Utilizing SCADA-Log Data to Improve Normal Behavior Models for Wind Turbine Condition Monitoring

Vorgelegt von
Mohamed Samy Elsisi
407923

Von der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades
Master of Science
- M.Sc. -
genehmigte Abschlussarbeit.

Gutachter : Prof. Dr. Klus-Robert Müller
Prof. Dr. Thomas Wiegand
Betreuer : M.Sc. Simon Letzgus

Eidesstattliche Versicherung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Berlin, den 05. Juni 2023 .....

Mohamed Samy Elsisi

Abstract

With the increasing deployment of renewable energy assets, automated condition monitoring solutions are crucial for scaling up wind turbine portfolios. This thesis aims to bridge the gap between SCADA signals and SCADA logs in wind turbine condition monitoring by incorporating SCADA log data into normal behavior models. By mining SCADA log data, subtle patterns and dependencies can be identified which can enhance the accuracy and robustness of the models. Advanced machine learning algorithms, including deep learning and anomaly detection techniques, are employed to detect abnormal behaviors and potential faults. The research contributes to improved fault detection, condition assessment, and predictive maintenance strategies, leading to enhanced operational efficiency and reliability of wind turbines.

Zusammenfassung

Mit dem zunehmenden Einsatz von Windkraftanlagen im Bereich der erneuerbaren Energien sind automatisierte Zustandsüberwachungslösungen von entscheidender Bedeutung für die Vergrößerung des Portfolios von Windturbinen. Diese Arbeit zielt darauf ab, die Lücke zwischen SCADA-Signalen und SCADA-Logs bei der Zustandsüberwachung von Windkraftanlagen zu schließen, indem SCADA-Logdaten in normale Verhaltensmodelle integriert werden. Durch die Auswertung von SCADA-Logdaten können subtile Muster und Abhängigkeiten identifiziert werden, was die Genauigkeit und Robustheit der Modelle verbessert. Fortgeschrittene Algorithmen des maschinellen Lernens, einschließlich Deep Learning und Techniken zur Erkennung von Anomalien, werden zur Erkennung von abnormalem Verhalten und potenziellen Fehlern eingesetzt. Die Forschung trägt zu einer verbesserten Fehlererkennung, Zustandsbewertung und vorausschauenden Wartungsstrategien bei, was zu einer höheren Betriebseffizienz und Zuverlässigkeit von Windkraftanlagen führt.

Contents

Chapter 1

Introduction

Contents

1.1 Background

In 2020, renewable energy represented 22.1% of energy consumed in the EU [European Commission 2023a]. This percentage is expected to increase drastically in the upcoming years with the target, set by the European Commission, of at least 32% by the year 2030 [European Commission 2023b]. With the increasing number of renewable energy assets being deployed every year, automated condition monitoring solutions are needed for operators to be able to scale up their portfolio of assets. Monitoring wind turbine health and performance is critical for early fault detection, maintenance planning, and optimizing wind farm operations.

Several manufacturers have created so-called condition monitoring systems (CMS). These monitor a variety of essential metrics such as drive train vibration, oil quality, and temperatures in some of the main components. Such devices are typically deployed as an addition to the regular wind turbine design. Even though the financial value of CMS's early defect detection has been demonstrated [Yang 2014], their high prices [Yang 2013] have discouraged operators from implementing them. Most of the utility-scale wind turbines come, however, with a Supervisory Control and Data Acquisition (SCADA) system by default. SCADA systems provide significant insights into wind turbine operational behavior. They record various types of data related to the operation and performance of the turbine which can be divided into two main categories: SCADA *signals* and SCADA *logs*. The SCADA signals provide real-time readings collected from various sensors installed in the turbine that reflect the current state of operation in terms of power production, wind speed, rotor speed, component temperatures,... The frequency and the number of signals provided by the SCADA system vary based on the turbine's manufacturer, model, and technology. The SCADA logs, on the other hand, capture alarms and events recorded by the SCADA system in the form of text in a non-fixed frequency. Some approaches utilize both CMS and SCADA data to perform condition monitoring tasks (e.g., [Feng 2011]), however, several other approaches for condition monitoring were developed in recent years that rely solely on the SCADA data, given its

low cost as it normally doesn't require additional hardware installation. For a comprehensive review of different methods for wind turbine condition monitoring using SCADA data, see [Tautz-Weinert 2017].

One of the methods used for condition monitoring using SCADA data is Normal Behavior Modeling (NBM). NBM uses the idea of detecting anomalies from normal operation by empirically modeling a measured parameter, used to reflect the condition of a specific part of the turbine, based on a training phase (usually during a healthy state of the turbine). During operation, the difference between the measured and the modeled/predicted signal is used as an indicator for a possible fault. A difference of 0, with some tolerance, reflects normal conditions, whereas a difference greater or less than a defined threshold reflects changed conditions or failures. Utilizing the signals provided by the SCADA systems in normal behavior models were proven capable of (early) detecting failures in wind turbines. For instance, both Zhang et al. [Zhang 2014] and Bangalore et al. [Bangalore 2015] applied machine learning techniques to SCADA signals data to develop anomaly detection models for fault diagnosis and prediction of the gearbox bearings of wind turbines. The results demonstrate that their condition-monitoring approaches are capable of indicating damage in the components being monitored in advance.

Other methods focus on utilizing logs to detect anomalies in the underlying system. Some of these approaches are general-purpose and can be applied to any computer system. For example, Brown et al. [Brown 2018] developed recurrent neural network (RNN) language models augmented with attention for anomaly detection in system logs that are generally applicable to any computer system and logging source. Similarly, Lyu et al. [Lyu 2019] developed an open-source framework with a set of tools that can be used for automated log parsing, anomaly detection, and impactful problem identification using machine learning. Other approaches specifically focus on leveraging the SCADA logs, especially alarms, to detect anomalies in wind turbines. Rahman et al. [Rahman 2016] use rare sequential pattern mining[1] to find anomalies in SCADA networks. They suggest that because anomalous events occur rarely in a system and the architecture and actions of SCADA systems do not change frequently, some anomalies can be found via uncommon sequential pattern mining. This anomaly detection might be useful for detecting intrusions or erroneous system behaviors. Andrade et al. [Andrade 2022] proposed methods that utilize unsupervised machine-learning clustering techniques to profile alarm patterns, identify abnormal events, and improve the detection of anomalies in industrial processes in the context of network operators and grid outages.

1.2 Motivation & Objectives

While both SCADA signals and logs are leveraged in both areas of study, both approaches are performed in isolation, which leaves operators with only two options:

[1] Rare sequential pattern mining is a data mining technique used to discover infrequent patterns in sequential data.

Either use one condition monitoring system or have to rely on multiple sources of information to make informed operational decisions.

In one of his recent publications, Letzgus [Letzgus 2020] presented methods from the Natural Language Processing (NLP) domain that help to find meaningful representations of SCADA log messages and sequences. His methods encode messages from the SCADA log into vector representation by creating one-hot vectors or applying the Correlated Occurrence Analogue to Lexical Semantic with temporal information (COALS-t) algorithm. This permits and facilitates the successful implementation of machine-learning-based condition monitoring models.

Additionally, Leahy et al. [Leahy 2017] demonstrated a method to label the SCADA signals by identifying stoppages that occurred in the turbines and that are recorded as alarms in the SCADA log. This data was then used to perform fault detection using classification techniques.

Inspired by the work done in the literature, the primary objective of this thesis is to bring both "worlds" of SCADA signals- and log-based anomaly detection together by incorporating SCADA log data into wind turbine condition monitoring models. By mining SCADA log data, we aim to identify subtle patterns, correlations, and dependencies that may contain information about operation conditions or control events which could help improve the accuracy and robustness of normal behavior models in case of events unexplainable by the SCADA signals. Furthermore, the utilization of advanced machine learning algorithms, such as deep learning and anomaly detection techniques, will enhance the detection and prediction capabilities for abnormal behaviors and potential faults. We present different methods to utilize the SCADA logs and incorporate them into machine-learning normal behavior models by generating SCADA log-based vectors that can be used as input features and labels that can be used to filter the SCADA signals being fed into the models. In addition to that, we propose a simple-yet-effective method to visualize relevant alarms and warnings found in the SCADA logs which encourages operators to deal with one system only to monitor the state of the turbines.

The findings of this research are expected to contribute to the field of wind turbine condition monitoring by providing enhanced techniques for normal behavior modeling. The developed models can serve as a basis for effective fault detection, condition assessment, and predictive maintenance strategies, ultimately leading to increased reliability, reduced downtime, and improved operational efficiency of wind turbines.

Chapter 2 will provide an overview of the methodology employed, along with the relevant literature, on wind turbine condition monitoring, SCADA-log data analysis, and machine learning techniques (including data preprocessing and feature extraction) applied to wind turbine condition monitoring. Chapter 3 will showcase the experimental results and discuss the performance of the proposed models. Finally, Chapter 4 will summarize the findings, draw conclusions, and provide recommendations for future research.

Chapter 2

Methods

Here, we extensively explain the methods we used and propose to utilize SCADA log messages in wind turbine normal behavior machine learning models. We start by introducing the dataset used to run the experiments. Then, we define the concept of normal behavior modeling and the machine learning models and their architecture used in this work. Finally, we introduce two different approaches to utilizing SCADA logs with ways of using them as input features in machine learning models, as s filter for the SCADA signals, or to visualize relevant warnings.

2.1 Dataset

In this section, we will describe the dataset used in this work to train, test and validate the models.

We used open-source data published on the *EDP OpenData* web platform [EDP 2018] and that was made available for research purposes. The data was collected from the SCADA systems of five different Vestas wind turbines (Turbine 01, 06, 07, 09 and 11) in the same wind park between the years 2016 and 2017 and is made up of the following four subsets: *Signals, Logs, Failures, and Metmast.* We will, however, only describe three sets since *Metmast* was not used in this work. A full description of the turbines' characteristics (e.g., rated power, cut-in speed, rotor diameter,. . .) and their power curve is documented in Appendix A.

2.1.1 Signals

The *Signals* dataset contains 10-min aggregated data (Mean, STandard Deviation (STD), Minimum (Min), and Maximum (Max) values) collected from the wind turbines' power meters and sensors installed at the major components such as gearbox, generator, and transformer (see Figure 2.1 for a demonstration of a turbine's hardware and the location of major components). These built-in sensors measure quantities such as temperatures, angles, wind and rotational speeds, power production,. . .

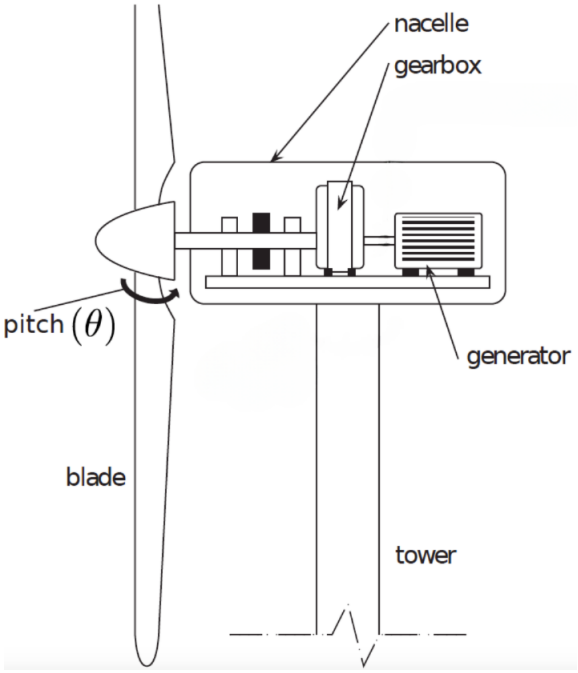


Figure 2.1: Diagram of a wind turbine side view with labeled main components. Figure adapted from [Boersma 2017]

This dataset was the most crucial for this work since it provides information that reflects the status of the turbine operation which is needed to perform SCADA-based automated condition monitoring and predictive maintenance.
Table 2.1 shows some of the 81 signals included in this dataset and Figure 2.2 shows a sample of selected signals collected from Turbine 01 along with some statistics that describe the whole dataset.

Type of signal	Signals
Temperature (°C)	Generator, Generator bearings, Hydraulic group oil, Gearbox oil, Gearbox bearing on the high-speed shaft, Nacelle, High Voltage (HV) transformer, Ambient temperature,...
Production value	Active power in Wh, Reactive power in VArh, Power according to the grid in kW,...
Angle (°)	Blades pitch angle (θ)

Table 2.1: Example signals found in the Signals dataset

Timestamp	Avg Temp in Nacelle (°C)	Avg Ambient Temp (°C)	Avg Temp in Generator Bearings (°C)	Avg Generator RPM	Total active power (Wh)
2016-01-01 00:00:00+00:00	28.00	18.00	41.00	1249.00	4313.00
2016-01-01 00:10:00+00:00	28.00	18.00	41.00	999.70	1735.00
2016-01-01 00:20:00+00:00	29.00	18.00	41.00	774.00	9704.00
2016-01-01 00:30:00+00:00	28.00	18.00	40.00	1257.10	22673.00
2016-01-01 00:40:00+00:00	28.00	18.00	40.00	1257.70	16506.00
2016-01-01 00:50:00+00:00	28.00	18.00	40.00	1328.50	53374.00
Count	104671.00	104671.00	104671.00	104671.00	104671.00
Min	18.00	5.00	18.00	0.00	-4662.00
Mean	29.86	20.05	45.78	1041.27	86189.91
Max	48.00	41.00	93.00	1684.70	371256.00
STD	5.36	5.57	15.99	623.40	109617.91

Figure 2.2: Sample dataset of selected signals from Turbine 01 and some statistics that describe the full dataset

2.1.2 Logs

Some events are logged by the SCADA system in non-fixed intervals. The events recorded by the system are divided into three categories: Alarm log, Warning log, and Operation and System log. According to the VestasOnline Enterprise user manual [Vestas 2016], alarms are system notifications that alert operators to an error scenario that has forced a wind turbine to cease normal operation and transition to one of three operational states: Pause, Stop, or Emergency (one of the following three acknowledgments is needed to resume operation: Local acknowledgment from the controller unit of the turbine, Remote acknowledgment from VestasOnline®, or Automatic acknowledgment), whereas warnings are system messages that indicate an irregularity that requires attention but does not cause the turbine to immediately cease normal operation and exit the Run state.
Operational logs are used to track a system's normal operation and to keep track of events and activities that have occurred. These logs can be used by operators for troubleshooting purposes.
System logs are used to monitor the operation and health of the system's hardware and software components. These logs can be used to identify system problems, such as hardware failures or software faults, and can aid in diagnosing and resolving these problems. Table 2.5 shows samples of logs from each category found in the EDP dataset.

