# 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 01. Mai 2023 ..............................................

Mohamed Samy Elsisi

## Abstract

English version of the German “Zusammenfassung”.

## Zusammenfassung

Deutsche Version des Englischen “Abstracts”.

# 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. This need gets more relevant in the case of operating offshore wind farms, where the cost of maintenance relative to the levelized cost of energy (LCOE) is significantly higher compared with onshore [Tautz-Weinert 2017]. Several approaches for condition monitoring were developed in the recent years that use SCADA1 data given its low cost (normally requiring no additional sensors). One of the methods used for condition monitoring using SCADA data is Normal Behavior Modelling (NBM). NBM uses the idea of detecting anomalies from normal operation by empirically modelling 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 modelled/predicted signal is used as indicator for a possible fault. A difference of 0, with some tolerance, reflects normal conditions, whereas a difference greater or less than 0 reflects changed conditions or failures [Tautz-Weinert 2017].

## 1.2 Motivation

While NBMs using SCADA data were proven capable of predicting failures [Tautz-Weinert 2017], they are treated as black box by the operators since they don't provide any insights regarding the root cause of the failure. Incorporating SCADA log data2 to NBM could help tackle this problem by providing some insights to an anomaly detected by the model in case a relevant warning or failure message was logged by the SCADA system around the same time. It was also shown that incorporating SCADA logs containing information about operation conditions or control events could help improve the accuracy of the model in case of events unexplainable by the input signals [Letzgus 2020].//

Logs were "Never" treated as input feature in NBMs

Chapter 2

# State of the art

**Contents**

NBM// What was done in the topic of logs?

## 2.1 Standalone Chapter or better as part of the introduction?

Chapter 3

# 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.

## 3.1 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 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.

### 3.1.1 Signals

The *Signals* dataset contains 10-min aggregated data (mean, standard deviation, minimum, and maximum values) collected from the wind turbines' power meters and sensors installed at the major components such as gearbox, generator and transformer (see Figure 3.1 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,. . .

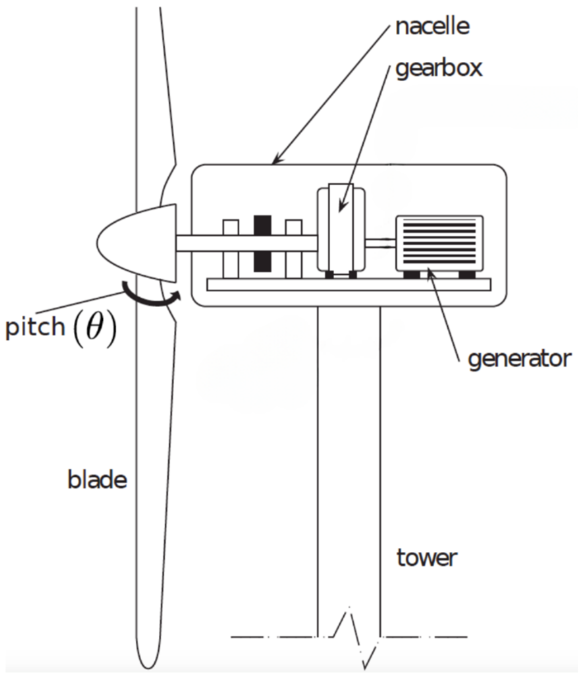


Figure 3.1: Diagram of a wind turbine side view with labeled main components. Figure adapted from [Boersma 2017]

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 3.1 shows some of the 81 signals included in this 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 ($\theta$) |

Table 3.1: Example signals found in the Signals dataset

### 3.1.2 Logs

In this dataset, events logged by the SCADA system are collected 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.

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

Table 3.2: Sample log events found in the Logs dataset

### 3.1.3 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.

## 3.2 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 3.2 depicts the concept of model-based 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.

