

Enhancing Aviation Maintenance with Explainable AI

Master of Science Thesis

J.Y. Andringa

Technische Universiteit Delft



Enhancing Aviation Maintenance with Explainable AI

Master of Science Thesis

by

J.Y. Andringa

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on 25-03-2024.

Student number: 4662822
Project duration: March 2023 – March 2024
Thesis committee: (Chair) Dr. B. (Bruno) Santos
(Supervisor) Dr. M. (Marcia) Baptista
(Examiner) Dr. N. (Nan) Yue

An electronic version of this thesis is available at <http://repository.tudelft.nl/>.

Acknowledgements

While these might be the first words you will read, these are the last words I have written for my master thesis, and thus also the last words I have written as a student at the TU Delft. This thesis is the culmination of my years at the TU Delft, where I have grown academically, professionally, and personally. This thesis you hold in your hands (or are reading on a screen) is filled to the brim with my thoughts, struggles, epiphanies, and the knowledge I have gained during my time here.

To guide me in this process, I can not thank Marcia Baptista enough for supporting me and guiding me in making this work a reality. I walked into her office with zero knowledge about AI, Machine Learning or maintenance, except from the fact that I knew I wanted to learn all about it, and contribute to this field. She helped me gain the necessary knowledge and expertise, and also shared my enthusiasm, which was ever growing for this field. I also want to thank Bruno Santos for providing his support, ideas and feedback, without which this work would not have reached half the quality it is now.

Lastly, I want to acknowledge the incredible support I have had not only during my time working on my thesis, but also during my time as a student in general. Thank you to the life long friends I have made at the VSV, to the amazing people in our thesis 'melting pot' 2.56, to my family who gave me the opportunity to have such an amazing student life, and to my girlfriend Manouk, who always managed to put a smile on my face even during the most tough times of this thesis, you gave me the motivation and energy to keep pushing on.

While I am a bit saddened that my time as a student is coming to a close, I can't wait to see what the future brings. One thing I know for certain though, my time in Delft will forever and always stay with me.

Jilles Andringa
Delft, March 2024

Contents

List of Figures	vii
List of Tables	ix
List of Abbreviations	xi
Introduction	xiii
I Scientific Paper	1
II Literature Study	
previously graded under AE4020	19
1 Introduction	21
2 Prognostics and Health Management	23
2.1 Approaches to PHM	23
2.1.1 Reliability based	23
2.1.2 Model based	24
2.1.3 Data-driven	24
2.1.4 Hybrid	24
2.2 Experimental data sets	24
2.2.1 C-MAPSS.	24
2.2.2 N-CMAPSS.	26
2.2.3 Li-ion Battery Aging Data set.	26
2.2.4 Bearing data set	26
2.3 RUL estimation methods in aviation	27
2.3.1 Machine Learning approaches in RUL estimation	28
2.3.2 Statistical approaches in RUL estimation	29
2.4 Deterministic vs. Probabilistic methods.	29
2.5 Performance indicators	31
2.5.1 Average Bias	32
2.5.2 Sample Standard Deviation	32
2.5.3 (Root) Mean Squared Error.	32
2.5.4 Mean Absolute Percentage Error	32
2.5.5 Prognostic Horizon	32
2.5.6 $\alpha - \lambda$ performance	33
2.5.7 Relative Accuracy	33
2.5.8 Cumulative Relative Accuracy	33
2.5.9 Convergence.	33
2.6 Challenges, shortcomings and future work	34
3 Bayesian methods	37
3.1 Bayes' Rule	37
3.2 Bayesian models in machine learning.	38
3.2.1 Naive Bayes	38
3.2.2 Bayesian Linear Regression	39
3.2.3 Bayesian Networks.	40
3.2.4 Bayesian Neural Networks	40
3.2.5 Gaussian Process Regression.	40
3.2.6 Relevance Vector Machines	41

3.3	Method comparison	41
3.4	Application of chosen methods	42
3.4.1	BNN	42
3.4.2	GPR	43
3.4.3	RVM	43
4	Explainable AI	45
4.1	Need for XAI	45
4.2	Explainable AI methods	46
4.3	Application of XAI in PHM	46
4.4	Counterfactual explanations	46
4.4.1	CFE properties	47
4.4.2	Distance measurement	47
4.4.3	Counterfactual explanation methods	48
4.5	Selection of CFE method	50
4.6	DiCE	51
4.7	CFE performance indicators	51
5	Research Proposal	53
5.1	Research gap	53
5.2	Research Goals	53
5.3	Research Questions	53
5.4	Project methodology & timeline.	54
5.4.1	Phase 1.	54
5.4.2	Phase 2.	55
5.4.3	Phase 3.	55
5.4.4	Phase 4.	55
5.4.5	Phase 5 & 6.	55
5.5	Gantt Chart	55
	Bibliography	59

List of Figures

2.1	RUL estimation taxonomy	23
2.2	Experiment setup of bearing data-set [85]	27
2.3	System health over four regimes (note that the System Health Index ranges from 0-1 and transition values are taken arbitrarily) [122]	28
2.4	Data driven methods overview	28
2.5	Visual representation of BRNN network used in [11]	30
2.6	Use of uncertainty in RUL estimation. Top plot shows error distribution being shifted w.r.t. the scoring function. The bottom plots show the associated total cost. [6]	31
2.7	$\alpha - \lambda$ performance evaluation using $\alpha = 0.20$ and $\lambda = 0.5$ [98]	33
2.8	Example of convergence of 3 cases that converge at different rates [98]	34
3.1	Different types of Bayesian machine learning models	38
3.2	BLR example with 1, 3 and 20 data points (top, middle, bottom) based on example provided by [53]	39
3.3	Example of a Gaussian Process Regression (GPR) with a Radial Basis Function (RBF) kernel. 10 observed data points are given and 20 functions are sampled from the posterior. The blue area indicates the 3σ confidence intervals. [115]	41
4.1	Model agnostic counterfactuals in XAI taxonomy as constructed by Chou et al. [19]	49
5.1	Methodology and timeline overview	54

List of Tables

2.1	Sensor measurement description	25
2.2	Summary of different data sets	26
4.1	Examples of commonly used XAI methods categorized by local vs global and model agnostic vs specific	46
4.2	Summary of CFE algorithms and their properties	50

List of Abbreviations

1-NN	1-Nearest Neighbour	MAPE	Mean Absolute Precision Error
ADNN	Adjacent Difference Neural Network	MCMC	Markov Chain Monte Carlo
ANN	Artificial Neural Network	ML	Machine Learning
ARIMA	Autoregressive Integrated Moving Average	MLE	Maximum Likelihood Estimation
BLR	Bayesian Linear Regression	MLP	Multi Layer Perceptron
BN	Bayesian Networks	MLR	Multivariate Linear Regression
BNN	Bayesian Neural Networks	MSE	Mean Squared Error
BRNN	Bayesian Recurrent Neural Network	NB	Naive Bayes
C-MAPSS	Commercial Modular Aero-Propulsion System Simulation	NN	Neural Network
CFE	Counterfactual Explanation	pdf	Probability Density Function
CNN	Convolutional Neural Networks	PF	Particle Filter
CRA	Cumulative Relative Accuracy	PH	Prognostic Horizon
DARPA	Defence Advanced Research Projects Agency	PHM	Prognostics and Health Management
DiCE	Diverse Counterfactual Explanations	RA	Relative Accuracy
DL	Deep Learning	RBF	Radial Basis Function
DNN	Deep Neural Network	ReLU	Rectified Linear Unit
DPP	Determinantal Point Processes	RFR	Random Forest Regression
EKF	Extended Kalman Filter	RMSE	Root Mean Squared Error
EOD	End of Discharge	RNN	Recurrent Neural Networks
EOL	End of Life	RUL	Remaining Useful Life
EOP	End of Prediction	RVM	Relevance Vector Machine
GM	Grey Model	SMC	Sequential Monte Carlo
GP	Gaussian Process	SSD	Sample Standard Deviation
GPR	Gaussian Process Regression	SVM	Support Vector Machine
GRU	Gated Recurrent Units	SVR	Support Vector Regression
HMC	Hamiltonian Monte Carlo	UKF	Uncented Kalman Filter
JANET	Just Another Network	UUT	Unit Under Test
KF	Kalman Filter	VI	Variational Inference
LSTM	Long Short Term Memory	VRNN	Vanilla Recurrent Neural Network
MAD	Median Absolute Deviation	WP	Wiener Process
		XAI	Explainable Artificial Intelligence

Introduction

Aircraft maintenance plays a central role in the aviation industry. Without well maintained aircraft, it is impossible to uphold the high safety and reliability standards we impose on the aviation industry. Aircraft and their components are often scheduled for regular maintenance checks to ensure that no critical damage can occur during flight operations. This process is expensive and time consuming, but necessary. To improve this process, the field of Prognostics and Health Management (PHM) is continuously making innovations. The goal of this field is to increase maintenance efficiency, reduce costs, and ensure the reliability of aircraft (components).

Many advancements have been made in this field, especially in cooperation with the field of Machine Learning (ML). Using ML, we are better able to predict when maintenance is actually needed, based on the real condition of aircraft components. This method, known as predictive maintenance, could significantly reduce unnecessary downtime and save resources by preventing the premature replacement of parts.

However, applying ML to aircraft maintenance comes with challenges, such as making accurate predictions, understanding how the models make these predictions, and the need for large amounts of data. To address these issues, this study focuses on using Bayesian models to better understand prediction uncertainties, and use the field of explainable AI (XAI) to make the ML predictions more interpretable, and use XAI as a data augmentation technique to increase the amount of training data without additional costs.

At the core of this research is the use of Counterfactual explanations, a form of XAI, not just to improve the interpretability of ML models but also to enhance their performance in predicting the Remaining Useful Life (RUL) of aircraft components. By combining predictive maintenance with these advanced ML techniques, this thesis aims to provide more accurate and understandable maintenance predictions, ultimately leading to more effective and efficient aircraft maintenance practices.

This thesis report is organized as follows : In Part I, the scientific paper is presented. Part II contains the relevant Literature Study that supports the research.

I

Scientific Paper

Enhancing Aviation Maintenance with Explainable AI: A Bayesian Approach to Counterfactual Explanations for Remaining Useful Life Estimation^{*}

Jilles Andringa^a (MSc. Student)

^aFaculty of Aerospace Engineering, Delft University of Technology, Kluyverweg 1, Delft, 2629 HS, the Netherlands

ARTICLE INFO

Keywords:

Prognostics and Health Management
Remaining Useful Life
C-MAPSS
Bayesian Uncertainty
Bayesian Neural Networks
Counterfactual Explanations
Interpretability
Data Augmentation

ABSTRACT

Machine learning models have improved Prognostics and Health Management (PHM) in aviation, notably in estimating the Remaining Useful Life (RUL) of aircraft engines. However, their 'black-box' nature limits transparency, critical in safety-sensitive aviation maintenance. Explainable AI (XAI), particularly Counterfactual (CF) explanations, offers a way to explain model decisions by suggesting alternative scenarios for different outcomes. Additionally, Bayesian models enhance predictions by quantifying uncertainty, yet the combination of CF explanations and Bayesian methods is largely unexplored. This study investigates counterfactual methods within a Bayesian framework to improve the explainability of RUL estimation and improve model performance. For this, a Bayesian Long Short-Term Memory (LSTM) model was applied to the C-MAPSS data-set. This research uniquely applies CF explanations in two ways, with the goal of offering insights into how varying operational conditions could affect the RUL, and to improve the model's performance by generating additional augmented data with reduced uncertainty for the model to train on. Preliminary results show that CF explanations are able to provide insights and suggestions for RUL improvement. Also, the addition of the augmented data using the CF uncertainty reduction method has shown to improve the models predictive performance, confirming the viability of this approach as a data augmentation method.

1. Introduction


Aircraft maintenance is a costly endeavour, both regarding time and money. Furthermore, aircraft are held to the highest maintenance standards to ensure the safety of passengers, crew and staff. Maintenance is often scheduled at regular intervals to ensure component degradation or faults are caught on time. While this approach is relatively simple yet effective, it often leads to unnecessary aircraft down time for maintenance and causes perfectly functioning components to be replaced prematurely. To make this process more efficient, the field of Prognostics and Health Management (PHM) uses predictive maintenance, which makes maintenance decisions based on data/model driven predictions. To ensure the high quality standards set for the aviation industry, these models should pose a high degree of accuracy and reliability.

For this purpose, Machine Learning (ML) models are often employed due to their ability to take in a large amount of data from different sources and find degradation patterns on their own, provided there is enough training data. While these models have proven to be quite accurate in their predictions, they pose some key issues. Firstly, they provide point estimates for their predictions, which tend to be overconfident. To address this issue, the field of Bayesian modelling provides methods to also quantify uncertainty in model predictions. Secondly, ML models suffer from a

"black box" nature, which limits the transparency of their predictions. The field of explainable AI (XAI) attempts to address this issue by developing methods to make ML models more explainable and interpretable. One of these methods is Counterfactual (CF) explanations, which aim to provide more insight into a model and its predictions by suggesting alternative scenarios that would have led to a different outcome. Finally, ML models require large quantities of data to train on in order to be accurate in their predictions. Acquiring this data is often a costly and time consuming endeavour. To mitigate this, data augmentation, where already acquired data is altered to be used as 'new' data, can be applied to increase the amount of training data without the need for new real-world data.

In this research, we aimed to use CF explanations as an approach to both improve the interpretability and performance of RUL prediction models. In doing so, we combined the fields of predictive maintenance, Bayesian uncertainty and counterfactual explanations in a novel manner, where the strength of each respective field contributes to the others. At the core of this collaboration between fields lies the Counterfactual Explanations, which we used as our main tool in improving the interpretability and performance of the predictive models. To achieve this, we set out to answer the proposed research question "*How can Bayesian uncertainty and Counterfactual Explanations be used in predictive maintenance to improve interpretability and predictive performance?*". This research question can be further subdivided into two main aspects, improving interpretability and improving predictive performance.

^{*} This research is the final part of achieving a MSc. in aerospace engineering at the Faculty of Aerospace Engineering in Delft.

 (J. Andringa) ORCID(s):

To answer this, we developed a Bayesian Long Short Term Memory (LSTM) model which predicts the Remaining Useful Life (RUL) of aircraft components. For this research, we use the C-MAPSS dataset as a case study, containing simulated sensor data of a set of aircraft engines. This RUL prediction model was used together with the DiCE model to generate CF explanations. To achieve a higher level of interpretability, we use the DiCE model to gain more insight from the RUL prediction model, where we want to find explanations on how to improve the RUL of aircraft engines, which can be used in maintenance strategies. To achieve a better performance from the predictive model, we combined the Bayesian uncertainty with the CF explanations to introduce a new method of data augmentation, where the training set is expanded using inputs generated from CF explanations.

The remainder of this paper is structured as followed. Section 2 will cover the theoretical background and previous literature on related work. Section 3 will go over the methodology used in this research, followed by the description of the C-MAPSS case study in Section 4. Finally, we will discuss the results in Section 5, finishing with Section 6 where we will conclude this research and propose recommendations for future work.

2. Related work

In this section, the theoretical background and previous work used in this research will be discussed. In subsection 2.1, the field of Prognostics and Health Management will be discussed, focusing on Remaining Useful Life (RUL) estimation and the current methods applied to this field. In subsection 2.2 the Long Short Term Memory (LSTM) model will be briefly introduced and discussed, followed by subsection 2.3 where the fundamentals of Bayesian models including previous work applied to PHM. Finally, in subsection 2.4, the field of counterfactual (CF) explanations will be introduced followed by data augmentation methods in subsection 2.5.

2.1. Remaining Useful Life estimation

Prognostics and Health Management (PHM) is an engineering discipline focused on reducing maintenance costs while improving maintenance performance through the implementation of assessment, prognosis, diagnosis, and management strategies. At its core lies failure prognostics, which entails predicting a systems future behaviour with the goal of predicting as accurately as possible if, when, and how a system will fail. This includes (but is not limited to) predicting the Remaining Useful Life (RUL) of a system. [43].

In general, there are four main approaches to RUL estimation, reliability based, model based, data-driven and hybrid, as described by [52]. Reliability based approaches are considered the simplest, as they look at historical maintenance and failure events to predict future events while not

taking into account the actual degradation indicators. Model based approaches (also known as physics based approaches), are a more widely used PHM approach due to the accuracy, precision and real time performance, however they require a deep understanding of the physics of a system and quickly deteriorate in performance as the complexity of a system increases [11]. Data-driven methods offer a preferable alternative to model-based approaches for prognostics system development due to their ease of development, lower cost, and minimal need for understanding system physics, leveraging AI and ML for high reliability and effective computation [68]. One of the benefits of data-driven methods is the ability to implement various data streams and features such as a variety of sensors [66]. However, their optimal performance depends on the availability of substantial historical and current data, with accuracy improving as the dataset of failure events increases. Lastly, the hybrid approach, as the name suggests, aims to combine the strengths of both model based and data-driven approaches. In this research, the data-driven approach was used due to the relative ease of implementation and the amount of historical data available.

An overview of the main data-driven methods applied in RUL estimation is provided by Ansari et al. [1]. They divide the various methods into two main categories, statistical and machine learning. Statistical methods make use of empirical knowledge and data to build statistical models for RUL estimation. They are considered relatively simple, accurate and easy to implement. Models used in previous work are for instance the Auto-regressive Integrated Moving Average (ARIMA) technique (used by Li et al. [35], Chen et al. [8], and Zhou Y. et al. [71]), the Grey Model (GM) (used by Zhou Z. et al. [72], Gu et al. [22] and Zhou D. et al. [70]), the Wiener Process (WP) (used by Lei et al. [32], Tang et al. [54], Feng et al. [12] and Xu et al. [67]), and Entropy analysis (used by Hu et al. [26]).

Machine learning (ML) models on the other hand, work by mapping the inputs of a system to an output, this mapping of input to output varies per machine learning method. The main advantage many ML methods have is their ability to extract and complex and non-linear relationships in its prediction model [17]. In previous RUL estimation work, various ML models have been applied, such as Naive Bayes (used by Ng. et al. [42], Jafari et al. [27] and Galal et al. [15]), Support Vector Regression (used by Wang et al. [60] and Patil et al. [47]), Relevance Vector Machines (used by Wang et al. [59] and [37]), Gaussian Process Regression (used by Li et al. [33], Liu and Chen [38] and Li et al. [34]) and Deep Learning (used by [46, 10, 34, 9, 39, 50, 63, 73, 25, 69, 28, 49]). In this research, we employ a deep learning method, chosen for its capacity to discern complex, non-linear relationships within large amounts of data [29] and the substantial presence of preceding work available to build upon.

2.2. LSTM models

Long Short-Term Memory (LSTM) networks are a specialized type of Recurrent Neural Networks (RNN) designed to learn long-term dependencies in data sequences. Unlike standard feed-forward neural networks, LSTMs have a unique structure that includes memory cells, enabling them to store information over extended periods. This characteristic makes them particularly effective for applications involving time-series data. They improve upon traditional RNNs by overcoming the vanishing/exploding gradient problem, where during back-propagation, gradients tend to either vanish or explode over time when looking at distant events in the data. [21] The main structure of an LSTM network can be seen in Figure 1.

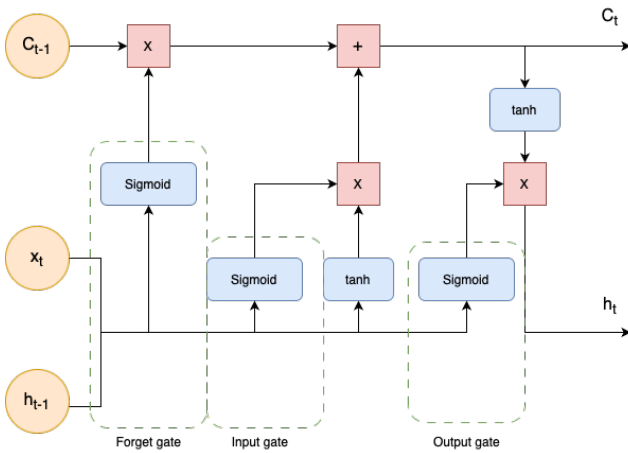


Figure 1: LSTM network structure

An LSTM model takes in the input vector x_t of a time step t in the data sequence and combines it with the hidden state of the previous time step h_{t-1} . This information goes into the forget gate, which decides how much information from the cell state C_{t-1} (the memory of the network) to remember or forget. After this, the input gate decides how much of the new information from x_t and h_{t-1} to add to the cell state, creating the current cell state C_t . Finally, in the output gate, x_t and h_{t-1} used to decide how much information from C_t will be used in the next hidden state h_t . In the end, the final hidden state also acts as the final output of the LSTM network.

While evaluating several Deep Neural Network (DNN) models have shown varying results in performance, the LSTM appears to be a favored method for RUL estimation [1]. It has been applied to RUL estimation in previous work by (among others) Park et al. [46], Choi et al. [10], Chinomona et al. [9], Liu et al. [34] and Zraibi et al. [73].

2.3. Bayesian models

Bayesian models are based on Bayes' theorem, which is a fundamental principle in probability theory and statistics. Mathematically it is expressed as seen in Equation 1, with H

being the hypothesis and E being the evidence. It describes the probability of an event, based on prior knowledge of conditions that might be related to the event [56].

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)} \quad (1)$$

Applying Bayes' rule to machine learning, as shown by Bishop et al. [5], Equation 1 is rewritten into the form of Equation 2, where the goal is to find the *distributions* of the model parameters w given the data D (so $P(w|D)$). This probabilistic framework allows for ML models using Bayesian methods to quantify uncertainty along with their predictions, instead of providing only point estimates.

$$P(w|D) = \frac{P(D|w)P(w)}{P(D)} \quad (2)$$

One of the challenges of Bayesian models relates to the marginal likelihood parameter $P(D)$, which represents the probability of seeing data D regardless of the hypothesis. In other words, it is the combined likelihood of the observed data under all possible model parameters, mathematically denoted by Equation 3. Calculating this parameter exactly is often intractable, as such there are approximation methods necessary to approximate this parameter. For instance Markov Chain Monte Carlo (MCMC) [18], Variational Inference (VI) [6], or Monte Carlo Dropout [14].

$$P(D) = \int P(D|w)P(w)dw \quad (3)$$

There are various Bayesian models that are able to make use of Bayes' rule and quantify uncertainty (e.g. Naive Bayes [62], Bayesian linear regression [40], Bayesian Networks [48], Gaussian Process Regression [61] and Relevance Vector Machines [55]), however this research will focus on Bayesian Neural Networks (BNN).

BNNs differ from their deterministic counterparts by representing their weights and biases as distributions rather than point values. Training a BNN involves updating its prior parameter distributions using the training data, resulting in a posterior distribution of parameters used for generating predictions and quantifying uncertainty [65].

BNNs have been applied to RUL estimation in previous work. Benker et al. [3] used a Bayesian DNN and CNN (Convolutional Neural Network) while Cacaes et al. [7] used and compared multiple Bayesian RNN networks.

2.4. Counterfactual Explanations

The field of explainable AI (XAI) aims to open the 'black box' of machine learning models and make the workings and outputs of these models more explainable, justifiable, and more useful. For the field of PHM, Nor et al. [44] performed a comprehensive literature review of XAI in PHM. They

show that XAI is mainly applied in the form of inherently interpretable models, rule and knowledge based models and attention mechanisms. One example of XAI being applied to RUL estimation comes from Hong et al. [24], who applied it to the C-MAPSS using the SHAP explanation model. Other previous work regarding XAI in PHM are for instance from Sundar et al. [53] using the LIME model, and Onchis et al. [45] who used LIME and SHAP.

Counterfactual explanations is an XAI method introduced by Wachter et al. [58]. It is a model agnostic and local XAI method, meaning it provides explanations based on solely the inputs and outputs of a model without looking at the underlying model architecture, and it provides explanations for separate predictions rather than the global model behaviour [57]. CF explanations work by asking the question "What if?", for instance "What if the input was not X but Y ? What is the output?", or the other way around; "What if the output was Y instead of X , what would have been the input?".

In general, a counterfactual explanation can be defined as a perturbation of the input \mathbf{x} to generate a different output y . This perturbed input can then be seen as a counterfactual example \mathbf{c} . In mathematical form (see Equation 4), our objective is to minimize the y loss such that a different prediction is generated, while also minimizing the distance between the original input \mathbf{x} and the counterfactual input \mathbf{c} (referred to as proximity).

$$\mathbf{c} = \arg \min_{\mathbf{c}} [y_{\text{loss}}(f(\mathbf{c}), y) + |\mathbf{x} - \mathbf{c}|] \quad (4)$$

In this research, we are interested in questions such as "What feature change will result in a certain in/decrease of the predicted RUL?", or "What feature change will result in a more certain prediction?", making CF explanations an ideal method to apply. Furthermore, to the best of the author's knowledge, CF explanations have yet to be applied to RUL estimation.

2.5. Data augmentation

Data augmentation for deep learning involves artificially increasing the size and diversity of a training dataset by creating modified versions of the existing data. This can include transformations like rotating, flipping, scaling images, or adding noise to audio. The goal is to improve model robustness and performance by exposing it to a wider range of variations without needing additional real-world data [20]. In a review by Wen et al. [64], various data augmentation methods applied to time series data is discussed. These methods range from relatively simple time domain augmentations such as window cropping, dynamic time warping or Gaussian noise injection, to more advanced methods such as decomposition based methods or statistical generative models. Counterfactual explanations can also be applied as a data augmentation method, as is done in previous work. For example, Kacprzak et al. [30], apply CF explanations to an

image classification problem, while Hasan et al. [23] apply it to the Adult-Income dataset. In this research we propose an adaptation of this approach, where we aim to apply CF explanations generated using Bayesian uncertainty as a form of data augmentation to a time series dataset.

3. Methodology

In this section, we will go into the methodology and setup used to conduct this research. The training and application of the BNN model used in this research will be discussed in subsection 3.1. We then go into our CF generation model in subsection 3.2, followed by our method for using CF explanations to provide insights on how to increase the RUL in subsection 3.3. We then discuss our data augmentation method where we combine Bayesian uncertainty with CF explanations to improve model performance, explained in subsection 3.4. Finally, we will discuss the performance metrics used in this research in subsection 3.5.

3.1. BNN model setup

The setup of the BNN model was inspired by Caceres et al. [7]. They compare the performance of various Bayesian RNN models applied to predicting the RUL using the C-MAPSS data set. For the FD001, the LSTM model had the best performance. As such this model was applied in this research.

The model architecture, as shown in Figure 2, consists of 1 LSTM layer, followed by 2 dense layers of 32 and 16 neurons respectively. Each input window of size (30 x 14) is fed into the LSTM layer sequentially. The LSTM layer consists of 32 units, resulting in 32 vectors that are outputted. Of these vectors, the final hidden state value is taken and fed into a dense layer of 32 neurons and subsequently into a layer of 16 neurons, eventually leading to a single output representing the RUL. Each weight and bias is represented by a distribution rather than a deterministic value, giving the model its probabilistic features.