Type of log event	Sample log event
Alarm log	*"High temperature brake disc"* *"High pres offlin:____RPM/ ___°C"*
Warning log	*"Yaw Position is changed: __°"* *"Low Battery Nacelle"*
Operation and System log	*"External power ref.:____kW"* *"GearoilCooler _, gear: ___°C"* *"Pause pressed on keyboard"*

Table 2.2: Sample log events found in the EDP Logs dataset

According to our analysis, we found around 180 different templates of log events in the ERP Logs dataset. A template, as demonstrated in Table 2.5, describes a certain event and may or may not be parameterized. E.g., *"External power ref.:2000kW"* and *"External power ref.:1392kW"* report a similar event (*"External power ref.:____kW"*) but with different parameters (kW production) and, hence, will be considered only once when calculating the total number of unique templates found.

As information is only logged when an event occurs, this dataset does not have a fixed frequency and it's hence difficult to statistically describe all the different log templates found. Since our experiments were focused on generator bearings-related failures, we will describe further some of the log events found from the different categories that are related to the generator component. Those events were utilized in one of our proposed methods (see 2.3.1). In the Operation and System log of Turbine 09, the events *"Gen. int. vent. _, temp:___°C"* (Event class I) and *"Gen. ext. vent. _, temp:___°C"* (Event class II) occurred 1735 and 2026 times, respectively, with the following frequencies:

	Event class I	Event class II
Min frequency	1 second	1 second
Mean frequency	10 hours and 6.47 minutes	8 hours and 39.48 minutes
Max frequency	9 days and 40.82 minutes	9 days and 41 minutes

Table 2.3: Measured frequencies of selected classes of log messages found in the Operation and System log of Turbine 09

Figure 2.3 shows a sample of those events.

TimeDetected	Remark
2016-01-01 05:18:41+00:00	Gen. int. vent. 0, temp: 34°C
2016-01-01 07:05:28+00:00	Gen. int. vent. 1, temp: 50°C
2016-01-01 12:29:33+00:00	Gen. int. vent. 2, temp: 70°C
2016-01-02 07:07:13+00:00	Gen. int. vent. 1, temp: 54°C
2016-01-02 08:44:35+00:00	Gen. int. vent. 0, temp: 34°C
...	...
2017-12-29 04:58:03+00:00	Gen. int. vent. 1, temp: 55°C
2017-12-30 03:39:26+00:00	Gen. int. vent. 0, temp: 35°C
2017-12-30 06:54:34+00:00	Gen. int. vent. 1, temp: 50°C
2017-12-30 08:42:33+00:00	Gen. int. vent. 0, temp: 35°C
2017-12-31 12:10:09+00:00	Gen. int. vent. 1, temp: 50°C

Event: *"Gen. int. vent. __, temp:___◦C"*

TimeDetected	Remark
2016-01-01 02:13:33+00:00	Gen. ext. vent. 1, temp: 49°C
2016-01-01 05:18:41+00:00	Gen. ext. vent. 0, temp: 34°C
2016-01-01 07:05:28+00:00	Gen. ext. vent. 1, temp: 50°C
2016-01-01 12:01:22+00:00	Gen. ext. vent. 2, temp: 65°C
2016-01-02 07:24:35+00:00	Gen. ext. vent. 1, temp: 49°C
...	...
2017-12-30 08:42:33+00:00	Gen. ext. vent. 0, temp: 35°C
2017-12-30 13:47:36+00:00	Gen. ext. vent. 1, temp: 30°C
2017-12-30 19:59:57+00:00	Gen. ext. vent. 0, temp: 28°C
2017-12-31 12:10:09+00:00	Gen. ext. vent. 1, temp: 50°C
2017-12-31 14:36:38+00:00	Gen. ext. vent. 2, temp: 65°C

Event: *"Gen. ext. vent. __, temp:___◦C"*

Figure 2.3: Sample messages found in the Operation and System Log of Turbine T09 from event class I and II, respectively

In the Alarm and Warning log of Turbine 09, the events *"Hot generator _ _ _ °C _ _ _ _ _ _ kW"* (Event class III) and *"High temp. Gen bearing _:_ _ _ °C"* (Event class IV) occurred 899 and 31 times, respectively, with the following frequencies:

	Event class III	Event class IV
Min frequency	2 seconds	1 hour and 13 minutes
Mean frequency	10 hours and 27.31 minutes	91 hours and 16.2 minutes
Max frequency	≈ 281 days	≈ 25 days

Table 2.4: Measured frequencies of selected classes of log messages found in the Alarm and Warning log of Turbine 09

Figure 2.4 shows a sample of those events.

2.1.3 Failures

The Failures dataset contains the history of failures, inspections, or maintenance that occurred in the turbines and was manually recorded by technicians. Each record reports the time of the event, component (e.g., Generator, Hydraulic group,..), and a text description of the failure or event (e.g., "Generator replaced", "Oil leakage in Hub",..).
This dataset was used in backtesting to validate the models' capability of detecting failures early. Table B.1 lists all the recorded failures found in the EDP dataset.

TimeDetected	Remark
2016-07-25 13:16:01+00:00	Hot generator 145°C 0kW
2016-07-25 13:20:40+00:00	Hot generator 145°C 0kW
2016-07-25 13:20:50+00:00	Hot generator 145°C 0kW
2016-07-25 13:21:40+00:00	Hot generator 145°C 0kW
2016-07-25 13:26:00+00:00	Hot generator 145°C 0kW
...	...
2017-08-20 17:52:05+00:00	Hot generator 145°C 0kW
2017-08-20 17:52:28+00:00	Hot generator 145°C 0kW
2017-08-20 17:52:54+00:00	Hot generator 145°C 0kW
2017-08-20 18:00:33+00:00	Hot generator 145°C 0kW
2017-08-20 18:01:09+00:00	Hot generator 145°C 0kW

Event: *"Hot generator ___○C _____kW"*

TimeDetected	Remark
2016-06-07 16:58:42+00:00	High temp. Gen bearing 1: 99°C
2016-06-21 14:45:57+00:00	High temp. Gen bearing 1: 99°C
2016-06-21 16:20:39+00:00	High temp. Gen bearing 1: 99°C
2016-06-21 17:58:34+00:00	High temp. Gen bearing 1: 99°C
2016-06-22 15:01:24+00:00	High temp. Gen bearing 1: 99°C
...	...
2016-09-01 14:01:58+00:00	High temp. Gen bearing 1: 99°C
2016-09-03 15:02:04+00:00	High temp. Gen bearing 1: 99°C
2016-09-28 15:04:56+00:00	High temp. Gen bearing 1: 99°C
2016-09-29 13:30:08+00:00	High temp. Gen bearing 1: 99°C
2016-09-29 19:01:40+00:00	High temp. Gen bearing 1: 99°C

Event: *"High temp. Gen bearing _:___ ○C"*

Figure 2.4: Sample messages found in the Alarm and Warning Log of Turbine T09 from event class III and IV, respectively

2.2 Normal behavior modeling

According to Tautz-Weinert and Watson [Tautz-Weinert 2017], Normal Behavior Modeling (NBM) detects anomalies in normal operation by empirically modeling an observed parameter based on a training phase. Figure 2.5 depicts the concept of model-based condition monitoring. During operation, an anomaly is detected by deducting the value of the modeled signal ($\hat{y}$(t)) from the measured one (y(t)) and comparing the residual (e(t)) with a predefined threshold. If the threshold is exceeded, this signal is labeled as an anomaly.

There are two primary approaches for NBM: Full Signal ReConstruction (FSRC), in which only signals other than the target are utilized to predict the target, and AutoRegressive with eXogenous Input Modeling (ARX), in which previous values of the target are also employed.
Having defined NBM on an abstract level, we demonstrate next the machine learning models we used to generate modeled signals (y(t)) from input signals (x(t)) and then explain the anomaly detection approach we used. Given that the structure of the available signals (e.g., the number of signals and their frequency) varies based on the turbine's model, manufacturer and sensors installed, we defined the dataset used in this work in the previous section.

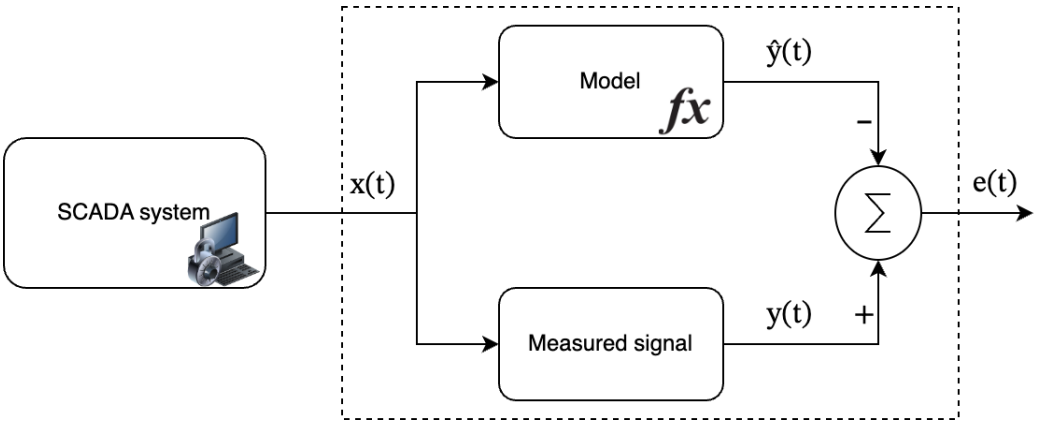


Figure 2.5: NBM with the input signals from the SCADA system (x(t)), measured signal (y(t)), modeled signal ($\hat{y}$(t)), and resulting error (e(t))

2.2.1 Machine Learning in NBM

From a wide range of machine learning model types, NBM focuses on regression models [Fahrmeir 2021]. Regression models are part of the supervised learning family, where the algorithm is trained on labeled data and the input features are mapped to corresponding output labels. As opposed to classification models, where the algorithm predicts *classes*, a regression model predicts numerical *values* (dependent variables) from the input features (independent variables).
According to Tautz-Weinert and Watson [Tautz-Weinert 2017], there are mainly

three types of NBM regression models used in the research field: *Linear and polynomial models*, *Artificial Neural Networks* and *Fuzzy Systems*. Given their simplicity, we used linear models in the early phases of this work. Later on, we started using ANNs for their capability of capturing non-linear dependencies in the data. We define these two types of models in detail in the next subsections. A fuzzy system [Jang 1997] is an artificial intelligence system that employs fuzzy logic [Zadeh 1965]. Fuzzy logic is a mathematical framework for dealing with uncertainty and imprecision. The input and output variables in a fuzzy system are represented by fuzzy sets, which are collections of values with degrees of membership rather than tight boundaries. The associations between the input variables and the output variables are then specified using fuzzy rules. These rules are often represented as "if-then" statements, with the "if" section defining the input conditions and the "then" part defining the output actions. The training of fuzzy systems was not within the scope of this work. However, we propose testing our methods on them in the future works section (see 4.2.5).

2.2.1.1 Linear regression

Sir Francis Galton proposed the idea of linear regression in 1894 [Galton 1894]. Linear regression is used for analyzing the linear relationship between one or more independent variables and a dependent variable. The dependent variable must be continuous, whereas the independent variables can be continuous or categorical. For a dependent variable Y and a set of n independent variables X_1 through X_n, the linear regression equation is defined as follows:

$$Y = m_1X_1 + m_2X_2 + ... + m_nX_n + C \tag{2.1}$$

where m_1 through m_n and C are constants. Figure 2.6 shows an example of linear regression for a single independent variable.

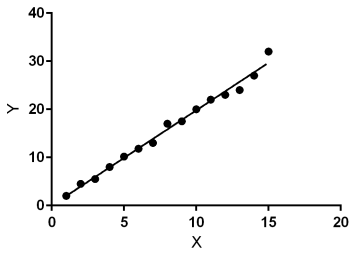


Figure 2.6: Example linear regression with one dependent variable: $Y = mX + b$, where m and b are constants

When the relationship between the dependent variable and the independent variables is assumed to be linear, linear regression is usually used. Linear regression is easy to use and understand, and it can be used to make predictions or find relationships between variables.

In the example of normal behavior modeling for a wind turbine component, the dependent variable can be defined as the component's temperature and the independent variables as a set of weather and turbine conditions measures (e.g., wind speed, ambient temperature, production value, other components' temperatures,..) that have either a direct or indirect effect on the target component. NBM in its most basic form is based on linear or polynomial models [Tautz-Weinert 2017]. Garlick et al. [Garlick 2009] employed a linear ARX model to detect generator bearings failures in bearing temperature measurements. Schlechtingen and Santos [Schlechtingen 2011] developed an FSRC linear condition monitoring model for the generator bearings' temperature.
Although multiple linear regression models were also shown capable of fitting the data with high accuracy in many other applications (e.g., [Wang 2019]), they are, by definition, not capable of capturing more complex non-linear dependencies. In addition to that, linear regression may not be appropriate when there are a significant number of independent variables. Artificial Neural Networks may be a better approach in these situations [Lee 2017].

