Remaining Useful Life Prediction using Gaussian Process Regression Model

Katarina Vuckovic[1], and Shashvat Prakash[2]

[1,2] *Raytheon Technologies - Collins Aerospace, Windsor Locks, CT, 06096, USA*
katarina.vuckovic@collins.com
shashvat.prakash@collins.com

Abstract

This paper evaluates the merits of a multi-variable Gaussian process regression (GPR) model for remaining useful life (RUL) estimation. The paper presents an optimization method that trains the GPR model to find the best kernel type and hyper-parameter combination. Furthermore, the paper evaluates the performance of the GPR model for small training datasets and with a reduction (missing) of input features. A comparison is made to the multi-layer perceptron (MLP) neural network which forms the basis of deep learning models. To illustrate model performance, an air filter clogging RUL dataset is used. The performance results show that both GPR and MLP models have similar sensitivity to training set size but GPR also computes the uncertainty. Empirically, MLP is more robust to a test set with a missing input while the data suggests that the GPR performs better when the training data also did not include the same input.

1. Introduction

Remaining useful life (RUL) of a component is a metric that defines the expected duration over which a component or system can function in accordance with its specifications. Typically, the RUL is estimated in units of time or numbers of cycles until failure. Estimation of RUL is a vital aspect of predictive maintenance and prognostic health management (PHM). Additionally, RUL estimations can also be applied to the analysis of reliability, efficiency, productivity and safety (Hu, Miao, Si, Pan, & Zio, 2022). Due to the wide range of applications, RUL estimation is gaining interest in many different domains of engineering.

As multiple sensors are becoming a commodity on many components and systems, the volume of available sensor data is growing (*Aircraft Sensors Market: 2021-28: Industry size , growth report*, 2021). This enables the expansion of data-driven techniques in PHM applications. To this end, many researchers are proposing data-driven PHM solutions in recent years (Huang, Di, Jin, & Lee, 2017). Particularly, deep learning techniques are attracting significant attention especially when dealing with highly non-linear and complex features (Y. Wang, Zhao, & Addepalli, 2020). Increasingly, if the failure mode is marginally observable, enough data can create the relevant reference signal.

Machine learning methods like deep learning offer some advantages over traditional physics based feature engineering, especially when future behavior can be expected to follow previously observed behavior. Yet, these techniques do have some undesirable properties. There are two main challenges with the neural networks (NNs) that are inherent to deep learning. The first challenge is the data availability. Deep learning models require large datasets to avoid the problem of under-fitting (Pei, 2021). While operational data is generally abundant, failures are relatively rare. If the RUL differs significantly based on the operating conditions (i.e. pressure, temperature, etc.), then the NN will inevitably underfit the data due to a small dataset. Furthermore, the only way to understand the nature of underfit is to require more data.The second problem with NN is that they provide a ”black-box” solution. In other words, the functioning mechanism is not transparent and cannot be interpreted easily (Chen et al., 2021). The ambiguous model interpretation may raise confidence issues within the PHM community especially when dealing with counter-intuitive outcomes. This is especially true for predictions that cannot be explained by the physical knowledge of the system.

Some of these concerns may be addressed by using Gaussian process regression (GPR) models. The GPR models have the ability to learn a wide range of complex non-linear functions by tuning a small number of hyper-parameters. Since the models have only a few parameters to optimized, they tend to generalize better on small datasets while also determining the confidence bounds of any estimation. This poses a significant advantage over NN models that contain many parameters and require large datasets to train. Regarding the traditional physics based engineering models, the GPR poses an advantage in the ability to model complex functions. Typically, feature engineering models are parametric and do not adapt as well to complex non-linear relationships.

To illustrate the benefits of the GPR model, the RUL of dust filters are examined (Mauthe, Hagmeyer, & Zeiler, 2021). These types of filters are utilized in a variety of industries such as automotive, aviation, and facility management. The dust filters fail once there is sufficient clogging due to the accumulation of dust particles. Since these filters are typically expendable components that require frequent replacement (Eker, Camci, & Jennions, 2013), RUL predictions of can improve maintenance efficiency, reduce delays due to failures, and improve system reliability.

This paper is organized as follows. Section 2 begins with a literature review of RUL estimation models. This is followed by an overview of the GPR and MLP models in Section 3. Section 4 covers the dataset used to illustrate the performance of the models as well as the model training approach. In section 5, the performance metrics for the two models are presented and discussed. Finally, Section 6 summarizes the work and highlights the main points.