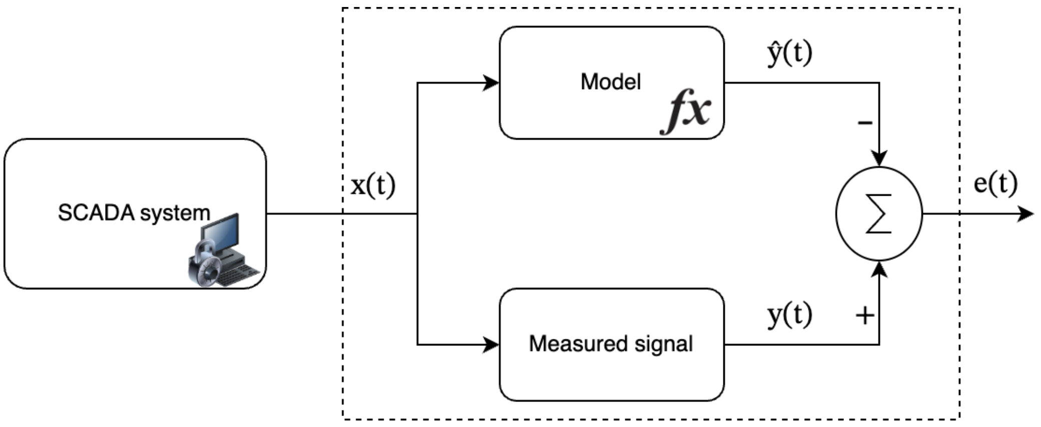


Figure 3.2: NBM with the input signals from the SCADA system (x(t)), measured signal (y(t)), modeled signal ($\hat{y}$(t)) and resulting error (e(t))

### 3.2.1 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 poly-*

*nomial models*, *Artificial Neural Networks (ANNs)* 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. Training fuzzy systems was not within the scope of this work. However, we propose testing our methods on them in the future works section (see 5.2).

#### 3.2.1.1 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:

$$Y = m_1X_1 + m_2X_2 + ... + m_nX_n + C \tag{3.1}$$

where $m_1$ through $m_n$ and $C$ are constants. Figure 3.3 shows an example of linear regression for a single independent variable.

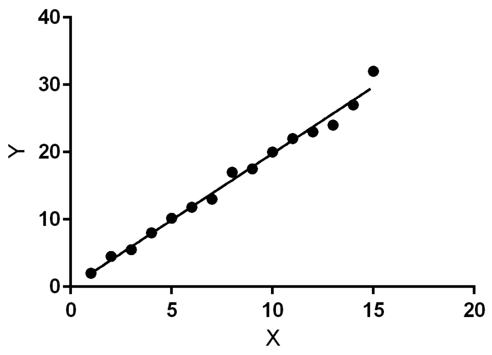


Figure 3.3: Example linear regression with one dependent variable: $Y = mX + b$, where m and b are constants

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 (ANNs) may be a better approach in these situations.

#### 3.2.1.2 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:

$$Y = f(\sum_{i=1}^{n} w_i X_i) \tag{3.2}$$

The sigmoid function, the rectified linear unit (ReLU) function, and the hyperbolic tangent function are examples of common activation functions. Figure 3.4 shows a diagram of a perceptron.

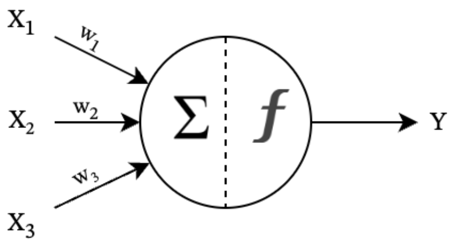


Figure 3.4: Example perceptron with three inputs

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 is 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 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 computes 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 4.1), we decided to train the normal behavior models on a feed-forward neural network having the architecture shown in Fig. 3.5 and ReLU (firstly introduced by Fukushima [Fukushima 1980]) as an activation function.

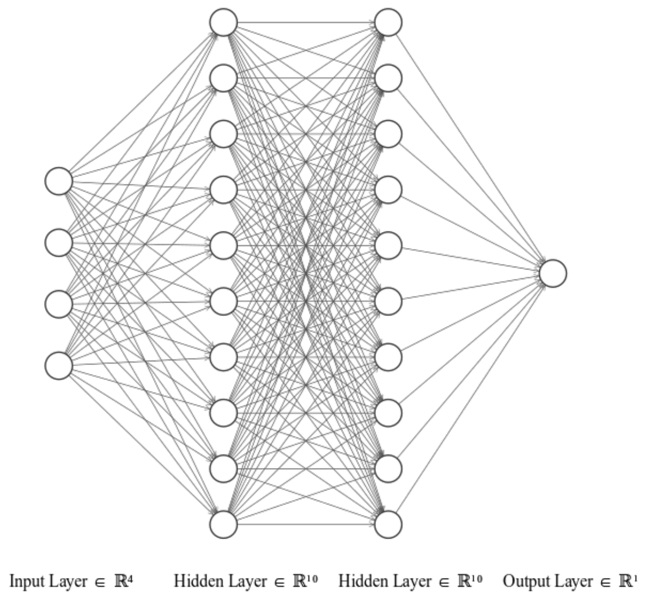


Figure 3.5: Architecture of normal behavior neural network model used in this work. *The input layer shape will vary based on the experiment and the number of input features.*