2.2.1.2 Artificial Neural Networks

Artificial Neural Networks (ANNs) are computational models that are inspired by the structure and function of biological neural networks in the brain [Haykin 1999]. They are made up of interconnected nodes (artificial neurons) that process and send data. Pattern recognition, computer vision, natural language processing, and robotics have all made extensive use of ANNs (for a comprehensive review of deep learning and neural networks, see [Schmidhuber 2015], [Goodfellow 2016]). An artificial neuron, also known as a perceptron, is the fundamental building unit of a neural network. It is a mathematical function that accepts one or more input values and outputs a single value [Rosenblatt 1958]. The input values are weighted, and the neuron applies an activation function to the total of the weighted inputs. The output value is subsequently passed on to the network's other neurons. The activation function determines the neuron's output based on the input value(s) and weights. For a set of inputs X_1 through X_n, weights w_1 through w_n and activation function f, the output of a perceptron Y is calculcated as follows:

$$Y = f(\sum_{i=1}^{n} w_i X_i) \tag{2.2}$$

The sigmoid function, the rectified linear unit (ReLU) function, and the hyperbolic tangent function are examples of common activation functions. Figure 2.7 shows a diagram of a perceptron.

In the context of deep learning, an ANN consists of one input layer, one output layer, and one or more *hidden* layers.
A hidden layer is a layer of neurons that are not connected directly to either the input or output layers. It is referred to as "hidden" because its neurons are not visible to the outside world, implying that its calculations are not directly apparent

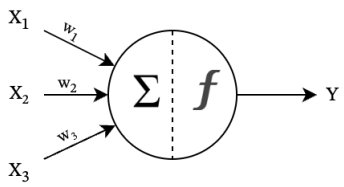


Figure 2.7: Example perceptron with three inputs

from input or output.
Information goes from the input layer, through one or more hidden layers, and then to the output layer in a *feedforward* neural network, which is a type of ANN. Each layer of neurons runs computations on the input data and sends the results to the next layer. The hidden layers extract and alter information from input data that can be utilized to make predictions or choices.
The number of hidden layers in an ANN is a hyperparameter that can be tuned during the training process. The number of hidden layers and neurons in each layer is determined by the task's complexity, the amount of accessible data, and the required level of accuracy.
After obtaining better results with it compared to linear regression (see Experiment 3.1), we decided to train the normal behavior models on a feed-forward neural network having the architecture shown in Fig. 2.8 using ReLU (firstly introduced by Fukushima [Fukushima 1980]) as an activation function in the hidden layers and a linear activation function (input = output) in the output layer. The output of the ReLU activation function is zero for any negative input, and for any positive input, the output is equal to the input.

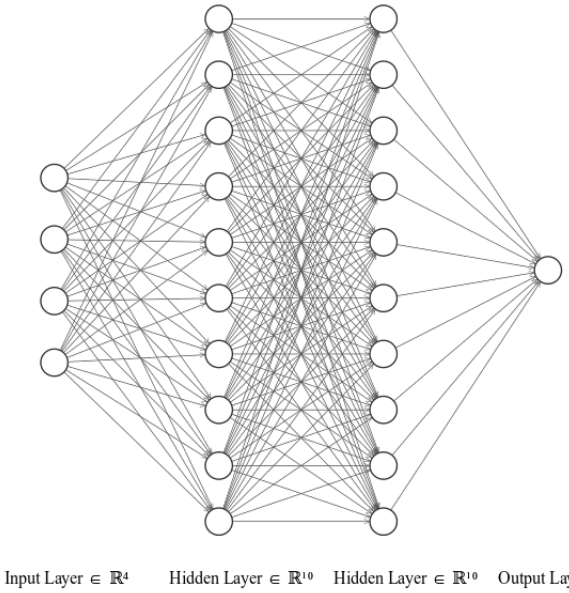


Figure 2.8: Architecture of normal behavior neural network model used in this work. *The input layer shape will vary based on the experiment and the number of input features.*

2.2.2 Reconstruction-based Anomaly Detection

The main idea behind training and improving normal behavior models is to allow our models to detect anomalies more accurately. An anomaly is defined as an occurrence or observation that differs from what is expected, usual, or typical. There are several techniques in the research field that can be used to detect anomalies. Ruff et. al. [Ruff 2021] provided a comprehensive review of different anomaly detection techniques which they divided into two groups: Shallow and Deep Methods. Shallow anomaly detection refers to the use of traditional machine learning algorithms such as One-Class Support Vector Machine (OCSVM), Principal Component Analysis (PCA), Isolation Forest, Local Outlier Factor (LOF) and more, whereas deep anomaly detection refers to the use of deep learning algorithms such as Autoencoder, Variational Autoencoder (VAE), Generative Adversarial Networks (GANs) and other techniques.
In this work, we focus on Reconstruction-based anomaly detection techniques utilized by normal behavior models, which can be considered either shallow or deep depending on the architecture of the normal behavior model. In reconstruction-based anomaly detection, by comparing the observed data to a reference set, such as historical data or a pre-defined model, anomalies can be found. Positive and negative anomalies are also possible. In the context of wind turbine condition monitoring and when mainly monitoring temperatures of the system, we focus on positive anomalies because a component that is overheating—due to wear and tear, oil leakage, faulty fan, or other reasons—is likely to fail. There is, however, no unified method in the research field to identify a data point as an anomaly. Brandao et al. ([Brandao 2010], [Brandao 2015]) used a fixed value of the mean absolute error as an anomaly threshold in their gearbox and generator fault detection model, even though this number was particular and no longer valid following maintenance procedures. Schlechtingen and Santos [Schlechtingen 2011] used daily average prediction errors in generator bearings temperature to trigger alarms. Zhang and Wang [Zhang 2014] used a hard threshold of 1.5°C for the residual to identify anomalies in the main shaft rear bearing temperature. Bangalore and Tjernberg ([Bangalore 2015], [Bangalore 2013b], [Bangalore 2013a]) used a Mahalanobis distance to compare residual and target distributions from the training period to find anomalies in gearbox bearings temperatures. The Mahalanobis distance was averaged over three days and compared to a training result-defined threshold.

As there is no standard way to identify anomalies in temperatures in the context of condition monitoring for wind turbines using normal behavior models, we experimented with several methods to do that and, finally, decided to set the anomaly threshold to the maximum prediction error seen in the training period. This way it is guaranteed that the normal behavior models will not label any data point in the training dataset as an anomaly (complying with the assumption that the turbine was operating in a healthy state during the training phase of the model) while having the threshold dynamically set based on the setup (e.g., input and output features,

training period, condition of the turbine during the training phase,...) without having to incorporate any domain knowledge related to the specific component to-be-monitored. This also helped better compare different architectures of normal behavior models and the effect of incorporating the proposed log features, not only in terms of prediction accuracy but also in terms of the quality and frequency of anomalies identified (a model that better fits the training data will have a tighter anomaly threshold).

2.2.2.1 Anomaly vs Alarm

In our approach, we differentiate between *Anomalies* and *Alarms*. An anomaly is a data point that deviates from "normal", whereas an alarm is a proactive way of communication that gets triggered when the operator's attention is urgently needed. The reason why we propose not to send an alarm every time an anomaly is detected by the system is that we want our system to limit the number of false alarms as they are costly and counterproductive.
As opposed to anomalies, which are tracked on a 10-min basis, we base alarms on daily events. If the number of anomalies found from the start of a day up until a given point in time exceeds a certain threshold, an alarm is triggered. We set the *alarm threshold* to the maximum number of anomalies that occurred per day during the training period when using an *anomaly threshold* set to the 99^{th} percentile of the distribution of the training prediction errors. To summarize, an alarm can be defined as an anomaly in the number of system anomalies found per day.

2.2.3 Feature selection

The way the independent variables are chosen is usually done by measuring the correlation coefficients between available features in a dataset and the target feature and then selecting the features having a high correlation coefficient. Depending on the problem setting, other features can be also considered based on domain knowledge, especially when dealing with a mechanical system as in the case of this work. A good example of this would be the incorporation of the ambient temperature measurement as an input feature—even if it does not highly correlate with the target feature—to make sure that the model generalizes when trying to predict a component's temperature throughout the year, by considering the effect of seasonality (temperatures are expected to be higher in summer than in winter).
In this work, we selected input features based on both domain knowledge and correlation coefficients. We used Kendall's method to measure the rank correlation [Kendall 1938]. In contrast to Pearson's correlation coefficient, Kendall's rank correlation can capture both linear and non-linear dependencies between two variables by measuring the monotonic relationship [El-Hashash 2022]. Kendall's correlation factor (τ) is calculated as follows:

$$\tau = \frac{C - D}{C + D} \tag{2.3}$$

where C is the number of concordant pairs and D is the number of discordant pairs. Concordant pairs are observations in which the rankings of both variables increase or decrease in the same direction, whereas discordant pairings are observations in which the ranks of both variables increase or decrease in opposing ways. The value of τ can range from -1 to 1, with -1 indicating a perfect negative relationship, 0 indicating no relationship, and 1 indicating a perfect positive relationship.
As a result of our analysis, alongside the generated log embeddings/features (discussed in Section 2.3), the following sensor signals were used as input features to the generator bearings' normal behavior and condition monitoring models: *Average generator Revolutions Per Minute (RPM)*, *Average temperature in the nacelle (°C)*, *Total active power (Wh)* and *Average ambient temperature (°C)*. For the power curve condition monitoring models (discussed in 2.3.1.2), the *Average windspeed within average timebase (m/s)* and *Average ambient temperature (°C)* were the only features used to predict the power production values of the turbine.

2.3 Log analysis

In this section, we will describe the different approaches we propose to utilize SCADA log messages and incorporate them into normal behavior models.
Most machine-learning architectures can only work with vector-shaped numerical inputs. Given that there are limited resources in the research field on how to generate numerical vectors from wind turbine SCADA system logs (as discussed in the introduction section), we introduce two methods that were proven capable of not only generating embeddings for machine-learning normal behavior models but also improving their accuracy (see chapter 3): a domain-knowledge-based method and utilizing an open-source framework for analyzing log data called LogPAI. We will discuss each method in detail.

2.3.1 Domain-knowledge-based method

2.3.1.1 Creating log embeddings

We scanned through the different log messages available in the dataset looking for information that reflects the turbine state and might help the normal behavior model fit the data more accurately. Since many normal behavior models monitor mechanical parts' temperature to mimic their thermal behavior, we narrowed the search down to operation and system logs that reflect events causing a change of temperature in major components. We, then, ended up with a category of logs that show the states of internal or external ventilators of some components (see table 2.5). Being parts of the cooling systems of major components, fans or ventilators affect the components' temperature significantly. Therefore, log messages related to these are promising candidates.

Log text template	Log text sample
Gen. ext. vent. _, temp:___°C	Gen. ext. vent. 2, temp:65°C
Gen. int. vent. _, temp:___°C	Gen. int. vent. 1, temp:50°C
HV Trafo. vent. _, temp:___°C	HV Trafo. vent. 0, temp:2°C
Nac.vent._, nac/gear:___/___°C	Nac.vent.3, nac/gear:43/ 54°C

Table 2.5: Example log text templates with sample texts

Indeed, our analysis showed a clear relationship between the state of a ventilator and the temperature of its turbine component. As shown in Fig. 2.9, at low temperatures of the generator bearings, the internal ventilator will switch off. The bearings will then heat up which, in turn, causes the ventilator to turn on which cools the bearings down, and so on.

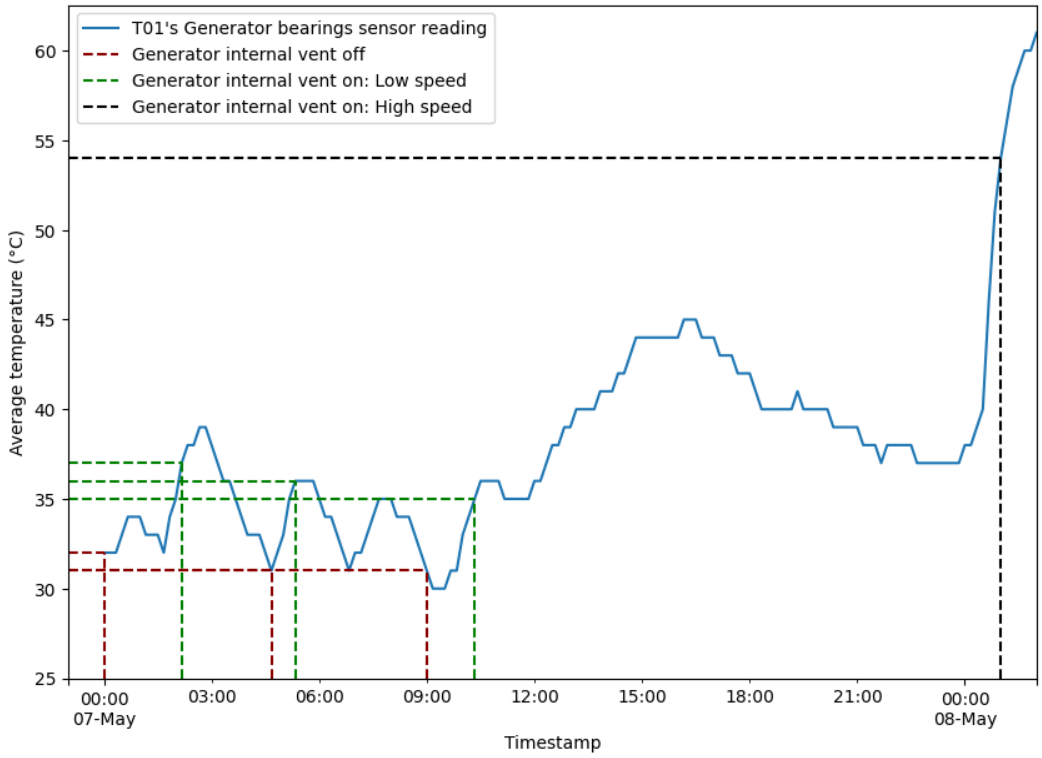


Figure 2.9: Generator internal vent control signals and their effect on the generator bearings' temperature

Analyzing the log texts of interest (e.g., *Gen. ext. vent. 2, temp:65°C*), we deduce that they provide three pieces of information: 1. Description of the ventilator (e.g., *Gen. ext. vent.*), 2. State of the ventilator (*0, 1, 2 or 3*), 3. The temperature of the turbine component the ventilator is installed in (e.g., *65°C*). Since the component temperature is regularly provided as a SCADA sensor signal, we decided to focus on the other two parts of the log messages. Our proposed method simply filters log messages containing the word "vent." and creates a new feature for every ventilator found in the data having its state as a value.
In contrast to the signals data fixed rate of occurrence (10 min), the generated log embeddings have an inconsistent frequency (the SCADA system creates a new log entry only when a ventilator changes states). We join both datasets by taking the value of the last occurrence in the log embeddings vector within a 10-minute window relative to a signal reading. Gaps in the log feature columns in the resulting dataset are then filled by propagating the last valid observation forward to the next valid (a ventilator has the same state as long as it hasn't changed). Figure 2.10 demonstrates an example of the join operation described.