2. LITERATURE REVIEW

This section provides an overview of the GPR models studied in the context of RUL prediction. There are several papers that explore RUL prediction with GPR models. The simplest approaches use a single input parameter, derived either as a single feature or as a combination of contributing features into a single Health Indicator (HI). More complex methods employ tiered GPR models with multiple inputs.

A case study on the RUL of lithium batteries proposes a fuzzy evaluation-Gaussian process regression (FE-GPR) that combines fuzzy logic with GPR model. The FE-GPR uses the battery capacity as the single input feature to estimate the RUL (Kang et al., 2020). Another study on the RUL predictions of Electric Energy Metering Equipment (EEME) used a single health index input based on the environmental stress (N. Li et al., 2021). In (Liu & Chen, 2019), a novel method that combines HIs with multiple GPR models is presented for forecasting the RUL. The authors present a rather complex framework where they first use three separate single feature GPRs to generate HI predictions which are then used as inputs to a multiple input feature GPR model that predicts the RUL. A similar approach is presented in few other works where the short-term state of health (SOH) is first estimated using GPR model. The SOH is then used as an input to another GPR model that predicts the long-term RUL (Jia et al., 2020), (X. Li, Wang, & Yan, 2019) and (X. Li, Yuan, & Wang, 2020). Another interesting multi step framework is proposed wherein the first step uses a Gaussian process classification (GPC) model to determine if component is healthy or degraded based on HIs. Then, another multiple input GPR model is used to predict the RUL (Benker, Bliznyuk, & Zaeh, 2021). Finally, in a RUL study on lubricating oil, a multiple output GPR (MO-GPR) model is used that correlates the historical degradation trends with current degradation trends (Tanwar & Raghavan, 2020).

The main contribution of this work is the optimization of the GPR model over multiple kernel types for a range of hyperparameter values. The optimizations yields that the best kernel for the analyzed dataset is the *Matérn32*. The second contribution is in the comparison of the GPR model with the MLP NN. The comparison is conducted across various training dataset sizes and when input features are missing.

3. MODELS

3.1. Gaussian Process Regression

A Gaussian Process (GP) is a generalization of the multivariant Gaussian probability distribution. Whereas the probability distribution describes random variables which are scalars or vectors, the GP as a stochastic process governs the properties of a function (Rasmussen & Williams, 2005). The GPR refers to a statistics based methodology where the Gaussian processes are used as prior models for the Bayesian regression functions that are fitted to the observed data (Särkkä, 2019). Given a set of observed data points, there is an infinite number of possible functions that can fit these points. In GPR, the GPs conduct regression by defining a distribution over these infinite number of functions. The GPR then uses the distribution as prior knowledge to make predictions and provide the analysis of uncertainties. Given an input and output training pair dataset $\mathcal{D} = (\mathbf{x}_n, y_n)$, the GPR model is defined as

$$y_n = f(\mathbf{x}_n) + \epsilon_n, \epsilon_n \sim \mathcal{N}(0, \sigma_n^2 \mathbf{I}) \tag{1}$$

where f is drawn from a Gaussian process with mean μ and covariance K, $f(\mathbf{x}_n) \sim \mathcal{GP}(\mu, K)$, ϵ_n is assumed to be an independent, identically distributed Gaussian noise added to the the system, and n refers to the n^{th} observation (Damianou & Lawrence, 2013).

Given training input data $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]$ and its corresponding observations $\mathbf{y} = [y_1, y_2, ..., y_n]$, the joint distribution of the observed values $\mathbf{y}$ and the function values at the new testing point y^* associated with test set $\mathbf{X}_*$, is defined as (J. Wang, 2021):

$$\begin{pmatrix} \mathbf{y} \\ y^* \end{pmatrix} \sim \mathcal{N}(0, \begin{bmatrix} \mathbf{K} + \sigma_n^2 I & \mathbf{K}_* \\ \mathbf{K}_*{}^T & \mathbf{K}_{**} \end{bmatrix}) \tag{2}$$

where $\mathbf{K} = K(\mathbf{X}, \mathbf{X})$, $\mathbf{K}_* = K(\mathbf{X}, \mathbf{X}_*)$, $\mathbf{K}_{**} = K(\mathbf{X}_*, \mathbf{X}_*)$, and $\mathbf{K} + \sigma_n^2 I$ is the variance of (1). If there are