Implementation of this model was done using the Bayesian Torch library by Krishnan et al. [31], who developed a library for Bayesian neural network layers and uncertainty estimation in Deep Learning. Finding the optimal distributions when training the Bayesian models was done using Variational Inference (VI), which aims to find the best fitting distribution by minimizing the Kullback-Leibler (KL) divergence between the distributions. The Bayesian Torch library uses a Gaussian distribution in the VI process. Aside from the model architecture, Caceres et al. [7] also performed a grid search to find the optimal model hyperparameters, these were also applied in this research, as seen in Table 2. During training, a decaying learning rate was used to smooth the learning in the later epochs, which decreased the learning rate to 70% of its original value in 60 epochs. Additionally, an early stopping method was also applied that cuts off the training process when the validation loss does not improve for a set amount of epochs, which is done to prevent

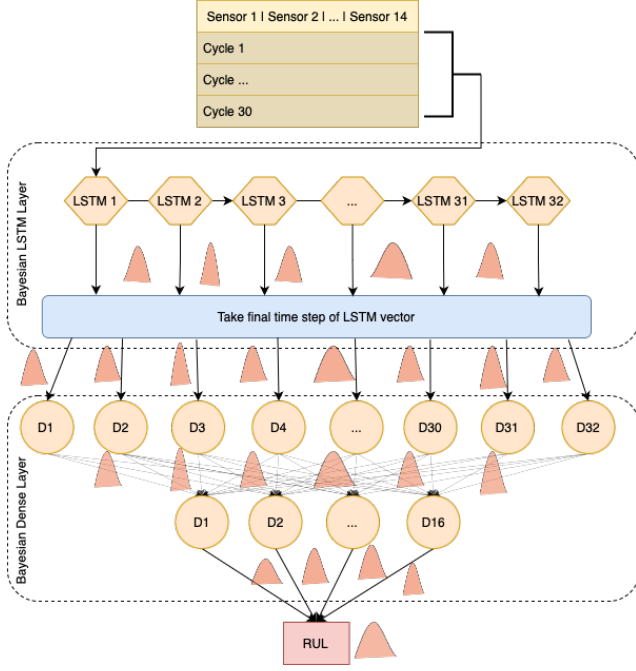


Figure 2: BNN architecture. 1 LSTM layer followed by 2 dense layers.

Table 1

FD001 dataset division for training, testing and CF generation

Set	# Engines	# Samples
Train	50	8461
Evaluate	10	1963
Remaining (CF)	40	7313

Table 2

Hyperparameters for BNN model development & training

Hyperparameter	Value	Hyperparameter	Value
BRNN Neurons	32	LR Decay [epochs]	60
Dense Layers	2	End LR	70%
Dense Neurons	32, 16	Validation split	20%
Epochs	100	Δ_{min}	0.25
Learning Rate	1e-3	Patience [epochs]	5

overfitting. For this research, if the loss of the validation set (which is a randomly sampled set of 20% of the training data) did not improve with $\Delta_{min} = 0.25$ for 5 epochs in a row, the training process was terminated and the weights resulting in the best loss were restored.

Our initial BNN model is trained on a training set of 50 out of the 100 engines and evaluated on 10 engines. The remaining 40 were used later for CF generation. We do not use these 40 engines to train the model on initially in order to prevent data contamination and overfitting. The dataset division is summarized in Table 1. To compare the effect of our de-noising method vs. including the noise in the data, we also trained a model on the noisy data.

When performing inference on the model, the weights and biases are randomly sampled from their distributions in each run. So, to generate the RUL distribution, we perform a small Monte Carlo simulation for each input file, resulting in a distribution of RUL outputs at each time step.

3.2. DiCE

The model of choice for generating the counterfactuals used in this research is the Diverse Counterfactual Explanations (DiCE) model, developed by Mothilal et al. [41], chosen among others for its ease of implementation, extensive documentation, compatibility with ML models, customizability and overall performance.

DiCE expands upon the basic counterfactual concept as described in Equation 4, by making an adapted version as seen in Equation 5. In this equation, the first term encourages the counterfactual input to produce a different output ($f(\mathbf{c}_i) = y$). The second term aims to keep the counterfactual input as close to the original input as possible, the third term seeks diversity among the k counterfactuals, and λ_1 and λ_2 are hyperparameters. DiCE iterates over the loss function until it converges and meets the desired condition (achieving a different output) or for a maximum of 5,000 steps, at which point it returns no result. It's worth noting that all \mathbf{c}_i values are initialized randomly.

$$C(\mathbf{x}) = \arg \min_{\mathbf{c}_1, \dots, \mathbf{c}_k} \left[\frac{1}{k} \sum_{i=1}^k \text{yloss}(f(\mathbf{c}_i), y) + \frac{\lambda_1}{k} \sum_{i=1}^k \text{dist}(\mathbf{c}_i, \mathbf{x}) - \lambda_2 \text{dpp_diversity}(\mathbf{c}_1, \dots, \mathbf{c}_k) \right] \quad (5)$$

3.3. Counterfactual RUL explanations

The DiCE model was applied to this research as shown in Figure 3. Firstly, the (30x14) input was converted to a (1x420), as DiCE is not able to handle 2D inputs such as our time series data. After this, DiCE altered the inputs slightly and checks if the desired output was achieved with this new input using the BNN model. If so, the altered input was accepted as a valid CF explanation and was then converted back to the original shape of (30x14). If not, the input was repeatedly altered until it either became a valid counterfactual, or it did not, after which the DiCE model outputs "No CF found".

For this research, some significant alterations and additions have been applied to the DiCE model. First of all, as mentioned above, the input data needed to be flattened to a 1D input. This causes every value to be considered as its own independent input, rather than part of a time series. An addition was made such that the data points in the flattened array were linked by their time stamp in the time series. As such, changing a feature value at one point in the time series will have an effect for later time points. Also, the desired output was changed from being a set value range to being relative to the original output value (e.g., desired output =

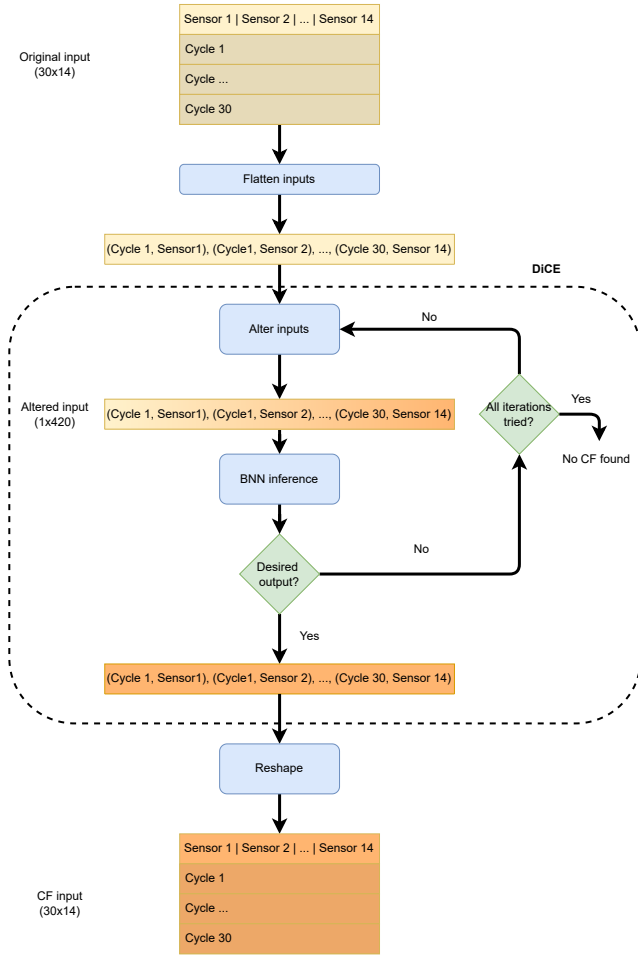


Figure 3: DiCE model flowchart

original output + [10-11] cycles).

The first application of counterfactuals in this research had the goal of finding explanations on how to increase the engine RUL. For generating counterfactual explanations for the RUL, we took the mean of the prediction distribution as the output to adjust. By generating inputs that would have lead to a higher/lower RUL, we aimed to find explanations on how the life of the engine could have been extended.

After applying the DiCE model, it successfully modified nearly all original input files to achieve a specified desired output, with the exception of approximately three files for which DiCE could not generate a valid counterfactual. Given the significant overlap among the sliding windows, we can ascertain whether multiple counterfactual explanations for the same time step are consistent with each other.

3.4. Counterfactual uncertainty & model retraining

The second application of counterfactuals in this research has the goal of improving the models predictive performance. Generally, the more data you have available for a model to train on, the better its performance will be. As

acquiring more data can be a long and costly endeavour, we propose a method to synthetically create more training data based on the data on hand by creating counterfactual inputs. In this case, we will generate counterfactual inputs with a lower uncertainty than the original. Our hypothesis is that by doing this, we make the input data 'sharper', leaving us with more certain features. These counterfactuals are generated similarly to the RUL counterfactuals, except we take the standard deviation (STD) σ as the output, after which we run DiCE with the request to decrease it. For this, we ask DiCE to decrease the STD by 15-25% compared to the STD produced in the original prediction. Note that here we do not use the time-series modification of the DiCE model, as we want to give the DiCE model as much flexibility as possible to find the sources of uncertainty in the data.

For this application, we create a data pipeline as described in Figure 4. We split the FD001 data set in three distinct parts. Of the 100 engines, 50 are used to train an initial BNN model, as described in subsection 3.1, and 10 will be used as an evaluation set. The remaining 40 engines will be used to create the counterfactual explanations. Firstly, the initial BNN model will be used by DiCE to generate the counterfactuals using the 40 remaining engines. These CF inputs will be verified to check their effect of the uncertainty reduction. Next, a BNN model will be trained using the 50 training engines including the remaining 40 non-CF engines, this model will serve as our baseline, as it is trained on all available training data (excluding the evaluation set). Then, a new BNN model will be trained using the aforementioned 50+40 engines from the original data-set, including a set of CF inputs generated using the 40 non-CF engines, increasing the number of training samples which should lead to a higher performance.

Aside from the CF explanations with reduced uncertainty, we also consider the CF explanations generated using the method described in subsection 3.3, which have alternative inputs resulting in a higher/lower RUL prediction. By doing this, we can compare the effect of the synthetic data created by the different CF explanation methods. The 5 models to be compared are:

- *Baseline*: Trained using the original 50 training engines + 40 non-CF engines. This is the baseline model using all the available data to train aside from the evaluation set. (15,744 samples)
- *Augmented Uncertainty CF*: Trained using the original 50 training engines + 40 non-CF engines + CF inputs with reduced uncertainty based on the 40 non-CF engines. (26,151 samples)
- *Augmented RUL CF (increasing)*: Trained using the original 50 training engines + 40 non-CF engines + CF inputs with increased RUL based on the 40 non-CF engines. (23,087 samples)

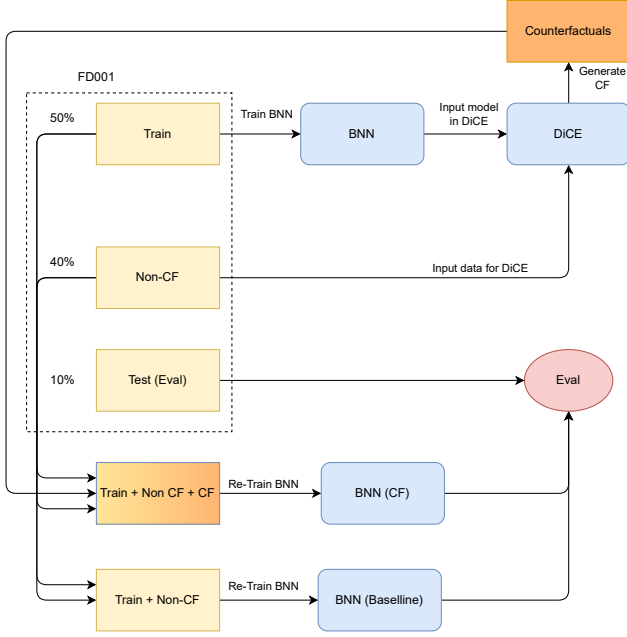


Figure 4: Counterfactual re-training data flow

- *Augmented RUL CF (decreasing)*: Trained using the original 50 training engines + 40 non-CF engines + CF inputs with decreased RUL based on the 40 non-CF engines. (23,087 samples)
- *Augmented RUL CF (combined)*: Trained using the original 50 training engines + 40 non-CF engines + CF inputs with increased and decreased RUL based on the 40 non-CF engines. (30,400 samples)

3.5. Performance metrics

In order to analyze the performance of our model and results, we introduce a selection of performance metrics. Firstly, we apply the Root Mean Squared Error (RMSE), as it is a commonly used error metric with advantages such as being in the same unit as the predictions, allowing for more interpretability, and penalizing larger error more due to its square term. It is defined as shown in Equation 6.

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (RUL_{pred} - RUL_{true})^2} \quad (6)$$

Another metric often used in RUL estimation is the $\alpha - \lambda$ accuracy [19]. This metric evaluates if the model's predictions lie within certain bounds defined by α at a certain time point defined by λ . The score is calculated based on Equation 7 for a given time point λ . As we use a Bayesian model that provides distributions instead of single RUL values, we will look at the fraction of the distribution that lies within the α bounds. For this research we take $\alpha = 0.2$ as this is a commonly used value in literature [4].

$$\alpha - \lambda \text{Accuracy} = \begin{cases} 1, & (1 + \alpha) \cdot RUL \leq RUL_{pred} \leq (1 - \alpha) \cdot RUL \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

Lastly, an asymmetric scoring function introduced by Saxena et al. [51], who created the C-MAPSS data set, will be applied. This scoring function, as shown in Equation 8, punishes late RUL predictions more than early RUL predictions. Here, τ represents the error between the predicted and true RUL.

$$s(\tau) = \begin{cases} s_1(\tau) = e^{-\frac{\tau}{13}} - 1, & \text{for } \tau < 0 \\ s_2(\tau) = e^{\frac{\tau}{10}} - 1, & \text{for } \tau \geq 0 \end{cases} \quad (8)$$

4. Case study: C-MAPSS

For this research, we used the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dataset as a case study to apply our proposed methods. C-MAPSS is a tool created by NASA [13] which can simulate a large commercial turbofan engine. Using this tool, a data set was created for the 2008 international conference on Prognostics and Health Management (PHM08) where attendees were challenged to create their best RUL prediction methods. Currently it is a widely used data set in RUL estimation research. [51]

The data set is structured as followed, there are four separate sub data sets (FD001, FD002, FD003, FD004), which vary in number of engines, operating conditions and failure modes. For the goal of this research, the relatively simple FD001 set will suffice. It has 100 train/test trajectories, one operating condition and one failure mode. Each engine within the dataset can be regarded as a member of an identical fleet of engines. Each engine contains a time-series set of data, where the amount of time steps represents an operating cycle of the engine. We assume that every engine operates at its standard capacity and begins to deteriorate at some point in the time series. However, the initial wear state of each engine remains unknown. Once a specific degradation threshold is reached, the engine is considered nonoperational and has effectively reached End of Life (EOL). Furthermore, the dataset is affected by a certain level of noise.

For this research, we performed some pre-processing steps on the dataset before applying it to our models. For each cycle per engine, the data set contains the following [Engine number, cycle number, operational setting 1-3, sensor measurement 1-21]. As FD001 only has one operating condition, we remove the operational settings from the data set. Also, we remove the cycle number to prevent later overfitting. This leaves us with the raw sensor data found

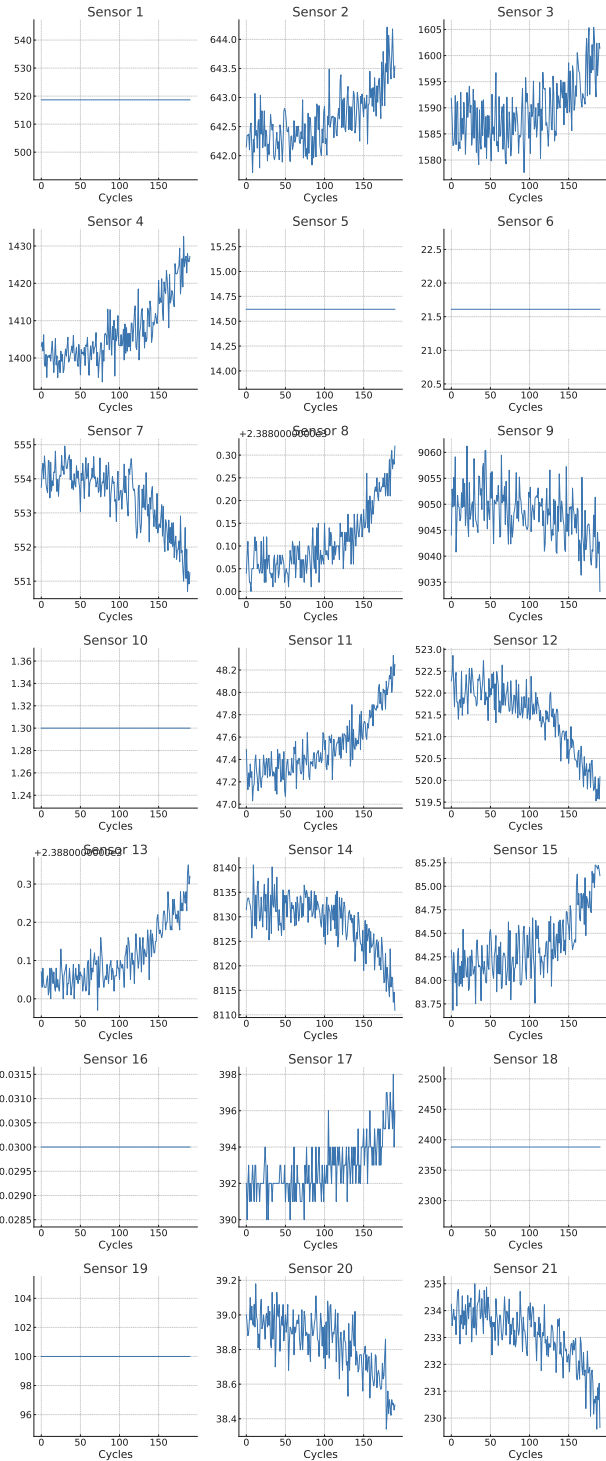


Figure 5: Raw sensor data of engine 1

in Figure 5.

The next steps in the pre-processing process consist of firstly, removing the unnecessary sensors, where it is clear that sensors 1,5,6,10,16,18 and 19 do not provide any useful input, leaving 14 useful sensors. All sensor names can be

found in Table 6. Secondly, de-noising the sensor trajectories, where each sensor was subjected to a Savitzky-Golay smoothing and differentiation filter [16] using a 3rd degree polynomial. Thirdly, all sensor inputs were normalized to a scale of [-1,1], to ensure each feature is interpreted the same by the model.

The final step of the pre-processing process takes inspiration of Caceres et al. [7], who also uses the C-MAPSS data set for RUL prediction. They propose a sliding window approach as shown in Figure 6, where the ground truth RUL is calculated by counting the amount of cycles remaining until the end of the data set per engine. A window size of 30 cycles will be used, ensuring each RUL prediction will be based on not only the current cycle, but the 30 cycles before, which can allow an LSTM network to look for degradation trends over time. By using a sliding window, we increase the amount of training data and ensure all inputs are of equal size. For the ground truth RUL we also incorporate a piece-wise linear correction, used by Benker et al. [3]. This limits the maximum ground truth RUL to 120 cycles, which attempts to prevent the model from trying to find fault modes in the healthy regime of the engine lifetime, but rather focus on finding degradation patterns more towards the EOL region. In previous work, L.D. Liberia [36] followed a similar approach to these pre-processing steps, who's implementation into Python will be used as a basis for this research.

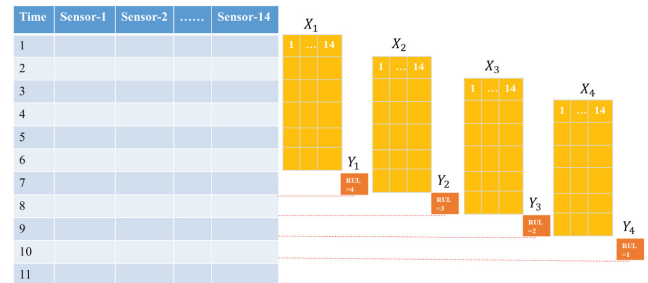


Figure 6: Sliding window representation of data set [7]

Performing all steps for 100 engines with varying lifetimes results in 17731 samples of size (30, 14), one of which can be seen in Figure 7. Note that for the remainder of this research, the de-noised as well as the noisy data sets will be used, as we also want to see how noisy data affects the CF explanation generation.

5. Results & Discussion

In this section the results will be discussed. First, we go into the performance of the BNN LSTM model developed for RUL prediction in subsection 5.1. Secondly, in subsection 5.2 we will go into the results and findings of the CF generation method with the goal of gathering more insights and explanations from the BNN model using the DiCE CF model. Finally, in subsection 5.3, we will go into the results

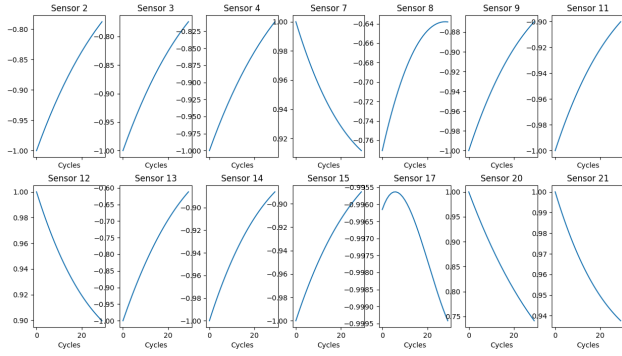


Figure 7: Processed sensor data of sliding window 1 of engine 1

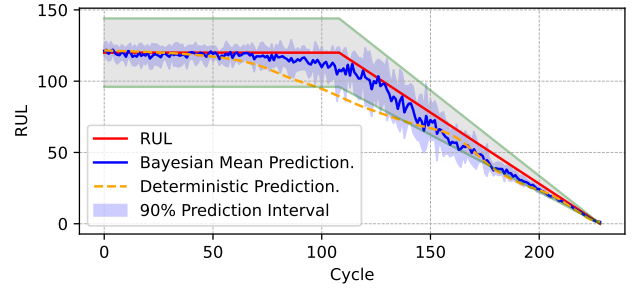
of our proposed method of using counterfactuals to reduce the uncertainty of the input data, and subsequently use this data to retrain the BNN model.

5.1. BNN predictive capabilities

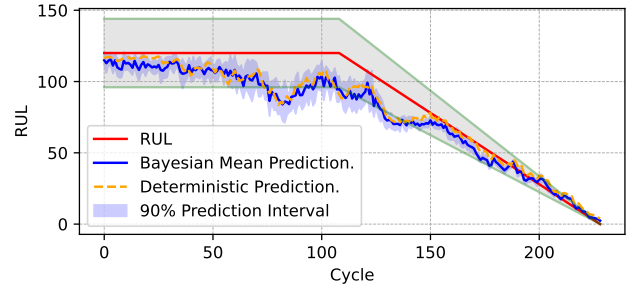
The initial BNN model, has been trained as described in subsection 3.1. Also, for comparison, a deterministic NN model has been trained using the same architecture as the BNN model. We take test engine 4 as an example case, seen in Figure 8. Note that the remaining test engines performed similarly. This example give us an indication of how the model performs over the entire life cycle of the engine. Looking at the model trained on the de-noised data, seen in Figure 8a, we see the predicted RUL values closely following the true RUL values and remains largely inside the $\alpha - \lambda$ bounds throughout the life cycle (indicated by the green lines and grey area). We do see however that the uncertainty of the BNN increases slightly around the piece-wise linear transition point, before converging again to a more certain prediction towards the EOL. This illustrates how the model initially searches for a degradation pattern within the data, leading to predictions with higher uncertainty. Once it identifies the trend, the predictions become more certain. Comparing this to the deterministic predictions, which also seems to diverge around the transition point, we simply observe a larger error with no idea of how certain the model is, showing the main advantage of using a Bayesian model.

Looking at the model trained on the noisy data, seen in Figure 8b, we see a that the model has a slightly higher error than that of the de-noised data. Most notably, while we might expect the RMSE error to be higher than the de-noised data, the uncertainty is smaller, meaning that the model is wrong while being quite certain it is right. While this behaviour is not favorable for a prediction model, the error and $\alpha - \lambda$ still show that this model is still a capable prediction model.

The overall performance of all evaluation samples can be seen in Table 3. In general, the model trained on the de-noised data performs better than the one trained on the noisy data. The only notable exception is the average uncertainty, indicated by the standard deviation (STD), which is lower for



(a) Trained using de-noised data. Bayesian RMSE: 5.81 cycles, Deterministic RMSE: 13.51 cycles



(b) Trained using noisy data. Bayesian RMSE: 15.34 cycles, Deterministic RMSE: 13.83 cycles

Figure 8: BNN model performance of test engine 4. $\alpha = 0.2$

Table 3

Overall test performance of BNN model over 10 test engines

Metric	De-noised	Noisy
RMSE	10.95	13.67
STD	7.40	4.70
Total score	3662.34	6919.16
Predictions in $\alpha = 0.2$	66%	63%

the noisy data. As mentioned above, this effect is not likely to result from the noisy data simply conveying less uncertainty, but could rather originate from other effects. One of these effects could be that the model is overfitted to the noise in the training data, leading to overly confident predictions that do not generalize well to new, unseen data. For these reasons, only the de-noised input data was considered in the remainder of this research.

5.2. RUL counterfactuals

As described in subsection 3.3, we run the DiCE model to create CF inputs which results in a different RUL. We are interested in what inputs would generate a better RUL, but also which would generate a worse RUL. After running DiCE with an output requirement range of $[-11, -10]$ and $[+10, +11]$, we have generated two sets of counterfactual inputs that should output approximately 10-11 cycles lower and higher respectively. Looking at Figure 9, we verify that using the counterfactual inputs generally results in the desired altered outputs, with some exceptions where the

DiCE model was not able to find a valid counterfactual.

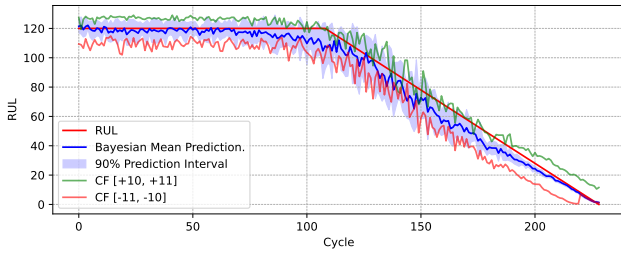


Figure 9: RUL prediction of engine 4 including CF output for [-11,-10] cycles decreased (red) and [10,11] cycles increased (green) RUL.

Looking at Figure 10, we see the (de-noised) sensor inputs for the entire life cycle of engine 1. This engine is taken as an example, however the overall trends are consistent over the entire evaluation data set. The green dots indicate alterations to the original inputs such that the RUL is increased by 10-11 cycles at that time point, and vice-versa for the red points. The clustering and overlap of points is due to the fact that each sliding window input is altered to its own counterfactual input, and each sliding window has a large overlap with its neighbouring windows. This does however allow us to analyse whether or not the various CF explanations of the same time point coincide with each other. For instance, a dense green cluster is likely to indicate that altering the input there should lead to a better RUL. This also allows us to more easily spot outliers and trends, where a subset of extreme or contradicting CF explanations are more easily identified.