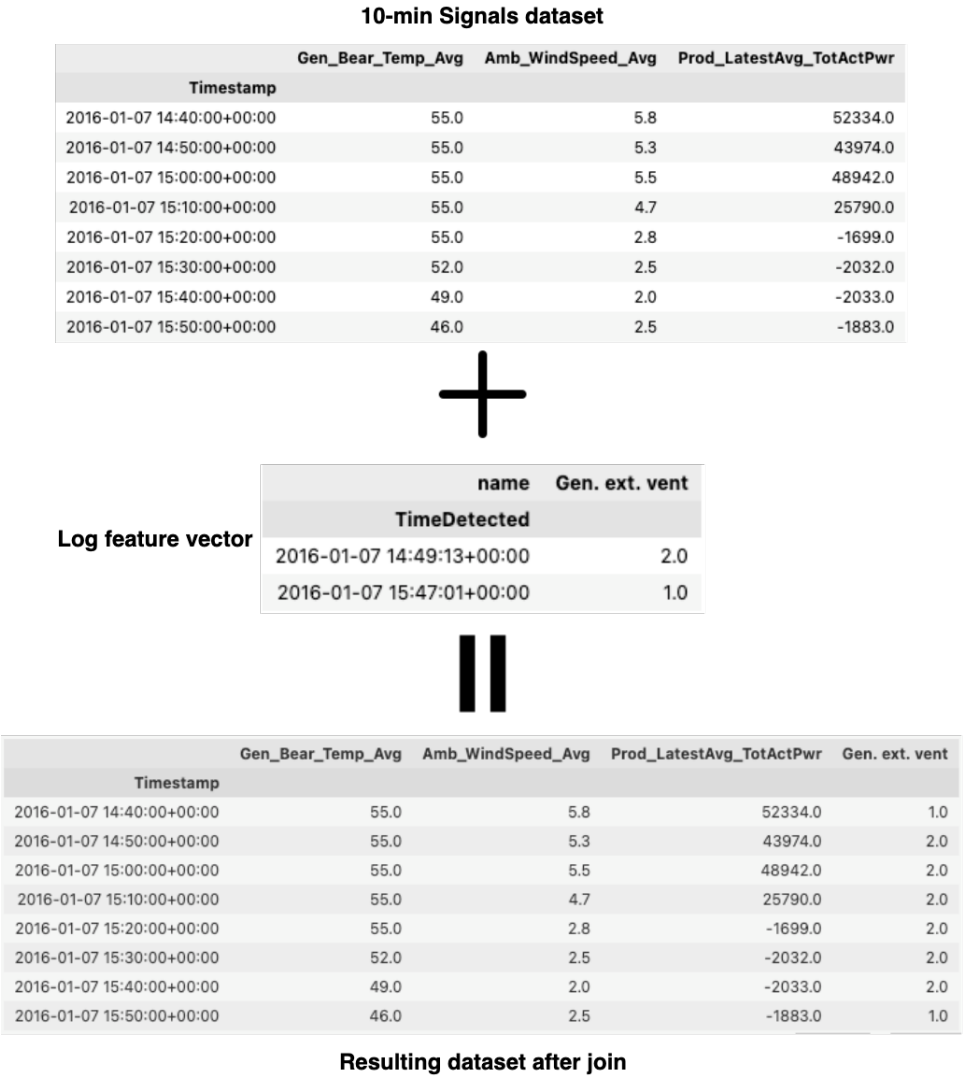


Figure 2.10: Demonstration of the join operation between the signals 10-min dataset and a log embeddings vector

Gaps occurring on the first records of the resulting dataset are, however, not filled by this method. These are then filled in the next step by taking an inverse value of the first-occurring non-empty value. This inverse value simply represents an estimation of the previous state of a ventilator: e.g., 0 or 2 if the first recorded value was equal to 1. There are cases where the state of a ventilator changes more than once in a 10-minute window. We handle those by calculating *time-based weighted averages* in windows of 10 minutes and replace the values—representing the ventilator states—of those individual timeframes with the weighted averages. The Time-based

Weighted Average (TWA) is calculated as follows:

$$TWA = \sum_{i}^{n} v_i * w_i \tag{2.4}$$

where n is equal to the number of occurrences in a (10-min) time window, v_i value (ventilator state) at the i$^{\text{th}}$ occurrence, and w_i is the weight assigned to the i$^{\text{th}}$ occurrence and is calculated, for a fixed time window (in seconds) TW and the number of seconds since the start of the time window at the i$^{\text{th}}$ occurrence S_i, as follows:

$$w_i = \begin{cases} \frac{S_i}{TW} & \text{if } n = 0 \\ \frac{S_i - S_{i-1}}{TW} & \text{otherwise} \end{cases} \tag{2.5}$$

Measuring the Kendall correlation factor between the generated log embeddings and all the signals of the turbines, we found that for every temperature signal, there is at least one log embedding feature that, on average, highly correlates ($Rank > 0.5$) with it.

2.3.1.2 Data labeling and filtering

In this approach, we developed a method to improve SCADA-data-driven wind turbine power curve models (for a comprehensive review of the various modeling techniques used to predict the power output of wind turbines and their applications in wind-based energy systems, see [Sohoni 2016]). As shown in Figure 2.11.a, there are times when the wind speed is above the turbine's cut-in speed[1] (4 m/s in this case), though the turbine's blades won't spin or will spin at lower rates than normal. This could happen due to several reasons, including grid curtailment[2], the turbine being in a service state and/or manually stopped by the operator, or the turbine is simply underproducing due to technical failures. From a condition monitoring perspective, being informed that the turbine is underperforming when in an operating state is crucial as it's a sign of a potential failure in one or more of the turbine's components. That is why, we introduce this method that focuses on isolating the data points collected from the turbine's SCADA system when it's in operative mode. Handling the effect of grid curtailment on the turbine's performance is beyond the scope of this work, it's, however, discussed further in Subsection 4.2.3 of the Future Works.

We start by extracting the log messages that report the current state of operation; namely, logs containing one of the following regular expressions:

[1] A turbine's cut-in speed is the wind speed at which the turbine's blades will start spinning, as provided by the manufacturer in the turbine's datasheet

[2] Grid curtailment refers to the intentional reduction or restriction of power generation from renewable energy sources due to limitations in the capacity of the electricity grid to accommodate the generated electricity.

- *"Run"*,
- *"(Stop|Pause).*kW.*RPM"*, or
- *"new SERVICE state"*

The SCADA signals are then merged with the extracted log messages, using the same join strategy described in 2.10, and booleanly labeled based on the following logic:

- Turbine's state of operation = *"Run"*, if the log message contains the expression *"Run"* or *"new SERVICE state: 0"*
- Turbine's state of operation = *"Stop"*, if the log message contains the expression *"(Stop|Pause).*kW.*RPM"* or *"new SERVICE state: 1"*

Figure 2.11.b shows a sample power curve after labeling the data points based on the proposed method. As shown, most of the time when the turbine isn't spinning, while the wind speed exceeds the cut-in speed, the data points are labeled in red and would be filtered out. In addition to that, some periods when the turbine is in a transition phase from a running to a stopping state will be filtered out. Those are also times when the turbine isn't necessarily underperforming but is gradually slowing down until it comes to a complete stop. One could argue that this proposed method could be simply replaced by filtering the data based on the turbine's rotor rotational speed (if speed equals zero, then the turbine is not in operative mode). However, doing so wouldn't cover the transitional phases of the turbine, but simply filter out the data at times when the turbine wasn't spinning regardless of the reason. This also includes times when the turbine isn't spinning due to a technical failure while in operation or due to low wind speeds, which are of high interest when it comes to monitoring the turbine's state.

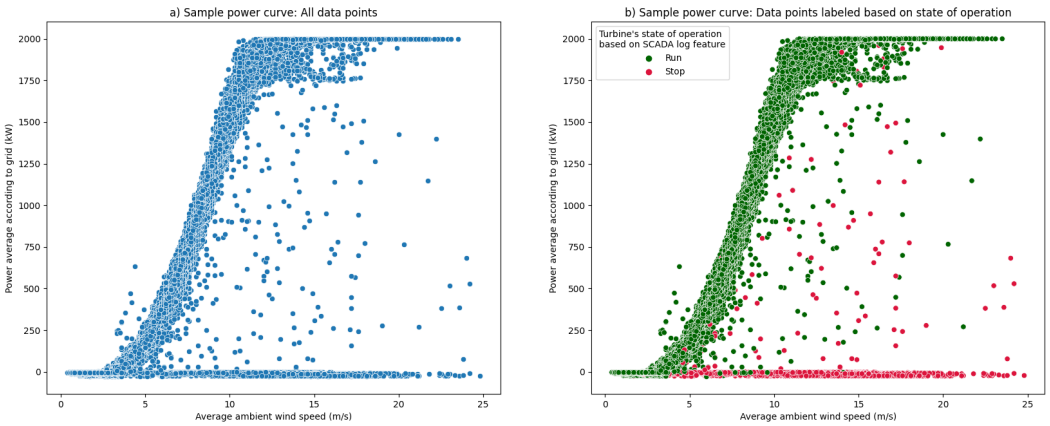


Figure 2.11: Turbine 01 power curve with log-feature-based labels

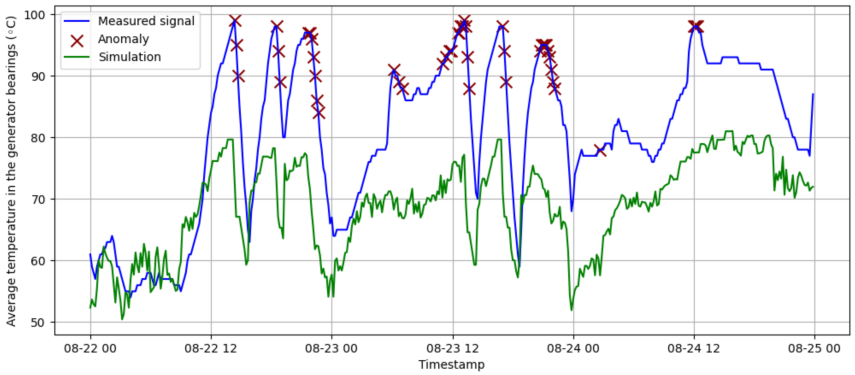
The log-based feature we introduced showed a clear improvement in the accuracy of power curve models (see experiment 3.4) when used to filter the data being input to the normal behavior model (using data points having *"Run"* as the state of operation exclusively). It also led to better results compared to simply filtering by the rotor speed.

2.3.1.3 Visualization of warnings

Here, we introduced a straightforward yet effective way of visualizing (e.g., on an operation dashboard) messages from the Alarm and Warning logs that are relevant to faults detected or predicted by normal behavior models and that are worth being reported to the operators.
When the normal behavior model detects a fault in a certain turbine component, the SCADA logs are queried for messages reporting high temperatures in this component during the same time window (e.g., last hour, last 12 hours, current day,...). If found, these messages could be included in the system reports that get sent to the operators to inform them of the detected failure. This gives more visibility and credibility to the detected/predicted failure by the system.
Here, we use the following regular expression to filter relevant warnings and alarms in the SCADA log:

$$((?=.*\text{Hot})(?=.*\{\}))|((?=.*\text{High temp})(?=.*\{\})) \tag{2.6}$$

where the placeholder ({}) holds the abbreviation of the target major component as found in the signals dataset (e.g., "Gen" for Generator and "Gear" for Gearbox). Figure 2.12 shows an example of how these messages could be displayed on a simple operation dashboard.




```
---------Alarms & Warnings found in the SCADA log---------
------------------------2016-08-22------------------------
Hot generator 146°C      0kW @ 2016-08-22 13:50:21+00:00
 High temp. Gen bearing 1: 99°C @ 2016-08-22 14:15:07+00:00
 High temp. Gen bearing 1: 99°C @ 2016-08-22 18:24:55+00:00

------------------------2016-08-23------------------------
High temp. Gen bearing 1: 99°C @ 2016-08-23 13:12:42+00:00
 High temp. Gen bearing 1: 99°C @ 2016-08-23 16:53:19+00:00

------------------------2016-08-24------------------------
Hot generator 146°C      0kW @ 2016-08-24 15:16:29+00:00
 Hot generator 145°C      0kW @ 2016-08-24 15:56:55+00:00
 Hot generator 145°C      0kW @ 2016-08-24 15:57:00+00:00
 Hot generator 145°C      0kW @ 2016-08-24 15:57:19+00:00
 Hot generator 145°C      0kW @ 2016-08-24 15:57:21+00:00
 Hot generator 145°C      0kW @ 2016-08-24 15:57:25+00:00
```

Figure 2.12: Example dashboard showing actual temperatures of the generator bearings (measured signal) against simulated signals by a condition monitoring model (simulation). Anomalous data points are marked in red and relevant logs found are displayed at the bottom

2.3.2 Utilizing LogPAI

LogPAI (Log Analytics Powered by AI) is a study project and open-source platform for analyzing and managing log data [Lyu 2019]. Tsinghua University researchers started the project, which focuses on developing efficient algorithms and tools for log analysis, anomaly detection, and log data visualization. LogPAI includes a complete suite of log analysis and processing tools such as *Logparser, Loglizer, and Logreduce*. These applications can assist users in preprocessing and parsing raw log data, detecting anomalies and patterns, and summarizing log data concisely and understandably. We decided to utilize LogPAI's Logparser ([Zhu 2018], [He 2016a]) and Loglizer [He 2016b] to respectively parse and create numerical features from SCADA logs in a more generic and automated way. While Logparser extracts structured information from unstructured log data generated by software systems automatically, Loglizer includes many feature extraction approaches for capturing the relevant information in log data and transforming it into a feature vector representation suitable for machine learning-based log analysis.