n training points and n test points, then $\mathbf{K}_* = K(\mathbf{X}, \mathbf{X}_*)$ denotes the $n \times n$ matrix of the covariances evaluated at all pairs of training and test points. Similar analogy applies for $\mathbf{K}$ and $\mathbf{K}_{**}$. In other words, covariance matrix $\mathbf{K}$ consist of covariance function entries $K_{ij} = k(\mathbf{x_i}, \mathbf{x_j})$. Then, the posterior predictive distribution is defined by the mean and variance functions of the posterior. The mean $\overline{y^*}$ and variance $\mathbb{V}[y^*]$ are evaluated as (J. Wang, 2021):

$$\overline{y^*} = \mathbf{K}_*^{\mathbf{T}}(\mathbf{K} + \sigma_n^2 I)^{-1}\mathbf{y}. \tag{3}$$

$$\mathbb{V}[y^*] = \mathbf{K}_{**} - \mathbf{K}_*^{\mathbf{T}}(\mathbf{K} + \sigma_n^2 I)^{-1}\mathbf{K}_*. \tag{4}$$

As it may be observed, the GPR is entirely defined by the mean and covariance functions (*kernel*). Without any prior knowledge, the mean is generally assumed to be zero and if necessary, shifted accordingly after the model is trained. Therefore, the adjustment of the mean is trivial. The main focus of the GPR design is selecting the appropriate kernel and optimizing its hyper-parameters. The kernel function describes the correlation of the Gaussian process. The most common choice of kernel is the square exponential (SE) (also known as Radial Basis Function (RBF)). Even though, this kernel generally performs well in many instances, there are other kernels that may perform better for some datasets. Therefore, optimization is applied on GPR over five different kernel types to find the kernel that best fits sample dataset.

Table 1. Covariance Functions

Name	Kernel $k(\mathbf{x}_i, \mathbf{x}_j)$
Squared Exponential	$\sigma^2 exp(-\frac{r^2}{2l^2})$
Exponential	$\sigma^2 exp(-\frac{r}{2l^2})$
Rational Quadratic	$\sigma^2(1 + \frac{r}{2\alpha l^2})-\alpha$
Matérn32	$\sigma^2(1 + \frac{\sqrt{3}r}{l})exp(-\frac{\sqrt{3}r}{l})$
Matérn52	$\sigma^2(1 + \frac{\sqrt{5}r}{l} + \frac{5r^2}{3l^2})exp(-\frac{\sqrt{5}r}{l})$

The kernels are listed in Table 1 where σ is the standard deviation of the signal, l is the characteristic length scale, α is the positive-valued scale-mixture parameter, and

$$r = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)}. \tag{5}$$

3.2. Multilayer Perceptron

The MLP is a type of NN that is created by stacking multiple perceptrons (Géron, 2019). The architecture of the MLP consists of an input layer, one or more hidden layers, and an output layer as illustrated in Figure 1. The number of perceptrons in the input layer is equivalent to the number input parameters. The hidden layers can contain any number of perceptrons and any number of layers. Finally, the number of perceptions in the output layer is equivalent to the number of output parameters.

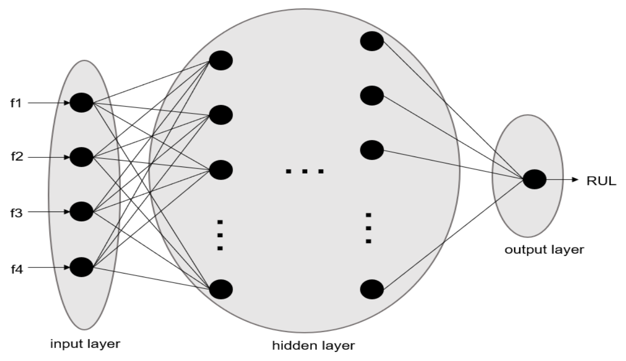


Figure 1. MLP Training Model

4. SIMULATIONS

4.1. Dataset

To illustrate the performance of the RUL predictions, analysis is performed on the filter clogging dataset[1] (Mauthe et al., 2021). Almost all industries depend on some sort of filtration process and filter clogging tends to be one of the main reasons for filter degradation and removal (Sutherland & Chase, 2011). Therefore, this dataset is a good example of a real world scenario.