Looking at Figure 10, we see that in general, the CF explanations are more extreme towards the earlier part of the life cycle (with notable outliers such as seen in sensor 9). This can be explained in a comparable manner to the piece wise linear correction as discussed in Section 4, where we do not want the model to try and find degradation patterns that are not yet there. This could lead to the more erratic behaviour of the CF explanations, as they try to change the model outcome.

Looking at the CF inputs for the sensors, we can identify trends that could help us create explanations on how to affect the RUL of an engine. Let us take a selection of sensors to expand upon. We look at sensor 2, sensor 17, and sensor 20 as seen in Figure 11. Figure 11a shows all the CF input points of each respective sensor over the engine lifetime, which is already able to show some clear trends. To aid interpretability, we introduce Figure 11b, where the average difference from the original input is displayed (note that the first and last 30 cycles are cut off, this is to ensure that the average is always taken over 30 samples).

Using Figure 11, we are able to make observations and interpretations regarding the CF explanations. Firstly,

looking at sensor 2 and sensor 17, we see that for both sensors Figure 11a and Figure 11b indicate that a lower sensor value would lead to a better RUL and vice versa. While this information does not directly indicate how to improve the RUL of the engine, maintenance engineers could use this information to focus their efforts on the effects that cause the sensors to increase in value.

Looking at sensor 20, a similar pattern emerges where higher sensor values generally suggest a better RUL, except near the EOL, where it suddenly flips. This could be due to two main factors: firstly, the DiCE model’s use of random data perturbations, combined with the high-dimensional nature of the inputs (14 sensors with time series data), might lead to counter-intuitive model responses. Secondly, difficulty in generating decreasing RUL explanations towards EOL arises since a RUL below 0 is beyond the dataset’s scope.

This phenomenon highlights a primary issue with our method: limited insight into correlated inputs. The DiCE model aims for sparsity by changing only 3-5 sensors at a time for each counterfactual (CF) explanation, implying that each CF per sensor is part of a broader CF involving multiple sensors. Our sliding window approach, featuring significant overlap, helps mitigate this by smoothing the effect across the engine’s entire life cycle, offering a more comprehensive explanation for each sensor. However, this approach does obscure the interconnections between sensors. When these effects are consistent (i.e. different CF explanations suggest the same increase/decrease alteration for a given sensor), we see results such as seen in sensor 2 and sensor 17, when they are inconsistent (i.e. different CF explanations suggest different alterations for a given sensor), we might see effects such as seen in sensor 20.

Nevertheless, the proposed counterfactual explanation method has shown that more information and insights can be extracted from the input data aside from the RUL predictions. Also, the FD001 subset used is known for its relatively simplistic trends in the sensor data, simplifying the identification of direct relationships between sensor readings and engine health. Its simplicity allows for the assumption that reversing the sensor trends could directly improve the predicted RUL, i.e. if a sensor value increases towards EOL, decreasing it should extend the RUL. We see a similar trend when looking at Figure 10, where CF explanations increasing the RUL tend to oppose the original sensors trend and vice versa. We can thus verify that the CF approach is generally able to find correct and logical CF explanations, aside from the effects discussed above.

5.3. Uncertainty CF & re-training

Following the steps as described in subsection 3.4, we first verify the reduced uncertainty by evaluating both sets using the initial BNN model, the results of which can be seen in Table 4. Here we can make a couple of observations, the

Combining Counterfactuals with Bayesian Uncertainty

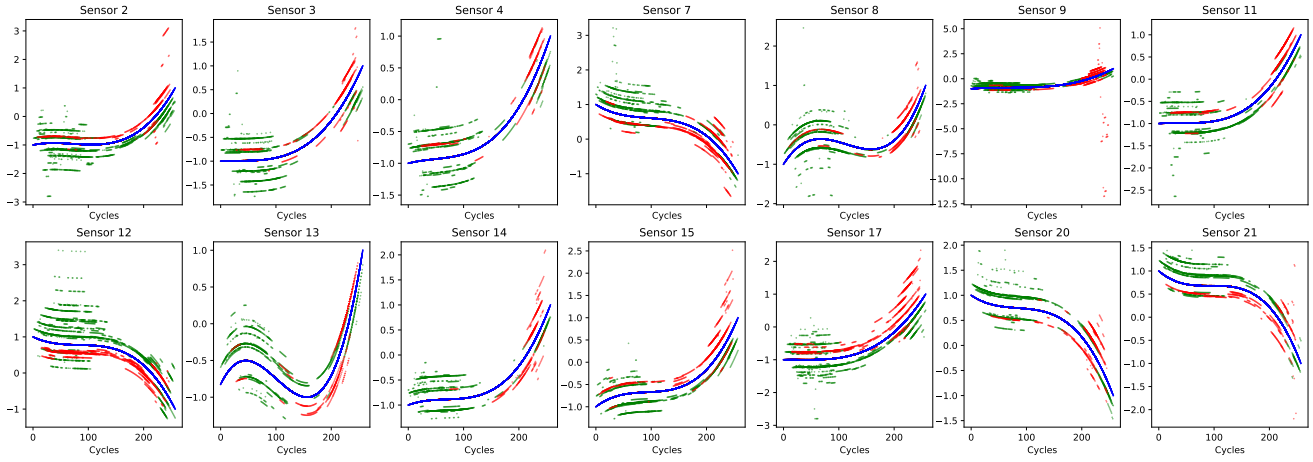
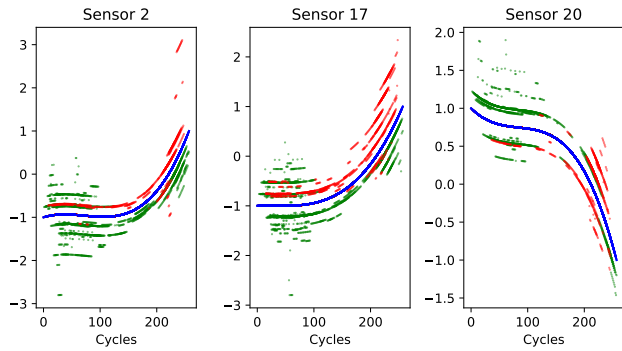
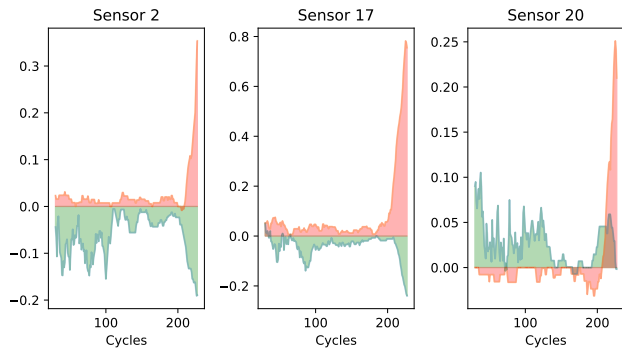


Figure 10: CF input of engine 4. Center line = original input, red = [-11, -10] worse RUL, green = [10, 11] better RUL



(a) Raw CF explanations



(b) Average input difference between original and CF explanations

Figure 11: CF input of sensor 2, 17 and 20 of engine 4. Center line = original input, red = [-11, -10] worse RUL, green = [10, 11] better RUL

first being that the overall average STD has been decreased. However, when looking over the life cycle of the engines, we see that the STD alterations are not consistent and vary highly over time. This shows that the DiCE model is not able to consistently decrease the uncertainty of the input data. This behaviour is likely due to the DiCE model not being optimized to handle uncertainty as a target in its CF explanations. Nevertheless, we were still able to achieve an

Table 4

Average standard deviation over engine life cycle of the 40 original (Non-CF) and CF input engines.

	(-, 120)	(120, 60)	(60, 30)	(30, 10)	(10, 0)	Overall average
Non-CF	5.93	9.70	5.34	1.28	0.58	7.12
CF	6.83	6.93	7.80	7.85	7.70	7.06

overall uncertainty reduction in the augmented input data.

Using the CF inputs with reduced uncertainty and the CF inputs with increased/decreased RUL we trained the 5 models as described in subsection 3.4. A comparison of the model performance can be seen in Figure 12, and is also summarised in Table 5. (An expanded version of Figure 12 can be seen in Figure 13, which also shows the performance per section of the engine life cycle.) From here, it is clear what the effect is of adding the additional CF and non-CF data to the model performance.

In general, it is clear that the best performing model is the *Augmented Uncertainty CF* model. This shows that the addition of augmented data in the form of counterfactuals with reduced uncertainty can improve the model performance compared to the baseline model. We attribute this performance improvement to not only the fact that the model had more data points to train on, but also to the reduced uncertainty CF application.

This aspect is further highlighted in the performance of the augmented models using the CF inputs with increased/decreased RUL. Despite having access to a larger dataset, the improvement in performance across all three

Table 5
Performance comparison of 5 trained models

	Baseline	Augmented Uncertainty CF	Augmented RUL CF (increasing)	Augmented RUL CF (decreasing)	Augmented RUL CF (combined)
Average RMSE	9.56	8.47	9.60	9.01	10.26
Average STD	8.68	7.38	7.78	7.42	7.19
Average distribution $\alpha=0.2$	74%	79%	70%	78%	68%
Score	2802.69	2312.69	2815.19	2391.36	3148.24

models is not notably significant. This can be attributed to the nature of the augmented data, which closely resembles the original dataset. For example, if a CF input extends the RUL from 30 to 40 cycles, but the original dataset already includes a similar data point at 40 cycles, the addition of this augmented data point does not substantially benefit the model.

However, an exception is observed with the *Augmented RUL CF (decreasing)* model, which also performs better than our baseline. This distinction arises, firstly, because reducing the RUL creates a broader valid range for CF inputs. Especially in scenarios involving the 120 cycle limit, where the model, although not trained on values exceeding 120, shows a better alignment with values below this limit. Secondly, towards the engine's End of Life (EOL), decreasing the RUL introduces more examples of conditions with higher engine degradation, which is a critical area for the predictive model.

6. Conclusions & Recommendations

The goal of this research was to find and apply methods to combine counterfactual explanations with Bayesian uncertainty to improve the interpretability and performance of RUL estimation models. To achieve this, we set out the answer our research question: "*How can Bayesian uncertainty and Counterfactual Explanations be used in predictive maintenance to improve interpretability and predictive performance?*" The first step taken was to develop a Bayesian LSTM model and train it on the C-MAPSS data-set. For this research, we have succeeded in developing and training a model capable of predicting the RUL over the life-cycle of a series of engines with a high accuracy, including the ability to quantify uncertainty. The pre-processing steps taken, most notably the de-noising of the input data, showed to have a positive effect on the model performance when comparing to the noisy input data. However, the model trained on the noisy data showed a relatively low uncertainty for relatively high errors, which has been attributed in this research to the model overfitting to the noise in the data. If for future research one does not want to apply de-noising, we recommend

addressing this issue by pursuing options such as varying the training hyperparameters, pre-processing steps, or other relevant topics. Furthermore, we recommend expanding the data-sets used for future research to achieve more realistic scenarios and predictions, as the C-MAPSS FD001 data-set is known for being a relatively simplistic RUL estimation data-set.

In this research, utilizing the BNN model enabled the generation of counterfactual explanations that effectively predicted higher and lower RUL for engines. These explanations, derived from altered sensor inputs, showed trends that impact RUL both positively and negatively, serving as a practical guideline for enhancing engine lifetime. However, the analysis revealed variability in the predictability of sensor impacts on RUL, attributed to the high-dimensionality, interconnectedness, and temporal nature of the sensor data. The DiCE model, while proving a valuable proof of concept, faces challenges with this complex data architecture. Another contributor is the sliding window approach, which does provide a framework to verify if multiple CF explanations agree on the input alterations, but also causes potentially contradictory CF explanations for identical time points. For future research, it is recommended to explore alternative models better suited to handling such data complexities or to devise strategies for simplifying the data architecture to mitigate these issues. Despite these issues, the approach offers valuable insights into how sensor data correlates with engine health, providing a basis for maintenance strategies and further research into predictive maintenance modeling.

The final part of this research attempted to use counterfactual explanations as a data augmentation method to generate more data points for model training. This was done by using the Bayesian uncertainty quantified by the BNN model to generate CF explanations with a reduced measure of uncertainty. These uncertainty CF inputs, along with CF inputs with increased/decreased RUL, were added to the training data of a set of 5 models. Analysing the performance of these models shows that the addition of the CF inputs with reduced uncertainty improves the overall

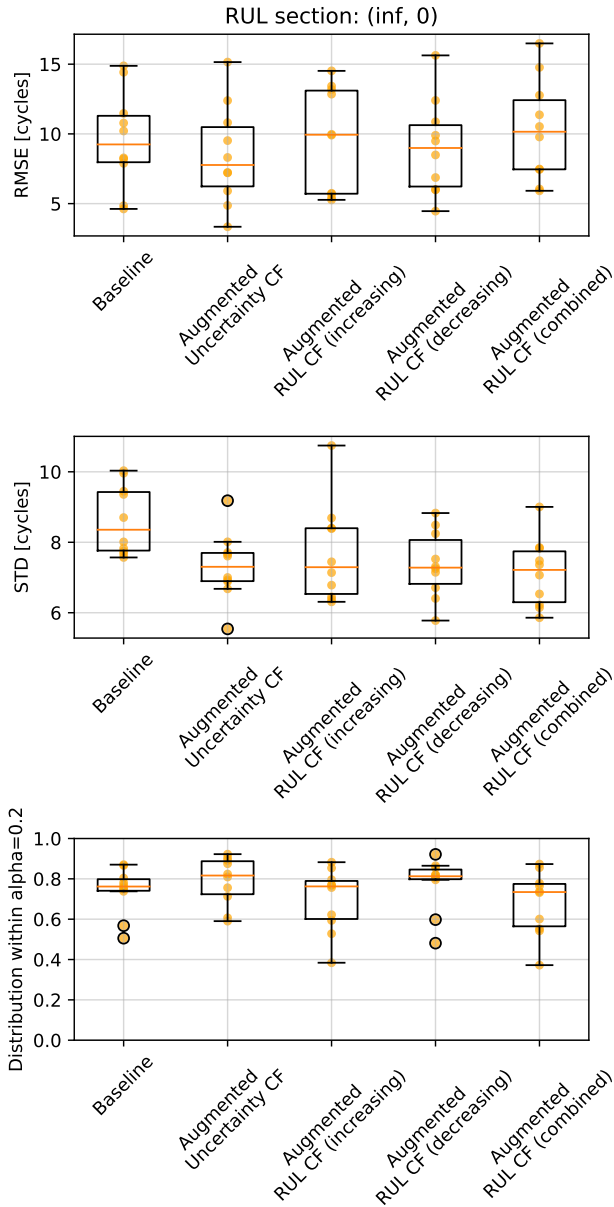


Figure 12: Performance of 5 trained models: RMSE (top), STD (middle), $\alpha - \lambda$ score (bottom)

model performance, confirming that this CF data augmentation method is a viable approach to model performance enhancement. For applying this method in future research, it is advised to implement a CF generation model tailored specifically for reducing uncertainty, such as the Counterfactual Latent Uncertainty Explanations (CLUE) model [2], in order to get a more consistent uncertainty reduction over the engine life cycle. Also, we recommend applying this method to a more complex data-set to get a better indication the performance differences of the data augmentation method, as currently all models evaluated perform quite well due to the relatively simple data-set. Lastly, we recommend looking into the fraction of CF inputs to real inputs in order to find the optimal amount of augmented CF data to add to the

training set in order to maximize the added performance while minimizing the risk of overfitting.

All in all, this study has successfully applied a counterfactual explanation approach to enhance interpretability and explainability in RUL estimation methods, offering a new perspective for maintenance strategies and PHM research. Additionally, integrating Bayesian uncertainty with CF explanations has shown to be a promising approach for data augmentation, enhancing the performance of our predictive model without relying on additional real-world data.

A. Appendix

Table 6: Sensor descriptions

Sensor index	Description	Units
1	Total temperature at fan inlet	$^{\circ}R$
2	Total temperature at low-pressure compressor (LPC) outlet	$^{\circ}R$
3	Total temperature at high-pressure compressor (HPC) outlet	$^{\circ}R$
4	Total temperature at LPC outlet	$^{\circ}R$
5	Pressure at fan inlet	psia
6	Total pressure in bypass duct	psia
7	Total pressure at HPC outlet	psia
8	Physical fan speed	rpm
9	Physical core speed	rpm
10	Engine pressure ratio	-
11	Static pressure at HPC outlet	psia
12	Ratio fuel flow to Ps30	pps/ps
13	Corrected fan speed	rpm
14	Corrected core speed	rpm
15	Bypass ratio	-
16	Burner fuel-air ratio	-
17	Bleed enthalpy	-
18	Demanded fan speed	rpm
19	Demanded corrected fan speed	rpm
20	High-pressure turbine (HPT) coolant bleed	lbm/s
21	Low-pressure turbine (LPT) coolant bleed	lbm/s

Combining Counterfactuals with Bayesian Uncertainty

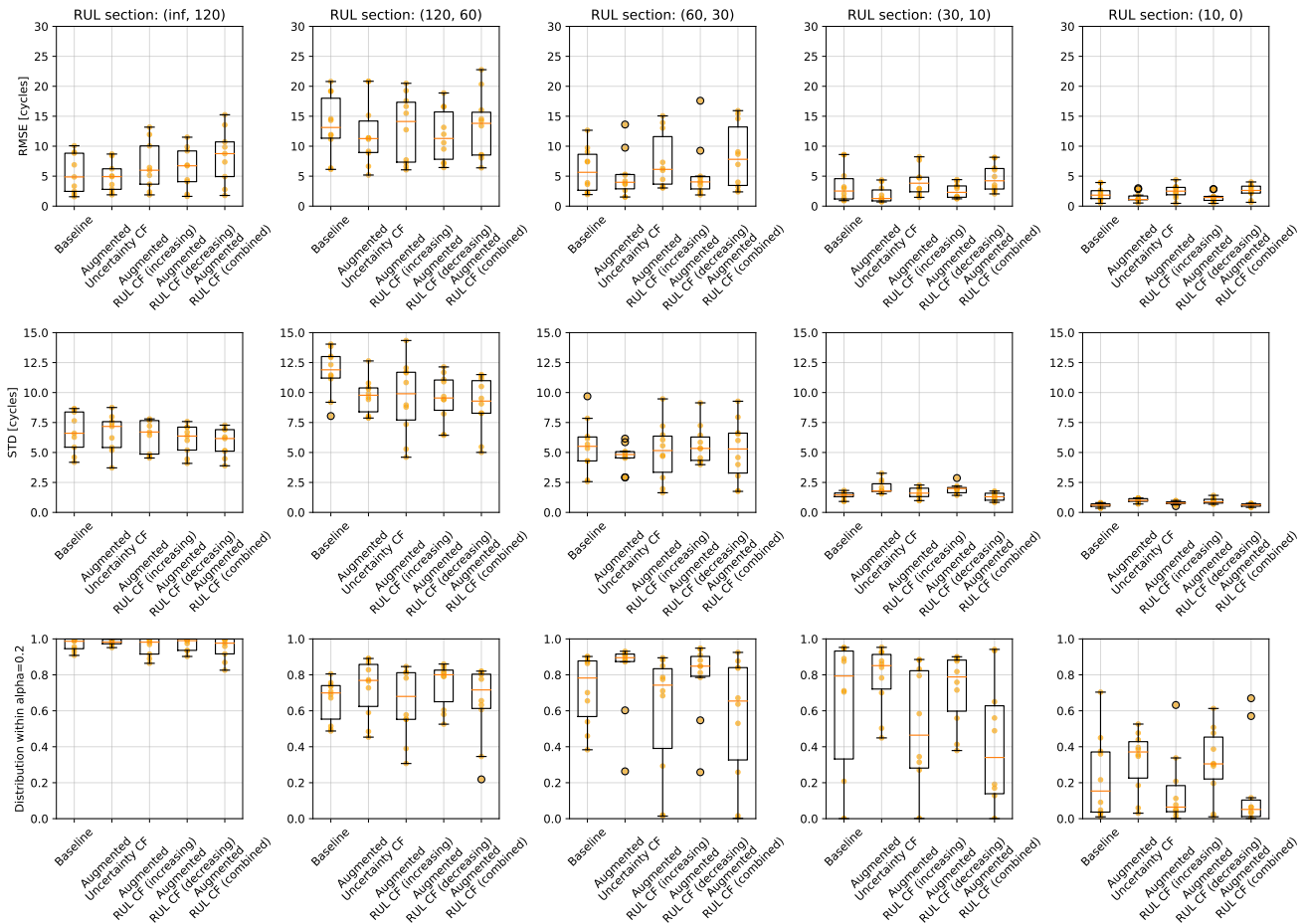


Figure 13: Overall performance of the 5 trained models per section of the life cycle.

References

- [1] Ansari, S., Ayob, A., Hossain Lipu, M.S., Hussain, A., Saad, M.H.M., 2022. Remaining useful life prediction for lithium-ion battery storage system: A comprehensive review of methods, key factors, issues and future outlook. *Energy Reports* 8, 12153–12185. doi:10.1016/J.EGYR.2022.09.043.
- [2] Antoran, J., Bhatt, U., Adel, T., Weller, A., Hernández-Lobato, J.M., 2021. Getting a {clue}: A method for explaining uncertainty estimates, in: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=XSLF1XFq5h>.
- [3] Benker, M., Furtner, L., Semm, T., Zaeh, M.F., 2021. Utilizing uncertainty information in remaining useful life estimation via Bayesian neural networks and Hamiltonian Monte Carlo. *Journal of Manufacturing Systems* 61, 799–807. doi:10.1016/J.JMSY.2020.11.005.
- [4] Biggio, L., Wieland, A., Chao, M.A., Kastanis, I., Fink, O., 2021. Uncertainty-aware remaining useful life predictor. *arXiv preprint arXiv:2104.03613*.
- [5] Bishop, C.M., Nasrabadi, N.M., 2006. *Pattern recognition and machine learning*. volume 4. Springer.
- [6] Blei, D.M., Kucukelbir, A., McAuliffe, J.D., 2017. Variational inference: A review for statisticians. *Journal of the American statistical Association* 112, 859–877.
- [7] Caceres, J., Gonzalez, D., Zhou, T., Drogue, E.L., 2021. A probabilistic Bayesian recurrent neural network for remaining useful life prognostics considering epistemic and aleatory uncertainties. *Structural Control and Health Monitoring* 28. doi:10.1002/stc.2811.
- [8] Chen, L., Xu, L., Zhou, Y., 2018. Novel approach for lithium-ion battery on-line remaining useful life prediction based on permutation entropy. *Energies* 11, 820.
- [9] Chinomona, B., Chung, C., Chang, L.K., Su, W.C., Tsai, M.C., 2020. Long short-term memory approach to estimate battery remaining useful life using partial data. *Ieee Access* 8, 165419–165431.
- [10] Choi, Y., Ryu, S., Park, K., Kim, H., 2019. Machine learning-based lithium-ion battery capacity estimation exploiting multi-channel charging profiles. *Ieee Access* 7, 75143–75152.
- [11] Elattar, H.M., Elminir, H.K., Riad, A.M., Riad, A.M., 2016. Prognostics: a literature review. *Complex & Intelligent Systems* 2(2), 125–154. doi:10.1007/S40747-016-0019-3.
- [12] Feng, J., Kvam, P., Tang, Y., 2016. Remaining useful lifetime prediction based on the damage-marker bivariate degradation model: A case study on lithium-ion batteries used in electric vehicles. *Engineering Failure Analysis* 70, 323–342.
- [13] Frederick, D.K., Decastro, J.A., Litt, J.S., 2007. *User's Guide for the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS)* URL: <http://www.sti.nasa.gov>.
- [14] Gal, Y., Ghahramani, Z., 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning, in: *international conference on machine learning*, PMLR. pp. 1050–1059.
- [15] Galal, M.A., Hussein, W.M., El-din abdel Kawy, E., Sayed, M.M., 2019. Satellite battery fault detection using naive bayesian classifier, in: *2019 IEEE Aerospace Conference, IEEE*. pp. 1–11.
- [16] Gallagher, N.B., 2020. Savitzky-golay smoothing and differentiation filter. *Eigenvector Research Incorporated*.
- [17] Ge, M.F., Liu, Y., Jiang, X., Liu, J., 2021. A review on state of health estimations and remaining useful life prognostics of lithium-ion batteries. *Measurement* 174, 109057.