2.3.2.1 Preprocessing of logs using Logparser

Logparser identifies and extracts relevant log events from raw logs ([Zhu 2018], [He 2016a]). It takes a log template-based method, grouping similar log messages to automatically learn and identify common structures and patterns in log messages. It then creates log templates that represent the unique structure of the log data and utilizes them to parse and extract structured information from fresh log messages. Logparser is also adaptable, allowing users to create their log templates to meet their requirements.
From the list of parsers available in the toolkit (e.g., LenMa [Shima 2016], LFA [Nagappan 2010], LogSig [Tang 2011],...), we decided to use *Drain* [He 2017] given that it is an online parser, which means it can process the SCADA logs in real-time as they are generated. The Drain algorithm groups similar log messages together and extracts structured events from them using a clustering-based approach by applying a two-stage approach: log parsing and event extraction. In the first stage, Drain employs a regular expression-based log parser to split raw log messages into a set of log keys and their related values. The log keys are unique identifiers for each type of log message, whereas the log values are the specific information connected with each log message. It then uses a clustering-based approach in the second stage to group similar log messages together and extracts structured events from them. Finally, Drain creates a template for each cluster that summarizes the relevant information contained in the log messages once the log messages have been clustered. The way this is done is by comparing the log keys and values of each log message using a similarity metric and assigning them to the best appropriate cluster based on their similarity scores. The similarity metric $simSeq$ used by the algorithm is

defined as follows:

$$simSeq = \frac{\sum_{i=1}^{n} equ(seq_1(i), seq_2(i))}{n}, \tag{1}$$

where seq_1 represents a log message and seq_2 represents the log event of a log group/cluster. $seq(i)$ is the ith token of the sequence, n is the log message length of the sequences and equ is defined as follows:

$$equ(t_1,\ t_2) = \begin{cases} 1 & \text{if } t_1 = t_2 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Applying Drain on the SCADA log data at hand by specifying its log format "$<TimeDetected>,<TimeReset>,<UnitTitle>,<Content>,<UnitTitleDestination>$", we get output structured log data (see Fig. 2.13 for an example) that the Loglizer can process to generate numerical features.

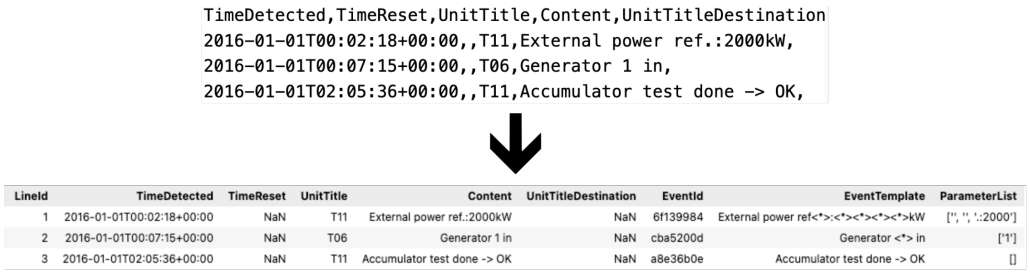


Figure 2.13: Sample raw logs and their corresponding structured logs after being parsed by Logparser (Drain)

P.S. We filter out all log messages having the template *"External power ref.:___kW"* before passing the logs to the Logparser as they weren't found to provide relevant information regarding the state of the turbine in the application of condition monitoring of its generator (bearings). In addition to that, given their high volume and frequency, they were found to worsen the quality of log embeddings generated by Loglizer when applied to our models.

2.3.2.2 Creating log embeddings using Loglizer

Loglizer is a machine learning-based technique to log analysis created by Tsinghua University academics [He 2016b]. Several critical components comprise the technique, including feature extraction, feature selection, and anomaly detection. Loglizer collects numerous features from log messages in the first stage, such as token frequencies, Term Frequency-Inverse Document Frequency (TF-IDF) scores, and structural features. These features are then used to train a machine-learning model to learn the normal behavior of the system. In the second stage, Loglizer uses a feature selection algorithm to identify the most essential features. In the final

step, Loglizer uses a machine learning algorithm to detect anomalous log messages. The algorithm is trained on a labeled dataset that contains both normal and anomalous log messages. During testing, the algorithm uses the learned model to predict whether each new log message is normal or anomalous based on the extracted features.

Loglizer's *Feature Extraction* component supports various feature extraction techniques, such as Bag-of-Words, TF-IDF, and Word2Vec, to capture the essential information contained in log data. We utilized the Loglizer feature extractor, using TF-IDF [Sparck Jones 1972] for term weighting, to generate numerical features from the parsed logs.

In natural language processing and information retrieval, TF-IDF is frequently used to assess how relevant a term is to a particular document within a collection of documents [Das 2021]. The TF component counts the number of times a phrase appears in a document and gives terms with more frequent occurrences a larger weight. The IDF component calculates the rarity or frequency of each term across all documents in a collection and gives less frequent terms a larger weight. The equation for calculating the TF-IDF score of a term in a document can be expressed as follows:

$$TF - IDF(t, d, D) = TF(t, d) * IDF(t, D) \tag{2.7}$$

where t is a term in a document, d is a document in a collection of documents, D is the entire collection of documents, $TF(t, d)$ is the term frequency of term t in document d (the number of times term t appears in document d) and $IDF(t, D)$ is the inverse document frequency of term t across all documents in collection D, defined as:

$$IDF(t, D) = log(\frac{N}{df(t)}) \tag{2.8}$$

, where N is the total number of documents in D and $df(t)$ is the number of documents in D that contain the term t.

We applied the feature extractor on the parsed *Event IDs*—given that they uniquely describe the different events recorded in the SCADA log—and used the generated features as input features to our normal behavior models. We also experimented with generating additional features by applying the Loglizer feature extractor on the parsed *Parameter lists*; those are used to parametrize the event templates which are, in turn, used to generate the Event IDs. However, our models didn't show any improvements as a result of incorporating those additional features. Hence, we decided to stick to the Event ID-based generated features only in our experiments.

Chapter 3

Experiments

In this chapter, we describe a set of experiments we ran to quantitatively and qualitatively measure the effect of incorporating the log embeddings introduced into normal behavior models when applied to both *healthy* and *faulty* turbines. For a normal behavior model monitoring a component of a turbine, we considered this turbine *faulty* if a failure was reported, in the failures dataset, related to this specific component of this turbine. It is considered *healthy* if no failures, relating to this turbine's component, were reported.
To compare different models in an identical setup, we use the following metrics:

- **Root Mean Squared Error ($RMSE$)**: It is a commonly used metric to evaluate the performance of a predictive model or an estimator. The $RMSE$ is calculated as the square root of the mean of the squared differences between the predicted ($y_{predicted}$) and actual values (y_{actual}), or as follows:

$$RMSE = \sqrt{\frac{1}{n} * \sum_{i}^{n} (y^i_{predicted} - y^i_{actual})^2} \tag{3.1}$$

  where n is the number of data points in a dataset. The RMSE is expressed in the same units as the original data. As a rule of thumb: The lower the RMSE, the better the model fits the data.

- **Numbers of anomalies and alarms detected during a given period**: We use these numbers to measure the capability of a model to detect/predict a failure. The number of anomalies detected reflects the total number of anomalous data points, whereas the number of alarms detected counts only the number of operation days of a turbine where the system notified the operator of a potential failure by sending an alarm (if the number of anomalies detected in a day exceeds a certain threshold, as explained in 2.2.2.1). When compared to another model, we consider a model more *capable* of predicting failures if it detects more anomalies and/or sends more alarms during the time of abnormal operation of a faulty turbine given that it reported no anomalies or alarms during the normal operation of the same turbine. In other words, we compare these metrics between models when applied to the test data of a faulty turbine, assuming that the data used to train these models was collected from the turbine in a period when it was operating in a healthy state; hence no anomalies should be detected in this period.

- **Timestamps of the first anomaly detected and alarm sent**: Used to compare the capability of different models to early-detect failures, when applied to a faulty turbine. The earlier the first anomaly is detected or the first alarm is sent the better.

All the condition monitoring normal behavior models used in our experiments were trained to monitor the generator bearings of a turbine (i.e., having the average temperature in the generator bearings as a target), whereas the power curve normal behavior models monitor the average power production of a turbine according to the grid (in kW). The input features used are listed in 2.2.3.

3.1 Benchmark NBM architecture

In the early stages of this work, we trained linear regression models due to their lightweight and low computational power needed. Knowing that they are incapable of capturing non-linear relationships in the data, we assumed that the linear regression models would be outperformed by feed-forward neural networks when it comes down to fitting the signals data of a healthy turbine. To test this hypothesis and select a specific architecture to be used as a benchmark NBM model in other experiments, we did this simple experiment to compare the $RMSE$ scores of both models.
This experiment was conducted on a healthy turbine (Turbine 01). Both models were trained on signals data collected between 01/09/2016 and 30/08/2017 and tested on data collected between 01/09/2017 and 31/12/2017.
As shown in Table 3.1, the feed-forward network outperformed the linear regression model—as expected—and was used as a baseline in all the other experiments.

Metric	Linear regression	Feed-forward network
Training RMSE	5.29	4.86
Testing RMSE	5.80	5.78

Table 3.1: Experiment I results: RMSEs measured and used to compare the benchmark models

3.2 Effect of incorporating log embeddings into NBM for condition monitoring when applied to a healthy turbine (T01)

3.2.1 Setup

This experiment aims to quantitatively and qualitatively measure the effect of incorporating SCADA-log-based embeddings into the baseline normal behavior model when applied to a healthy turbine. (see method 2.3.1 and 2.3.2.2)

We ran this experiment three times: one time using the baseline normal behavior model (i.e., using only SCADA signals as input features), repeated once after adding the log embeddings generated based on domain knowledge as input features (denoted as *Model-DK*), and another time after incorporating LogPAI-generated log embeddings (denoted as *Model-PAI*). The models were trained on data collected between 01/09/2016 and 30/08/2017 and tested on data collected between 01/09/2017 and 31/12/2017.

As shown in Table 3.2, all log embeddings generated based on domain knowledge highly correlate with the target feature and were hence used as input features in *Model-DK* in addition to the selected SCADA signals. As for the log embeddings generated using LogPAI, only a few features were found to relatively highly correlate (correlation factor greater than 0.3 or less than -0.3) with the target feature (see Table 3.3) and were selected, in addition to the selected SCADA signals, as input features in *Model-PAI*.

Feature	Correlation
Generator external ventilator	0.713057
Generator internal ventilator	0.730726
High-voltage transformer ventilator	0.513700
Nacelle ventilator	0.514112

Table 3.2: Measures of Kendall's correlation between the domain knowledge-based log embeddings and the target feature in Turbine 01

Feature	Correlation
LogPAI Feature 1	-0.335313
LogPAI Feature 2	-0.320031
LogPAI Feature 3	0.015902
LogPAI Feature 4	-0.220749
LogPAI Feature 5	0.083943
LogPAI Feature 6	-0.303429
LogPAI Feature 7	0.191848
LogPAI Feature 8	-0.045460
LogPAI Feature 9	-0.077507
LogPAI Feature 10	-0.157537
LogPAI Feature 11	-0.102636
LogPAI Feature 12	0.018344
LogPAI Feature 13	0.244219
LogPAI Feature 14	-0.129601
LogPAI Feature 15	0.361669
LogPAI Feature 16	-0.010470

Table 3.3: Measures of Kendall's correlation between the log embeddings generated based on the event IDs using the LogPAI framework and the target feature in Turbine 01. Selected features are highlighted.

At this stage, the main goal is to test whether those highly-correlating features would improve the baseline model, or, they provide redundant information that could be indirectly deduced from the signal features.

3.2.2 Results

3.2.2.1 Performance

As reported in Table 3.4, the incorporated log embeddings, both in *Model-DK* and *Model-PAI*, improved the RMSE scores of the models. This shows that those features provided additional information to the model that wasn't available in the selected SCADA signals, which shows that our proposed methods were indeed capable of retrieving valuable information from the SCADA logs.

Metric	***Baseline***	***Model-DK***	***Model-PAI***
Training RMSE	4.864	4.087	4.617
Testing RMSE	5.784	5.195	5.607

Table 3.4: Experiment results: RMSEs measured and used to compare between the *Baseline* model, *Model-DK*, and *Model-PAI* when applied to Turbine 01

3.2.2.2 Anomaly detection

Applying the models to the testing dataset, 12, 15, and 8 data points were labeled anomalous by the *Baseline* model, *Model-DK*, and *Model-PAI*, respectively. Those anomalous data points were spread over 4 days for all three models. Whether to consider these data points as false positives or not is arguable. One could consider them as false positives based on the premise that the turbine was operating in a healthy state. In this case, *Model-PAI* would be ranked highest in terms of anomaly detection. However, these data points could indeed be anomalous and be signaling a fault that might happen in the future. Here, *Model-DK* would show a higher sensitivity to anomalous data points.
Given that the dataset provided doesn't include data from the following years (2018+), we weren't able to test the different possibilities and will leave the interpretation open to the reader.

No alarms were reported in all three models. This result shows that none of the anomalies detected was considered critical enough for the operator to be notified, which aligns with the assumption that the turbine is healthy. A summary of the discussed results is shown in Table 3.5.