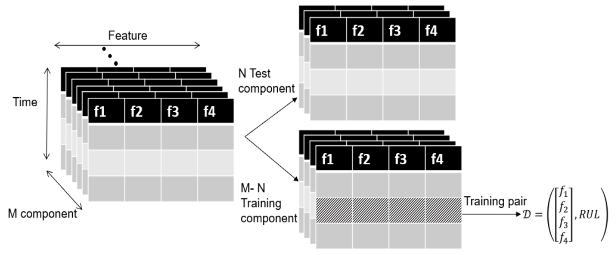

Feature
Time
M component
f1 f2 f3 f4
N Test component
M- N Training component
Training pair
$\mathcal{D} = \left(\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}, RUL\right)$

Figure 2. Dataset Preparation

This dataset was generated in a laboratory environment using an air filtration test bench. The dataset records 1) the pressure drop across the filter, 2) the amount of dust introduced, 3) the size of the dust particles (three different sizes), and 4) the flow rate across the filter. These four features are recorded at a regular sampling rate of 10 Hz. The feature measurements and the RUL at a given time instance represent one input-output training pair as illustrated in Figure 2. The total dataset contains $M = 50$ filters (components) which are split into N testing and $M - N$ training filters.

Since one of the goals is to test the performance for varying sizes of training datas, the dataset is split into 3 groups, each with a training set representing 20%, 50%, and 80% (i.e. 10 filters, 25 filters, and 40 filters) of the overall dataset. Each group contains 5 randomly selected batches used for cross-validation. For 20% training, the first 10 components in the dataset were selected for batch 1 while the remaining 40 were used for validation. For batch 2, the second 10 components were selected for testing and so on. For 80% training, the

[1]Dataset is available at https://www.kaggle.com/prognosticshse/datasets

testing and training sets were reversed from the 20% training batches. Finally, the 50% training, were selected as follows: first 25 components (batch 1), second 25 components (batch 2), components 14-38 (batch 3), components 1-13 and 39-50 (batch 4), and components 6-30 (batch 5).

4.2. GPR Model

The GPR model consists of an offline phase, during which the kernel functions and their hyper-parameters are optimized, and an online phase, during which the RUL prediction is computed using (3). The confidence interval is obtained from the variance in (4) as illustrated in Figure 3. Referring to (1), the input vector $\mathbf{x}$ contains the four features at a given time instance, while the output y is the RUL of that component.

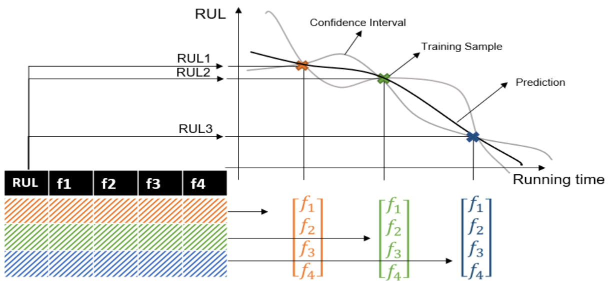


Figure 3. GPR Model

4.2.1. Offline Hyper-parameter Training

For any given kernel in Table 1, Bayesian optimization is used to find the optimal hyper-parameters given by the vector $\Theta = [\alpha, \sigma^2, l]$. The log likelihood function

$$\log(P(\mathbf{y}|\mathbf{X})) = \frac{\mathbf{y}^T(\mathbf{K}+\sigma_n^2 I)^{-1}\mathbf{y}}{2} + \frac{\log|\mathbf{K}+\sigma_n^2 I|}{2} + \frac{n\log(2\pi)}{2} \tag{6}$$

is maximized to find the optimal hyper-parameters given by the vector Θ (J. Wang, 2021)

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmax}} \log(P(\mathbf{y}|\mathbf{X},\Theta)). \tag{7}$$

The optimization over multiple kernels and multiple hyper-parameters values are shown in Figure 4. The kernels from Table 1 are listed on the *Kernel Function* axis. Furthermore, the *Sigma* axis represents standard deviation σ_n for the noise in (1). Finally, the vertical axis represents the objective cost function, which is being minimized as part of the optimization. In this case, the log likelihood function defined in (6) should be maximized, so the objective cost function is the negative of the log likelihood function (6). The optimization is complete when either the objective cost function converges or the model reaches its maximum number of 15 iterations.

The objective function converges on the *Matérn32* kernel as the estimated optimal solution. The *squared exponential* which is used as a default kernel in many GPR models, definitely shows a sub-optimal solution compared to *Matérn32*.