- [18] Gelfand, A.E., Smith, A.F., 1990. Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association* 85, 398–409.
- [19] Goebel, K., Saxena, A., Saha, S., Saha, B., Celaya, J., 2011. Prognostic performance metrics. *Machine learning and knowledge discovery for engineering systems health management* 147, 20.
- [20] Goodfellow, I., Bengio, Y., Courville, A., 2016. *Deep learning*. MIT press.
- [21] Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R., Schmidhuber, J., 2016. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems* 28, 2222–2232.
- [22] Gu, W., Sun, Z., Wei, X., Dai, H., 2014. A new method of accelerated life testing based on the grey system theory for a model-based lithium-ion battery life evaluation system. *Journal of Power Sources* 267, 366–379.
- [23] Hasan, M.G.M.M., Talbert, D.A., 2021. Counterfactual examples for data augmentation: A case study, in: *The International FLAIRS Conference Proceedings*.
- [24] Hong, C.W., Lee, C., Lee, K., Ko, M.S., Kim, D.E., Hur, K., 2020a. Remaining useful life prognosis for turbofan engine using explainable deep neural networks with dimensionality reduction. *Sensors* 20, 6626.
- [25] Hong, J., Lee, D., Jeong, E.R., Yi, Y., 2020b. Towards the swift prediction of the remaining useful life of lithium-ion batteries with end-to-end deep learning. *Applied energy* 278, 115646.
- [26] Hu, X., Jiang, J., Cao, D., Egardt, B., 2015. Battery health prognosis for electric vehicles using sample entropy and sparse bayesian predictive modeling. *IEEE Transactions on Industrial Electronics* 63, 2645–2656.
- [27] Jafari, M., Brown, L.E., Gauchia, L., 2018. A bayesian framework for ev battery capacity fade modeling, in: *2018 IEEE Transportation Electrification Conference and Expo (ITEC)*, IEEE, pp. 304–308.
- [28] Jiao, R., Peng, K., Dong, J., 2020. Remaining useful life prediction of lithium-ion batteries based on conditional variational autoencoders-particle filter. *IEEE Transactions on Instrumentation and Measurement* 69, 8831–8843.
- [29] Jordan, M.I., Mitchell, T.M., 2015. Machine learning: Trends, perspectives, and prospects. *Science* 349, 255–260.
- [30] Kacprzak, I., Glocker, B., Monteiro, M., Ribeiro, F.D.S., Xia, T., Kainz, B., 2023. Counterfactual data augmentation for deep learning predictive models.
- [31] Krishnan, R., Esposito, P., Subedar, M., 2022. Bayesian-torch: Bayesian neural network layers for uncertainty estimation. <https://github.com/Intellabs/bayesian-torch>. URL: <https://doi.org/10.5281/zenodo.5908307>, doi:10.5281/zenodo.5908307.
- [32] Lei, Y., Li, N., Guo, L., Li, N., Yan, T., Lin, J., 2018. Machinery health prognostics: A systematic review from data acquisition to rul prediction. *Mechanical systems and signal processing* 104, 799–834.
- [33] Li, L., Wang, P., Chao, K.H., Zhou, Y., Xie, Y., 2016. Remaining useful life prediction for lithium-ion batteries based on gaussian processes mixture. *PloS one* 11, e0163004.
- [34] 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.
- [35] Li, X., Zhang, L., Wang, Z., Dong, P., 2019. Remaining useful life prediction for lithium-ion batteries based on a hybrid model combining the long short-term memory and elman neural networks. *Journal of Energy Storage* 21, 510–518.
- [36] Libera, L.D., 2019. A comparative study between bayesian and frequentist neural networks for remaining useful life estimation in condition-based maintenance. *arXiv preprint arXiv:1911.06256*. arXiv:1911.06256.
- [37] Liu, D., Zhou, J., Pan, D., Peng, Y., Peng, X., 2015. Lithium-ion battery remaining useful life estimation with an optimized relevance vector machine algorithm with incremental learning. *Measurement* 63, 143–151.
- [38] 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.
- [39] Liu, Y., Zhao, G., Peng, X., 2019. Deep learning prognostics for lithium-ion battery based on ensembled long short-term memory networks. *IEEE Access* 7, 155130–155142.
- [40] Minka, T., 2000. Bayesian linear regression. Technical Report. Citeseer.
- [41] Mothilal, R.K., Sharma, A., Tan, C., 2020. Explaining machine learning classifiers through diverse counterfactual explanations, in: *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pp. 607–617.
- [42] Ng, S.S., Xing, Y., Tsui, K.L., 2014. A naive bayes model for robust remaining useful life prediction of lithium-ion battery. *Applied Energy* 118, 114–123.
- [43] Nguyen, V.D., Kefalas, M., Yang, K., Apostolidis, A., Olhofer, M., Limmer, S., Bäck, T., 2019. A review: Prognostics and health management in automotive and aerospace. *International Journal of Prognostics and Health Management* 10. doi:10.36001/ijphm.2019.v10i2.2730.
- [44] Nor, A.K.M., Pedapati, S.R., Muhammad, M., Leiva, V., 2021. Overview of explainable artificial intelligence for prognostic and health management of industrial assets based on preferred reporting items for systematic reviews and meta-analyses. *Sensors* 21, 8020.
- [45] Onchis, D.M., Gillich, G.R., 2021. Stable and explainable deep learning damage prediction for prismatic cantilever steel beam. *Computers in Industry* 125, 103359.
- [46] Park, K., Choi, Y., Choi, W.J., Ryu, H.Y., Kim, H., 2020. Lstm-based battery remaining useful life prediction with multi-channel charging profiles. *Ieee Access* 8, 20786–20798.
- [47] Patil, M.A., Tagade, P., Hariharan, K.S., Kolake, S.M., Song, T., Yeo, T., Doo, S., 2015. A novel multistage support vector machine based approach for li ion battery remaining useful life estimation. *Applied energy* 159, 285–297.
- [48] Pearl, J., 2011. *Bayesian networks*.
- [49] Ren, L., Zhao, L., Hong, S., Zhao, S., Wang, H., Zhang, L., 2018. Remaining Useful Life Prediction for Lithium-Ion Battery: A Deep Learning Approach. *IEEE Access* 6, 50587–50598. doi:10.1109/ACCESS.2018.2858856.
- [50] Rouhi Ardeshiri, R., Ma, C., 2021. Multivariate gated recurrent unit for battery remaining useful life prediction: A deep learning approach. *International Journal of Energy Research* 45, 16633–16648.
- [51] Saxena, A., Goebel, K., Simon, D., Eklund, N., 2008. Damage propagation modeling for aircraft engine run-to-failure simulation, in: *2008 International Conference on Prognostics and Health Management, PHM 2008*. doi:10.1109/PHM.2008.4711414.
- [52] Si, X.S., Wang, W., Hu, C.H., Zhou, D.H., 2011. Remaining useful life estimation - A review on the statistical data driven approaches. doi:10.1016/j.ejor.2010.11.018.
- [53] Sundar, S., Rajagopal, M.C., Zhao, H., Kuntumalla, G., Meng, Y., Chang, H.C., Shao, C., Ferreira, P., Miljkovic, N., Sinha, S., et al., 2020. Fouling modeling and prediction approach for heat exchangers using deep learning. *International Journal of Heat and Mass Transfer* 159, 120112.
- [54] Tang, S., Yu, C., Wang, X., Guo, X., Si, X., 2014. Remaining useful life prediction of lithium-ion batteries based on the wiener process with measurement error. *energies* 7, 520–547.
- [55] Tipping, M.E., 2001. Sparse bayesian learning and the relevance vector machine. *Journal of machine learning research* 1, 211–244.
- [56] V. Stone, J., 2014. Bayes' rule: a tutorial introduction to Bayesian analysis. *Choice Reviews Online* 51. doi:10.5860/choice.51-3301.
- [57] Verma, S., Boonsanong, V., Hoang, M., Hines, K.E., Dickerson, J.P., Shah, C., 2020. Counterfactual explanations and algorithmic recourses for machine learning: A review. *arXiv preprint arXiv:2010.10596*.
- [58] Wachter, S., Mittelstadt, B., Russell, C., 2017. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.* 31, 841.

- [59] Wang, D., Miao, Q., Pecht, M., 2013. Prognostics of lithium-ion batteries based on relevance vectors and a conditional three-parameter capacity degradation model. *Journal of Power Sources* 239, 253–264.
- [60] Wang, H., Li, J., Yang, F., 2014. Overview of support vector machine analysis and algorithm. *Application Research of Computers* 31, 1281–1286.
- [61] Wang, J., 2020. An intuitive tutorial to gaussian processes regression. arXiv preprint arXiv:2009.10862 .
- [62] Webb, G.I., Keogh, E., Miikkulainen, R., 2010. Naive bayes. *Encyclopedia of machine learning* 15, 713–714.
- [63] Wei, M., Gu, H., Ye, M., Wang, Q., Xu, X., Wu, C., 2021. Remaining useful life prediction of lithium-ion batteries based on monte carlo dropout and gated recurrent unit. *Energy Reports* 7, 2862–2871.
- [64] Wen, Q., Sun, L., Yang, F., Song, X., Gao, J., Wang, X., Xu, H., 2020. Time series data augmentation for deep learning: A survey. arXiv preprint arXiv:2002.12478 .
- [65] Wilson, A.G., Izmailov, P., 2020. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems* 33, 4697–4708.
- [66] Xiongzi, C., Jinsong, Y., Diyin, T., 2011. Remaining useful life prognostic estimation for aircraft subsystems or components: A review. *IEEE* doi:10.1109/ICEMI.2011.6037773.
- [67] Xu, C., Cleary, T., Wang, D., Li, G., Rahn, C., Wang, D., Rajamani, R., Fathy, H.K., 2021. Online state estimation for a physics-based lithium-sulfur battery model. *Journal of Power Sources* 489, 229495.
- [68] Zhao, Z., Bin Liang, Wang, X., Lu, W., 2017. Remaining useful life prediction of aircraft engine based on degradation pattern learning. *Reliability Engineering & System Safety* 164, 74–83. doi:10.1016/J.RESS.2017.02.007.
- [69] Zhou, D., Li, Z., Zhu, J., Zhang, H., Hou, L., 2020. State of health monitoring and remaining useful life prediction of lithium-ion batteries based on temporal convolutional network. *IEEE Access* 8, 53307–53320.
- [70] Zhou, D., Xue, L., Song, Y., Chen, J., 2017. On-line remaining useful life prediction of lithium-ion batteries based on the optimized gray model gm (1, 1). *batteries* 3, 21.
- [71] Zhou, Y., Huang, M., 2016. Lithium-ion batteries remaining useful life prediction based on a mixture of empirical mode decomposition and arima model. *Microelectronics Reliability* 65, 265–273.
- [72] Zhou, Z., Huang, Y., Lu, Y., Shi, Z., Zhu, L., Wu, J., Li, H., 2014. Lithium-ion battery remaining useful life prediction under grey theory framework, in: 2014 Prognostics and System Health Management Conference (PHM-2014 Hunan), IEEE. pp. 297–300.
- [73] Zraibi, B., Okar, C., Chaoui, H., Mansouri, M., 2021. Remaining useful life assessment for lithium-ion batteries using cnn-lstm-dnn hybrid method. *IEEE Transactions on Vehicular Technology* 70, 4252–4261.

II

Literature Study
previously graded under AE4020

1

Introduction

In today's dynamic world, ensuring the reliability and optimal performance of complex systems such as aircraft is of paramount importance. The discipline of Prognostics and Health Management (PHM) has emerged as a powerful engineering field aimed at minimizing maintenance costs through the effective utilization of assessment, prognosis, diagnosis, and health management systems. At the heart of PHM lies the concept of failure prognostics, which involves predicting the future behavior of systems, including their remaining useful life (RUL) [75]. Accurately estimating the RUL is a critical aspect of PHM, as it enables informed maintenance decision-making and resource allocation [110].

The goal of this study is to review the current literature surrounding PHM (mainly focused towards RUL estimation), Bayesian models and XAI. This review will be a first step for the subsequent MSc thesis at the Faculty of Aerospace Engineering at Delft University of Technology, which will aim to develop an explainable RUL estimation model using a Bayesian framework. The methodology of this thesis will be further discussed in chapter 5.

Chapter 2 delves into the significance of RUL estimation within the realm of PHM. Recognizing the RUL as an unknown random variable, this chapter explores how available data sources can be harnessed to estimate it. By transforming raw data into actionable information, PHM provides a valuable framework for optimizing maintenance strategies and minimizing costly downtime.

Chapter 3 explores an innovative approach to machine learning known as Bayesian methods. Unlike traditional methods that often produce deterministic outputs, Bayesian methods preserve uncertainty, allowing for a more nuanced representation of the relationship between input data and output predictions. By embracing uncertainty, Bayesian models provide richer insights into the accuracy and reliability of their results [21]. This chapter highlights the advantages of Bayesian methods over frequentist approaches and emphasizes their potential to enhance decision-making processes, particularly in safety-critical contexts.

The advent of Artificial Intelligence (AI) has revolutionized various industries and everyday life, empowering us to make reliable predictions and uncover intricate relationships within vast and complex data-sets [124]. However, the opaqueness of AI models, often referred to as the "black box" perception, has raised concerns regarding the interpretability and trustworthiness of their outputs. Chapter 4 explores the field of explainable AI (XAI), which seeks to shed light on the inner workings and reasoning behind AI models' predictions. By providing insights into the decision-making process, XAI techniques aim to make AI models more trustworthy, transparent, and accountable. In the context of PHM, where consequential decisions rely on predictive models, the integration of XAI not only enhances our understanding but also elevates the trustworthiness and utility of these models.

2

Prognostics and Health Management

Prognostics and Health management (PHM) is an engineering discipline with the goal of minimizing maintenance costs by the use of assessment, prognosis, diagnosis and health management systems. PHM is used for its capability to translate raw data into useful information which can be used in maintenance decision making. At the core of PHM lies failure prognostics, which involves predicting the future behaviour of a system including the system's remaining useful life (RUL) [75]. The RUL is an important aspect in PHM and will be discussed further in this review. In PHM, the RUL is seen as an unknown random variable which must be estimated using available data sources [101]. In this chapter, the general approaches to PHM will be discussed in section 2.1, followed by an overview of popular data sets used in prognostic model development in section 2.2. After this, current RUL estimation approaches from literature will be discussed in section 2.3, followed by an overview of performance indicators used to evaluate developed models in section 2.5. Lastly, the current challenges and shortcomings will be discussed in section 2.6

2.1. Approaches to PHM

In general, approaches to PHM can be categorized into four categories, reliability based, model based, data-driven and hybrid, see Figure 2.1. This section will expand on these four approaches using the overview provided by Xiao et al. [101], describing in general their approach to PHM, their use cases, their strengths and their shortcomings.

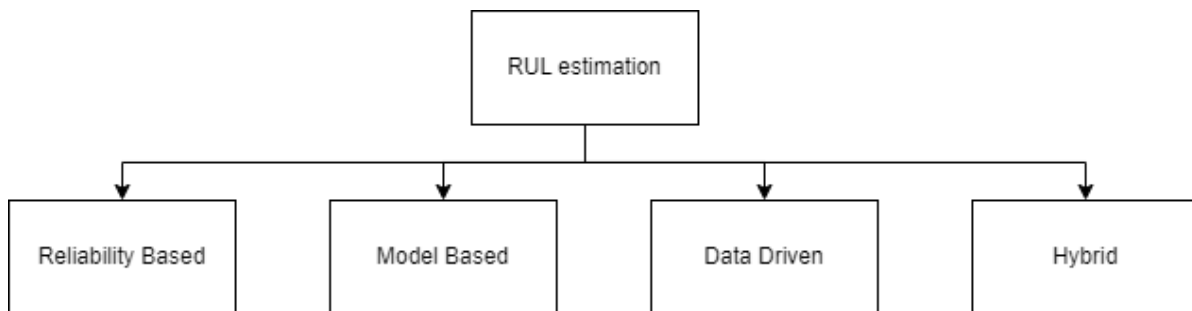


Figure 2.1: RUL estimation taxonomy

2.1.1. Reliability based

Reliability based approaches are considered the simplest approaches to PHM. They base their prognostics on the distribution of past event records of the systems they are trying to predict. In a practical sense, historical repair and failure data are used to predict the future failure of identical systems, actual degradation indicators are not taken into account. This makes reliability based approaches unfavorable, especially for critical systems. They are thus mainly used for non-critical, unmonitored, mass produced components.

2.1.2. Model based

Model based approaches, also known as physics based approaches, are a widely utilized approach in PHM due to its accuracy, precision and real-time performance [25]. This approach makes use of physical/mathematical models developed to represent a system that is to be predicted. This model will be used to predict system degradation and failures in real time. However, in order to establish a reliable model, a deep understanding of the physics of the system is required, and the model performance quickly decreases as the system complexity increases. An advantage of this approach however is relatively low (failure) data requirement compared to other approaches. The most common model-based methods are Kalman filters (KF), extended Kalman filters (EKF), unscented Kalman filters (UKF) and particle filters (PF).

Of these methods, the PF method is the most popular and considered to be state-of-the-art in the prognostics field [101]. It is used to estimate the state of a system that is changing over time based in (noisy and incomplete) measurements. The PF method is a Sequential Monte Carlo (SMC) technique which implements a recursive Bayesian filter to update the system state. The method works by representing the possible states of the system as a set of particles, which is initially randomly sampled from a probability density function (pdf). As more and more measurements are taken, the particles are updated by assigning weights according to their consistency with the actual measurement values. The particles are then re-sampled according to these assigned weights, after which the particles are propagated forward in time to represent the next time step (this is done using a model of the systems dynamics). These steps can be repeated to track the state of a system over time [46].

2.1.3. Data-driven

Data-driven approaches are, in contrast to model based approaches, easier to be developed and implemented in practical applications as it is often difficult to obtain a reliable physical model. The lower cost of algorithm development and the little knowledge required about the physics of the system to be analysed makes this the preferred approach for many prognostics system developers [126]. Data-driven methods provide a high reliability an effective computation, mainly using techniques in the field of Artificial Intelligence and Machine Learning (ML). However, data-driven methods do require large amounts of both historical and current observation data to perform optimally. In general, the more failure events in the data, the higher the estimation accuracy. Data-driven approaches in PHM are generally classified as either a statistical approach or a machine learning approach.

Statistical approaches use statistical parameters (mean, variance, etc.) to make predictions on probabilistic distributions (known or unknown). Some examples of statistical methods include maximum-likelihood estimation, hypothesis testing, Bayesian networks and hidden Markov model.

Machine learning approaches use acquired data to make predictions. They convert the data, e.g. health and failure data, into useful information by training a machine learning model. Some examples of machine learning methods include neural networks, decision tree, random forest and support vector machines.

2.1.4. Hybrid

The hybrid approach, also known as fusion approach, combines the model based and data-driven approaches. This approach aims to fuse the strengths of both approaches while minimizing their respective limitations in order to better estimate system health states and more accurately predict the RUL. This method aims to compensate for a lack of knowledge about the system's physics and lack of data.

2.2. Experimental data sets

In the field of PHM, it is important to have experimental data-sets in order to develop and test new methods and models. This section will go over some of the most used data sets in the PHM field. The data-sets evaluated are the C-MAPSS, N-CMAPSS, Li-ion Battery Aging and bearing data-set.

2.2.1. C-MAPSS

The Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) is a simulation tool created to simulate a large commercial turbofan engine [30]. This tool was used by NASA to create a challenge during the 2008 international conference on Prognostics and Health Management (PHM08), where four training/testing

data sets and two challenge data sets were provided and attendees were challenged to develop methods to achieve the best RUL predictions [96].

The data-sets are structured as followed, each data-set consists of multiple engines denoted by the engine number, each engine operates under 3 specified operating settings, followed by 21 sensors, every row in the data set denotes 1 operating cycle:

- Unit: Engine number [-]
- t: time [cycles]
- Op1: operational setting 1
- op2: operational setting 2
- op3: operational setting 3
- s1: Sensor measurement 1
- s2: Sensor measurement 2
-
- s21: sensor measurement 21

According to Frederick et al. [30], the C-MAPSS simulator can take 14 variable inputs, consisting of for instance fuel flow, fan efficiency, fan pressure ratio, etc., and then generates 26 outputs consisting of for instance fan speed, net thrust, various pressure and temperature readings, etc. Physical properties measures by the sensors is shown in Table 2.1 [97].

Table 2.1: Sensor measurement description

Sensor index	Description	Units	Sensor index	Description	Units
1	Total temperature at fan inlet	°R	12	Ratio fuel flow to Ps30	pps/ps
2	Total temperature at low-pressure compressor (LPC) outlet	°R	13	Corrected fan speed	rpm
3	Total temperature at high-pressure compressor (HPC) outlet	°R	14	Corrected core speed	rpm
4	Total temperature at LPC outlet	°R	15	Bypass ratio	-
5	Pressure at fan inlet	psia	16	Burner fuel-air ratio	-
6	Total pressure in bypass duct	psia	17	Bleed enthalpy	-
7	Total pressure at HPC outlet	psia	18	Demanded fan speed	rpm
8	Physical fan speed	rpm	19	Demanded corrected fan speed	rpm
9	Physical core speed	rpm	20	High-pressure turbine (HPT) coolant bleed	lbm/s
10	Engine pressure ratio	-	21	Low-pressure turbine (LPT) coolant bleed	lbm/s
11	Static pressure at HPC outlet	psia			

Each engine in the data-set can be seen as one engine in a fleet of identical engines. It is assumed that every engine is operating at normal capacity and starts to degrade at some point in the time series, however, it is unknown what the initial state of wear is of each engine. After a certain threshold of degradation is reached, the engine is deemed nonoperational and has thus reached its end of life. Additionally, the data is contaminated with a certain amount of noise. An overview of the different data sets can be seen in Table 2.2, where it can be seen that the main differences between them lie in the variation of the amount of operating conditions and the amount of fault modes.

Table 2.2: Summary of different data sets

Data Set: FD001 Train trajectories: 100 Test trajectories: 100 Conditions: ONE (Sea Level) Fault Modes: ONE (HPC Degradation)	Data Set: FD002 Train trajectories: 260 Test trajectories: 259 Conditions: SIX Fault Modes: ONE (HPC Degradation)
Data Set: FD003 Train trajectories: 100 Test trajectories: 100 Conditions: ONE (Sea Level) Fault Modes: TWO (HPC Degradation, Fan Degradation)	Data Set: FD004 Train trajectories: 248 Test trajectories: 249 Conditions: SIX Fault Modes: TWO (HPC Degradation, Fan Degradation)

In the training set, the RUL is calculated by looking at how many cycles are still remaining in the data-set per engine, as after the final cycle the engine is considered nonoperational. In the test sets however, the last cycle does not represent the end of life (EOL). The goal is to estimate the RUL after this cycle, which can be compared by a separate file containing the true RUL values for the test set. Aside from the training and test data sets provided in the C-MAPSS data set, there are also two challenge data sets which do not contain any RUL information. This set was used in the competition to rank the competitors. The score is calculated by NASA after uploading your results to their website.

2.2.2. N-CMAPSS

The N-CMAPSS data-set, provided by the NASA Prognostics Center of Excellence [14], is very similar to the C-MAPSS data set described in subsection 2.2.1, where a set of run-to-failure data of a small fleet of aircraft engines is provided. The N-CMAPSS data-set improves on this by running the engines under realistic flight conditions instead of a set sample of conditions. This can test RUL estimation models in a more dynamic and realistic environment.

2.2.3. Li-ion Battery Aging Data set

Similar to the C-MAPSS data set, NASA also has a data set representing a run to failure experiment of multiple Li-Ion batteries, which are run through three operational profiles (charge, discharge and Electrochemical Impedance Spectroscopy). The experiments were run until the batteries reached their EOL, which was a 30% fade in compared to their original capacity. The data provided gives information about aspects such as the environmental conditions, cycle number, cycle operation (charge, discharge or impedance), voltage, current, capacity, etc. The data set was created for the development of prognostics models, specifically those that can manage uncertainty due to the different state of life for every battery. The aim is to use this data to make reliable RUL estimations for both End of Discharge (EOD) and End of Life (EOL) [95].

2.2.4. Bearing data set

The bearing data-set, also provided by NASA [55], consists of data acquired during a series of run to failure tests of a set of 4 bearings. In the experiment, 4 Rexnord ZA-2115 double row bearings are installed on a shaft rotating at a constant 2000 RPM with a radial load of 6000 lbs applied to the shaft and bearings, see Figure 2.2. In the data, there are 3 data-sets, data-set 1 contains measurements of 2 accelerometers (x and y axes) per bearing, while data-sets 2 and 3 contain measurements of 1 accelerometers per bearing.

Each data-set consists of individual files containing 1 second snapshots recorded at specific time intervals (+- every 10 min), with each snapshot containing 20,480 data points with a sampling rate of 20 kHz. The experiment showed that all failures occurred after exceeding the designed life time of 100 million revolutions.

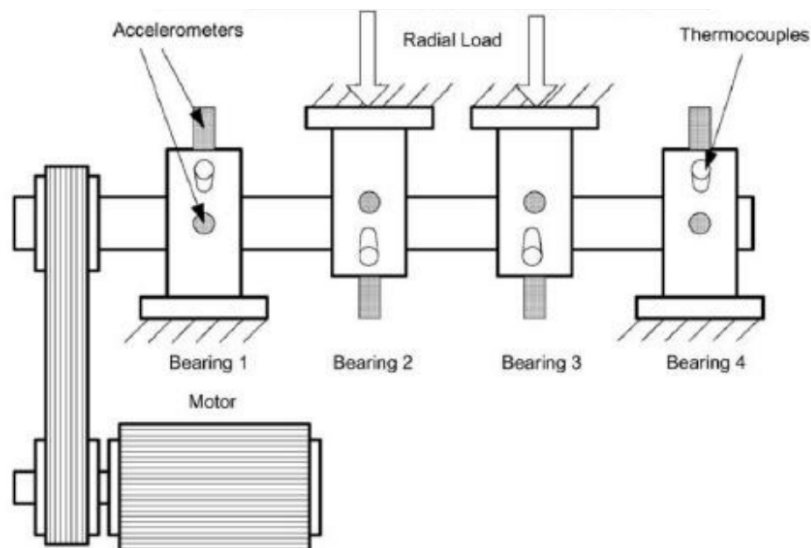


Figure 2.2: Experiment setup of bearing data-set [85]

2.3. RUL estimation methods in aviation

The aircraft industry requires a high degree of reliability in its aircraft in order to efficiently and safely operate. The industry generally achieves this high standard by implementing scheduled maintenance to all its aircraft and their components. Scheduling maintenance in advance can be seen as a pro-active / preventive form of maintenance, which aims to prevent any system/component from failing while in use. This does however often result in unnecessary maintenance when a component is still perfectly fine. Unscheduled maintenance is also generally undesirable, as this means a system/component is suddenly in need of maintenance or replacement, potentially grounding the aircraft, but could even be more detrimental if a critical system fails during operation. In both cases, the aircraft has a risk of experiencing unnecessary down-time, which is very costly [52]. This is where predictive maintenance comes in, where the use of time and components is optimized by using predictions on the RUL of systems/components to schedule maintenance rather than a standard interval [75].

According to Figure 2.3, the health of a system can be classified to be in one of four regimes, healthy, caution, repair and failure. Looking at Figure 2.3, the actual RUL is defined as a random variable with a pdf. The pdf of the RUL is dependent on factors such as age, operating environment and the observed condition monitoring. It is therefore required to have sensors monitoring various aspects of the systems of interest in real time to produce a reliable RUL estimation [122].

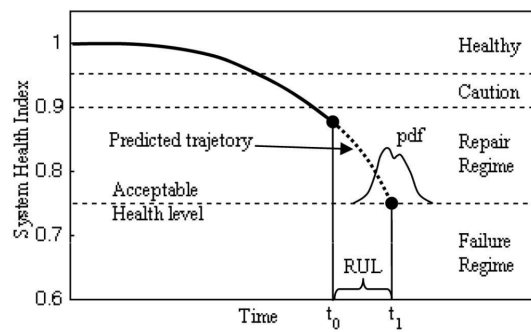


Figure 2.3: System health over four regimes (note that the System Health Index ranges from 0-1 and transition values are taken arbitrarily) [122]

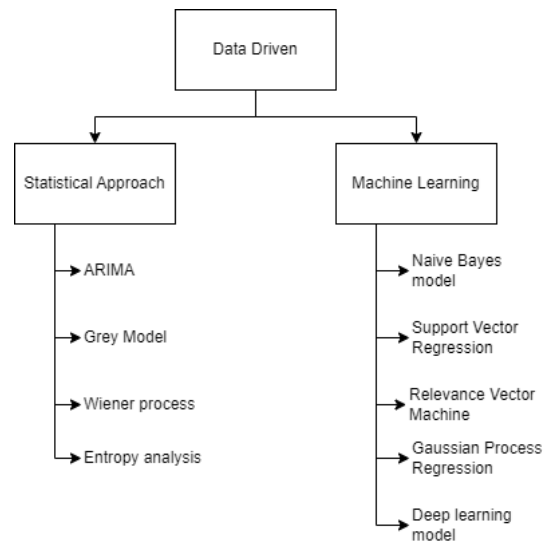


Figure 2.4: Data driven methods overview

The remaining part of this section will review various approaches to RUL estimation found in literature. The approaches will mainly focus on data-driven methods, as they only rely on past observed data and statistical models, making them generally more simple to implement than model driven approaches provided there is enough historical data to work with [65, 101]. An overview of the methods discussed can be seen in Figure 2.4.