Metric	***Baseline***	***Model-DK***	***Model-PAI***
#Anomalous data points	12	15	8
..firstly detected on	04/12/2017	15/11/2017	04/12/2017
..lastly detected on	27/12/2017	27/12/2017	27/12/2017
#Anomalous days	4	4	4
..of which warning logs were found	0	0	0
#Alarms sent	0	0	0

Table 3.5: Summary of experiment results

3.3 Effect of incorporating log embeddings into NBM for condition monitoring when applied to a faulty turbine (T09)

3.3.1 Setup

This experiment aims to quantitatively and qualitatively measure the effect of incorporating SCADA-log-based embeddings into the baseline normal behavior model when applied to a faulty turbine (see method 2.3.1, 2.3.1.3, and 2.3.2.2).
For this turbine (T09), several failures relating to the generator bearings were reported starting on 07/06/2016 and ending on 17/10/2016 by replacing the generator bearings (see Table B.1). In this case, not only do we want to measure the performance of our models fitting the data during the presumably healthy (training) period, but also we want to test their capability of detecting the recorded failures early and notifying the operator in cases of major anomalies in the operation of the turbine's component.
We ran this experiment three times: one time using the baseline normal behavior model (i.e., using only SCADA signals as input features), repeated once after adding the log embeddings generated based on domain knowledge as input features (denoted as *Model-DK*), and another time after incorporating LogPAI-generated log embeddings (denoted as *Model-PAI*). The models were trained on data collected between 01/01/2016 and 15/02/2016 and tested on data collected between 16/02/2016 and 18/10/2016. Due to the limits of the dataset at hand, only two and a half months of data could be used to train the models since, according to our analysis, this is the period where the turbine was assumably still operating in healthy conditions. Due to the shortage in the training dataset, some features in the testing dataset were found to be out-of-distribution (see Fig. 3.1). This fact made the interpretation of the results a bit more challenging, it, however, helped test the robustness of the different models' architectures.

As shown in Table 3.6, all log embeddings generated based on domain knowledge highly correlate with the target feature and were hence used as input features in *Model-DK* in addition to the selected SCADA signals. As for the log embeddings generated using LogPAI, only two features were found to relatively highly correlate (correlation factor greater than 0.3 or less than -0.3) with the target feature (see Table 3.7) and were selected, in addition to the selected SCADA signals, as input features in *Model-PAI*.

At this stage, the main goal is to test whether the generated log embeddings would also improve the performance of the baseline model when applied to a faulty turbine. In addition to that, we want to test the effect of these features on improving the model's capability of failure early detection. For that, we start by measuring the RMSE scores to compare the models' performances and then analyze the anomalies detected in the testing period and the yielded alarms.

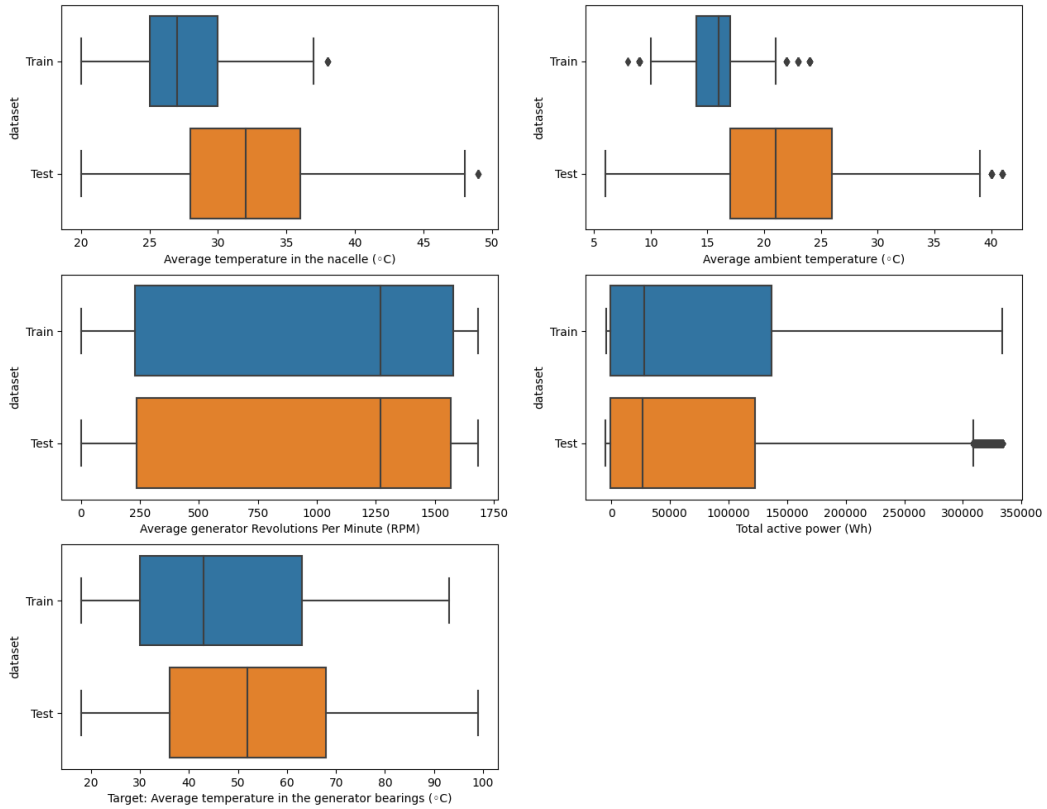


Figure 3.1: Boxplots showing the distribution of signals collected from Turbine 09 sensors used to train and test the models

3.3.2 Results

3.3.2.1 Performance

Again, it was shown that the incorporated log embeddings, both in *Model-DK*, and *Model-PAI*, improved the RMSE scores of the models (see Table 3.8).

Metric	***Baseline***	***Model-DK***	***Model-PAI***
Training RMSE	8.562	8.081	8.386
Testing RMSE	9.084	8.936	9.029

Table 3.8: Experiment results: RMSEs measured and used to compare between the *Baseline* model, *Model-DK*, and *Model-PAI* when applied to Turbine 09

3.3.2.2 Anomaly & Early fault detection

In total, 48, 217, and 236 anomalies were reported by the *Baseline* model, *Model-DK*, and *Model-PAI*, respectively. Those anomalous data points were spread over 29 days for the *Baseline* model, 42 days for *Model-DK*, and 53 days for *Model-*

Feature	Correlation
Generator external ventilator	0.505995
Generator internal ventilator	0.656839
High-voltage transformer ventilator	0.500316
Nacelle ventilator	0.480353

Table 3.6: Measures of Kendall's correlation between the domain knowledge-based log embeddings and the target feature in Turbine 09

Feature	Correlation
LogPAI Feature 1	-0.279547
LogPAI Feature 2	0.026935
LogPAI Feature 3	-0.320831
LogPAI Feature 4	0.019996
LogPAI Feature 5	-0.296431
LogPAI Feature 6	0.018961
LogPAI Feature 7	0.231462
LogPAI Feature 8	-0.214312
LogPAI Feature 9	0.172929
LogPAI Feature 10	0.324340
LogPAI Feature 11	-0.131364
LogPAI Feature 12	-0.011957
LogPAI Feature 13	-0.080340
LogPAI Feature 14	-0.158273
LogPAI Feature 15	-0.126988
LogPAI Feature 16	-0.074183

Table 3.7: Measures of Kendall's correlation between the log embeddings generated based on the event IDs using the LogPAI framework and the target feature in Turbine 09. Selected features are highlighted.

PAI. From these anomalous days, relevant alarm and warning logs were found (see 2.3.1.3 on how these logs are retrieved) on 18 days for both *Model-DK* and *Model-PAI* and only 14 days for the *Baseline* model. In addition to that, both *Model-DK* and *Model-PAI* detected the first anomaly one day earlier than the *Baseline* model (16/02/2016 versus 17/02/2016).

This result only shows the major effect of the log embeddings—whether the ones generated by applying domain knowledge or using LogPAI—on the behavior of the model during unhealthy conditions of the turbine. The way we interpret this result is as follows: By adding supplementary information regarding the turbine's control signals found in the SCADA log, the model could provide a better simulation of the turbine in healthy conditions and, hence, detect abnormalities more easily during unhealthy states of operation.

Using the method described in 2.2.2.1, only *Model-DK* sent alarms with a total of six alarms starting on 19/02/2016 and ending on 23/08/2016. This is due to the fact that the peak number of anomalies detected per day was significantly higher for *Model-DK* (between 13 and 28) compared to *Model-PAI* (12) and the *Baseline* model (3 only). On one hand, we believe this result is due to the limited sample size used to train the model. However, on the other hand, it shows the higher robustness

of *Model-DK*: the limited training dataset was enough for the model to report higher numbers of anomalies detected per day during the validation period, high enough for it to trigger several alarms.
A summary of the discussed results is shown in Table 3.9.

Metric	***Baseline***	***Model-DK***	***Model-PAI***
#Anomalous data points	48	217	236
..firstly detected on	17/02/2016	16/02/2016	16/02/2016
..lastly detected on	29/09/2016	06/10/2016	11/10/2016
#Anomalous days	29	42	53
..of which warning logs were found	14	18	18
#Alarms sent	0	6	0
..firstly on		19/02/2016	
..lastly on		23/08/2016	

Table 3.9: Summary of experiment results: Anomaly and early detection of faulty generator bearings which were replaced on 17/10/2016

3.4 Effect of log-based data filtering on NBM for power curve modeling

3.4.1 Setup

The main goal of this experiment is to test the effectiveness of method 2.3.1.2 in improving normal behavior models having power production as the target, also known as power curve models. To do so, we trained a normal behavior model on data collected from Turbine 01 sensors between September 2016 and August 2017 and tested it on data collected between September 2017 and December 2017. We used only the ambient wind speed (m/s) and ambient temperature (°C) signals as input features and the average power production according to the grid (kW) as the target feature.
We trained three different models having the same architecture but using different datasets: Using all the raw signals collected; denoted as *Baseline*, using raw signals when the turbine was spinning only (i.e., rotor speed greater than zero); denoted as *Model-Spin*, and using raw signals when the turbine was operating in a "Run" state only based on the SCADA log; denoted as *Model-Run*.

Figure 3.2 shows how the data points are labeled based on the filters explained. One could see that *Model-Spin* doesn't consider all data points where the turbine isn't spinning (red and yellow points) including when it's in a "Run" state but the wind speed hasn't reached the cut-in speed yet (yellow points). Although *Model-Run* will also not consider some of the data points where the turbine is not spinning (red points), it will still consider the yellow points. In addition to that, it will exclude data points where the turbine is in a "Stop" state but still spinning (most probably

still in the transition phase after gradually applying the brakes). The *Baseline* is simply trained on all the data points.

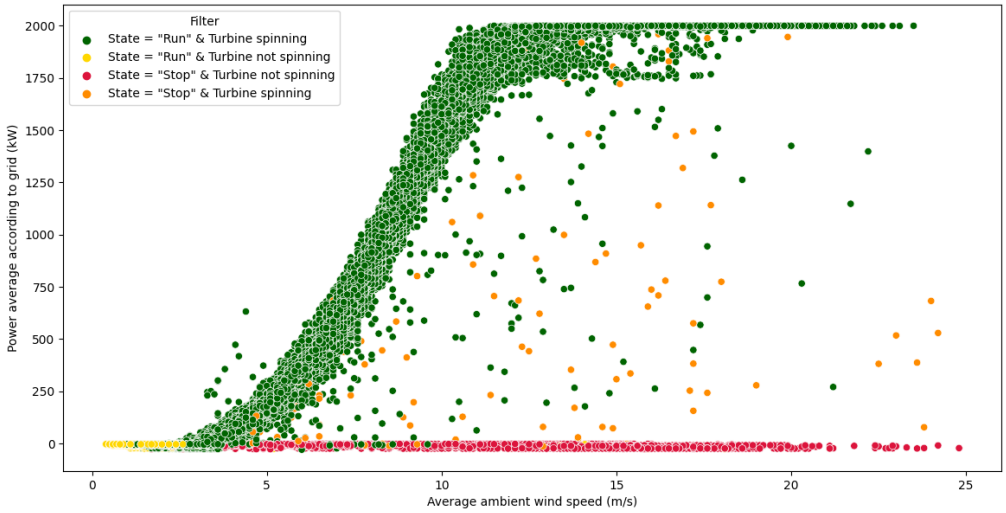


Figure 3.2: Turbine 01 power curve: Data points labeled based on different filters

3.4.2 Results

Here, we simply compared the models' performances by measuring their RMSE scores. Comparing the models' anomaly detection and performance evaluation capabilities wasn't in the scope of this work, as we mainly focused on analyzing the generator-related condition monitoring models. We did however reach these results while analyzing different strategies to utilize the SCADA logs in the context of SCADA-based condition monitoring and decided they were worth documenting.

Table 3.10 shows the measured RMSE scores of the three models. As shown, filtering the data improved the model's performance drastically. The filtering based on the turbine's state retrieved from the SCADA logs (*Model-Run*) yielded the best results in this setting. This shows that the generated labels provide valuable insights into the turbine's state and can be used instead of a simple filter by the rotor's speed, especially if the latter isn't available as a SCADA signal for a given turbine. Figure 3.3 also shows the better power curve fit both *Model-Spin* and *Model-Run* have compared to the *Baseline* model when applied to the testing dataset.

