | 512 | 16 |
| 0.580981 | 0.075628 | 1.6969563 | 0.868671 | 0.8921684 | 0.5 | 4 | 6 | 4 | 0.001 | 256 | 512 | 16 |
| 0.46805 | 0.110536 | 2.5450878 | 1.273071 | 1.09854698 | 0.1 | 2 | 2 | 8 | 0.001 | 256 | 512 | 16 |
| 0.452457 | 0.089694 | 1.9632151 | 1.854704 | 1.33045721 | 0.3 | 2 | 2 | 8 | 0.001 | 256 | 512 | 16 |
| 0.474477 | 0.084572 | 1.98403 | 1.694444 | 1.27287209 | 0.5 | 2 | 2 | 8 | 0.001 | 256 | 512 | 16 |
| 1.057922 | 0.384868 | 2.842375 | 2.073566 | 1.41439855 | 0.1 | 4 | 6 | 8 | 0.001 | 256 | 512 | 16 |
| 0.648012 | 0.026251 | 1.298223 | 1.243674 | 1.08172321 | 0.3 | 4 | 6 | 8 | 0.001 | 256 | 512 | 16 |
| 0.798509 | 0.025152 | 1.1225076 | 1.227384 | 1.0744195 | 0.5 | 4 | 6 | 8 | 0.001 | 256 | 512 | 16 |
| 0.444468 | 0.105021 | 2.2947099 | 1.978076 | 1.3765111 | 0.1 | 2 | 2 | 10 | 0.001 | 256 | 512 | 16 |
| 0.450622 | 0.114811 | 2.3024795 | 1.825843 | 1.31167936 | 0.3 | 2 | 2 | 10 | 0.001 | 256 | 512 | 16 |
| 0.491623 | 0.12756 | 2.1538444 | 1.290164 | 1.09249043 | 0.5 | 2 | 2 | 10 | 0.001 | 256 | 512 | 16 |
| 0.496313 | 0.090289 | 2.0181305 | 0.76725 | 0.83911151 | 0.1 | 4 | 6 | 10 | 0.001 | 256 | 512 | 16 |
| 1.058133 | 0.381438 | 2.8319802 | 2.202143 | 1.45914352 | 0.3 | 4 | 6 | 10 | 0.001 | 256 | 512 | 16 |
| 0.592331 | 0.022197 | 1.2225087 | 1.22262 | 1.07520902 | 0.5 | 4 | 6 | 10 | 0.001 | 256 | 512 | 16 |
| 1.05906 | 0.404464 | 2.8944004 | 2.644672 | 1.60364997 | 0.1 | 2 | 2 | 4 | 0.01 | 256 | 512 | 16 |
| 1.05732 | 0.375343 | 2.8192887 | 1.966333 | 1.37599242 | 0.3 | 2 | 2 | 4 | 0.01 | 256 | 512 | 16 |
| 0.703288 | 0.02748 | 1.0976866 | 1.22715 | 1.07439256 | 0.5 | 2 | 2 | 4 | 0.01 | 256 | 512 | 16 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1.05705 | 0.388685 | 2.8533072 | 2.524736 | 1.56580901 | 0.1 | 4 | 6 | 4 | 0.01 | 256 | 512 | 16 |
| 1.057212 | 0.398474 | 2.878124 | 2.614446 | 1.59419787 | 0.3 | 4 | 6 | 4 | 0.01 | 256 | 512 | 16 |
| 1.057074 | 0.417471 | 2.9297423 | 2.884966 | 1.67689812 | 0.5 | 4 | 6 | 4 | 0.01 | 256 | 512 | 16 |
| 1.057536 | 0.3953 | 2.8694527 | 2.485007 | 1.5530709 | 0.1 | 2 | 2 | 8 | 0.01 | 256 | 512 | 16 |
| 0.6359 | 0.039386 | 0.9522426 | 1.299375 | 1.1074332 | 0.3 | 2 | 2 | 8 | 0.01 | 256 | 512 | 16 |
| 0.791351 | 0.061399 | 0.8274699 | 1.080527 | 1.00376701 | 0.5 | 2 | 2 | 8 | 0.01 | 256 | 512 | 16 |