2.3.1. Machine Learning approaches in RUL estimation

A machine learning model maps the inputs of a system to an output. Due to the nature of various machine learning techniques, various complex non-linear relationships can be extracted and utilized [34]. In RUL estimation, commonly used ML techniques are for example Naive Bayes (NB), Support Vector Regression (SVR), Relevance Vector Machines (RVM), Gaussian Process Regression (GPR) and Deep Learning (DL)[5]. There are also other machine learning techniques such as Random Forest Regression [16] and k-nearest neighbours [64] that are used in RUL estimation, but will not be discussed further here.

Naive Bayes is considered to be the simplest form of Bayesian modeling, using Bayes theorem as a discriminant function, where the term naive comes from the fact that all input variables are considered independent from each other [74]. It works by first taking a prior output distribution based on the output distribution in the data. It updates this prior distribution to a posterior distribution using Bayes Theorem (described in section 3.1) [121]. In literature, this method has been applied to RUL estimation. Ng, et al. [74] used the NB model to predict the RUL for Li-ion batteries, while Jafari et al. [43] used an enhanced NB model to predict capacity fade degradation in an EV battery. Lastly, Galal et al. [33] used a supervised NB classifier to detect faults in satellite batteries. All papers report satisfactory results, showing the capabilities of the NB model. However, all papers also indicate that the NB model uses significantly simplifying assumptions, and that accuracy can significantly be increased if more complex, dependent, data can be used.

Support Vector Regression is based on the concept of Support Vector Machine (SVM), which is mainly used for binary classification. It works by finding a best fit for a hyperplane that reduces its cost function the most, but still maximizes the margin between the decision boundary and the data points. By using the Kernel Trick, which augments the original data into a higher dimension (without actually transforming the data), it can also handle non-linear regression [125]. In literature, SVR based methods are used to develop prognostic models for for instance Li-ion batteries. Wang et al. [114] uses the energy efficiency and battery working temperature as critical health indicators (HI), and trains an SVR based model to capture the capacity degradation curve. Patil et al. [82] uses the voltage and temperature as the critical features, and uses an SVR based method to predict the RUL. One of the advantages of SVR is its ability to handle minor sample, non-linear, high dimensional data [114]. However, one of the disadvantages of this method is the high computational cost, its dependence on suitable hyperparameter selection, and the fact that its predictions are not probabilistic [5, 106].

Relevance Vector Machine has the same basic form as an SVM, but uses a Bayesian framework. RVMs were introduced by Tipping [106] as an improvement of SVM. It adopts a fully probabilistic framework where a prior distribution is introduced over the model weights and is updated iteratively, where the most probable values are taken for each weight. Due to this, it is often found that the posterior distribution of weights sees many weights go to zero, thus only keeping the relevant, non-zero, weights. As these weights are associated with kernel functions, the reduction of these kernel functions also decreases the computational complexity. Wang et al. [112] utilized this increase in computational efficiency for the RUL estimation of a battery. Liu et al. [61] also used the RVM for its increase in computational accuracy, but did not yet dive into the uncertainty reduction of the RUL estimations. RVM generally reduces the computational cost when compared to SVM, it however still remains generally complex when the data set becomes significantly large. Also, due to many weights going to zero, there is a risk of losing relevant support vectors in the process [5]. It does however remain a very flexible machine learning algorithm with the capability of preserving uncertainty.

Gaussian Process Regression works by taking a prior distribution of functions that could describe the input data. These functions are dependent on the chosen kernel function and the selection of these functions is a Gaussian process. After data is added, the functions are updated using Bayes method, giving a posterior distribution of functions that fit the data [87]. For RUL estimation, Li et al. [58], Liu and Chen [62], and Li et al. [62] all introduce GPR based RUL estimation approaches for Li-ion batteries. Although results were satisfactory, the approaches did often struggle with the computational complexity when working with large data sets, and appropriate parameter selection is advised [5].

Deep Learning is based on a neural network (NN) framework, where artificial neural networks (ANN) consist of a single input, hidden and output layer, deep neural networks (DNN) consist of multiple hidden layers, allowing them to extract more complex features and relationships [10]. Variations of DNN architectures have been widely used in RUL estimation. A widely adapted approach uses the Long Short Term Memory (LSTM) framework, which itself is a variation of a Recurrent Neural Network (RNN). Due to its high performance when working with time series data, it is often used in RUL estimation [17, 18, 60, 63, 81]. Other DNN approaches use the Gated Recurrent Unit (GRU) [93, 118], Convolutional Neural Networks (CNN) [41, 128, 132] and autoencoders [45, 88]. From the literature mentioned before, it was found that the performance of the DNN approaches varied, the approaches using a general DNN, GRU and autoencoder suffered from a insufficient prediction accuracy, while approaches such as the CNN suffered from a high model complexity. The LSTM approaches appeared to be the favored approach for this RUL estimation case [5]. It should be noted that DNN models do require a large amount of data to be properly trained and are prone to overfitting [12], however there are techniques to reduce this overfitting tendency such as the dropout technique, as applied by Khumprom and Yodo [50].

2.3.2. Statistical approaches in RUL estimation

Another data-driven approach in RUL estimation is the statistical approach, where empirical knowledge and available data is used to build a statistical model for RUL estimation. Statistical models are considered simple, accurate and easy to implement [5]. Some statistical methods used in RUL estimation are for instance the Autoregressive Integrated Moving Average (ARIMA) technique (used by Li et al. [59], Chen et al. [15], and Zhou Y. et al. [130]), the Grey Model (GM) (used by Zhou Z. et al. [131], Gu et al. [37] and Zhou D. et al. [129]), the Wiener Process (WP) (used by Lei et al. [56], Tang et al. [105], Feng et al. [27] and Xu et al. [123]), and Entropy analysis (used by Hu et al. [42]).

While statistical methods have shown to be competent RUL estimators, there are some drawbacks. One of the main drawbacks of statistical methods is its reliance on simplifying assumptions regarding the degradation process, while machine learning methods (such as GPR and SVR) do not require such assumptions and can thus capture more complex, non-linear relationships in the input data [116]. Therefore, machine learning approaches are mainly considered from this point.

2.4. Deterministic vs. Probabilistic methods

When considering a model, it can either be deterministic or probabilistic. In a deterministic model, deterministic values are used to compute an answer that is simply given in the form of a value, classification, etc., where the exact answer never changes if the same input is given. Probabilistic methods on the other hand

also consider the uncertainties related to a lack of knowledge (epistemic uncertainty) and of the inherent randomness of a system (aleatory uncertainty) [22]. The addition of considering uncertainty allows for a model to show how (un)sure it is of its prediction, making it significantly more useful for when decisions need to be made based on the model output compared to a model with a deterministic output, which tend to be over confident of its results [11].

Considering the RUL estimation methods discussed in section 2.3, there are already some models which make use of the uncertainty information, namely the NB, RVM and GPR models. These models all use a form of Bayesian modeling, which will be further discussed in chapter 3. In addition to these models that inherently use Bayesian methods to conserve uncertainty, there are also approaches that make use of DNN architectures, which are converted to Bayesian models using specific training techniques [66]. For RUL estimation, this is for instance done by Caceres et al. [11], where a Frequentist (a.k.a. deterministic) RNN is converted to a Bayesian RNN (BRNN) by training using Bayes-by-backprop with the Flipout method. This resulted in a network capable of outputting a Gaussian mean and variance instead of a single value for the RUL estimation. The network architecture is visualized by Figure 2.5. The resulting conclusion of this paper was that; BRNN models were capable of predicting the correct RUL value withing a 90% confidence interval for nearly all predictions, uncertainty in the predictions could be observed to increase with increasing data complexity, and that the Bayesian models performed comparably or even better than their Frequentist counterparts.

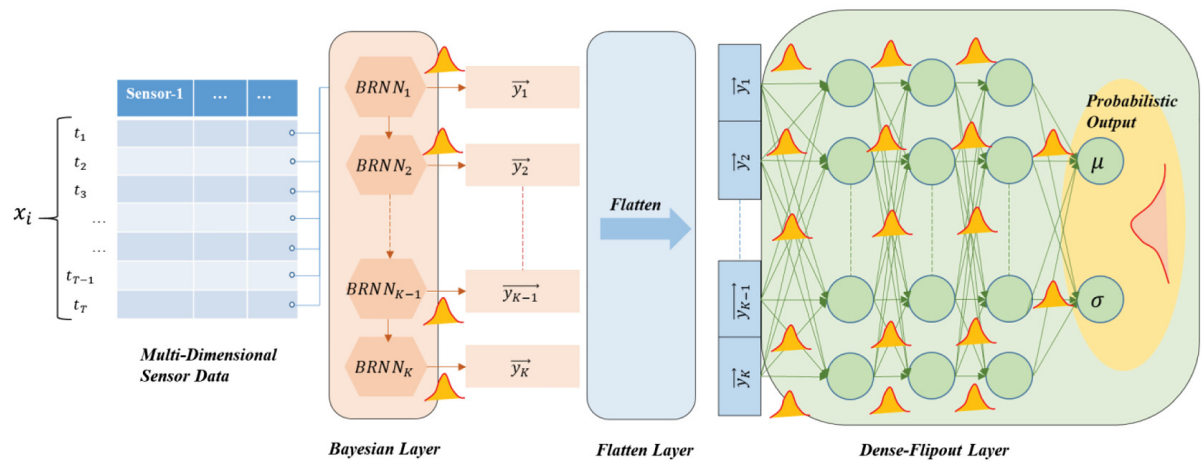


Figure 2.5: Visual representation of BRNN network used in [11]

Similarly, Benker et al. [6] converts a DNN and CNN to their Bayesian counterparts using Hamiltonian Monte Carlo (HMC) and Variational Inference (VI). Comparing the Bayesian and non-Bayesian DNN & CNN, their performance is again comparable, with the Bayesian models sometimes even outperforming the non-Bayesian models. This shows that, even without explicitly using the uncertainty information provided by the Bayesian models, they still are able to provide a more accurate RUL prediction (likely due to their natural regularization). Benker et al. [6] also propose a novel method of utilizing the uncertainty information from the RUL prediction.

Assuming a true RUL y^* and predicted RUL y , then we can define the deviation as $\tau = y^* - y$. Assuming τ follows a Gaussian distribution, then we have $p(\tau|\mu_\tau, \sigma_\tau) = \mathcal{N}(\tau|0, \sigma_\tau)$. In this case, the expected value of y is now equal to the true RUL y^* . This can be re-parametrized by implementing $\mu_\tau = b\sigma_\tau$, which shift the mean away from 0.

The authors also implement a cost function, given by Equation 2.1 which penalizes unexpected failures ($\tau \geq 0$) more than unnecessarily early maintenance ($\tau < 0$). The total cost c_{total} can then be calculated by multiplying the cost function with the distribution $p(\tau|b\sigma_\tau, \sigma_\tau)$. The goal is then to find a b , such that c_{total} is minimized. This process can be seen in Figure 2.6 where it is clear that shifting the distribution by $b\sigma_\tau$ can reduce the total cost.

$$s(\tau) = \begin{cases} s_1(\tau) = e^{-\frac{\tau}{13}} - 1, & \text{for } \tau < 0 \\ s_2(\tau) = e^{\frac{\tau}{10}} - 1, & \text{for } \tau \geq 0 \end{cases} \quad (2.1)$$

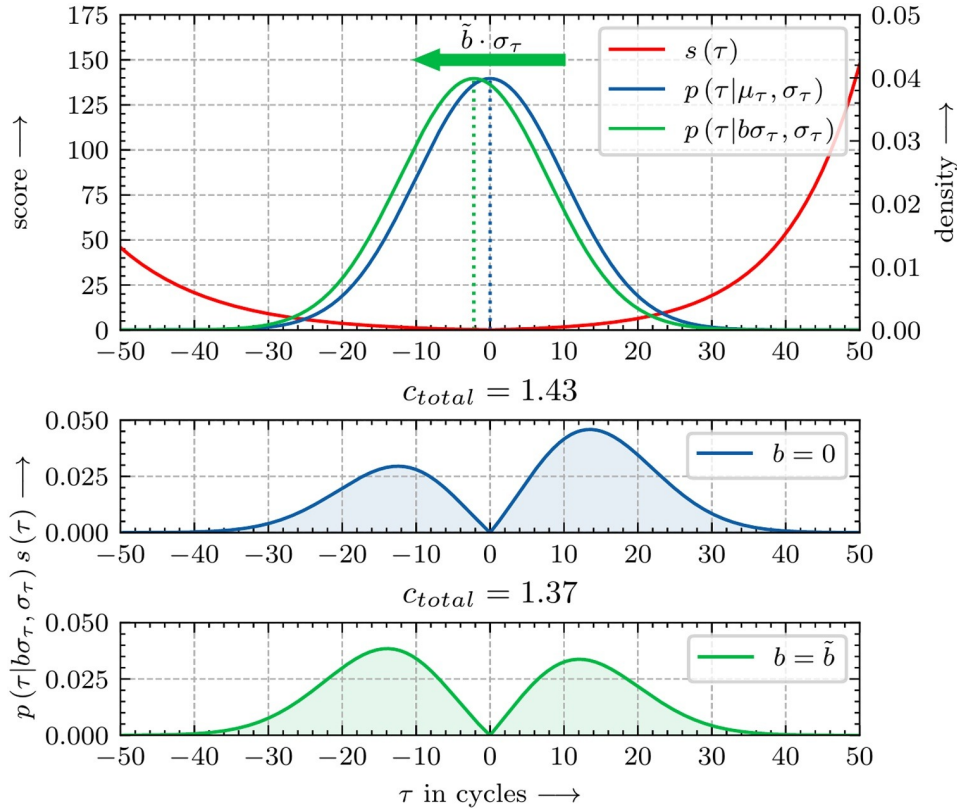


Figure 2.6: Use of uncertainty in RUL estimation. Top plot shows error distribution being shifted w.r.t. the scoring function. The bottom plots show the associated total cost. [6]

All in all, when comparing probabilistic (Bayesian) methods with deterministic methods, Wilson et al. [120] summarizes the main compelling argument for probabilistic methods:

“The key distinguishing property of a Bayesian approach is marginalization instead of optimization, where we represent solutions given by all settings of parameters weighted by their posterior probabilities, rather than bet everything on a single setting of parameters.”

2.5. Performance indicators

When developing a model, it is important to know how well your model performs and compares to other models. The use of a performance indicator can help represent the performance of your model, however there are multiple options to choose from which look at different aspects of a model. This section will cover a few of the industry standard performance indicators and some more novel approaches, more specific to PHM. An overview of these performance indicators is provided by Saxena et al. [98], who performed a survey on current evaluation metrics used in the field of prognostics. Some general units used in this section are:

- UUT: Unit Under Test
- $\Delta(i)^l$: The error between predicted and true RUL at time step i for UUT l .
- EOP: End of Prediction, indicating the models EOL prediction.
- EOL: The true End of Life.
- P: Time index when the first prediction was made.

- $r(i)^l$: The predicted RUL at time step i and UUT l given that data is available up to time step i .
- $r_*(i)^l$: The true RUL at time step i and UUT l given that data is available up to time step i .
- ℓ : Set of all time points where predictions are made.

2.5.1. Average Bias

The average bias metric is a method used to determine the accuracy of a prediction. It is defined by Equation 2.2, which averages the prediction error for all ℓ predictions. However, this metric does cancel out positive and negative errors and does not account for the influence of outliers.

$$B_\ell = \frac{1}{\ell} \sum_{i=1}^{\ell} \Delta(i) \quad (2.2)$$

2.5.2. Sample Standard Deviation

The Sample Standard Deviation (SSD) is a metric to evaluate the precision of a model. It is defined by Equation 2.3, which measures the variability with respect to the mean. $\Delta(i)$ represents the prediction error and m represents the mean of the sample set of errors. It should be noted that the SSD only applies to Gaussian distributions.

$$SSD = \sqrt{\frac{\sum_{i=1}^{\ell} (\Delta(i) - m)^2}{\ell - 1}} \quad (2.3)$$

2.5.3. (Root) Mean Squared Error

The Mean Squared Error (MSE) is a metric can indicate both the precision and accuracy of a model, it is defined by Equation 2.4. As it is not unitless, it is difficult to directly compare the MSE of two different models. Due to its square, it takes both positive and negative errors into account and penalizes larger errors significantly more than smaller errors, however it is sensitive to non-normal data and to the presence of outliers.

$$MSE = \frac{1}{\ell} \sum_{i=1}^{\ell} \Delta(i)^2 \quad (2.4)$$

More frequently used in the field of statistics and machine learning is a variation of the MSE, the Root Mean Squared Error (RMSE), see Equation 2.5. It simply takes the root of the MSE, but this does ensure that the units of the RMSE coincide with that of the original data.

$$RMSE = \sqrt{\frac{1}{\ell} \sum_{i=1}^{\ell} \Delta(i)^2} \quad (2.5)$$

2.5.4. Mean Absolute Percentage Error

The Mean Absolute Percentage Error (MAPE) takes into account not only the prediction error, but also when this prediction was made. In the case of the MAPE, errors made closer to EOL are weighted more heavily than those further away. The MAPE is defined by Equation 2.6 which returns a unitless percentage value, making it ideal for comparing multiple different models. However, the MAPE does have its shortcomings, it is only relevant for ratio-scaled data which has a meaningful zero, and prediction errors that exceed the EOL run the risk of being severely penalized compared to errors that are less than the EOL.

$$MAPE = \frac{1}{\ell} \sum_{i=1}^{\ell} \left| \frac{100\Delta(i)}{r_*(i)} \right| \quad (2.6)$$

2.5.5. Prognostic Horizon

The Prognostic Horizon (PH) is an indicator on how far a model can look ahead and still provide a reliable prediction. It is given by Equation 2.7 where it looks at the difference between the End of Prediction (EOP), which indicates the predicted EOL, and the current time step i , bound by an allowable error bound α .

$$PH = EOP - i \quad \text{with} \quad i = \min \left\{ j \mid (j \in \ell) \wedge \left(r_*(1 - \alpha) \leq r^l(j) \leq r_*(1 + \alpha) \right) \right\} \quad (2.7)$$

For example, if $\alpha = 5\%$, then the PH will indicate how far ahead the model can predict the EOL of a system within 5% of the actual EOL.

2.5.6. $\alpha - \lambda$ performance

The $\alpha - \lambda$ metric can be used to evaluate if a model adheres to specified performance levels. The λ is the time horizon indicating the percentage of time between the first prediction P and the EOL. The α is used to test the accuracy of a model, it can be seen as the margin of error in which the RUL predictions lie with respect to the true RUL. Looking at Equation 2.8, it can be seen how the predicted RUL $r^l(t_\lambda)$ is evaluated at time step λ , where $t_\lambda = i_P + \lambda(i_{EOP} - i_P)$, within the α bounds of the true RUL $r_*(t)$. An example can be seen in Figure 2.7.

$$[1 - \alpha] \cdot r_*(t) \leq r^l(t_\lambda) \leq [1 + \alpha] \cdot r_*(t) \quad (2.8)$$

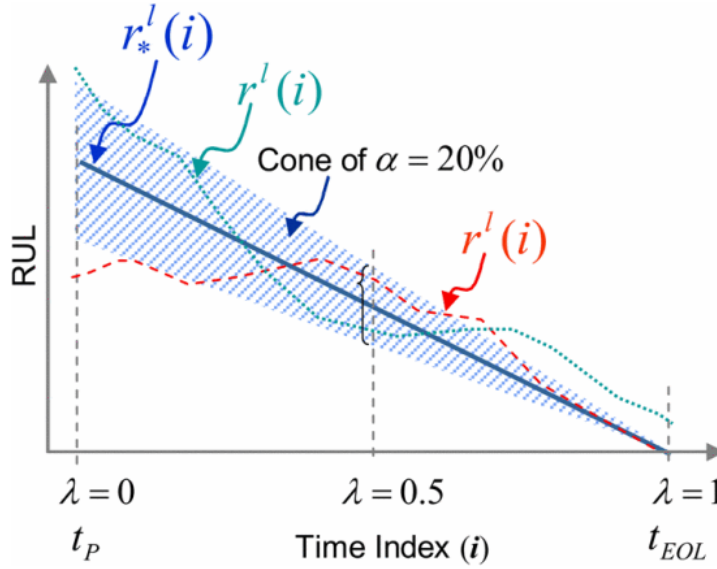


Figure 2.7: $\alpha - \lambda$ performance evaluation using $\alpha = 0.20$ and $\lambda = 0.5$ [98]

2.5.7. Relative Accuracy

The Relative Accuracy (RA) is comparable to the $\alpha - \lambda$ performance, with the key difference being that rather than evaluating if the predictions fall within a certain accuracy range, the accuracy range itself is determined. The higher the RA, the more accurate the prediction. The RA is calculated by using Equation 2.9, where $t_\lambda = i_P + \lambda(i_{EOP} - i_P)$.

$$RA_\lambda = 1 - \frac{|r_*(t_\lambda) - r^l(t_\lambda)|}{r_*(t_\lambda)} \quad (2.9)$$

2.5.8. Cumulative Relative Accuracy

The RA is evaluated at specified time points λ . If we want to have a metric that can take into account the RAs at different time points, we can aggregate the RAs into the Cumulative Relative Accuracy (CRA). This is done according to Equation 2.10, where the weights w are assigned such that higher weights are given to predictions closer to the EOL.

$$CRA_\lambda = \frac{1}{\ell} \sum_{i=1}^{\ell} w(r^l) RA_\lambda \quad (2.10)$$

2.5.9. Convergence

The convergence metric aims to indicate how the accuracy and/or precision of a model improves over time. The convergence metric is given by Equation 2.11, where (x_c, y_c) indicates the centroid of the area under the curve. The lower the distance indicated by C_M , the faster the convergence. An example is given in Figure 2.8,

where 3 cases can be seen with different convergence metrics. The metric $M(i)$ in this case represents the performance of a system and in this case should be minimized.

$$\begin{aligned}
 C_M &= \sqrt{(x_c - t_p)^2 + y_c^2} \\
 x_c &= \frac{\frac{1}{2} \sum_{i=P}^{EOP} (t_{i+1}^2 - t_i^2) M(i)}{\sum_{i=P}^{EOP} (t_{i+1} - t_i) M(i)} \\
 y_c &= \frac{\frac{1}{2} \sum_{i=P}^{EOP} (t_{i+1} - t_i) M(i)^2}{\sum_{i=P}^{EOP} (t_{i+1} - t_i) M(i)}
 \end{aligned} \tag{2.11}$$

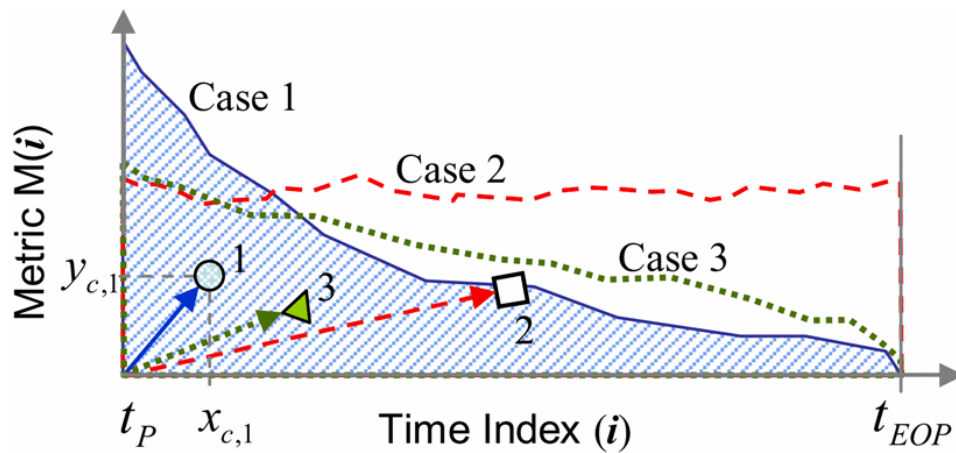


Figure 2.8: Example of convergence of 3 cases that converge at different rates [98]

2.6. Challenges, shortcomings and future work

There have been many innovations in the field of PHM, and specifically in RUL prediction. From literature reviewed in section 2.3, it can be seen that a variety of RUL estimation approaches have been developed and tested, confirming their ability to predict the RUL of a system with a certain amount of reliability. The scope of this literature review has focused mainly on machine learning approaches to RUL estimation, where both deterministic and probabilistic methods were evaluated. From section 2.4, the probabilistic methods prove to provide more useful information regarding the uncertainty of a prediction with respect to the deterministic methods, while also outperforming them in some cases. From this it is clear that in future work, probabilistic methods should be considered for a more widespread implementation in RUL estimation. So, from now on only probabilistic (Bayesian) methods will be considered for implementation.

Also, in section 2.2, the commonly used data-sets were discussed that are used for PHM model validation. Of these data-sets, the C-MAPSS data-set has been chosen to work with for the continuation of this research. The C-MAPSS data-set is chosen for its large yet manageable size, its multiple operating conditions, its noisy data and due to its wide application in the PHM sector [86].

However, aside from the fact that the added uncertainty information provided by the Bayesian models can be useful in the decision making process, no information is given as to how the models come up with their RUL estimation. This 'black box' behaviour is often seen as a drawback to the field of machine learning, where information goes in and an answer comes out but it is unclear why the model gave the answer [94]. In the field of PHM and RUL estimation, decisions are expected to be made based on the predictions generated by the used models, which have a significant influence on the performance of critical systems and thus also on the safety of passengers. To make the models more trustworthy, methods need to be applied to extract

why the model gives a certain prediction and if this is logical. This is where the field of explainable AI (XAI) comes in which will be discussed more in depth in chapter 4.

3

Bayesian methods

Bayesian methods differentiate themselves from other machine learning methods due to their ability to conserve uncertainty. When input data needs to be converted to an output by the use of machine learning, traditional methods will simply give an output, e.g. True/False when classifying, or a discrete value when using regression. Bayesian methods allow for the uncertainty present in the input to be shown in the output, transforming the output to e.g. 80% True and 20% False for classification, or a distribution of possible values for regression. This allows for more insight into the accuracy of the model's results [21]. Especially when compared to Frequentist approaches, which tend to be overconfident in their results due to their deterministic expressions. This overconfidence can potentially lead to unreliable decisions being made by the people using this model, which can have detrimental consequences when regarding safety critical equipment. In this chapter, Bayes' rule will be introduced in Equation 3.1 followed by an overview of various bayesian models in section 3.2. After this, the methods will be compared in section 3.3 followed by an overview of the application of these methods in section 3.4.

3.1. Bayes' Rule

Bayes' rule is widely used in fields ranging from statistics, engineering, and machine learning, all the way to medicine or even basic decision making. The rule is based on updating *prior* beliefs based on evidence in order to get the *posterior* belief. The general form of Bayes rule is given by Equation 3.1, where H is your hypothesis and E is the evidence [107].

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)} \quad (3.1)$$

To give an example, say you test positive for a rare disease which occurs in 0.1% (so 0.001) of the population. Now, the accuracy of the test is such that the test correctly identifies 99% (so 0.99) of the people who have the disease. This could give you the idea that it is 99% certain that you have the disease. However, let's apply Bayes' rule to this situation. Looking term for term, $P(E|H)$ is the probability that the evidence is true given that the hypothesis is true, in this case $P(\text{positive test}|\text{disease}) = 0.99$. $P(E)$ is the total probability that the evidence is true, regardless if the hypothesis is true or not, so expanding it gives:

$$\begin{aligned} P(E) &= P(E|H)P(H) + P(E|\neg H)P(\neg H) \\ &= P(\text{positive test}|\text{disease})P(\text{disease}) + P(\text{positive test}|\text{no disease})P(\text{no disease}) \\ &= 0.99 \cdot 0.001 + (1 - 0.99) \cdot (1 - 0.001) = 0.01098 \end{aligned}$$

Now, generally the hardest value to give is that for $P(H)$, a.k.a. the *prior*. In this case, assuming we do not know the test results, we guess that $P(H) = P(\text{disease}) = 0.001$, which is a realistic guess given we do not know the test results and that is the overall probability you have the disease.