Metric	***Baseline***	***Model-Spin***	***Model-Run***
Training RMSE	179.200	61.707	51.309
Testing RMSE	139.972	59.988	50.251

Table 3.10: Experiment results: RMSEs measured and used to compare between the *Baseline* model, *Model-Spin* and *Model-Run* when applied to Turbine 01

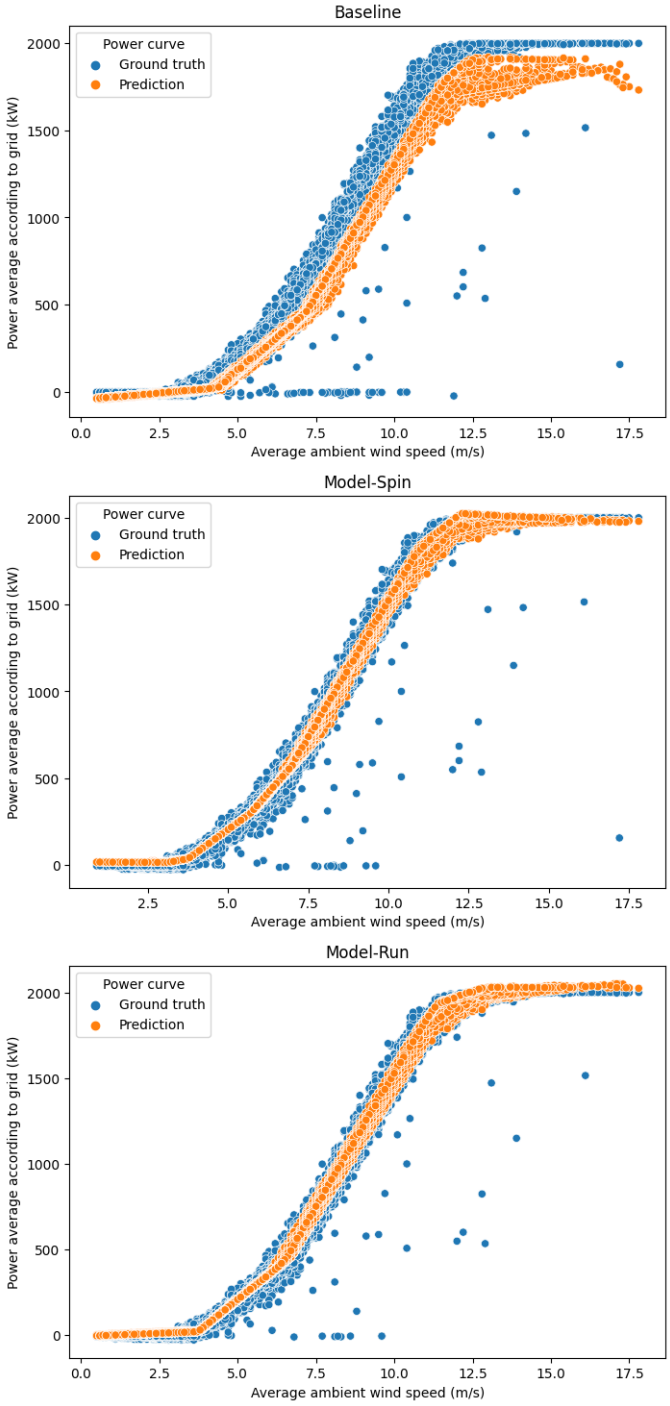


Figure 3.3: True versus predicted power curves of the *Baseline* model, *Model-Spin*, and *Model-Run*, respectively, when applied to the testing dataset of Turbine 01

Chapter 4

Conclusions and Future Works

4.1 Conclusions

In this study, titled "Utilizing SCADA-Log Data to Improve Normal Behavior Models for Wind Turbine Condition Monitoring," we aimed to leverage SCADA event log data to enhance the performance of condition-monitoring machine learning models, improve anomaly detection, and provide valuable insights for wind turbine operation and maintenance. This section presents the key findings and contributions of the research.

Firstly, we successfully generated log embeddings based on the SCADA event log, which served as informative input features for the condition monitoring machine learning models. The utilization of these log embeddings led to significant improvements in the models' fitting to the target feature, enhancing their ability to detect anomalies. The incorporation of SCADA-log data as input features demonstrated its effectiveness in improving the performance of wind turbine condition monitoring systems. Two methods were employed to generate log embeddings as input features for machine learning models.
The first method utilized domain knowledge, incorporating expert insights and understanding of wind turbine operations to develop the log embeddings. This approach capitalized on the specific characteristics and patterns present in the SCADA event log data, allowing for an accurate representation of normal behavior.
The second method involved the utilization of an open-source framework known as LogPAI, which facilitated the generation of log embeddings. By employing this framework, the study leveraged existing resources and techniques, benefiting from the advancements in log analysis and embedding generation.

Furthermore, we implemented a method for labeling and filtering data used in power curve models, leveraging the SCADA event log. By incorporating the insights gained from the log data, we were able to optimize the accuracy and reliability of power curve models, contributing to more accurate predictions of wind turbine performance and improved decision-making processes in wind farm operations.

In addition to the improvements in machine learning models and power curve modeling, we also developed a simple operation dashboard that visualizes relevant warnings and alarms extracted from the SCADA event log. This dashboard provides a comprehensive overview of the turbine's operational status and facilitates the timely identification of potential issues. By highlighting relevant events and trends, it empowers operators to make informed decisions and take proactive measures to ensure the optimal performance and maintenance of wind turbines.

Overall, the findings of this study demonstrate the significant potential of utilizing SCADA event log data for wind turbine condition monitoring. The integration of log embeddings as input features enhances the performance of machine learning models, while the utilization of SCADA data in power curve modeling improves the accuracy of performance predictions. Additionally, the developed operation dashboard allows for effective visualization and monitoring of relevant alarms and warnings. These contributions collectively contribute to the advancement of wind turbine monitoring and maintenance practices, ultimately leading to increased operational efficiency and reduced downtime.

The insights gained from this research lay the foundation for future studies in the field of wind turbine condition monitoring. Further exploration of the potential of SCADA-log data integration, optimization of machine learning models, and the development of advanced visualization techniques can foster continued advancements in the industry.

4.2 Future Works

The preceding research has provided valuable insights into the utilization of SCADA-log data for improving normal behavior models in wind turbine condition monitoring. However, several avenues for future exploration and development could contribute to further advancements in this field. This section presents potential areas of focus for future research, aiming to expand the current knowledge and address the existing gaps. By considering these future works, researchers and practitioners can continue to enhance the accuracy, robustness, and applicability of condition-monitoring techniques for wind turbines.

4.2.1 Expanding Dataset Size for Enhanced Model Training and Generalization

An important aspect to consider in future research is the expansion of the dataset used for training the condition monitoring models. While the dataset utilized in this work provided valuable insights and yielded promising results, its size was inherently limited. Therefore, one potential avenue for future exploration involves gathering a larger and more diverse dataset encompassing a wider range of wind turbine operational scenarios.

By incorporating a larger dataset, the machine learning models can potentially capture a more comprehensive representation of the normal behavior patterns and variations present in wind turbine operations. This increased data volume would allow for more robust training, improved generalization, and enhanced anomaly detection capabilities.

To obtain a larger dataset, collaborative efforts among industry stakeholders, research institutions, and wind farm operators may be necessary. Sharing

anonymized SCADA-log data across multiple sites and leveraging international collaborations could facilitate the creation of a more extensive and representative dataset.

Furthermore, the availability of a larger dataset would enable the evaluation and comparison of the proposed methods and techniques on a broader scale. Robustness and reliability assessments could be conducted across multiple wind farm installations, considering variations in turbine models, geographical locations, and environmental conditions. This would provide a more comprehensive understanding of the efficacy and potential limitations of the developed models and algorithms.

Therefore, future research endeavors should prioritize the acquisition and utilization of larger datasets to refine and validate the proposed condition monitoring models. By encompassing a broader range of operational scenarios, the models can be further optimized and their performance can be thoroughly evaluated, ultimately enhancing their practical applicability in real-world wind turbine condition monitoring systems.

4.2.2 Unveiling Deeper Insights through Domain-Knowledge-Based Analysis of SCADA-Log Data

Moreover, the domain-knowledge-based method employed for extracting log embeddings presents an opportunity for further analysis and extraction of additional insights from the SCADA-log data. By delving deeper into the domain knowledge and refining the embedding extraction process, it is possible to uncover more intricate patterns and correlations within the data. Future research should consider exploring advanced techniques, such as feature engineering or domain-specific preprocessing, to enhance the information captured by the log embeddings and extract deeper insights that can contribute to a more comprehensive understanding of wind turbine behavior and performance.

4.2.3 Incorporating Grid Curtailment Information to Enhance Labeling and Filtering in Power Curve Models

In addition, the domain-knowledge-based method employed for labeling and filtering data points in the application of power curve models can be further enhanced by incorporating valuable information about grid curtailment, which can be extracted from the SCADA log. By considering grid curtailment events, characterized by instances where the turbine's power output is deliberately limited due to grid constraints or other external factors, the labeling and filtering process can be refined to account for these specific conditions.

This additional information from the SCADA log about grid curtailment events can contribute to a more precise identification and classification of data points in

the power curve models. For example, the log event "External power ref.:1392kW" shows that the turbine was curtailed and can only produce up to 1392 kW (instead of 2000 kW). By factoring in the influence of grid curtailment, the models can better capture the true normal behavior of the wind turbine under varying operational conditions, resulting in improved accuracy and reliability.

4.2.4 Extending Methodology to Monitor Other Wind Turbine Components

Additionally, it is crucial to explore the applicability of the developed methods to components beyond the generator bearings. While this study focused on the condition monitoring of generator bearings, there are other critical components within wind turbines that require monitoring for efficient and reliable operation. Future research should aim to test and validate the proposed methods on various turbine components, such as gearbox, pitch system, yaw system, or tower structural elements, to assess their effectiveness and adaptability in detecting anomalies and predicting failures across the entire turbine system.

4.2.5 Potential Application of Fuzzy Systems as Normal Behavior Models

Furthermore, as part of future work, the exploration of fuzzy systems as alternative normal behavior models holds significant promise. Fuzzy systems have demonstrated their effectiveness in capturing and representing complex and uncertain relationships in various domains [Tautz-Weinert 2017]. By incorporating fuzzy logic and linguistic variables, these systems can provide a flexible and interpretable framework for modeling the normal behavior of wind turbines based on SCADA-log data.

The utilization of fuzzy systems in wind turbine condition monitoring could offer advantages such as robustness to data variations and the ability to handle imprecise or incomplete information. By integrating fuzzy systems into the existing framework, researchers and practitioners can further improve the accuracy and reliability of anomaly detection algorithms, enhancing the overall performance of condition monitoring systems.

Exploring the feasibility and potential benefits of incorporating fuzzy systems as normal behavior models represents an intriguing avenue for future research in this field. This would involve designing appropriate fuzzy rule sets, membership functions, and inference mechanisms tailored specifically to the characteristics of SCADA-log data and wind turbine operations. Such investigations could deepen our understanding of the strengths and limitations of fuzzy systems in this context and contribute to the advancement of wind turbine condition monitoring techniques.

Therefore, investigating the application of fuzzy systems as normal behavior models stands as an important direction for future research, with the potential to further enhance the effectiveness and reliability of condition monitoring approaches in the wind energy industry.

4.2.6 Applicability of Methods to Other SCADA-Enabled Systems

Finally, the methods developed in this study hold the potential for application beyond the specific domain of wind turbine condition monitoring. Given the widespread use of SCADA systems across various industries, these methods can be tested and adapted to other systems that employ SCADA for data collection and monitoring purposes.

Appendix A

Wind Turbine Characteristics

Here, we list the characteristics of the turbines whose data was used in our experiments. The following table shows the technical characteristics of these Vestas turbines:

Power	Rated power (kW)	2,000
	Cut-in wind speed (m/s)	4
	Rated wind speed (m/s)	12
	Cut-out wind speed (m/s)	25
	Wind class (IEC)	IEC II (7.5 - 8.5 m/s)
Rotor	Diameter (m)	90
	Swept area (m^2)	6,362
	Number of blades	3
	Rotor speed, max (rpm)	14.9
	Tip speed (m/s)	70
	Power density 1 (W/m^2)	314.4
	Power density 2 (m^2/kW)	3.2
Gearbox	Type	Planetary/spur
	Stages (m/s)	3
Generator	Type	Asynchronous
	Speed, max (rpm)	2,016
	Voltage (V)	690
	Grid frequency (Hz)	50
Tower	Hub height (m)	80
	Type	Steel tube
	Shape	Conical
	Corrosion protection	Painted

Table A.1: Wind turbine characteristics

The following graph demonstrates the turbines' power curve provided by the manufacturer:

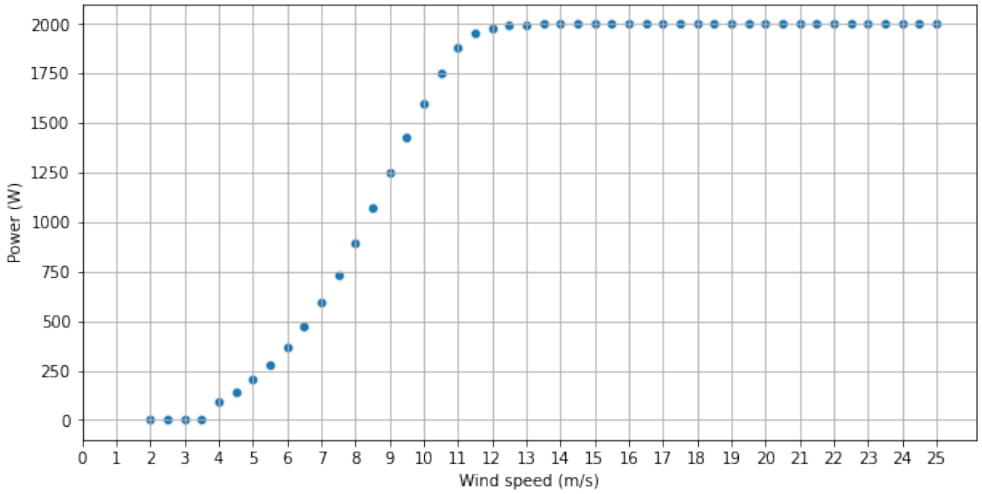


Figure A.1: Power curve of the Vestas turbines at an air density of 1.225 kg/m^3

Appendix B

Recorded Failures

Here, we show the full list of failures detected in all five turbines and recorded by the technicians or the service company. These failures were found and documented upon on-site inspections and were used to validate the capability of the normal behavior models to predict them early (enough) before actually occurring. Table B.1 shows the full list of failures.