| 1.056977 | 0.386199 | 2.8442955 | 1.349897 | 1.13000786 | 0.1 | 4 | 6 | 8 | 0.01 | 256 | 512 | 16 |
| 1.058228 | 0.40087 | 2.8845196 | 2.194631 | 1.45658946 | 0.3 | 4 | 6 | 8 | 0.01 | 256 | 512 | 16 |
| 1.057178 | 0.421223 | 2.9412785 | 2.687827 | 1.61704946 | 0.5 | 4 | 6 | 8 | 0.01 | 256 | 512 | 16 |
| 1.05935 | 0.395532 | 2.872653 | 1.779263 | 1.30624926 | 0.1 | 2 | 2 | 10 | 0.01 | 256 | 512 | 16 |
| 1.058977 | 0.371856 | 17.094133 | 34.33733 | 5.74881649 | 0.3 | 2 | 2 | 10 | 0.01 | 256 | 512 | 16 |
| 1.057758 | 0.419895 | 2.9362791 | 2.341187 | 1.50605798 | 0.5 | 2 | 2 | 10 | 0.01 | 256 | 512 | 16 |
| 1.058005 | 0.386447 | 2.8457181 | 2.654665 | 1.60676277 | 0.1 | 4 | 6 | 10 | 0.01 | 256 | 512 | 16 |
| 1.057132 | 0.380999 | 2.8301122 | 2.114997 | 1.42899251 | 0.3 | 4 | 6 | 10 | 0.01 | 256 | 512 | 16 |
| 1.056932 | 0.394585 | 2.8706338 | 2.469564 | 1.54809117 | 0.5 | 4 | 6 | 10 | 0.01 | 256 | 512 | 16 |
| 0.459104 | 0.063612 | 2.6015656 | 2.503328 | 1.53482842 | 0.1 | 2 | 2 | 4 | 0.0001 | 128 | 512 | 16 |
| 0.493114 | 0.061207 | 2.1203115 | 1.843794 | 1.32478523 | 0.3 | 2 | 2 | 4 | 0.0001 | 128 | 512 | 16 |
| 0.528982 | 0.023819 | 1.6984658 | 1.694561 | 1.27109218 | 0.5 | 2 | 2 | 4 | 0.0001 | 128 | 512 | 16 |
| 0.410251 | 0.050559 | 1.9056329 | 1.541326 | 1.19857049 | 0.1 | 4 | 6 | 4 | 0.0001 | 128 | 512 | 16 |
| 0.476205 | 0.022056 | 1.4972684 | 1.544311 | 1.19648898 | 0.3 | 4 | 6 | 4 | 0.0001 | 128 | 512 | 16 |
| 0.51369 | 0.03257 | 0.9463548 | 0.959718 | 0.94614416 | 0.5 | 4 | 6 | 4 | 0.0001 | 128 | 512 | 16 |
| 0.458019 | 0.025746 | 1.7240373 | 1.841258 | 1.31556976 | 0.1 | 2 | 2 | 8 | 0.0001 | 128 | 512 | 16 |
| 0.49162 | 0.043396 | 1.8546668 | 1.829129 | 1.32589078 | 0.3 | 2 | 2 | 8 | 0.0001 | 128 | 512 | 16 |
| 0.53072 | 0.026387 | 1.7880435 | 1.788565 | 1.30999959 | 0.5 | 2 | 2 | 8 | 0.0001 | 128 | 512 | 16 |
| 0.404368 | 0.051074 | 1.7365258 | 1.297164 | 1.10770893 | 0.1 | 4 | 6 | 8 | 0.0001 | 128 | 512 | 16 |
| 0.47418 | 0.025599 | 1.3621793 | 1.215274 | 1.0768224 | 0.3 | 4 | 6 | 8 | 0.0001 | 128 | 512 | 16 |
| 0.520481 | 0.05406 | 1.4537433 | 0.886727 | 0.89607745 | 0.5 | 4 | 6 | 8 | 0.0001 | 128 | 512 | 16 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.461898 | 0.034618 | 1.4916092 | 1.48747 | 1.19001961 | 0.1 | 2 | 2 | 10 | 0.0001 | 128 | 512 | 16 |