Filling everything into Equation 3.1, we get $P(H|E) = P(\text{disease}|\text{positive test}) = \frac{0.99 \cdot 0.001}{0.01098} \approx 0.09 = 9\%$ chance of having the disease given that you test positive, this is now our posterior belief. Now, 9% is significantly higher than our prior belief of 0.1%, but also significantly lower than our fear of 99%. This goes to

show that initial beliefs can be significantly updated by evidence using Bayes' rule.

When applying Bayes rule in machine learning, as shown by Bishop et al. [7], we can rewrite Equation 3.1 into a form more commonly used in the machine learning community, see Equation 3.2. Here, we assume to have a model with parameters w and a data-set D , where the goal is to find the posterior distribution of model parameters $P(w|D)$ that best fits the data-set D . This is done by first taking a prior distribution $P(w)$, representing our initial belief of what the likely range is of model parameters, and multiplying this by the likelihood $P(D|w)$, representing how likely it is that we see the data given the model parameters. Finally, to achieve a valid posterior distribution, we need to normalize over all possible parameter values, which is done by marginalizing over all possible parameter settings using the marginal likelihood parameter $P(D)$, which can be expanded to Equation 3.3.

$$P(w|D) = \frac{P(D|w)P(w)}{P(D)} \quad (3.2)$$

$$P(D) = \int P(D|w)P(w)dw \quad (3.3)$$

The learning process of finding the posterior distribution of parameters w is called Bayesian inference, which contrasts the Frequentist approach to finding a single optimal set of parameters that best fits the data (so maximizing $P(D|w)$ a.k.a. Maximum Likelihood Estimation (MLE)). Instead of optimization, the main focus of the Bayesian approach is the marginalization over the entire parameter space, which is often intractable. For this, methods such as Markov Chain Monte Carlo (MCMC), which samples using a large finite set of parameters, or variational inference (VI), which uses known distributions to approximate the posterior, are applied to achieve a practically feasible approximation to the posterior distribution.

3.2. Bayesian models in machine learning

There are many types of Bayesian models used for machine learning purposes. In this section, a selection of these models will be discussed with respect to their (potential) RUL estimation capabilities. The selected models are based on the Bayesian RUL estimation approaches discussed in section 2.3, with the addition of other relevant models found in literature relating to RUL estimation [76, 80]. An overview of the Bayesian models is given in Figure 3.1.

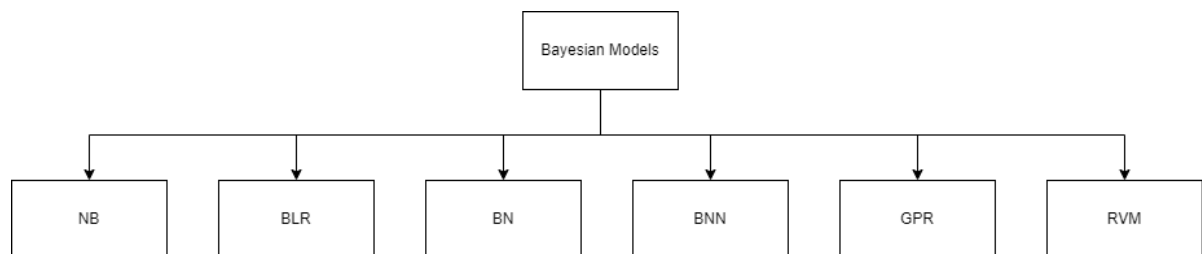


Figure 3.1: Different types of Bayesian machine learning models

3.2.1. Naive Bayes

The Naive Bayes (NB) algorithm is mainly used for classification purposes, where it uses Bayes theorem to calculate a posterior distribution of each class given the input features. It is considered *naive* as it assumes that all input features are independent of each other, which is a strong assumption that is often not true, but does seem to work well in practice nonetheless. NB is considered a simple, robust and computationally efficient Bayesian approach that performs competitively with other classifiers and can readily be applied to large data sets. [117, 121]. Variants of NB include Gaussian NB [79], Multivariate NB [26] and Bernoulli NB [102].

NB is a classification algorithm of itself, often applied in for instance spam filtering of emails [102]. When looking at applying it to regression tasks, it has been shown to perform considerably less than its classification counterpart. The cause of this lies in its Independence assumption, making it more restrictive for regression than for classification [29].

All in all, NB is a widely applied method due to its simplicity, robustness and computational efficiency. It has already been applied to RUL estimation in the past [33, 43, 74] with success. However, the main takeaway was that even though NB provided acceptable results, its simplifying assumptions do tend to restrict its predictive capabilities.

3.2.2. Bayesian Linear Regression

According to [69], in Bayesian Linear Regression (BLR), the relationship of the parameters in the model is assumed to be linear, but mapped using the Bayesian method. Assuming a relationship of input x to output y in the form of $y(x, \mathbf{w}) = w_0 + w_1 x$, in BLR the weights w_0 and w_1 are estimated using Bayesian inference. Assuming a Gaussian distribution of the prior, we get $p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$, where α represents the precision of the prior distribution. Using a mean of 0 ensures that smaller values of \mathbf{w} are favored. Now, if we also assume that the noise of the input to also follow a Gaussian distribution, we can also assume the posterior will follow a Gaussian distribution, given by $p(\mathbf{w}|\mathbf{t}, \alpha, \beta) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$, where \mathbf{t} are the observations of the input data, β is the precision of the input data and \mathbf{m}_N and \mathbf{S}_N represent the mean and covariance matrix of the posterior respectively. By sampling from the posterior, a set of linear models can be found fitting the data. By sampling enough, a distribution can be formed using the mean and variance of the samples.

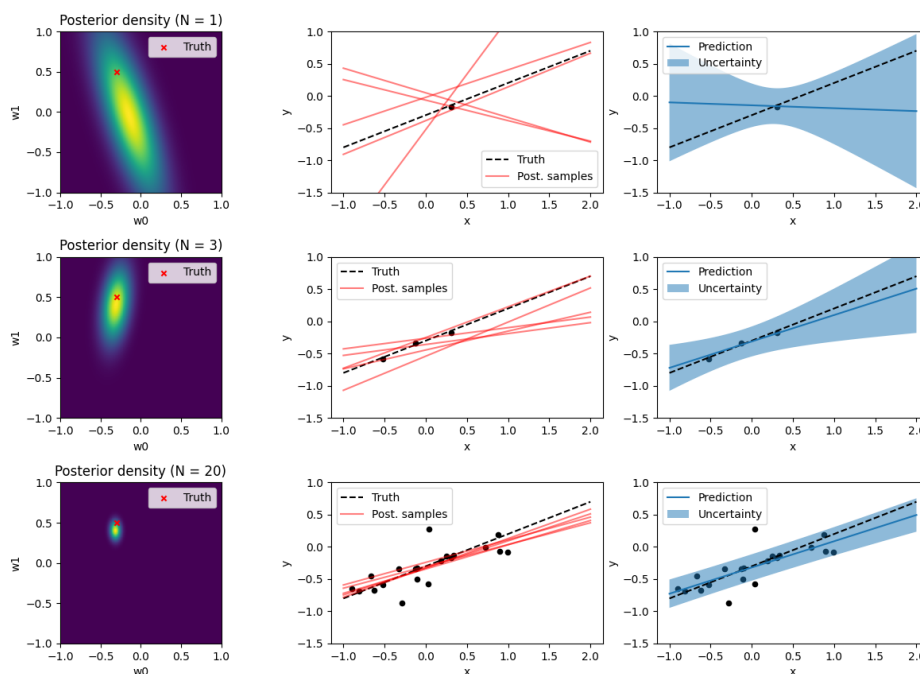


Figure 3.2: BLR example with 1, 3 and 20 data points (top, middle, bottom) based on example provided by [53]

An example provided by [53] can be seen in Figure 3.2, where 1, 3 and 20 data points are taken from a data set (which we know is a linear relationship with some noise). The left side shows the posterior density of w_0 and w_1 , which gets more sure when there are more data points available. The middle graphs show 5 random samples of the posterior while the right shows the entire distribution of the posterior over the data points. It can be seen that more data points make the regression model more sure of the relationship, but the uncertainty information still remains nonetheless, which is especially present when evaluating points further from the observations.

The example given shows a linear function, but also non linear functions can be used to map the input and output data. For instance, a set Gaussian basis functions can be fit to the data, with a linear set of parameters w , see Equation 3.4 for an example. This shows that also non-linear data can be mapped using BLR, however

its ability to capture highly nonlinear relationships is still limited [53].

$$y(x, \mathbf{w}) = w_0 + w_1 \exp\left(-\frac{(x - \mu_1)^2}{2\sigma_1^2}\right) + \dots + w_n \exp\left(-\frac{(x - \mu_n)^2}{2\sigma_n^2}\right) \quad (3.4)$$

All in all, BLR models are flexible tools with a wide range of applications, however they are computationally expensive and are sensitive to choice of prior distribution. [36].

3.2.3. Bayesian Networks

Bayesian Networks (BN) are graphical probabilistic models that represents the probabilistic relationships between a set of variables. Each node in the network represents a variable and describes the probability of its possible states given the states of its parent nodes. For example, if node A has parent nodes $B_1 \dots B_n$, then the conditional probability of A is given by $P(A|B_1, \dots, B_n)$. BN's are useful for describing or even discovering causal relationships in data [84].

Training a BN has two main steps, structure learning and parameter learning. Structure learning attempts to learn the relationships in the data and constructs the parent child relationships in the nodes. Parameter learning then learns the conditional probability of each node with respect to their parent nodes. Generally, BN is considered a relatively simple approach that is very useful for describing the relationships in the data, however they are considered to be computationally expensive and sensitive to the choice of prior distributions [44].

3.2.4. Bayesian Neural Networks

Conventional NN architectures use deterministic weights and biases to give deterministic predictions. A Bayesian Neural Network (BNN) on the other hand, according to Wilson et al. [120], are probabilistic models that incorporates uncertainty in its model parameters. In a BNN, each and every weight and bias is represented by a posterior distribution. However, as discussed in section 3.1, finding the complete posterior distribution is often intractable and thus approximation methods are needed to convert a conventional NN to a BNN. There are multiple of these methods to convert a conventional NN (like a CNN, LSTM, MLP, etc.) to a BNN. Frequently applied methods include MCMC (which takes a large but finite amount of samples of the posterior distribution), Monte Carlo Dropout (which randomly drops out neurons in training/testing and are run multiple times, giving an average and variance), VI (which approximates the posterior distribution using known tractable distributions) and deep ensembles (where identically structured neural networks are given different weights and their results are combined to obtain an average and variance)[32, 113, 120]. All in all, BNNs improve over conventional NNs by incorporating uncertainty in its parameters and results. They are very flexible and scaleable models, with the main drawback being its increased computational complexity for large data sets with respect to its conventional counterpart. This however is mitigated by recent advancements in variational inference techniques [39].

3.2.5. Gaussian Process Regression

A Gaussian Process (GP) is defined as a model that describes a probability distribution over possible functions that fit a set of points. Gaussian Process Regression (GPR) is a non-parametric approach to function prediction, meaning that it does not assume a specific functional form of the unknown function beforehand. The functional form is however initially guessed when establishing the prior distribution of functions to sample from. The prior distribution is dependent on the chosen kernel function, which aims to capture the relationship between the different data points (in general, similar data points yield similar kernel function outputs). A covariance matrix is made showing the relationships of the different data points based on the kernel function, which is in turn used to sample functions from a multivariate normal distribution. The prior distribution can then be updated using Bayes theorem to give a posterior distribution of functions which can be sampled to make predictions. Now, at any input point, the posterior distribution will show the uncertainty of its prediction at that given point [100]. An example of GPR being applied to 10 data points with the commonly used radial basis function (RBF) kernel can be seen in Figure 3.3.

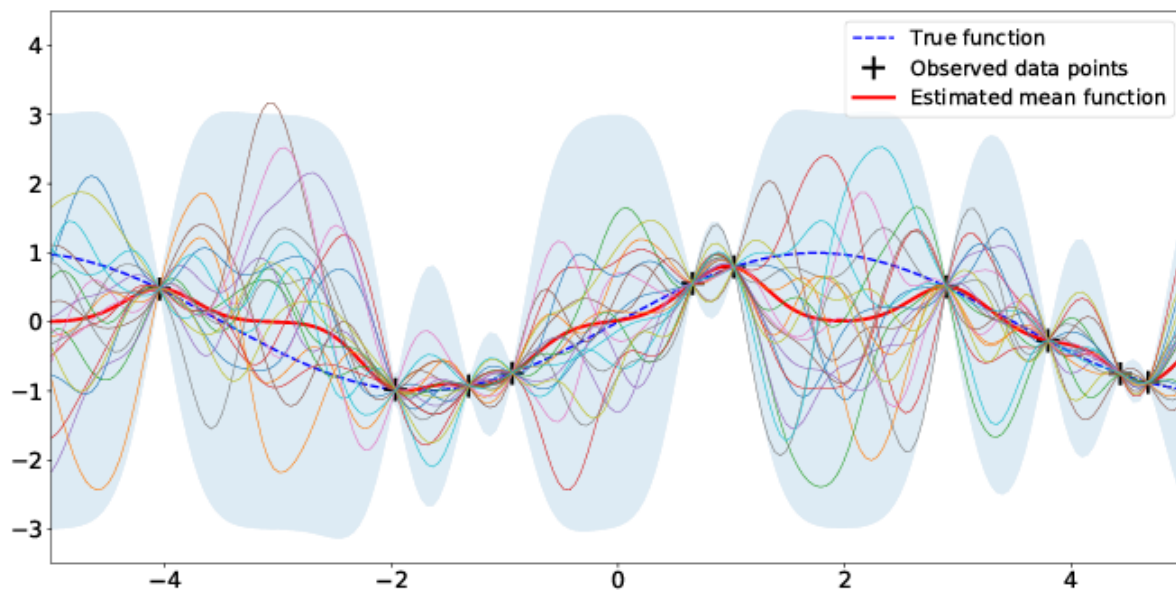


Figure 3.3: Example of a Gaussian Process Regression (GPR) with a Radial Basis Function (RBF) kernel. 10 observed data points are given and 20 functions are sampled from the posterior. The blue area indicates the 3σ confidence intervals. [115]

The main difference between GPR and BLR is the non-parametric property of the GPR. A BLR model attempts to fit a pre defined, finite, set of function parameters to a set of data points, meaning that the number of parameters is the limiting factor in the complexity of the model. A GPR model on the other hand has a set of parameters that grows with the size of the data-set, all the way to infinity, meaning that there are an infinite amount of functions to sample from, allowing for more complex relationships. However, a GPR does become very computationally expensive if the data set is large, with an overall computational complexity of $O(n^3)$ and a memory consumption that is quadratic [115].

3.2.6. Relevance Vector Machines

Relevance Vector Machines (RVM) are similar to Support Vector Machines (SVM), as they seek to find the best decision boundaries to separate different classes or predict continuous variables. The key difference between the two is that RVMs are Bayesian, providing a probabilistic prediction. RVMs model the output variables as a weighted linear combination of basis functions. RVMs are special due to their use of sparse weights, where it assumes that most of the weights are zero, leaving only a small subset of relevant basis functions. RVMs are trained by first taking a prior distribution of the weight vector, often a Gaussian distribution, which is updated to a posterior distribution using Bayes theorem based on the observations made. Due to the sparse set weights, the RVM automatically performs feature selection, making it an interpretable and efficient model. They are useful when the data-set is large and noisy, mainly when compared to other kernel based methods, however they are very sensitive to hyperparameter selection and do become computationally expensive if the data-set is too large [106].

3.3. Method comparison

This research will focus on the RUL estimation of turbofan engines using the C-MAPSS data-set. The data-set can be considered large consisting of multiple variables with noisy observations. The exact relationships and interdependencies between these variables is unknown. In order to take into account a large range of relationships in the data as possible, the model to be applied to this problem should be able to handle non-linear and complex relationships. This section will look into the models elaborated in section 3.2, and evaluate them based on the requirements for the proposed application.

Firstly, two of the discussed methods, Naive Bayes and Bayesian Networks, are considered not the most suitable for this application. This is due to their relatively simplistic nature. An NB is a simple and useful model, but its independence assumption between input variables may not be appropriate for the data-set in question, which likely contains many dependent input variables with complex relationships. BNs are

graphical models that are useful for finding and describing the joint distribution between input and output variables, however they become increasingly more expensive for large data-sets as BNs need to estimate the conditional probability between each variable combination. Additionally, as the amount of variables grows, the number of possible BN models grows exponentially [72]. For these reasons, BNs and NBs are not considered for further implementation.

Then, the remaining methods are BNN, RVM, GPR and BLR. All of these methods are capable of handling larger amounts of noisy data with complex relationships. Of these methods BLR and RVM are the most computationally efficient [106, 119]. BNNs and GPRs on the other hand are more flexible and can capture more complex non-linear relationships between the input and output [73, 119].

In this research, model accuracy is preferred over computational efficiency, meaning that the BNN and GPR model will be evaluated further. However, to compare the accuracy with a more efficient yet simplified model, RVM will also be evaluated further. RVM is chosen over BLR as RVM has already been applied to RUL estimation in the past (see section 2.1), whereas for BLR there is little to no literature regarding RUL estimation. Thus, BLR for RUL estimation is deemed out of scope for this research. In summary, the methods that will be further evaluated in this research are BNN, GPR and RVM.

3.4. Application of chosen methods

From section 3.3 it was concluded that the chosen methods to be further evaluated for application to this research will be BNN, GPR and RVM. This section will go into the earlier applications of these methods in RUL estimation and the what takeaways can be applied to this research.

3.4.1. BNN

BNNs have shortly been introduced earlier in section 2.4 and in section 3.2, with a mention of their application to RUL estimation as implemented by Caceres et al. [11] and Benker et al. [6]. In general, a BNN distinguishes itself from a normal NN by using distributions instead of single point values for its weights [66]. Caceres et al. used a Bayesian RNN (BRNN) in combination with a dense flipout layer. Benker et al. on the other hand uses a DNN and a CNN, both converted to their Bayesian counterparts. Commonly used approaches according to [13, 32] of converting a NN to a BNN are; Variational Inference (VI), Markov Chain Monte Carlo (MCMC), Dropout and Monte Carlo Dropout. Also, a more recent approach introduced by Blundell et al. [9] in 2015 called Bayes by Backprop can be considered. In general, these methods aim to find/approximate the posterior distribution of the model parameters. According to Gal et al. [31], while (Monte Carlo) Dropout models are often used in practical implementations of BNN due to their simplicity, VI, MCMC and Bayes by Backprop are more reliable and accurate. For this reason, we will only consider VI, MCMC and Bayes by Backprop methods.

Variational Inference is a method that approximates the posterior distribution of parameters by using a known family of distributions that are easier to work with (such as a Gaussian distribution), instead of trying to find the true posterior distribution. It tries to find the best approximation by optimizing the parameters of the chosen family of distributions by either minimizing a loss function called the Kullback-Lieber divergence, measuring how similar the approximate and true posterior distributions are, or optimizing the Evidence Lower Bound (ELBO). VI is a flexible and efficient technique, especially when applied to a large data-set [8].

MCMC for approximate inference is introduced by Gelfand and Smith [35] and in contrary to VI, aims to approximate the exact posterior distribution rather than represent it with a known distribution. It does this by constructing an ergodic Markov Chain whose stationary distribution is the posterior distribution, which is done by taking an initial sample of the posterior distribution (any relevant value in this case) and then generating a sequence of new samples using a transition kernel. This transition kernel defines the probability of going to a new state [92]. By sampling from this stationary distribution, a posterior distribution can be approximated. The main drawback of this method is its computational cost, especially for highly dimensional input data [8].

Bayes by Backprop treats the weights in the neural network as random variables with prior distributions. These prior distributions are updated using Bayes Rule to achieve a posterior distribution of weights over the network [9]. The prior distributions are often chosen to be a Gaussian distribution (however other distributions can also be used), as such it is comparable to VI due to its assumption of distribution family. Bayes by Backprop is a flexible and capable method, however its computational cost does significantly rise with larger data-sets.

VI, MCMC and Bayes by Backprop are all viable methods for posterior distribution approximation, with the main consideration between them being accuracy vs computational efficiency. As the C-MAPSS data-set to be evaluated is highly dimensional and large, the VI method can be considered to be the appropriate approach. However, Benker et al. [6] also perform a comparison of the performance of Hamiltonian Monte Carlo (HMC), a variant of MCMC, and VI for RUL estimation. They conclude that VI does outperform a HMC model, however a HMC model with pre trained parameters on the other hand outperforms VI. Comparing Bayes by Backprop an VI, the same consideration can be made with VI being more efficient but Bayes by Backprop being more accurate [39]. For now, the VI and Bayes by Backprop approach will mainly be considered.

3.4.2. GPR

GPR has already been discussed earlier in section 3.2 and their applications in PHM are mentioned in section 2.3. In summary, GPR models have the ability to sample over an infinite amount of functions that fit the data-set to provide an uncertainty estimate inbetween or beyond the data-set. The main considerations when applying a GPR model is the choice of kernel function and hyperparameter setting.

The Kernel function also known as the covariance function, describes the relationship between two input points, where the goal is to have similar input points lead to similar kernel function outputs, as this creates a smooth function [87]. The kernel function works essentially as our prior, as the choice of kernel function describes our belief of how the data is related. For instance, as summarized by Duvenaud [24], a simple linear kernel will sample linear functions in GPR model, while a periodic kernel will sample periodic functions in the GPR model. In the field of GPR, the Radial Basis Function (RBF) (also known as the Squared Exponential or Gaussian kernel function) has become the de-facto kernel function [24]. It is defined as shown in Equation 3.5, where $[x, x']$ indicates the 2 evaluated data points. The reason for its popularity is universality, it can be integrated against any function, it has an infinite amount of derivatives and only has 2 parameters. Kernels can also be combined to incorporate our prior beliefs even more accurately. For this research, the RBF function will likely be applied, this is however dependent on the performance found during development and may be altered.

$$cov[x, x'] = \sigma_f^2 \exp\left(-\frac{(x - x')^2}{2l^2}\right) \quad (3.5)$$

Hyperparameter tuning is the process of finding the optimal hyperparameter setting for a model. In GPR, the hyperparameters are the output variance σ_f^2 , which controls how far the functions vary from the mean, and specifically for the RBF function in this case, the length scale l , which stretches or shortens the functions (in general, it is not possible to extrapolate more than l length units away from the data) [24]. Tuning these parameters to their optimal setting is done using the log marginal likelihood given by Equation 3.6, where Θ^* is the set of optimal hyperparameters given by observations \mathbf{y} , inputs \mathbf{X} and old hyperparameters Θ [87].

$$\Theta^* = \operatorname{argmax} \log p(\mathbf{y}|\mathbf{X}, \Theta) \quad (3.6)$$

3.4.3. RVM

RVM models are introduced in section 3.2 and their applications in PHM is mentioned in section 2.3. In summary, RVM models aim to find the most relevant features of a data-set by using sparse weights that are set to zero for a large amount of input features. Similarly to GPR, they also make use of kernel functions and require hyperparameter tuning. In its most basic form, RVM is given by Equation 3.7, which is essentially a weighted sum of kernel functions $K(\mathbf{x}, \mathbf{x}_i)$ [106]. The choice of kernel function is a similar process as for the GPR, where the RBF kernel is also a common choice.

$$y(\mathbf{x}|\mathbf{w}) = \sum_{i=1}^N w_i K(\mathbf{x}, \mathbf{x}_i) + w_0 \quad (3.7)$$

The main hyperparameters to be optimized for RVM are those related to the chosen kernel function (as described for GPR), the noise variance (the assumed noise in the data, estimated from the data), and the hyperparameters controlling the sparsity of the solution. These hyperparameters can be optimized by using techniques such as cross-validation, maximum likelihood estimation, or Bayesian inference [106].

4

Explainable AI

Artificial intelligence has made significant leaps over the last decade, and is used more and more in everyday life but also in various industries. AI models have allowed us to make reliable predictions and identify complex relationships based on vast quantities of complex data. However, one drawback that AI suffers from is its 'black box' perception. After a model is developed, input data is converted to an output, but with no real understanding of what happens in-between. The field of explainable AI (XAI) aims to develop methods to achieve a deeper understanding as to what the output of an AI model actually means and why it generated this output. The reason for applying XAI to machine learning models is not only to provide a insights into the inner workings and reasoning of the models, but also to make the models more trustworthy. Especially in the field of PHM, where impactful decisions are to be made based on predictive models, a certain degree of explainability will greatly increase the trustworthiness and thus the usefulness of these models. In this chapter, the need for XAI will be discussed in section 4.1 followed by a short overview of XAI methods in section 4.2. Next the applicaiton of XAI in PHM will be discussed in section 4.3 followed by a more in depth introduction to counterfactual explanations in section 4.4. After this, a selection will be made for which counterfactual method to apply in section 4.5 followed by a more in depth description of this method in Equation 4.7. Lastly, the counterfactual performance indicators will be discussed in section 4.7.

4.1. Need for XAI

As discussed in chapter 2, the field of PHM has adapted the use of various ML techniques and considers AI as the current forefront of the field [124]. Furthermore, chapter 2 concluded with a mention to XAI and its necessity in the field of PHM. In a more broad sense, the field of XAI has spiked in interest in recent years, especially since the introduction of the XAI program by the Defense Advanced Research Projects Agency (DARPA) in 2017 [3]. According to Nor et al. [77], the main needs for XAI can be summarized as follows;

- Justification of the models decision by identifying issues and enhancing AI models.
- Obedience of the AI regulation and guidelines in usage, bias, ethics, dependability, accountability, safety, and security.
- Permission for users to confirm the models desirable features, promote engagement, obtain fresh insights into the model or data, and augment human intuition.
- Allowance for users to better optimize and focus their activities, efforts, and resources.
- Support for the model development when it is not yet considered as reliable.
- Encouragement for the cooperation between AI experts and external parties.

When looking at the field of PHM, needs are related to the justification and obedience of regulations. This is because PHM is related to high-investments and safety sensitive domains [77].

4.2. Explainable AI methods

XAI methods can generally be categorized using 4 categories; global vs local and model specific vs model agnostic [127]. Looking at global vs local, a global method explains the effect of the input features on the overall model by identifying the features that have the most impact on the model predictions, while a local method provides explanations for individual predictions by identifying the key features that led to a particular prediction. Looking at model specific vs model agnostic, model specific methods look at the specific model type and structure used to generate an explanation while model agnostic only looks at the input and outputs of the model, essentially treating the model in-between as a black box. Note that in this research, we are primarily interested in model agnostic methods due to our "black box" models. An overview of commonly used examples of XAI methods is given in Table 4.1.