Turbine	Component	Recorded at	Technician Remarks
01	Gearbox	2016-07-18 02:10:00	Gearbox pump damaged
	Transformer	2017-08-11 13:14:00	Transformer fan damaged
06	Hydraulic group	2016-04-04 18:53:00	Error in pitch regulation
	Generator	2016-07-11 19:48:00	Generator replaced
	Generator	2016-07-24 17:01:00	Generator temperature sensor failure
	Generator	2016-09-04 08:08:00	High temperature generator error
	Generator	2016-10-02 17:08:00	Refrigeration system and temperature sensors in generator replaced
	Generator	2016-10-27 16:26:00	Generator replaced
	Hydraulic group	2017-08-19 09:47:00	Oil leakage in Hub
	Gearbox	2017-10-17 08:38:00	Gearbox bearings damaged
07	Generator bearings	2016-04-30 12:40:00	High temperature in generator bearing (replaced sensor)
	Transformer	2016-07-10 03:46:00	High temperature transformer
	Transformer	2016-08-23 02:21:00	High temperature transformer. Transformer refrigeration repaired
	Hydraulic group	2017-06-17 11:35:00	Oil leakage in Hub
	Generator bearings	2017-08-20 06:08:00	Generator bearings damaged
	Generator	2017-08-21 14:47:00	Generator damaged
	Hydraulic group	2017-10-19 10:11:00	Oil leakage in Hub
09	Generator bearings	2016-06-07 16:59:00	High temperature generator bearing
	Generator bearings	2016-08-22 18:25:00	High temperature generator bearing
	Gearbox	2016-10-11 08:06:00	Gearbox repaired
	Generator bearings	2016-10-17 09:19:00	Generator bearings replaced
	Generator bearings	2017-01-25 12:55:00	Generator bearings replaced
	Hydraulic group	2017-09-16 15:46:00	Pitch position error related GH
	Gearbox	2017-10-18T08:32:00	Gearbox noise
11	Generator	2016-03-03 19:00:00	Electric circuit error in generator
	Hydraulic group	2016-10-17 17:44:00	Hydraulic group error in the brake circuit
	Hydraulic group	2017-04-26 18:06:00	Hydraulic group error in the brake circuit
	Hydraulic group	2017-09-12 15:30:00	Hydraulic group error in the brake circuit

Table B.1: List of failures recorded found in the EDP dataset

Bibliography

[Andrade 2022] J. R. Andrade, C. Rocha, R. Silva, J. P. Viana, Ricardo J. Bessa, C. Gouveia, B. Almeida, R. J. Santos, M. Louro, P. M. Santos and A. F. Ribeiro. *Data-Driven Anomaly Detection and Event Log Profiling of SCADA Alarms.* IEEE Access, vol. 10, pages 73758–73773, 2022. (Cited on page 2.)

[Bangalore 2013a] Pramod Bangalore and Lina Bertling Tjernberg. *Self evolving neural network based algorithm for fault prognosis in wind turbines: A case study.* pages 1–6, 06 2013. (Cited on page 16.)

[Bangalore 2013b] Pramod Bangalore and Lina Bertling Tjernberg. *An approach for self evolving neural network based algorithm for fault prognosis in wind turbine.* 2013 IEEE Grenoble Conference, pages 1–6, 2013. (Cited on page 16.)

[Bangalore 2015] Pramod Bangalore and Lina Bertling Tjernberg. *An Artificial Neural Network Approach for Early Fault Detection of Gearbox Bearings.* IEEE Transactions on Smart Grid, vol. 6, no. 2, pages 980–987, 2015. (Cited on pages 2 and 16.)

[Boersma 2017] Sjoerd Boersma, Bart Doekemeijer, Pieter Gebraad, Paul Fleming, Jennifer Annoni, Andrew Scholbrock, Joeri Frederik and J. W. Wingerden. *A tutorial on control-oriented modeling and control of wind farms.* pages 1–18, 05 2017. (Cited on page 6.)

[Brandao 2010] R.F.M. Brandao, José Carvalho and Fernando Maciel-Barbosa. *Neural networks for condition monitoring of wind turbines.* volume 6, pages 1–4, 01 2010. (Cited on page 16.)

[Brandao 2015] R.F.M. Brandao, José Carvalho and Fernando Maciel-Barbosa. *Intelligent System for Fault Detection in Wind Turbines Gearbox.* 06 2015. (Cited on page 16.)

[Brown 2018] Andy Brown, Aaron Tuor, Brian Hutchinson and Nicole Nichols. *Recurrent Neural Network Attention Mechanisms for Interpretable System Log Anomaly Detection.* In Proceedings of the First Workshop on Machine Learning for Computing Systems, MLCS'18, New York, NY, USA, 2018. Association for Computing Machinery. (Cited on page 2.)

[Das 2021] Mamata Das, Selvakumar Kamalanathan and Pja Alphonse. *A Comparative Study on TF-IDF Feature Weighting Method and Its Analysis Using Unstructured Dataset.* In International Conference on Computational Linguistics and Intelligent Systems, 2021. (Cited on page 28.)

[EDP 2018] EDP. *EDP Open Data.* https://opendata.edp.com/opendata/en/data.html, July 2018. Accessed: 2023-01-29. (Cited on page 5.)

[El-Hashash 2022] Essam El-Hashash and Raga Hassan. *A Comparison of the Pearson, Spearman Rank and Kendall Tau Correlation Coefficients Using Quantitative Variables.* Asian Journal of Probability and Statistics, vol. 20, pages 36–48, 10 2022. (Cited on page 17.)

[European Commission 2023a] European Commission. *Renewable energy statistics.* https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Renewable_energy_statistics, 03 2023. Accessed: 2023-03-26. (Cited on page 1.)

[European Commission 2023b] European Commission. *Renewable energy targets.* https://energy.ec.europa.eu/topics/renewable-energy/renewable-energy-directive-targets-and-rules/renewable-energy-targets_en, 2023. Accessed: 2023-03-26. (Cited on page 1.)

[Fahrmeir 2021] Ludwig Fahrmeir, Thomas Kneib, Stefan Lang and Brian D. Marx. Regression models, pages 23–84. Springer Berlin Heidelberg, Berlin, Heidelberg, 2021. (Cited on page 11.)

[Feng 2011] Yanhui Feng, Yingning Qiu, Christopher Crabtree, Hui Long and P.J. Tavner. *Use of SCADA and CMS signals for failure detection & diagnosis of a wind turbine gearbox.* European Wind Energy Association Conference EWEA 2011, 01 2011. (Cited on page 1.)

[Fukushima 1980] Kunihiko Fukushima. *Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position.* Biological Cybernetics, vol. 36, no. 4, pages 193–202, April 1980. (Cited on page 14.)

[Galton 1894] Sir Galton Francis. Natural inheritance. New York, Macmillan and co, 1894. https://www.biodiversitylibrary.org/bibliography/46339. (Cited on page 12.)

[Garlick 2009] William Garlick, Roger Dixon and Simon Watson. *A model-based approach to wind turbine condition monitoring using SCADA data.* 01 2009. (Cited on page 13.)

[Goodfellow 2016] Ian Goodfellow, Yoshua Bengio and Aaron Courville. Deep learning. MIT Press, 2016. http://www.deeplearningbook.org. (Cited on page 13.)

[Haykin 1999] S. Haykin and S.S. Haykin. Neural networks: A comprehensive foundation. International edition. Prentice Hall, 1999. (Cited on page 13.)

[He 2016a] Pinjia He, Jieming Zhu, Shilin He, Jian Li and Michael R. Lyu. *An Evaluation Study on Log Parsing and Its Use in Log Mining.* In 2016 46th

Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pages 654–661, 2016. (Cited on page 26.)

[He 2016b] Shilin He, Jieming Zhu, Pinjia He and Michael R. Lyu. *Experience Report: System Log Analysis for Anomaly Detection*. In 27th IEEE International Symposium on Software Reliability Engineering, ISSRE 2016, Ottawa, ON, Canada, October 23-27, 2016, pages 207–218. IEEE Computer Society, 2016. (Cited on pages 26 and 27.)

[He 2017] Pinjia He, Jieming Zhu, Zibin Zheng and Michael R. Lyu. *Drain: An Online Log Parsing Approach with Fixed Depth Tree*. In 2017 IEEE International Conference on Web Services (ICWS), pages 33–40, 2017. (Cited on page 26.)

[Jang 1997] J.S.R. Jang, C.T. Sun and E. Mizutani. Neuro-fuzzy and soft computing: A computational approach to learning and machine intelligence. MATLAB curriculum series. Prentice Hall, 1997. (Cited on page 12.)

[Kendall 1938] M. G. Kendall. *A NEW MEASURE OF RANK CORRELATION*. Biometrika, vol. 30, no. 1-2, pages 81–93, 06 1938. (Cited on page 17.)

[Leahy 2017] Kevin Leahy, Colm Gallagher, Ken Bruton, Peter O'Donovan and Dominic O' Sullivan. *Automatically Identifying and Predicting Unplanned Wind Turbine Stoppages Using SCADA and Alarms System Data: Case Study and Results*. volume 926, page 012011, 11 2017. (Cited on page 3.)

[Lee 2017] K.-Y Lee, K.-H Kim, J.-J Kang, S.-J Choi, Y.-S Im, Y.-D Lee and Y.-S Lim. *Comparison and analysis of linear regression & artificial neural network*. International Journal of Applied Engineering Research, vol. 12, pages 9820–9825, 01 2017. (Cited on page 13.)

[Letzgus 2020] Simon Letzgus. *Finding meaningful representations of SCADA-log information for data-driven condition monitoring applications*. 12 2020. (Cited on page 3.)

[Lyu 2019] Michael R. Lyu, Jieming Zhu, Pinjia He, Shilin He, Jinyang Liu, Zhuangbin Chen, Yintong Huo, Yuxin Su and Zibin Zheng. *LogPAI*. `https://logpai.com/`, 2019. (Cited on pages 2 and 26.)

[Nagappan 2010] Meiyappan Nagappan and Mladen A. Vouk. *Abstracting log lines to log event types for mining software system logs*. In 2010 7th IEEE Working Conference on Mining Software Repositories (MSR 2010), pages 114–117, 2010. (Cited on page 26.)

[Rahman 2016] Anisur Rahman, Yue Xu, Kenneth Radke and Ernest Foo. *Finding Anomalies in SCADA Logs Using Rare Sequential Pattern Mining*. In Jiageng Chen, Vincenzo Piuri, Chunhua Su and Moti Yung, editeurs, Network

and System Security, pages 499–506, Cham, 2016. Springer International Publishing. (Cited on page 2.)

[Rosenblatt 1958] F. Rosenblatt. *The perceptron: A probabilistic model for information storage and organization in the brain.* Psychological Review, vol. 65, no. 6, pages 386–408, 1958. (Cited on page 13.)

[Ruff 2021] Lukas Ruff, Jacob Kauffmann, Robert Vandermeulen, Gregoire Montavon, Wojciech Samek, Marius Kloft, Thomas Dietterich and Klaus-Robert Müller. *A Unifying Review of Deep and Shallow Anomaly Detection.* Proceedings of the IEEE, vol. PP, pages 1–40, 02 2021. (Cited on page 16.)

[Schlechtingen 2011] Meik Schlechtingen and Ilmar Ferreira Santos. *Comparative analysis of neural network and regression based condition monitoring approaches for wind turbine fault detection.* Mechanical Systems and Signal Processing, vol. 25, no. 5, pages 1849–1875, 2011. (Cited on pages 13 and 16.)

[Schmidhuber 2015] Jürgen Schmidhuber. *Deep learning in neural networks: An overview.* Neural Networks, vol. 61, pages 85–117, 2015. (Cited on page 13.)

[Shima 2016] Keiichi Shima. *Length Matters: Clustering System Log Messages using Length of Words.* 11 2016. (Cited on page 26.)

[Sohoni 2016] Vaishali Sohoni, S. Gupta and Rajesh Nema. *A Critical Review on Wind Turbine Power Curve Modelling Techniques and Their Applications in Wind Based Energy Systems.* Journal of Energy, vol. 2016, pages 1–18, 01 2016. (Cited on page 22.)

[Sparck Jones 1972] Karen Sparck Jones. *A statistical interpretation of term specificity and its application in retrieval.* Journal of documentation, vol. 28, no. 1, pages 11–21, 1972. (Cited on page 28.)

[Tang 2011] Liang Tang, Tao Li and Chang-Shing Perng. *LogSig: Generating system events from raw textual logs.* pages 785–794, 10 2011. (Cited on page 26.)

[Tautz-Weinert 2017] Jannis Tautz-Weinert and Simon Watson. *Using SCADA data for wind turbine condition monitoring - A review.* IET Renewable Power Generation, vol. 11, pages 382–394, 03 2017. (Cited on pages 2, 11, 13 and 42.)

[Vestas 2016] Vestas. *VestasOnline Enterprise User Guide.* `https://voe.vestas.com/public/UserManual.pdf`, 02 2016. Accessed: 2023-02-06. (Cited on page 7.)

[Wang 2019] Xiaofeng Wang, Guoliang Lu and Peng Yan. *Multiple regression analysis based approach for condition monitoring of industrial rotating machinery using multi-sensors.* In 2019 Prognostics and System Health Management Conference (PHM-Qingdao), pages 1–5, 2019. (Cited on page 13.)

[Yang 2013] Wenxian Yang, Richard Court and Jiesheng Jiang. *Wind turbine condition monitoring by the approach of SCADA data analysis.* Renewable Energy, vol. 53, pages 365–376, 2013. (Cited on page 1.)

[Yang 2014] Wenxian Yang, Peter J. Tavner, Christopher J. Crabtree, Y. Feng and Y. Qiu. *Wind turbine condition monitoring: technical and commercial challenges.* Wind Energy, vol. 17, no. 5, pages 673–693, 2014. (Cited on page 1.)

[Zadeh 1965] L.A. Zadeh. *Fuzzy sets.* Information and Control, vol. 8, no. 3, pages 338–353, 1965. (Cited on page 12.)

[Zhang 2014] Zhen-You Zhang and Ke-Sheng Wang. *Wind turbine fault detection based on SCADA data analysis using ANN.* Advances in Manufacturing, vol. 2, no. 1, pages 70–78, 2014. (Cited on pages 2 and 16.)

[Zhu 2018] Jieming Zhu, Shilin He, Jinyang Liu, Pinjia He, Qi Xie, Zibin Zheng and Michael R. Lyu. *Tools and Benchmarks for Automated Log Parsing.* CoRR, vol. abs/1811.03509, 2018. (Cited on page 26.)