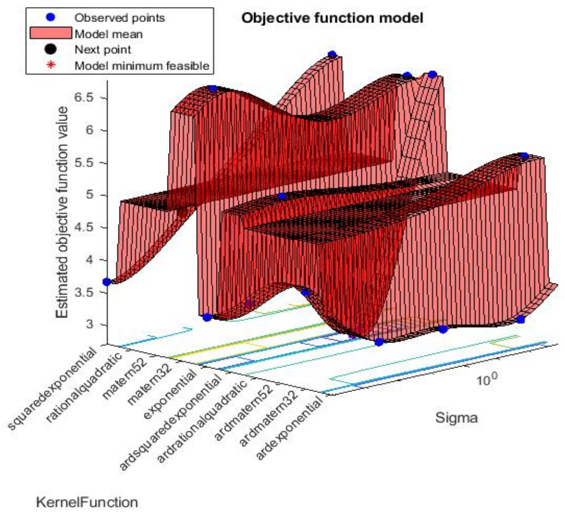


Figure 4. Optimization over multiple kernel types for training dataset 20% Batch 5

It is possible to separate the length-scale parameter for each predictor. This is known as the automatic relevance determination (ARD). The purpose of ARD is to regularize the solution space using a parameterized, data-dependent prior distribution that effectively prunes away redundant or superfluous features (Neal, 2012). The kernels are tested with and without the automatic relevance determination (ARD) and show that for this dataset the ARD does not improve the performance.

4.3. MLP Training

Referring to Figure 1, the input layer has 4 neurons (one for each feature). There are 3 hidden layers of size 150, 100, and 25. Finally, the output layer has 1 neuron which represents the RUL estimate. The MLP training network is designed to use a 'relu' activation function, 'adam' optimizer, learning rate of 0.0001, and maximum iteration of 300.

5. RESULTS AND DISCUSSION

5.1. Performance Metrics

The prediction is evaluated using three common performance metrics for RUL: 1) the mean absolute error (MAE), 2) the mean absolute percent error (MAPE), and 3) the root mean square error (RMSE) (Liu & Chen, 2019). The equations are defined in (8) - (10), respectively, where y_i is the true and y_i^* is the predicted RUL, and n is the total number of test samples.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - y_i^*| \tag{8}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}|\frac{y_i - y_i^*}{y_i}| * 100\% \tag{9}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - y_i^*)^2}{n}} \tag{10}$$

Each of the three performance metrics provides a different insight into the results. The MAE gives less weight to the outliers which means that the metric is less sensitive to them. This also means that it may not adequately reflect the performance when dealing with large error values. On the other hand, the RMSE is heavily dependent on large errors. Furthermore, both MAE and RMSE are absolute errors specific to the scale of the data. These metrics do not provide any insight into the performance of data at different scales. For such application it is better to use MAPE which is the absolute error normalized over the data. The MAPE generates a metric that can be used to compare results across different scales. However, the MAPE does have drawbacks. True value data points that are equal to zero have to be excluded from the dataset to avoid dividing by zero. In addition, errors at small values of y_i will have a large bearing on the result.

5.2. Performance Results

The three performance parameters for each batch as well as the averages and standard deviations (STDEV) over all five batches are presented in Tables 2-4. The MAE and MAPE are expressed in seconds while the MAPE is a relative error expressed in percentage. The average result in all instances except for MAPE for 50% of training data shows that the GPR outperforms the MLP model. Referring to Table 3, the largest error is coming from batch 1 with GPR MAPE at 121%. Careful analysis of the data points reveals that the GPR had higher error at small RUL, which is penalized heavier due to the normalization. Furthermore, the MAPE for GPR in batch 1 is significantly higher than the MAPE for the other batches in the same train category (column), thus skewing the average to be higher.

Table 2. Comparing GPR and MLP RUL estimation MAE.

MAE	Train 20%		Train 50%		Train 80%	
Batch	GPR	MLP	GPR	MLP	GPR	MLP
1	88.1 s	106.9 s	22.5 s	27.1 s	16.9 s	14.2 s
2	37.9 s	55.3 s	28.9 s	26.1 s	20.7 s	29.6 s
3	35.2 s	28.9 s	22.9 s	26.6 s	19.2 s	19.5 s
4	16.4 s	35.2 s	22.3 s	26.3 s	18.9 s	42.2 s
5	21.5 s	26.7 s	33.4 s	32.4 s	26.7 s	26.1 s
Average	39.8 s	50.6 s	22.0 s	27.7 s	20.5 s	26.3 s
STDEV	28.4 s	33.4 s	4.9 s	2.6 s	3.7 s	10.7 s