Table 4.1: Examples of commonly used XAI methods categorized by local vs global and model agnostic vs specific

	Model Agnostic	Model Specific
Local	LIME [90], Anchors [91], Counterfactual Explanations [109], Protodash [28]	Bayesian Networks [68], SVM [38], Decision Trees [67]
Global	SHAP [108], PDP [104], Integrated Gradients [99]	RuleFit [2], NNV [51], GBM feature importance [4]

4.3. Application of XAI in PHM

The application of AI in PHM has been growing over the past decades, and in even more recent years also the application of XAI [124]. A comprehensive literature review of the application of XAI in the field of PHM has been performed by Nor et al. [77]. This overview shows that currently the XAI applied is mainly based on interpretable models, rule and knowledge based models and attention mechanisms, but also that model agnostic methods are close behind. It is also concluded that XAI could become a preferred or even a required tool within PHM compared to standalone methods.

In the literature overview, also some applications are mentioned. One application, from Hong et al. [40], is particularly relevant as it aims to perform RUL estimation on the C-MAPSS data-set and uses a model agnostic method for the explainability. The authors propose a combination of multiple deep neural networks, namely 3 CNN layers, 3 LSTM layers and 2 Bi-LSTM layers. Using this model, the authors achieve an RMSE of 10.41 which can be considered a very high accuracy. They then use SHAP to explain the contribution of each individual feature to the output. Also other model agnostic applications are mentioned, such as Sundar et al. [103] who aim to predict fouling resistance with a deep neural network and achieve explainability using LIME, or Onchis et al. [78] who aim to detect damages on cantilever beams using deep neural networks and also achieving explainability using LIME and SHAP.

In summary, this overview shows the recent increase in interest of applying XAI in the field of PHM, but also covers some of its current shortcomings. Aspects such as lack of human role, explanation metrics and uncertainty management are recommended areas of further research. In this research, the aspect of uncertainty management will be further addressed.

Furthermore, for this research, we are interested in estimating the RUL of an aircraft engine using the C-MAPSS data-set, and then generating useful explanations. The explanations we are looking for are generally in the form of: "*What feature change will result in a certain in/decrease of the RUL?*". For the C-MAPSS data-set specifically, a question could be in the form of e.g. "*What if the inlet temperature would have been y instead of x, what would have been the RUL?*". This answering of the question "*What if things had been different?*" is the main philosophy of the field of counterfactual explanations [89]. As such, out of the methods mentioned in Table 4.1, Counterfactual Explanations (CFE) will be considered as the XAI method of this research.

4.4. Counterfactual explanations

As shown in Table 4.1, CFE is a model agnostic and local XAI method, meaning that it treats the ML model as a black box looking only at the in-, and outputs, and that it focuses on individual predictions rather than the overall predictive set. CFE was introduced by Wachter et al. [111], and is in its core a "*What if?*" applied to the results of, in this case, machine learning. For example, a person going to a bank for a loan. The ML

model used to determine whether or not to issue the loan takes as input [Income, Credit score, Education, Age]. The ML model determines that the loan is denied, but does not give an explanation. This is where XAI, and more specifically CFE comes in. With CFE, we can ask why the loan was denied and what can be done to change it to approved! For example, the CFE could say that the income is too low and that it needs to be increased by \$10k (i.e. *What if* the income was \$10k more?). This makes an ML model more interpretable and more useful. This section will go into the properties of CFEs, the distance measurement methods and the various CFE methods extracted from literature.

4.4.1. CFE properties

According to Verma et al. [109], Mothilal et al. [71] and Chou et al. [19], the main properties that a good CFE model should incorporate is validity, actionability, feasibility, sparsity, diversity, data manifold closeness and causality;

- **Validity:** A CFE is valid if it ensures the "new" prediction is in the desired class, while minimizing the amount of deviation from the original inputs. This deviation (or distance) can be the L1/L2 distance, quadratic distance, or another distance function.
- **Actionability:** In order to provide a useful CFE, the proposed change to the input needs to be actionable. For instance, in the example given above, requiring a change in the income is actionable, but altering the person's age is not.
- **Feasibility:** While minimizing the deviation is a priority for CFEs, a CFE can be considered unfeasible if it is close to the decision boundary, where the model is not as certain about its output. A feasible CFE should lie in a more well defined region of a model.
- **Sparsity:** A trade-off can be made between the number of features changed and the total amount of change to be made to obtain a valid CFE. Ideally, a CFE should change the smallest amount of features possible to obtain an effective CFE. This can be measured by for instance the L0/L1 norm.
- **Diversity:** If multiple CFEs are generated, then they should be diverse, providing significantly different explanations.
- **Data manifold closeness:** This aspect relates to the plausibility of the adjusted features provided by a CFE. A CFE would be difficult to trust if a combination of features is proposed that is unlike anything observed in the training set. So, this aspect aims to ensure that the proposed changes lie within a feasible range of the observations made.
- **Causality:** Features in a data-set are rarely completely independent of each other. To relate to the example given above, if the CFE would suggest to get a new educational degree, then the person's age should also increase by some amount. A realistic and actionable CFE should account for any known causal relationships between features.

Using the above aspects, a CFE can be posed as an optimization problem as shown in Equation 4.1. It can be seen here that it tries to maximize for a new desired output y' by generating new model predictions $f(x')$, while minimizing the deviation between the original data point x and new data point x' , incorporating sparsity, and ensuring that the new data point x' does not deviate too much from the training set χ , all while ensuring that x' is in the actionable space Ω .

$$\arg \min_{\substack{x' \in \Omega \\ \text{actionability}}} \max_{\lambda} \underbrace{\lambda(f(x') - y')^2}_{\text{desired output}} + \underbrace{d(x, x')}_{\text{deviation}} + \underbrace{g(x' - x)}_{\text{sparsity}} + \underbrace{l(x'; \chi)}_{\text{data manifold}} \quad (4.1)$$

4.4.2. Distance measurement

At the core of CFE lies the property of minimal change to the input to achieve the desired output [57]. This distance, or deviation $d(x, x')$ as described in Equation 4.1, can be described by multiple functions according to [19]. Commonly used distance functions are the L_p norms defined by Equation 4.2 given vector x . Different values for p give different norms with specific properties. In XAI the norms used are the L_0 - norm (which is

technically not a norm by definition), the L_1 -norm (also known as the Manhattan distance), the L_2 -norm (also known as the Euclidean distance) and the L_∞ -norm.

$$\|x\|_p = \left(\sum_{i=1}^n |x_i|^p \right)^{1/p} \quad (4.2)$$

L_0 -norm is defined by Equation 4.3 and has been explored in the field of XAI by primarily Dandl et al. [20] and Karimi et al. [49]. In practice, the L_0 norm does not measure the direct distance, rather it counts the nonzero elements of a vector. In this case, it is a measure of the amount of features that are changed from x to x' , enforcing sparsity. It is however not differentiable, making it hard to minimize in an optimization problem.

$$\|x\|_0 = \sqrt[0]{\sum_i x_i^0} \quad (4.3)$$

L_1 -norm is defined by Equation 4.4 and is argued to be the norm that provides the best solutions according to Wachter et al. [111]. It enforces sparsity as it gives equal penalties to all parameters. Its main drawback is that it can be hard to differentiate.

$$\|x\|_1 = \sum_i |x_i| \quad (4.4)$$

L_2 -norm is defined by Equation 4.5 and is also commonly used in XAI. It measures the shortest distance between two points and can detect larger errors than the L_1 -norm. It does not produce sparse vectors such as the L_0 and L_1 norms, however it is a smooth function that is differentiable and computationally efficient for optimisation.

$$\|x\|_2 = \sqrt{\sum_i x_i^2} \quad (4.5)$$

L_∞ -norm is defined by Equation 4.6 and is primarily explored in the XAI context by Karimi et al. [49]. It looks at the maximum change across the features and penalizes the cost of the largest feature change. It is entirely differentiable, except when two features have the same absolute value. It also leads to less sparse solutions compared to the L_0 and L_1 norms.

$$\|x\|_\infty = \sqrt[\infty]{\sum_i x_i^\infty} = \max(|x_i|) \quad (4.6)$$

4.4.3. Counterfactual explanation methods

Chou et al. [19] perform a comprehensive review of model-agnostic counterfactual approaches in XAI. In total, 23 algorithms were analyzed. As many of these algorithms originate from a similar theoretical background, a total of 7 categories are introduced representing the "*Master theoretical algorithms*" [23]. A taxonomy of these categories and algorithms is given in Figure 4.1.

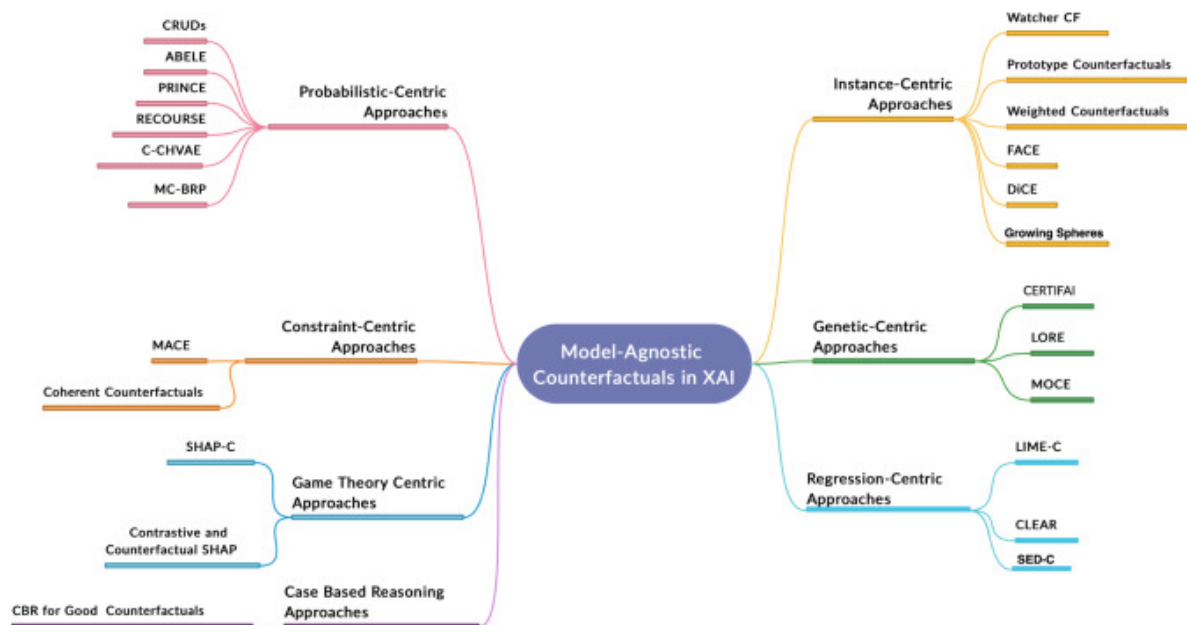


Figure 4.1: Model agnostic counterfactuals in XAI taxonomy as constructed by Chou et al. [19]

- Instance-Centric:** This approach corresponds to the algorithms derived from Lewis [57] and Wachter et al. [111]. These approaches are based on random feature permutations, where a distance function is used to find a CFE close to the original data-point. As these approaches seek novel loss functions and optimization algorithms, they are susceptible to fail the actionability, feasibility and diversity properties. However, there are some instance centric algorithms that use certain mechanisms to overcome these issues, such as FACE and DiCE.
- Constraint-Centric:** These algorithms are all modelled as constraint satisfaction problems. The advantage of this approach is that they are general and can satisfy different counterfactual properties such as feasibility, diversity and actionability.
- Genetic-Centric:** This approach uses genetic algorithms as an optimization method for CFE generation. Due to the properties of genetic algorithms such as feature crossover and mutation, these approaches generally satisfy the diversity property.
- Regression-Centric:** This approach generates CFEs by applying a regression model. It works by fitting a regression model to a newly generated data-point by permuting the features. The weights of this regression model can then be used as the explanations. These algorithms do however have difficulty satisfying the actionability and diversity properties and often have poor stability.
- Game Theory-Centric:** This approach relates to all algorithms that use Shapley values for their explanations. They are similar to the SHAP method (as mentioned in section 4.2), which generates explanations by varying all features and documenting their relative effect on the output. This approach extends the SHAP algorithm to incorporate counterfactuals. These algorithms often have difficulty with actionability and diversity properties, but often do have good stability.
- Case-Based Reasoning:** This approach is a memory based method, where previously solved and stored cases are adapted and used for the generation of new solutions. For CFEs, the case-based reasoning stores good counterfactual explanations, which can later be retrieved when a new CFE needs to be generated. These algorithms perform well on the validity, actionability, feasibility and diversity properties.
- Probabilistic-Centric:** This approach uses probabilistic methods to model the CFE generation problem. Methods such as random walks, Markov sampling, variational auto-encoders and probabilistic graphical models (PGM) can be used. According to Pearl [83], who used a PGM framework, causality is promoted and actionability and feasibility are ensured.

4.5. Selection of CFE method

Of the methods shown in Figure 4.1, one will be used in this research. An overview of these methods and their properties is provided in Table 4.2, which is based on the information provided by Chou et al. [19]. Using this table, we can make an initial selection of potential methods.

First of all, for the application proposed in this research, the most important properties are sparsity and diversity. This is because the explanations should be limited to a handful of potential feature changes to provide realistic proposals regarding aircraft engine maintenance, but should also provide multiple of these explanations for redundancy.

Secondly, the actionability property is of high importance, as this will ensure that the feature changes are realistic and that immutable features can be taken into account.

Thirdly, for the sake of research scope, there should be an existing code/library/package that can be implemented (with proper modifications). This will not only save time, but also provide a verified code base as a starting point.

Table 4.2: Summary of CFE algorithms and their properties

Category	Algorithm	Available code	Actionability	Sparsity	Diversity	Feasibility	Causality
Probabalistic	CRUDs			x	x	x	
	PRINCE	x		x			
	C-CHVAE	x		x			
	ABELE	x		x			
	RECOURSE	x	x	x	x	x	x
	MC-BRP	x		x	x		
Constraint	MACE	x	x	x	x	x	
	Coherent CF	x	x	x	x	x	
Game theory	SHAP-C	x					
	C and C SHAP						
Case based reasoning	CBR CF		x	x	x	x	
Instance	Watcher CF	x		x			
	Prototype CF	x		x			
	FACE	x		x	x	x	
	Weighted CF			x			
	Growing Spheres	x		x			
	DiCE	x	x	x	x	x	
Genetic	MOCE	x		x	x		
	CERTIFAI	x			x	x	
	LORE	x		x	x		
Regression	LIME-C	x					
	SED_C	x		x			
	CLEAR	x		x			

Using these requirements, the best algorithms that remain are RECOURSE, MACE, Coherent CF and DiCE. All of these algorithms should theoretically be sufficient for this research. For the final trade-off, we examined the open source code base of all the algorithms (RECOURSE [48], MACE [47], Coherent CF [1], DiCE [70]). From this examination, it was determined that DiCE is the most user friendly open source library to use and is easiest applied to this research. This is mainly due to three reasons, firstly, all the other code bases are mainly the functional code repository used for their respective papers while DiCE is designed to be a standalone package to be used for CFE applications. Secondly, DiCE is developed and supported by Microsoft, ensuring that the code base has been and will be well maintained and improved. Lastly, DiCE has an extensive documentation and guidance library. For these reasons, DiCE will be considered the main CFE generation algorithm.

4.6. DiCE

DiCE, or Diverse Counterfactual Explanations, is a CFE generating method proposed by Mothilal et al. [70], which focuses on generating diverse, feasible, close proximity and sparse explanations which also take into account user constraints (e.g. limiting feature changes or setting immutable features). These aspects are combined into an optimization function as given in Equation 4.7 with the goal of generating k diverse CFEs. This is comparable to the optimization function proposed in Equation 4.1, with the main difference being the $dpp_diversity$. This aspect aims to ensure diverse CFEs while mitigating non-preferred effects such as changing a large set of features (decreasing sparsity) or make big changes from the original input. So, diversity via Determinantal Point Processes (DPP) is used which is given by Equation 4.8. In general, a large DPP indicates a scattered and diverse set of CFEs while a small DPP indicates a clustered and similar set [54]. Equation 4.7 is optimized using gradient descent, where ideally $f(x') = y$ is achieved for every CFE. This may not always be possible due to the non-convex nature of the loss function, and thus a maximum amount of 5000 steps are run or until the loss function converges and the generated CFE is valid. All x' are initialized randomly.

$$C(x) = \arg \min_{x'_1 \dots x'_k} \frac{1}{k} \sum_{i=1}^k y_{loss}(f(x'_i), y) + \frac{\lambda_1}{k} \sum_{i=1}^k dist(x'_i, x) - \lambda_2 dpp_diversity(x'_1 \dots x'_k) \quad (4.7)$$

$$dpp_diversity = det(\mathbf{K}) \quad \text{with} \quad \mathbf{K}_{i,j} = \frac{1}{1 + dist(x'_i, x'_j)} \quad (4.8)$$

$$dist(x', x) = \frac{1}{n} \sum_{p=1}^d \frac{|x'^p - x^p|}{MAD^p} \quad (4.9)$$

When implementing the DiCE algorithm, there were some practical considerations made by the authors. These include the choice of yloss, choice of distance function, relative scale of features, sparsity enhancement and hyperparameter choice.

- **Choice of yloss:** The yloss function should not simply minimize the distance between $f(x')$ and y , as this is not always necessary. Rather, it should ensure that the counterfactual input vector crosses the threshold of $f()$ such that a new outcome is accepted. By default, the yloss function implemented in DiCE is a hinge-loss function that does not give a penalty if $f(x')$ is above the required threshold for the desired class y .
- **Choice of distance function:** This will define how the distance is measured between original input x and counterfactual input x' . For this, the mean of the L_1 distance can be taken as defined by Equation 4.9, where the MAD_p is the median absolute deviation for the p -th out of n continuous variables.
- **Relative scale of features:** The scale of features highly influences their relative impact on the objective function. As such, the features are scaled $[0, 1]$.
- **Enhancing sparsity:** Sparsity is enhanced by a post-hoc operation that restores the altered feature values to their original values until the predicted outcome $f(x')$ changes. The changed features are those whose difference is less than a certain threshold. For each feature the threshold is chosen to be the minimum of MAD and the bottom 10% percentile of the absolute difference between non-identical values from the median.
- **Hyperparameter choice:** This relates to the choice of λ_1 and λ_2 in Equation 4.7. Through a grid search with different values and evaluating the diversity and proximity of the generated CF examples, these have been set to $\lambda_1 = 0.5$ and $\lambda_2 = 1$.

4.7. CFE performance indicators

While the interest for CFEs has increased in recent years, the evaluations are done typically in a qualitative fashion. A set of evaluation metrics is proposed by Mothilal et al. [70] to be able compare the validity, proximity, sparsity and diversity of the generated CFEs.

Validity is measured by taking the fraction of the k generated CFEs that are actually counterfactuals, i.e. they correspond to a different, desired, outcome. This is given by Equation 4.10, where $f(x') > 0.5$ indicates that the CFE has crossed the threshold for generating a new output.

$$\%Valid\text{-CFEs} : \frac{|\{\text{unique instances of } C \text{ s.t. } f(x') > 0.5\}|}{k} \quad (4.10)$$

Proximity is measured using the distance function where simply the average distance over the k CFEs is taken. This is given by Equation 4.11. Note that in contrast to the normalized $([0,1])$ features used in Equation 4.1, this metric will use the original feature values for a better interpretability of the distances.

$$\text{Proximity} : 1 - \frac{1}{k} \sum_{i=1}^k \text{dist}(\mathbf{x}'_i, \mathbf{x}) \quad (4.11)$$

Sparsity is measured as the average number of changed features of k CFEs and d input features. It is given by Equation 4.12.

$$\text{Sparsity} : 1 - \frac{1}{kd} \sum_{i=1}^k \sum_{l=1}^d 1_{[x'_i \neq x_i^l]} \quad (4.12)$$

Diversity is measured similarly to proximity, with the difference being that instead of comparing CFEs to the original input, the distance is measured between each CFE which is then averaged, which is given in Equation 4.13. Also another metric for diversity is proposed, which measures the sparsity based diversity by looking at the fraction of features that are different between the CFEs, which is given by Equation 4.14.

$$\text{Diversity} : \frac{1}{C_k^2} \sum_{i=1}^{k-1} \sum_{j=i+1}^k \text{dist}(\mathbf{x}'_i, \mathbf{x}'_j) \quad (4.13)$$

$$\text{Count-Diversity} : \frac{1}{C_k^2 d} \sum_{i=1}^{k-1} \sum_{j=i+1}^k \sum_{l=1}^d 1_{[x'_i \neq x'_j^l]} \quad (4.14)$$

Aside from the metrics proposed above, the authors also propose a metric to help the user approximate the decision boundary. This will help in understanding the local decision boundary and help the user "guess" the working of the model. This metric works by training a secondary machine learning model to "mimic" the original ML model. The proposed model is a 1-nearest neighbour classifier (1-NN), chosen due to its simplicity and connection to people's decision making process using examples, that is trained on the generated CFEs and the original data-set. The 1-NN is compared to the original ML model to evaluate the accuracy.

5

Research Proposal

This chapter will focus on the next steps of this research based on what was found in the literature. First, the research gap will be discussed in section 5.1. Then, the research goals and questions will be proposed in section 5.2 and section 5.3 respectively, followed by the methodology and timeline given in section 5.4.

5.1. Research gap

In this literature review we have discussed three main aspects, PHM, Bayesian methods and XAI (specifically counterfactual explanations). From this we have learned that in the world of PHM, the application of machine learning methods for RUL prediction is widely applied, especially in research. The main obstacle preventing these methods from being applied more widespread in the industry is the "black box" nature of these models, which makes them difficult to understand and trust. XAI is a potential solution for this issue, which as of now has not been widely covered in research. On the other hand, the use of Bayesian methods that incorporate uncertainty in their predictions have been used in the field of PHM (e.g. in the form of Gaussian Processes or Naive Bayes), but have not been widely used in combination with XAI. Therefore, in summary, the main research gap this research will address is the combination of using XAI, specifically CFE, with Bayesian models applied to the field of PHM, specifically RUL estimation.

5.2. Research Goals

In this research, we aim to produce CFEs for Bayesian methods applied to, in this case, the C-MAPSS turbofan data-set. Afterwards we want to compare the quality of these explanations of the different Bayesian methods with each other with the goal of determining which method produces the most actionable and realistic explanations. The goals are summarized as bellow:

- **RG1:** Implement three different Bayesian models (GRF, BNN and RVM) to the C-MAPSS data-set.
- **RG2:** Implement counterfactual generation method (DiCE) on the Bayesian models.
- **RG3:** Define/Identify metrics to evaluate/asses the quality of the explanations.
- **RG4:** Evaluate and compare the quality of the counterfactual explanations per model.

5.3. Research Questions

The main research question of this research is:

What is the quality of counterfactual explanations for different Bayesian models in predictive maintenance applied to a turbofan case study?

The following sub-questions are proposed to perform this research:

1. How do we define a valid counterfactual with regard to RUL prediction?

2. How can the uncertainty information from the Bayesian models be used to create or enhance the counterfactual explanations?
3. What set of metrics best indicates the quality of a counterfactual explanation for the field of predictive maintenance using the uncertainty of a Bayesian model?
4. Which of the analyzed Bayesian models generates the highest quality counterfactuals according to the selected set of metrics?

5.4. Project methodology & timeline

This section will go over a high level outline of the project methodology and timeline proposed for this research. Starting off is the pre-processing of the C-MAPSS data, such as normalization, labeling and sensor selection. Next is the development of the Bayesian models, namely the RVM, GPR and BNN models. This research will start off with developing one of these models (for now the BNN model) initially and implement the CFE generation method (DiCE) on only this model. This will allow us to make adjustments in the development of the remaining Bayesian models to make them more compatible with the CFE generation model. During this process, various metrics will be implemented and evaluated to determine which set of metrics will provide an estimate of the quality of the CFEs. After an acceptable CFE generating performance has been achieved from the RVM and DiCE models, the GPR and RVM models will be developed using the experience gained from the development of the BNN model. These will also be run through the DiCE model where-after the results will be analyzed and reported. The aforementioned steps are split up into phases as explained below. A schematic overview can be seen in Figure 5.1.

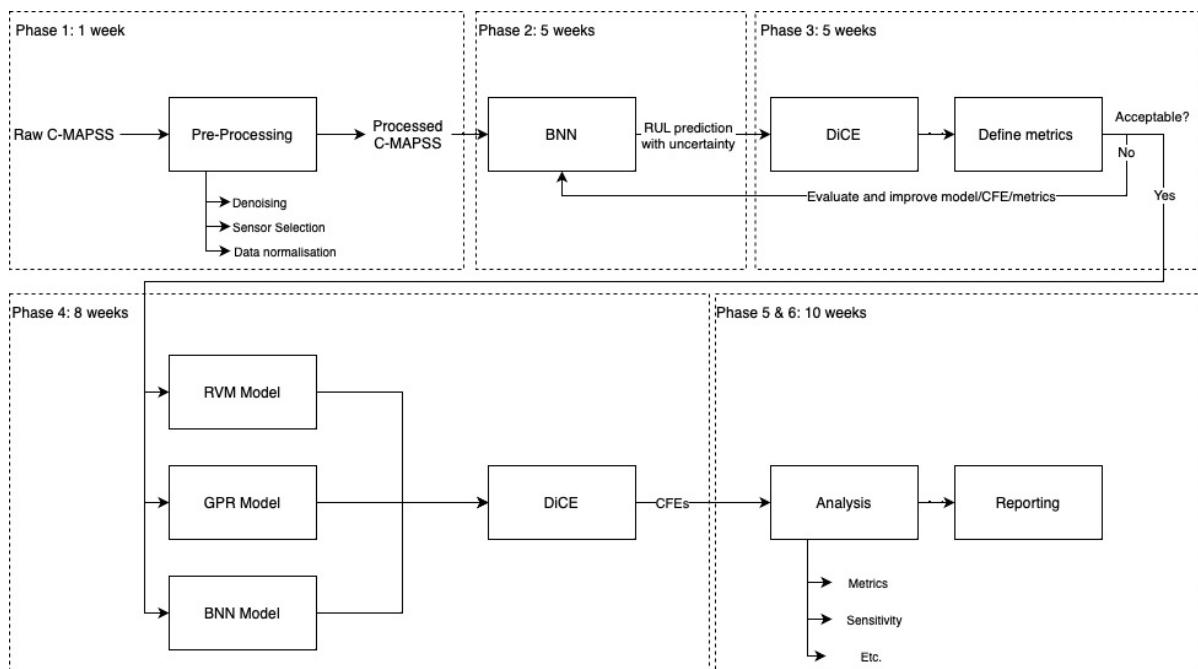


Figure 5.1: Methodology and timeline overview

5.4.1. Phase 1

Phase 1 will encompass the data preparation aspect, where the C-MAPSS data will be prepared for the following phases. This will consist of:

- Sensor selection, some sensors may not carry significant information, so only those who do will be selected.
- Data normalization, where the inputs will be scaled from 0-1, as this ensures that all features are equally represented in the model.
- Denoising, the sensor data has noise that may influence the model performance.