| 0.489707 | 0.03084 | 1.8916998 | 1.883806 | 1.33348799 | 0.3 | 2 | 2 | 10 | 0.0001 | 128 | 512 | 16 |
| 0.525829 | 0.058634 | 1.9031011 | 1.811853 | 1.31181359 | 0.5 | 2 | 2 | 10 | 0.0001 | 128 | 512 | 16 |
| 0.412688 | 0.054139 | 1.6283253 | 1.629846 | 1.25052202 | 0.1 | 4 | 6 | 10 | 0.0001 | 128 | 512 | 16 |
| 0.469209 | 0.038513 | 2.1093826 | 2.146458 | 1.40906727 | 0.3 | 4 | 6 | 10 | 0.0001 | 128 | 512 | 16 |
| 0.516608 | 0.028172 | 1.6739241 | 1.458616 | 1.16215754 | 0.5 | 4 | 6 | 10 | 0.0001 | 128 | 512 | 16 |
| 0.42025 | 0.127511 | 2.2400305 | 1.566706 | 1.22548831 | 0.1 | 2 | 2 | 4 | 0.001 | 128 | 512 | 16 |
| 0.440349 | 0.070262 | 1.8024977 | 1.561241 | 1.20879257 | 0.3 | 2 | 2 | 4 | 0.001 | 128 | 512 | 16 |
| 0.461527 | 0.058247 | 1.8156894 | 1.739636 | 1.26961446 | 0.5 | 2 | 2 | 4 | 0.001 | 128 | 512 | 16 |
| 0.469521 | 0.086865 | 1.8751297 | 1.585002 | 1.23124313 | 0.1 | 4 | 6 | 4 | 0.001 | 128 | 512 | 16 |
| 0.538716 | 0.06176 | 1.9260575 | 1.35382 | 1.13289773 | 0.3 | 4 | 6 | 4 | 0.001 | 128 | 512 | 16 |
| 0.559299 | 0.075773 | 1.8997074 | 1.121945 | 1.02695835 | 0.5 | 4 | 6 | 4 | 0.001 | 128 | 512 | 16 |
| 0.366448 | 0.096755 | 2.1329453 | 1.71813 | 1.26169944 | 0.1 | 2 | 2 | 8 | 0.001 | 128 | 512 | 16 |
| 0.436537 | 0.095132 | 1.8933973 | 1.693604 | 1.25938535 | 0.3 | 2 | 2 | 8 | 0.001 | 128 | 512 | 16 |
| 0.463038 | 0.094872 | 1.9095187 | 1.902846 | 1.34452701 | 0.5 | 2 | 2 | 8 | 0.001 | 128 | 512 | 16 |
| 0.48764 | 0.102999 | 1.8804339 | 1.698722 | 1.27566874 | 0.1 | 4 | 6 | 8 | 0.001 | 128 | 512 | 16 |
| 0.60719 | 0.089553 | 1.7842255 | 0.620374 | 0.7649138 | 0.3 | 4 | 6 | 8 | 0.001 | 128 | 512 | 16 |
| 0.556719 | 0.045163 | 1.4899809 | 1.29596 | 1.10465539 | 0.5 | 4 | 6 | 8 | 0.001 | 128 | 512 | 16 |
| 0.387705 | 0.080999 | 1.8894515 | 0.948467 | 0.93695265 | 0.1 | 2 | 2 | 10 | 0.001 | 128 | 512 | 16 |
| 0.434602 | 0.121697 | 2.1323991 | 1.654283 | 1.24428201 | 0.3 | 2 | 2 | 10 | 0.001 | 128 | 512 | 16 |
| 0.456061 | 0.073169 | 1.8767257 | 1.805697 | 1.30705094 | 0.5 | 2 | 2 | 10 | 0.001 | 128 | 512 | 16 |
| 0.534261 | 0.071901 | 1.7537938 | 1.688503 | 1.26877058 | 0.1 | 4 | 6 | 10 | 0.001 | 128 | 512 | 16 |
| 0.510786 | 0.137799 | 6.1271124 | 1.589318 | 1.23350978 | 0.3 | 4 | 6 | 10 | 0.001 | 128 | 512 | 16 |