On average both models perform better when the training dataset is larger as is expected of data-driven models. However, as the training dataset sizes are reduced, variation among the different batches increases. Some batches perform poorly while other batches have performances comparable to models with larger training datasets. For instance, referring to the training dataset of 20% in Tables 2 and 4, the MAE and RMSE for batch 1 is much higher than for the other four batches. The training dataset for batch 1 primarily contains samples of filters that are operating in environments with large dust particles. Therefore, it does not learn the correlation of the dust size to the RUL, and performs poorly when presented with a dust size other than the one in the training set. Furthermore, it cannot be expected that the model performs well on filters that are routinely subjected to smaller dust particles. On the other hand, for batches where the training dataset contains a variety of conditions, the performance improves significantly. Furthermore, the advantage of GPR over MLP is that in conditions where variety exits, the GPR can train a better model with a smaller dataset. For instance, batch 4 in the training dataset of 20% contains a variety of operating conditions. The performance of GPR for this batch is roughly two times better than MLP. The results show that, on average, both models improve by similar amounts when presented with more training data (last two columns of Tables 6 and 7). Yet, there are differences between the model improvement numbers when examined on a per-batch basis. This variation among the batches is best measured by the standard deviation which improves for the GPR model as the dataset size increases. However, the STDEV for MLP does not necessarily improve for larger datasest. In fact, for the MLP model, it is actually smaller for 50% than for 80% training dataset.

Table 3. Comparing GPR and MLP RUL estimation MAPE.

MAPE	Train 20%		Train 50%		Train 80%	
Batch	GPR	MLP	GPR	MLP	GPR	MLP
1	79%	93%	121%	40%	27%	26%
2	92%	143%	44%	29%	23%	27%
3	80%	43%	41%	42%	17%	14%
4	22%	43%	27%	23%	46%	58%
5	29%	31%	67%	56%	45%	47%
Average	61%	71%	61%	38%	31%	35%
STDEV	32.36%	46.9%	37.0%	12.7%	13.2%	17.7%

Table 4. Comparing GPR and MLP RUL estimation RMSE.

RMSE	Train 20%		Train 50%		Train 80%	
Batch	GPR	MLP	GPR	MLP	GPR	MLP
1	90.3 s	132.5 s	26.6 s	37.9 s	21.2s	16.9s
2	74.9 s	73.2 s	37.9 s	34.5 s	26.3 s	35.1 s
3	39.6 s	39.9 s	27.9 s	38.0 s	24.9 s	24.9 s
4	20.7 s	42.9 s	31.2 s	34.3 s	26.0 s	52.6 s
5	43.7 s	38.5 s	41.2 s	44.0 s	33.7 s	32.7 s
Average	53.8 s	65. s	32.96 s	37.7 s	26.4 s	32.4 s
STDEV	28.1 s	40.1 s	6.3 s	3.9 s	4.5 s	13.3 s

5.3. Predictive Variance

As the focus of this paper is on small datasets, this section analyzes the average predictive variance for the batches trained with 20% dataset. Using the variance in (4), the 95% confidence interval (CI) can be computed as

$$CI_{95\%} = \mathbf{y}^* \pm 1.96 * (\mathbb{V}[\mathbf{y}^*])^2. \tag{11}$$

An example of an estimated RUL with the 95% CI is illustrated in Figure 5. This example uses the GPR model trained with 20% of dataset from batch 1. The predicted RUL in the figure represents a sample component from the testing dataset. In this example, the true value is indicated with the solid straight line while the dashed line is the predicted RUL. The shaded gray region is the 95% CI computed from (11).

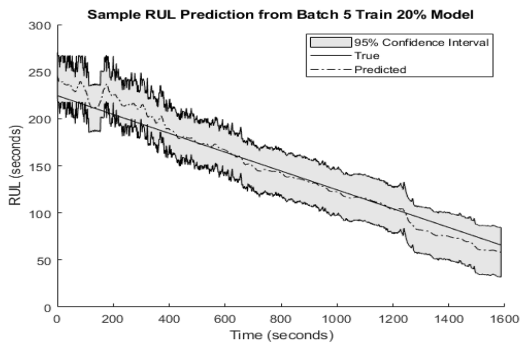


Figure 5. Example of a RUL prediction of a samples component.

Table 5 shows the average variance the training dataset of 20% and the percent of testing samples that are within the 95% CI. As it may be observed from the table, the percent of samples within the 95% CI is less than 95%. If the training dataset was properly balanced with a variety of filter operating conditions, the third column in Table 5 would have been 95% or higher. However, none of the five batches contains the right variety of conditions to properly map the entire solution space. This emphasizes the need to carefully select the samples in the training dataset.