5.4.2. Phase 2

In phase 2, the first Bayesian model will be developed for RUL prediction. It will consist of:

- Development of BNN model, this model is chosen as the first Bayesian model to be developed.
- Verification of BNN model.
- Evaluation and improvement of BNN model, where the predictive performance is evaluated and improvements can be implemented.

5.4.3. Phase 3

In phase 3, the CFE generation method, in this case DiCE, will be applied to the BNN model. It will consist of:

- Implementation of DiCE to the BNN model.
- Verification of DiCE model.
- Definition and implementation of metrics.
- Evaluation and improvement of DiCE and BNN model.

5.4.4. Phase 4

Using the experience of the application of the BNN model to the DiCE model, the GPR and RVM models will be developed and implemented to the DiCE model. This phase will consist of:

- Development of GPR and RVM models.
- Verification of GPR and RVM models.
- Implementation of DiCE to the GPR and RVM models.
- Implementation of additional CFE generation method (*Only if time allows*).

5.4.5. Phase 5 & 6

The remaining phases will mainly consist of the analysis of the results, metrics, sensitivity analysis, final documentation and feedback implementation.

5.5. Gantt Chart

For this research, a Gantt chart has also been made indicating the various steps that will be taken during the project timeline. Note that the green columns around July and December indicate planned vacations during the summer and Christmas holidays.

Bibliography

- [1] Diverse coherent explanations, 2021. <https://bitbucket.org/ChrisRussell/diverse-coherent-explanations/src/master/>.
- [2] Zakaria Abou El Houda, Bouziane Brik, and Sidi-Mohammed Senouci. A novel iot-based explainable deep learning framework for intrusion detection systems. *IEEE Internet of Things Magazine*, 5(2):20–23, 2022.
- [3] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160, 2018.
- [4] Afek Ilay Adler and Amichai Painsky. Feature importance in gradient boosting trees with cross-validation feature selection. *Entropy*, 24(5):687, 2022.
- [5] Shaheer Ansari, Afida Ayob, M. S. Hossain Lipu, Aini Hussain, and Mohamad Hanif Md Saad. Remaining useful life prediction for lithium-ion battery storage system: A comprehensive review of methods, key factors, issues and future outlook. *Energy Reports*, 8:12153–12185, 11 2022. ISSN 2352-4847. doi: 10.1016/J.EGYR.2022.09.043.
- [6] Maximilian Benker, Lukas Furtner, Thomas Semm, and Michael F. Zaeh. Utilizing uncertainty information in remaining useful life estimation via bayesian neural networks and hamiltonian monte carlo. *Journal of Manufacturing Systems*, 61:799–807, 10 2021. ISSN 0278-6125. doi: 10.1016/J.JMSY.2020.11.005.
- [7] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.
- [8] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.
- [9] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *32nd International Conference on Machine Learning, ICML 2015*, volume 2, 2015.
- [10] Steven L. Brunton, J. Nathan Kutz, Krithika Manohar, Aleksandr Y. Aravkin, Kristi Morgansen, Jennifer Klemisch, Nicholas Goebel, James Buttrick, Jeffrey Poskin, Adriana W. Blom-Schieber, Thomas Hogan, and Darren McDonald. Data-driven aerospace engineering: Reframing the industry with machine learning. *AIAA Journal*, 59(8), 2021. ISSN 1533385X. doi: 10.2514/1.J060131.
- [11] Jose Caceres, Danilo Gonzalez, Taotao Zhou, and Enrique Lopez Droguett. A probabilistic bayesian recurrent neural network for remaining useful life prognostics considering epistemic and aleatory uncertainties. *Structural Control and Health Monitoring*, 28(10), 2021. ISSN 15452263. doi: 10.1002/stc.2811.
- [12] Weipeng Cao, Xizhao Wang, Zhong Ming, and Jinzhu Gao. A review on neural networks with random weights. *Neurocomputing*, 275:278–287, 2018.
- [13] Daniel T Chang. Bayesian neural networks: Essentials. *arXiv preprint arXiv:2106.13594*, 2021.
- [14] M. Chao, C. Kulkarni, K. Goebel, and O. Fink. Aircraft engine run-to-failure dataset under real flight conditions. NASA Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, 2021.
- [15] Luping Chen, Liangjun Xu, and Yilin Zhou. Novel approach for lithium-ion battery on-line remaining useful life prediction based on permutation entropy. *Energies*, 11(4):820, 2018.
- [16] Xin Chen, Ge Jin, Siqi Qiu, Minglei Lu, and Danjiong Yu. Direct remaining useful life estimation based on random forest regression. In *2020 Global Reliability and Prognostics and Health Management (PHM-Shanghai)*, pages 1–7. IEEE, 2020.

- [17] Benvolence Chinomona, Chunhui Chung, Lien-Kai Chang, Wei-Chih Su, and Mi-Ching Tsai. Long short-term memory approach to estimate battery remaining useful life using partial data. *Ieee Access*, 8:165419–165431, 2020.
- [18] Yohwan Choi, Seunghyoung Ryu, Kyungnam Park, and Hongseok Kim. Machine learning-based lithium-ion battery capacity estimation exploiting multi-channel charging profiles. *Ieee Access*, 7: 75143–75152, 2019.
- [19] Yu-Liang Chou, Catarina Moreira, Peter Bruza, Chun Ouyang, and Joaquim Jorge. Counterfactuals and causability in explainable artificial intelligence: Theory, algorithms, and applications. *Information Fusion*, 81:59–83, 2022.
- [20] Susanne Dandl, Christoph Molnar, Martin Binder, and Bernd Bischl. Multi-objective counterfactual explanations. In *Parallel Problem Solving from Nature–PPSN XVI: 16th International Conference, PPSN 2020, Leiden, The Netherlands, September 5-9, 2020, Proceedings, Part I*, pages 448–469. Springer, 2020.
- [21] Cameron Davidson-Pilon. *Bayesian methods for hackers: probabilistic programming and Bayesian inference*. 2015. ISBN 9780133902914. <https://learning-oreilly-com.tudelft.idm.oclc.org/library/view/bayesian-methods-for/9780133902914/>.
- [22] Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic? does it matter? *Structural safety*, 31(2):105–112, 2009.
- [23] Pedro Domingos. *The master algorithm: How the quest for the ultimate learning machine will remake our world*. Basic Books, 2015.
- [24] David Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge, 2014.
- [25] Hatem M Elattar, Hamdy K Elminir, ũ A M Riad, and A M Riad. Prognostics: a literature review. *Complex & Intelligent Systems 2016 2:2*, 2(2):125–154, 6 2016. ISSN 2198-6053. doi: 10.1007/S40747-016-0019-3. <https://link.springer.com/article/10.1007/s40747-016-0019-3>.
- [26] Deniz Ertuncay and Giovanni Costa. Determination of near-fault impulsive signals with multivariate naive bayes method. *Natural Hazards*, 108(2):1763–1780, 2021.
- [27] Jing Feng, Paul Kvam, and Yanzhen Tang. Remaining useful lifetime prediction based on the damage-marker bivariate degradation model: A case study on lithium-ion batteries used in electric vehicles. *Engineering Failure Analysis*, 70:323–342, 2016.
- [28] Renato Miranda Filho, Anisio M Lacerda, and Gisele L Pappa. Explainable regression via prototypes. *ACM Transactions on Evolutionary Learning*, 2(4):1–26, 2023.
- [29] Eibe Frank, Leonard Trigg, Geoffrey Holmes, and Ian H Witten. Naive bayes for regression. *Machine Learning*, 41:5–25, 2000.
- [30] Dean K Frederick, Jonathan A Decastro, and Jonathan S Litt. User’s guide for the commercial modular aero-propulsion system simulation (c-mapss). 2007. <http://www.sti.nasa.gov>.
- [31] Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015.
- [32] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.
- [33] Mohamed Ahmed Galal, Wessam M Hussein, Ezz El-din abdel Kawy, and Mahmoud MA Sayed. Satellite battery fault detection using naive bayesian classifier. In *2019 IEEE Aerospace Conference*, pages 1–11. IEEE, 2019.
- [34] Ming-Feng Ge, Yiben Liu, Xingxing Jiang, and Jie Liu. A review on state of health estimations and remaining useful life prognostics of lithium-ion batteries. *Measurement*, 174:109057, 2021.

- [35] Alan E Gelfand and Adrian FM Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association*, 85(410):398–409, 1990.
- [36] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 1995.
- [37] Weijun Gu, Zechang Sun, Xuezhe Wei, and Haifeng Dai. A new method of accelerated life testing based on the grey system theory for a model-based lithium-ion battery life evaluation system. *Journal of Power Sources*, 267:366–379, 2014.
- [38] Khan Md Hasib, Farhana Rahman, Rashik Hasnat, and Md Golam Rabiul Alam. A machine learning and explainable ai approach for predicting secondary school student performance. In *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0399–0405. IEEE, 2022.
- [39] José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International conference on machine learning*, pages 1861–1869. PMLR, 2015.
- [40] Chang Woo Hong, Changmin Lee, Kwangsuk Lee, Min-Seung Ko, Dae Eun Kim, and Kyeon Hur. Remaining useful life prognosis for turbofan engine using explainable deep neural networks with dimensionality reduction. *Sensors*, 20(22):6626, 2020.
- [41] Joonki Hong, Dongheon Lee, Eui-Rim Jeong, and Yung Yi. Towards the swift prediction of the remaining useful life of lithium-ion batteries with end-to-end deep learning. *Applied energy*, 278:115646, 2020.
- [42] Xiaosong Hu, Jiuchun Jiang, Dongpu Cao, and Bo Egardt. Battery health prognosis for electric vehicles using sample entropy and sparse bayesian predictive modeling. *IEEE Transactions on Industrial Electronics*, 63(4):2645–2656, 2015.
- [43] Mehdi Jafari, Laura E Brown, and Lucia Gauchia. A bayesian framework for ev battery capacity fade modeling. In *2018 IEEE Transportation Electrification Conference and Expo (ITEC)*, pages 304–308. IEEE, 2018.
- [44] Finn V Jensen and Thomas Dyhre Nielsen. *Bayesian networks and decision graphs*, volume 2. Springer, 2007.
- [45] Ruihua Jiao, Kaixiang Peng, and Jie Dong. Remaining useful life prediction of lithium-ion batteries based on conditional variational autoencoders-particle filter. *IEEE Transactions on Instrumentation and Measurement*, 69(11):8831–8843, 2020.
- [46] Marine Jouin, Rafael Gouriveau, Daniel Hissel, Marie Cécile Péra, and Nouredine Zerhouni. Particle filter-based prognostics: Review, discussion and perspectives. *Mechanical Systems and Signal Processing*, 72-73:2–31, 5 2016. ISSN 0888-3270. doi: 10.1016/J.YMSSP.2015.11.008.
- [47] Amir-Hossein Karimi. Model agnostic counterfactual explanation, 2021. <https://github.com/amirhk/mace>.
- [48] Amir-Hossein Karimi. Recourse, 2021. <https://github.com/amirhk/recourse>.
- [49] Amir-Hossein Karimi, Gilles Barthe, Borja Balle, and Isabel Valera. Model-agnostic counterfactual explanations for consequential decisions. In *International Conference on Artificial Intelligence and Statistics*, pages 895–905. PMLR, 2020.
- [50] Phattara Khumprom and Nita Yodo. A data-driven predictive prognostic model for lithium-ion batteries based on a deep learning algorithm. *Energies*, 12(4):660, 2019.
- [51] Pieter-Jan Kindermans, Kristof T Schutt, Maximilian Alber, Klaus-Robert Muller, Dumitru Erhan, Been Kim, and Sven Dähne. Learning how to explain neural networks: Patternnet and patternattribution. *arXiv preprint arXiv:1705.05598*, 2017.
- [52] EMRE Kiyak. The importance of preventive maintenance in terms of reliability in aviation sector. *Recent Advances in Manufacturing Engineering*, 2011.

- [53] Martin Krasser, Feb 2019. <http://krasserm.github.io/2019/02/23/bayesian-linear-regression/>.
- [54] Alex Kulesza, Ben Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.
- [55] J. Lee, H. Qiu, G. Yu, J. Lin, and Rexnord Technical Services. Bearing data set. NASA Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, 2007. IMS, University of Cincinnati.
- [56] Yaguo Lei, Naipeng Li, Liang Guo, Ningbo Li, Tao Yan, and Jing Lin. Machinery health prognostics: A systematic review from data acquisition to rul prediction. *Mechanical systems and signal processing*, 104:799–834, 2018.
- [57] David Lewis. *Counterfactuals*. John Wiley & Sons, 2013.
- [58] Lingling Li, Pengchong Wang, Kuei-Hsiang Chao, Yatong Zhou, and Yang Xie. Remaining useful life prediction for lithium-ion batteries based on gaussian processes mixture. *PloS one*, 11(9):e0163004, 2016.
- [59] Xiaoyu Li, Lei Zhang, Zhenpo Wang, and Peng Dong. Remaining useful life prediction for lithium-ion batteries based on a hybrid model combining the long short-term memory and elman neural networks. *Journal of Energy Storage*, 21:510–518, 2019.
- [60] Xiaoyu Li, Changui Yuan, and Zhenpo Wang. 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, 2020.
- [61] Datong Liu, Jianbao Zhou, Dawei Pan, Yu Peng, and Xiyuan Peng. Lithium-ion battery remaining useful life estimation with an optimized relevance vector machine algorithm with incremental learning. *Measurement*, 63:143–151, 2015.
- [62] Jian Liu and Ziqiang Chen. Remaining useful life prediction of lithium-ion batteries based on health indicator and gaussian process regression model. *Ieee Access*, 7:39474–39484, 2019.
- [63] Yuefeng Liu, Guangquan Zhao, and Xiyuan Peng. Deep learning prognostics for lithium-ion battery based on ensembled long short-term memory networks. *IEEE Access*, 7:155130–155142, 2019.
- [64] Zhen Liu, Wenjuan Mei, Xianping Zeng, Chenglin Yang, and Xiuyun Zhou. Remaining useful life estimation of insulated gate bipolar transistors (igbts) based on a novel volterra k-nearest neighbor optimally pruned extreme learning machine (vkopp) model using degradation data. *Sensors*, 17(11):2524, 2017.
- [65] Theodoros H. Loutas, Dimitrios Roulias, and George Georgoulas. Remaining useful life estimation in rolling bearings utilizing data-driven probabilistic e-support vectors regression. *IEEE Transactions on Reliability*, 62(4):821–832, 12 2013. ISSN 00189529. doi: 10.1109/TR.2013.2285318.
- [66] David JC MacKay and David JC Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.
- [67] Basim Mahbooba, Mohan Timilsina, Radhya Sahal, and Martin Serrano. Explainable artificial intelligence (xai) to enhance trust management in intrusion detection systems using decision tree model. *Complexity*, 2021:1–11, 2021.
- [68] Bojan Mihaljević, Concha Bielza, and Pedro Larrañaga. Bayesian networks for interpretable machine learning and optimization. *Neurocomputing*, 456:648–665, 2021.
- [69] Thomas Minka. Bayesian linear regression. Technical report, Citeseer, 2000.
- [70] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 607–617, 2020.

- [71] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 607–617, 2020.
- [72] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.
- [73] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.
- [74] Selina SY Ng, Yinjiao Xing, and Kwok L Tsui. A naive bayes model for robust remaining useful life prediction of lithium-ion battery. *Applied Energy*, 118:114–123, 2014.
- [75] Van Duc Nguyen, Marios Kefalas, Kaifeng Yang, Asteris Apostolidis, Markus Olhofer, Steffen Limmer, and Thomas Bäck. A review: Prognostics and health management in automotive and aerospace. *International Journal of Prognostics and Health Management*, 10(2), 2019. ISSN 21532648. doi: 10.36001/ijphm.2019.v10i2.2730.
- [76] Jannie S Nielsen and John D Sørensen. Bayesian estimation of remaining useful life for wind turbine blades. *Energies*, 10(5):664, 2017.
- [77] Ahmad Kamal Mohd Nor, Srinivasa Rao Pedapati, Masdi Muhammad, and Víctor Leiva. Overview of explainable artificial intelligence for prognostic and health management of industrial assets based on preferred reporting items for systematic reviews and meta-analyses. *Sensors*, 21(23):8020, 2021.
- [78] Darian M Onchis and Gilbert-Rainer Gillich. Stable and explainable deep learning damage prediction for prismatic cantilever steel beam. *Computers in Industry*, 125:103359, 2021.
- [79] Marlis Ontivero-Ortega, Agustin Lage-Castellanos, Giancarlo Valente, Rainer Goebel, and Mitchell Valdes-Sosa. Fast gaussian naive bayes for searchlight classification analysis. *Neuroimage*, 163:471–479, 2017.
- [80] Zhenan Pang, Xiaosheng Si, Changhua Hu, Dangbo Du, and Hong Pei. A bayesian inference for remaining useful life estimation by fusing accelerated degradation data and condition monitoring data. *Reliability Engineering & System Safety*, 208:107341, 2021.
- [81] Kyungnam Park, Yohwan Choi, Won Jae Choi, Hee-Yeon Ryu, and Hongseok Kim. Lstm-based battery remaining useful life prediction with multi-channel charging profiles. *Ieee Access*, 8:20786–20798, 2020.
- [82] Meru A Patil, Piyush Tagade, Krishnan S Hariharan, Subramanya M Kolake, Taewon Song, Taejung Yeo, and Seokgwang Doo. A novel multistage support vector machine based approach for li ion battery remaining useful life estimation. *Applied energy*, 159:285–297, 2015.
- [83] Judea Pearl. *Causality: models, reasoning, and inference*, 1980.
- [84] Judea Pearl. *Bayesian networks*. 2011.
- [85] Hai Qiu, Jay Lee, Jing Lin, and Gang Yu. Wavelet filter-based weak signature detection method and its application on rolling element bearing prognostics. *Journal of sound and vibration*, 289(4-5):1066–1090, 2006.
- [86] Emmanuel Ramasso and Abhinav Saxena. Performance benchmarking and analysis of prognostic methods for cmapps datasets. *International Journal of Prognostics and Health Management*, 5(2):1–15, 2014.
- [87] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures*, pages 63–71. Springer, 2004.
- [88] Lei Ren, Li Zhao, Sheng Hong, Shiqiang Zhao, Hao Wang, and Lin Zhang. Remaining useful life prediction for lithium-ion battery: A deep learning approach. *IEEE Access*, 6:50587–50598, 7 2018. ISSN 21693536. doi: 10.1109/ACCESS.2018.2858856.

- [89] Alexander Reutlinger. Is there a monist theory of causal and noncausal explanations? the counterfactual theory of scientific explanation. *Philosophy of Science*, 83(5):733–745, 2016.
- [90] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*, 2016.
- [91] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.
- [92] Christian P Robert, George Casella, and George Casella. *Monte Carlo statistical methods*, volume 2. Springer, 1999.
- [93] Reza Rouhi Ardeshiri and Chengbin Ma. Multivariate gated recurrent unit for battery remaining useful life prediction: A deep learning approach. *International Journal of Energy Research*, 45(11):16633–16648, 2021.
- [94] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.
- [95] B. Saha and K. Goebel. Battery data set. NASA Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, 2007.
- [96] A. Saxena and K. Goebel. Turbofan engine degradation simulation data set. NASA Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, 2008.
- [97] Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 International Conference on Prognostics and Health Management, PHM 2008*, 2008. doi: 10.1109/PHM.2008.4711414.
- [98] Abhinav Saxena, José Celaya, Bhaskar Saha, Sankalita Saha, and Kai Goebel. Evaluating algorithm performance metrics tailored for prognostics. In *IEEE Aerospace Conference Proceedings*, 2009. doi: 10.1109/AERO.2009.4839666.
- [99] Rory Sayres, Ankur Taly, Ehsan Rahimy, Katy Blumer, David Coz, Naama Hammel, Jonathan Krause, Arunachalam Narayanaswamy, Zahra Rastegar, Derek Wu, et al. Using a deep learning algorithm and integrated gradients explanation to assist grading for diabetic retinopathy. *Ophthalmology*, 126(4):552–564, 2019.
- [100] Matthias Seeger. Gaussian processes for machine learning. *International journal of neural systems*, 14(02):69–106, 2004.
- [101] Xiao Sheng Si, Wenbin Wang, Chang Hua Hu, and Dong Hua Zhou. Remaining useful life estimation - a review on the statistical data driven approaches, 2011. ISSN 03772217.
- [102] Gurinder Singh, Bhawna Kumar, Loveleen Gaur, and Akriti Tyagi. Comparison between multinomial and bernoulli naive bayes for text classification. In *2019 International Conference on Automation, Computational and Technology Management (ICACTM)*, pages 593–596. IEEE, 2019.
- [103] Sreenath Sundar, Manjunath C Rajagopal, Hanyang Zhao, Gowtham Kuntumalla, Yuquan Meng, Ho Chan Chang, Chenhui Shao, Placid Ferreira, Nenad Miljkovic, Sanjiv Sinha, et al. Fouling modeling and prediction approach for heat exchangers using deep learning. *International Journal of Heat and Mass Transfer*, 159:120112, 2020.
- [104] Gero Szepannek. How much can we see? a note on quantifying explainability of machine learning models. *arXiv preprint arXiv:1910.13376*, 2019.
- [105] Shengjin Tang, Chuanqiang Yu, Xue Wang, Xiaosong Guo, and Xiaosheng Si. Remaining useful life prediction of lithium-ion batteries based on the wiener process with measurement error. *energies*, 7(2):520–547, 2014.
- [106] Michael E Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of machine learning research*, 1(Jun):211–244, 2001.

- [107] James V. Stone. Bayes' rule: a tutorial introduction to bayesian analysis. *Choice Reviews Online*, 51(06), 2014. ISSN 0009-4978. doi: 10.5860/choice.51-3301.
- [108] Guy Van den Broeck, Anton Lykov, Maximilian Schleich, and Dan Suciuc. On the tractability of shap explanations. *Journal of Artificial Intelligence Research*, 74:851–886, 2022.
- [109] Sahil Verma, Varich Boonsanong, Minh Hoang, Keegan E Hines, John P Dickerson, and Chirag Shah. Counterfactual explanations and algorithmic recourses for machine learning: A review. *arXiv preprint arXiv:2010.10596*, 2020.
- [110] Wlamir Olivares Loesch Vianna, Leonardo Ramos Rodrigues, and Takashi Yoneyama. Aircraft line maintenance planning based on phm data and resources availability using large neighborhood search. In *Annual Conference of the PHM Society*, volume 7, 2015.
- [111] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.
- [112] Dong Wang, Qiang Miao, and Michael Pecht. Prognostics of lithium-ion batteries based on relevance vectors and a conditional three-parameter capacity degradation model. *Journal of Power Sources*, 239: 253–264, 2013.
- [113] Hao Wang and Dit-Yan Yeung. Towards bayesian deep learning: A framework and some existing methods. *IEEE Transactions on Knowledge and Data Engineering*, 28(12):3395–3408, 2016.
- [114] HY Wang, JH Li, and FL Yang. Overview of support vector machine analysis and algorithm. *Application Research of Computers*, 31(5):1281–1286, 2014.
- [115] Jie Wang. An intuitive tutorial to gaussian processes regression. *arXiv preprint arXiv:2009.10862*, 2020.
- [116] Youdao Wang, Yifan Zhao, and Sri Addepalli. Remaining useful life prediction using deep learning approaches: A review. *Procedia manufacturing*, 49:81–88, 2020.
- [117] Geoffrey I Webb, Eamonn Keogh, and Risto Miikkulainen. Naive bayes. *Encyclopedia of machine learning*, 15:713–714, 2010.
- [118] Meng Wei, Hairong Gu, Min Ye, Qiao Wang, Xinxin Xu, and Chenguang Wu. Remaining useful life prediction of lithium-ion batteries based on monte carlo dropout and gated recurrent unit. *Energy Reports*, 7:2862–2871, 2021.
- [119] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.
- [120] Andrew G Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems*, 33:4697–4708, 2020.
- [121] Xindong Wu, Vipin Kumar, Quinlan J. Ross, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus Ng, Bing Liu, Philip S. Yu, Zhi Hua Zhou, Michael Steinbach, David J. Hand, and Dan Steinberg. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1), 2008. ISSN 02193116. doi: 10.1007/s10115-007-0114-2.
- [122] C Xiongzi, Y Jinsong, and T Diyin. Remaining useful life prognostic estimation for aircraft subsystems or components: A review. *IEEE*, 2011. <https://ieeexplore.ieee.org/abstract/document/6037773/>.
- [123] Chu Xu, Timothy Cleary, Daiwei Wang, Guoxing Li, Christopher Rahn, Donghai Wang, Rajesh Rajamani, and Hosam K Fathy. Online state estimation for a physics-based lithium-sulfur battery model. *Journal of Power Sources*, 489:229495, 2021.
- [124] Shen Zhang, Shibo Zhang, Bingnan Wang, and Thomas G. Habetler. Deep learning algorithms for bearing fault diagnosticsa comprehensive review. *IEEE Access*, 8:29857–29881, 2020. doi: 10.1109/ACCESS.2020.2972859.

- [125] Qi Zhao, Xiaoli Qin, Hongbo Zhao, and Wenquan Feng. A novel prediction method based on the support vector regression for the remaining useful life of lithium-ion batteries. *Microelectronics Reliability*, 85:99–108, 2018.
- [126] Zeqi Zhao, Bin Liang, Xueqian Wang, and Weining Lu. Remaining useful life prediction of aircraft engine based on degradation pattern learning. *Reliability Engineering & System Safety*, 164:74–83, 8 2017. ISSN 0951-8320. doi: 10.1016/J.RESS.2017.02.007.
- [127] Xiaoting Zhong, Brian Gallagher, Shusen Liu, Bhavya Kailkhura, Anna Hiszpanski, and T Yong-Jin Han. Explainable machine learning in materials science. *npj Computational Materials*, 8(1):204, 2022.
- [128] Danhua Zhou, Zhanying Li, Jiali Zhu, Haichuan Zhang, and Lin Hou. State of health monitoring and remaining useful life prediction of lithium-ion batteries based on temporal convolutional network. *IEEE Access*, 8:53307–53320, 2020.
- [129] Dong Zhou, Long Xue, Yijia Song, and Jiayu Chen. On-line remaining useful life prediction of lithium-ion batteries based on the optimized gray model gm (1, 1). *batteries*, 3(3):21, 2017.
- [130] Yapeng Zhou and Miaohua Huang. Lithium-ion batteries remaining useful life prediction based on a mixture of empirical mode decomposition and arima model. *Microelectronics Reliability*, 65:265–273, 2016.
- [131] Zhenwei Zhou, Yun Huang, Yudong Lu, Zhengyu Shi, Liangbiao Zhu, Jiliang Wu, and Hui Li. Lithium-ion battery remaining useful life prediction under grey theory framework. In *2014 Prognostics and System Health Management Conference (PHM-2014 Hunan)*, pages 297–300. IEEE, 2014.
- [132] Brahim Zraibi, Chafik Okar, Hicham Chaoui, and Mohamed Mansouri. Remaining useful life assessment for lithium-ion batteries using cnn-lstm-dnn hybrid method. *IEEE Transactions on Vehicular Technology*, 70(5):4252–4261, 2021.