### 3.2.2 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. An anomaly is commonly referred to as an outlier or an uncommon trend in data in numerous domains such as statistics, data analysis, and security. 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. Depending on the context, positive anomalies could suggest that something is performing better than predicted and negative anomalies indicate that something is underperforming (e.g., in the context of a company's sales figures), or the other way around. 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,. . . —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).

#### 3.2.2.1 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 $99^{\text{th}}$ percentile of the distribution of the number of anomalies that occurred per day during the training period when using an *anomaly threshold* set to the $99^{\text{th}}$ 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.
(TODO: Maybe soma visualization is needed here?)

### 3.2.3 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 your 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 dependency between two variables by measuring the monotonic relationship. In addition to that, variables don't have to be normally distributed when using Kendall's method.
(TODO: list features selected)

## 3.3 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. In summary, we introduce three different ways for utilizing SCADA log messages: Extracting input features for normal behavior models, Data filtering, and Visualization of warnings. We will explain each approach in depth.

### 3.3.1 Extracting log embeddings

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 (see chapter 2), we came up with two methods that were proven capable of not only generating embeddings for machine-learning normal behavior models but also improving their accuracy (see chapter 4): 1. our Novel method based on domain knowledge and 2. Utilizing an open-source framework for analyzing log data called LogPAI. We will discuss each method in detail.

#### 3.3.1.1 Novel method

**Background:**
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 normal behavior models monitor the state of a component by monitoring its temperature, 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 shows the states of internal or external ventilators of some components (see table 3.3). Being parts of the cooling systems of major components, fans or ventilators must affect the component's temperature.

| **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 3.3: 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. 3.6, 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.

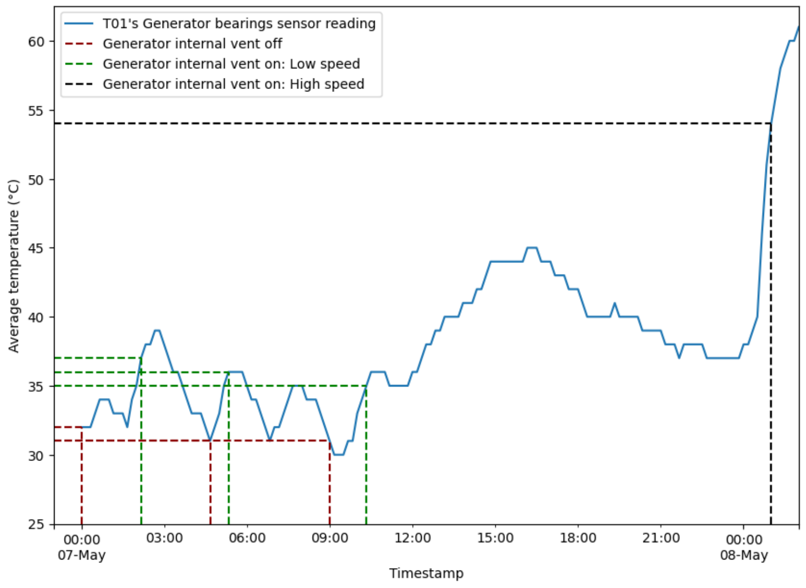
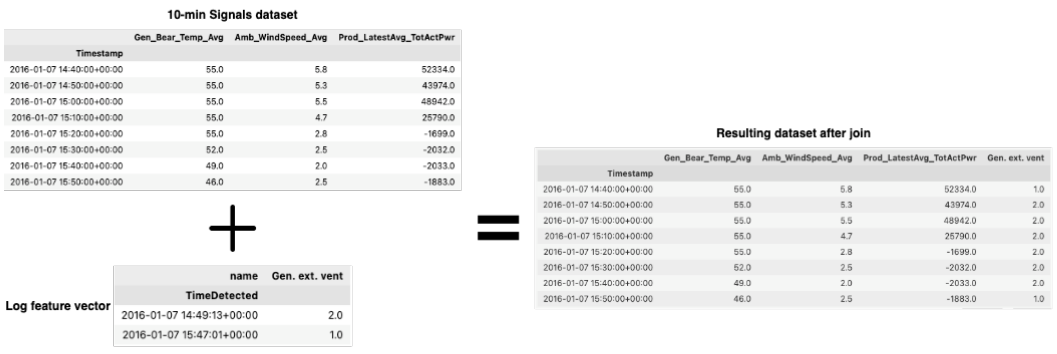