Table 5. The average predictive variance and number of samples within the predicted confidence interval for the batches trained with 20% of the dataset.

Batch	Variance	Percent of samples within 95% CI
1	4.14	42%
2	5.32	76%
3	4.44	66%
4	7.62	82%
5	4.02	59%

The largest variance in batch 4 means that this batch has the biggest variety of input samples which results in the lowest prediction error in GPR in Tables 2-4. This shows that batch 4 is the best batch to train on and MLP does not provide this information.

5.4. Missing Input

Another measure of performance is robustness to missing or reduced input data. In many practical applications, the inputs available to train a model may be limited. Accurately detecting particle size over time may not be feasible. Accordingly, both the GPR and MLP models were evaluated when (1) the dust particle size is contained in the training but not in the testing set, and (2) when dust particle size was missing from both training and testing sets. Relative degradation to a baseline performance underlies how sensitive each model type is to missing data. Table 6 and 7 summarize the results. When the 4-input model expects particle size but instead gets a constant '0' value, the MLP model degrades the least across the 5 batches. Accordingly, when the model is trained with three inputs, omitting the particle size, the GPR model degrades the least compared to the four input case across the 5 batches.

Table 6. Comparing performance GPR model trained with 4 features (baseline) to 1) Dust size features missing during testing, 2) Dust size missing during training and testing, 3) improved performance when training dataset is increased to 50%, and 4) improved performance when training dataset is increased to 80%.

Train Dataset Size	20%	20%	20%	50%	80%
Train Features	4	4	3	4	4
Test Features	4	3	3	4	4
Batch	RMSE	Percentage Change from RMSE			
1	90.3s	4%	-1%	-71%	-84%
2	74.9s	0%	-67%	-63%	-67%
3	39.9s	-19%	-7%	-4%	-34%
4	20.7s	46%	132%	50%	25%
5	43.7s	-2%	%	-6%	-23%
Average	53.8s	13%	11%	-19%	-35%

Table 7. Comparing performance MLP model trained with 4 features (baseline) to 1) Dust size features missing during testing, 2) Dust size missing during training and testing, 3) improved performance when training dataset is increased to 50%, and 4) improved performance when training dataset is increased to 80%.

Train Dataset Size	20%	20%	20%	50%	80%
Train Features	4	4	3	4	4
Test Features	4	3	3	4	4
Batch	RMSE	Percentage Change from RMSE			
1	132.7s	2%	-2%	-71%	-82%
2	73.2s	-19%	73%	-53%	-52%
3	39.9s	-28%	65%	-5%	-38%
4	43.2s	55%	-13%	-21%	22%
5	38.5s	-6%	122%	14%	-15%
Average	65.5s	1%	49%	-27%	-34%

6. CONCLUSION

This work proposed a GPR model to estimate the RUL. The primary focus was on the design of the GPR model which was trained to find the optimal kernel type and hyper-parameters. The performance of the GPR model was compared to the MLP NN using three different performance metrics. Furthermore, the performance of the models was evaluated for multiple training dataset sizes and for missing data. The results showed that training input size affect both GPR and MLP equally, but each model performs differently in cases of missing data. Where the data was expected but not provided, MLP is empirically less affected, and when the model is trained without the missing data as an input, GPR is less degraded. In general, the GPR has advantages in that it provides both the estimate and its confidence distribution and requires less tuning effort. Lastly, the analysis revealed the importance of a quality, well-balanced training dataset.

REFERENCES

Aircraft sensors market: 2021-28: Industry size , growth report. (2021).

Benker, M., Bliznyuk, A., & Zaeh, M. F. (2021). A gaussian process based method for data- efficient remaining useful life estimation. *IEEE Access*, *9*, 137470-137482. doi: 10.1109/ACCESS.2021.3116813

Chen, K., Kong, Q., Dai, Y., Xu, Y., Yin, F., Xu, L., & Cui, S. (2021). *Recent advances in data-driven wireless communication using gaussian processes: A comprehensive survey.*

Damianou, A., & Lawrence, N. D. (2013). Deep gaussian processes. In *Artificial intelligence and statistics* (pp. 207–215).

Eker, O., Camci, F., & Jennions, I. K. (2013). Filter clogging data collection for prognostics. In *Annual conference of the phm society* (Vol. 5).

Géron, A. (2019). *Hands-on machine learning with scikit-learn, keras, and tensorflow: Concepts, tools, and techniques to build intelligent systems.* " O'Reilly Media, Inc.".