| 0.584385 | 0.028799 | 1.1614923 | 1.100136 | 1.0244174 | 0.5 | 4 | 6 | 10 | 0.001 | 128 | 512 | 16 |
| 1.056942 | 0.393406 | 2.8672266 | 2.08312 | 1.41779494 | 0.1 | 2 | 2 | 4 | 0.01 | 128 | 512 | 16 |
| 1.058254 | 0.400152 | 2.8851302 | 2.841454 | 1.66387379 | 0.3 | 2 | 2 | 4 | 0.01 | 128 | 512 | 16 |
| 1.056405 | 0.384336 | 2.8427558 | 1.953294 | 1.37124598 | 0.5 | 2 | 2 | 4 | 0.01 | 128 | 512 | 16 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1.057136 | 0.384301 | 2.8391576 | 2.446289 | 1.54055595 | 0.1 | 4 | 6 | 4 | 0.01 | 128 | 512 | 16 |
| 1.05784 | 0.392976 | 2.8683884 | 1.665538 | 1.26196682 | 0.3 | 4 | 6 | 4 | 0.01 | 128 | 512 | 16 |
| 1.057394 | 0.386944 | 2.8466232 | 2.326271 | 1.5010972 | 0.5 | 4 | 6 | 4 | 0.01 | 128 | 512 | 16 |
| 0.842013 | 0.026017 | 1.3096631 | 1.209089 | 1.06588459 | 0.1 | 2 | 2 | 8 | 0.01 | 128 | 512 | 16 |
| 1.058113 | 0.37889 | 2.8240473 | 2.616587 | 1.59487033 | 0.3 | 2 | 2 | 8 | 0.01 | 128 | 512 | 16 |
| 0.853297 | 0.032162 | 1.0155057 | 1.161378 | 1.04326403 | 0.5 | 2 | 2 | 8 | 0.01 | 128 | 512 | 16 |
| 1.058404 | 0.385301 | 2.8416209 | 2.330007 | 1.50234103 | 0.1 | 4 | 6 | 8 | 0.01 | 128 | 512 | 16 |
| 1.057894 | 0.379649 | 2.8300004 | 2.650151 | 1.60535765 | 0.3 | 4 | 6 | 8 | 0.01 | 128 | 512 | 16 |
| 1.059091 | 0.407424 | 2.9025636 | 2.257285 | 1.47793984 | 0.5 | 4 | 6 | 8 | 0.01 | 128 | 512 | 16 |
| 1.057198 | 0.384693 | 2.8401334 | 2.764508 | 1.64058852 | 0.1 | 2 | 2 | 10 | 0.01 | 128 | 512 | 16 |
| 0.889859 | 0.084579 | 1.7873995 | 1.719805 | 1.28328824 | 0.3 | 2 | 2 | 10 | 0.01 | 128 | 512 | 16 |
| 1.058125 | 0.408532 | 2.9056633 | 2.224506 | 1.4668088 | 0.5 | 2 | 2 | 10 | 0.01 | 128 | 512 | 16 |
| 1.057514 | 0.391373 | 2.8613536 | 2.076397 | 1.41542268 | 0.1 | 4 | 6 | 10 | 0.01 | 128 | 512 | 16 |
| 1.057628 | 0.407204 | 2.9025607 | 2.126454 | 1.43299603 | 0.3 | 4 | 6 | 10 | 0.01 | 128 | 512 | 16 |
| 1.055698 | 0.419227 | 2.9347253 | 2.825806 | 1.65916502 | 0.5 | 4 | 6 | 10 | 0.01 | 128 | 512 | 16 |
| 0.480915 | 0.056268 | 2.3573694 | 2.062901 | 1.40069437 | 0.1 | 2 | 2 | 4 | 0.0001 | 64 | 512 | 16 |
| 0.523096 | 0.054637 | 1.7391291 | 1.736133 | 1.2825973 | 0.3 | 2 | 2 | 4 | 0.0001 | 64 | 512 | 16 |