Figure 3.6: Generator internal vent control signals and their effect on the generator bearings temperature

**Method:**
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. Temperature of the turbine component the ventilator is installed in (e.g., *65° C*).

Since the component temperature is regularly provided as a SCADA signal, we decided to focus on the other two parts of the log messages. Our method simply filters log messages containing the word "vent." and creates a new feature for every ventilator (1.) found in the data having its state (2.) 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-min 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).
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 embeddings feature that, on average, highly correlates ($Rank > 0.5$) with it.


10-min Signals dataset

|  | Gen_Bear_Temp_Avg | Amb_WindSpeed_Avg | Prod_LatestAvg_TotActPwr |
|---|---|---|---|
| Timestamp |  |  |  |
| 2016-01-07 14:40:00+00:00 | 55.0 | 5.8 | 52334.0 |
| 2016-01-07 14:50:00+00:00 | 55.0 | 5.3 | 43974.0 |
| 2016-01-07 15:00:00+00:00 | 55.0 | 5.5 | 48942.0 |
| 2016-01-07 15:10:00+00:00 | 55.0 | 4.7 | 25790.0 |
| 2016-01-07 15:20:00+00:00 | 55.0 | 2.8 | -1699.0 |
| 2016-01-07 15:30:00+00:00 | 52.0 | 2.5 | -2032.0 |
| 2016-01-07 15:40:00+00:00 | 49.0 | 2.0 | -2033.0 |
| 2016-01-07 15:50:00+00:00 | 46.0 | 2.5 | -1883.0 |

+

Log feature vector

| name | Gen. ext. vent |
|---|---|
| TimeDetected |  |
| 2016-01-07 14:49:13+00:00 | 2.0 |
| 2016-01-07 15:47:01+00:00 | 1.0 |

=

Resulting dataset after join

|  | Gen_Bear_Temp_Avg | Amb_WindSpeed_Avg | Prod_LatestAvg_TotActPwr | Gen. ext. vent |
|---|---|---|---|---|
| Timestamp |  |  |  |  |
| 2016-01-07 14:40:00+00:00 | 55.0 | 5.8 | 52334.0 | 1.0 |
| 2016-01-07 14:50:00+00:00 | 55.0 | 5.3 | 43974.0 | 2.0 |
| 2016-01-07 15:00:00+00:00 | 55.0 | 5.5 | 48942.0 | 2.0 |
| 2016-01-07 15:10:00+00:00 | 55.0 | 4.7 | 25790.0 | 2.0 |
| 2016-01-07 15:20:00+00:00 | 55.0 | 2.8 | -1699.0 | 2.0 |
| 2016-01-07 15:30:00+00:00 | 52.0 | 2.5 | -2032.0 | 2.0 |
| 2016-01-07 15:40:00+00:00 | 49.0 | 2.0 | -2033.0 | 2.0 |
| 2016-01-07 15:50:00+00:00 | 46.0 | 2.5 | -1883.0 | 1.0 |

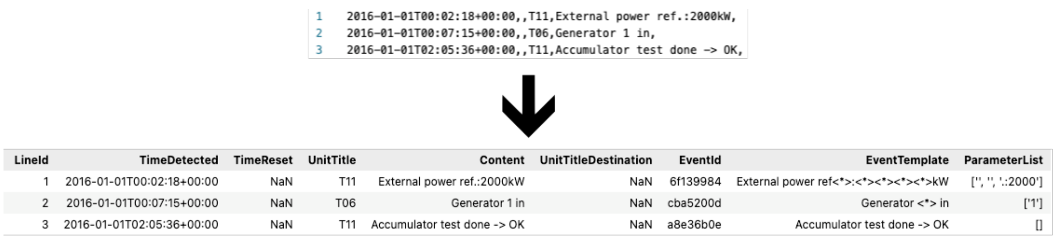
Figure 3.7: Demonstration of the join operation between the signals 10-min dataset and a log embeddings vector