Hu, Y., Miao, X., Si, Y., Pan, E., & Zio, E. (2022). Prognostics and health management: A review from the perspectives of design, development and decision. *Reliability Engineering System Safety*, *217*.

Huang, B., Di, Y., Jin, C., & Lee, J. (2017). Review of data-driven prognostics and health management techniques: lessons learned from phm data challenge competitions. *Machine Failure Prevention Technology*, *2017*, 1–17.

Jia, J., Liang, J., Shi, Y., Wen, J., Pang, X., & Zeng, J. (2020). Soh and rul prediction of lithium-ion batteries based on gaussian process regression with indirect health indicators. *Energies*, *13*(2). doi: 10.3390/en13020375

Kang, W., Xiao, J., Xiao, M., Hu, Y., Zhu, H., & Li, J. (2020). Research on remaining useful life prognostics based on fuzzy evaluation-gaussian process regression method. *IEEE Access*, *8*, 71965-71973. doi: 10.1109/ACCESS.2020.2982223

Li, N., Duan, J., Ma, J., Qiu, W., Zhang, W., & Teng, Z. (2021, nov). Gaussian process based remaining useful life prediction for electric energy metering equipment. *Journal of Physics: Conference Series*. doi: 10.1088/1742-6596/2125/1/012032

Li, X., Wang, Z., & Yan, J. (2019). Prognostic health condition for lithium battery using the partial incremental capacity and gaussian process regression. *Journal of Power Sources*, *421*, 56-67.

Li, X., Yuan, C., & Wang, Z. (2020). Multi-time-scale framework for prognostic health condition of lithium battery using modified gaussian process regression and nonlinear regression. *Journal of Power Sources*, *467*, 228358.

Liu, J., & Chen, Z. (2019). Remaining useful life prediction of lithium-ion batteries based on health indicator and gaussian process regression model. *IEEE Access*, *7*, 39474-39484. doi: 10.1109/ACCESS.2019.2905740

Mauthe, F., Hagmeyer, S., & Zeiler, P. (2021). Creation of publicly available data sets for prognostics and diagnostics addressing data scenarios relevant to industrial applications. *International Journal of Prognostics and Health Management*, *12*(2).

Neal, R. M. (2012). *Bayesian learning for neural networks* (Vol. 118). Springer Science & Business Media.

Pei, E. (2021). *On some similarities and differences between deep neural networks and kernel learning machines*. Rochester Institute of Technology.

Rasmussen, C. E., & Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning*. The MIT Press. doi: 10.7551/mitpress/3206.001.0001

Sutherland, K., & Chase, G. (2011). *Filters and filtration handbook*. Elsevier.

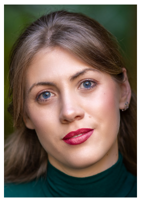
Särkkä, S. (2019). *The use of gaussian processes in system identification.*

Tanwar, M., & Raghavan, N. (2020). Lubricating oil remaining useful life prediction using multi-output gaussian process regression. *IEEE Access*, *8*, 128897-128907. doi: 10.1109/ACCESS.2020.3008328

Wang, J. (2021). *An intuitive tutorial to gaussian processes regression.*

Wang, Y., Zhao, Y., & Addepalli, S. (2020). Remaining useful life prediction using deep learning approaches: A review. *Procedia manufacturing*, *49*, 81–88.

BIOGRAPHIES



Katarina Vuckovic received her B.S. in Aerospace Engineering (2017), B.S. in Electrical Engineering (2017), and M.S. in Electrical Engineering (2019) from Florida Institute of Technology. She is currently pursuing her Ph.D. studies in Electrical Engineering at the University of Central Florida. She has been with has with Collins Aerospace for five years working as a systems engineer on wireless communication systems, aircraft automation applications, and prognostics and health management of aircraft components.

Shashvat Prakash is a Senior Principal Engineer at Raytheon Technologies, Collins Aerospace. He has degrees in Mechanical Engineering from University of Illinois (B.S.), Carnegie Mellon (M.S.), and Georgia Institute of Technology (Ph.D.). His experience spans computational design optimization at Carnegie Mellon, satellite attitude control at NASA Goddard Space Flight Center, combustion control at Georgia Institute of Technology, gas turbine engine control and combustion at General Electric's GE Aviation, and prognostic and health management at Collins. He is active in SAE's Integrated Vehicle Health Management HM-1 Committee.