| 0.572362 | 0.168892 | 2.8615983 | 1.091525 | 1.00060809 | 0.5 | 2 | 2 | 4 | 0.0001 | 64 | 512 | 16 |
| 0.440542 | 0.054055 | 1.8365844 | 1.612674 | 1.22717834 | 0.1 | 4 | 6 | 4 | 0.0001 | 64 | 512 | 16 |
| 0.505567 | 0.037508 | 1.9336575 | 1.760214 | 1.29701412 | 0.3 | 4 | 6 | 4 | 0.0001 | 64 | 512 | 16 |
| 0.543944 | 0.051195 | 1.9128712 | 0.698463 | 0.7692191 | 0.5 | 4 | 6 | 4 | 0.0001 | 64 | 512 | 16 |
| 0.480008 | 0.030151 | 1.4543918 | 1.531923 | 1.20695543 | 0.1 | 2 | 2 | 8 | 0.0001 | 64 | 512 | 16 |
| 0.525366 | 0.1246 | 2.8239927 | 2.448215 | 1.53059578 | 0.3 | 2 | 2 | 8 | 0.0001 | 64 | 512 | 16 |
| 0.571958 | 0.025759 | 0.9784019 | 0.976325 | 0.92480278 | 0.5 | 2 | 2 | 8 | 0.0001 | 64 | 512 | 16 |
| 0.453434 | 0.062523 | 1.9376516 | 0.756523 | 0.82955629 | 0.1 | 4 | 6 | 8 | 0.0001 | 64 | 512 | 16 |
| 0.503626 | 0.036632 | 1.6905237 | 1.507599 | 1.19610679 | 0.3 | 4 | 6 | 8 | 0.0001 | 64 | 512 | 16 |
| 0.556878 | 0.057789 | 1.550644 | 1.190931 | 1.04240358 | 0.5 | 4 | 6 | 8 | 0.0001 | 64 | 512 | 16 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.473133 | 0.028896 | 1.5814538 | 1.584361 | 1.20520568 | 0.1 | 2 | 2 | 10 | 0.0001 | 64 | 512 | 16 |

# BIOGRAPHICAL SKETCH

Sergio Martinez was born in McAllen, Texas on September 6th, 2000. He attended Vanguard Academy Rembrandt Secondary and graduated in 2018. He took part in the Dual Enrollment Engineering Academy (DEEA) at South Texas College in McAllen, Texas and graduated Magna Cum Laude with an Associate's of Science in Engineering in 2018. He attended the University of Texas – Rio Grande Valley for his undergraduate career and graduated Summa Cum Laude in 2020 with a Bachelor of Science in Mechanical Engineering. He was the Treasurer and a Design Lead for the Society of Automotive Engineers – RGV Baja Racing Team, where he designed, machined, and tested an offroad mini-Baja vehicle. During his Graduate career at the University of Texas – Rio Grande Valley, he worked as a Teacher's Assistant for the MECE 3330: Measurements and Instrumentation course at UTRGV and later as a Research Assistant at the University Transportation Center for Railway Safety. He earned his Master of Science in Mechanical Engineering at the University of Texas – Rio Grande Valley in May 2023. Sergio can be contacted by email at serge.mtz.757@gmail.com