#### 3.3.1.2 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.

**Preprocessing using Logparser:**
From the list of parsers available in the toolkit, 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. The research demonstrates that Drain is very good at dealing with enormous amounts of log data and extracting meaningful events from noisy and diverse log data. Several phases are involved in the Drain algorithm, including log parsing, log message clustering, and event extraction. Drain employs a fixed-depth tree to parse log messages into a set of log keys and their related values during the log parsing stage. 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. Drain uses a similarity measure to compare the log keys and values of each log message and allocates them to the best appropriate cluster based on their similarity scores during the log message clustering stage. Drain creates a template for each cluster that summarizes the relevant information contained in the log

messages once the log messages have been clustered. Overall, the Drain algorithm makes an important contribution to log data analysis and management by providing a scalable and effective approach for extracting structured events from unstructured log data. 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. 3.8 for an example) that the Loglizer can process to generate numerical features.



```
2016-01-01T00:02:18+00:00,,T11,External power ref.:2000kW,
2016-01-01T00:07:15+00:00,,T06,Generator 1 in,
2016-01-01T02:05:36+00:00,,T11,Accumulator test done -> OK,
```

| LineId | TimeDetected | TimeReset | UnitTitle | Content | UnitTitleDestination | EventId | EventTemplate | ParameterList |
|---|---|---|---|---|---|---|---|---|
| 1 | 2016-01-01T00:02:18+00:00 | NaN | T11 | External power ref.:2000kW | NaN | 6f139984 | External power ref<*>:<*><*><*><*>kW | ['', '', '.:2000'] |
| 2 | 2016-01-01T00:07:15+00:00 | NaN | T06 | Generator 1 in | NaN | cba5200d | Generator <*> in | ['1'] |
| 3 | 2016-01-01T02:05:36+00:00 | NaN | T11 | Accumulator test done -> OK | NaN | a8e36b0e | Accumulator test done -> OK | [] |

Figure 3.8: Sample raw logs and their corresponding structured logs after being parsed by Drain

**Creating numerical features using Loglizer**:
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' *Event IDs*.

### 3.3.2 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]). We start by extracting the log messages that report the current state of operation; namely logs containing one of the following expressions:

- *"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 3.3.1.1, and labeled based on the following logic:

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

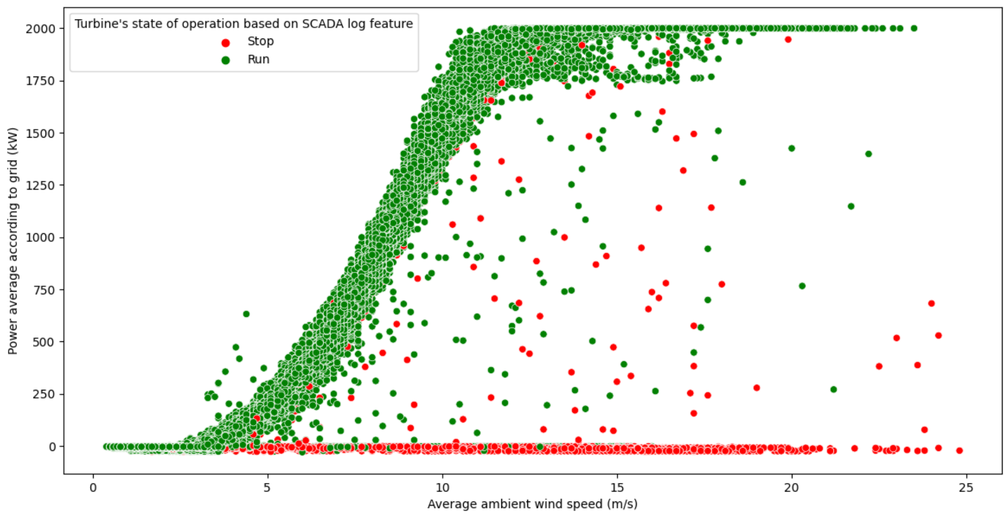


Figure 3.9: T01 power curve with log-feature-based labels

The log-based feature we introduced showed a clear improvement in the accuracy of power curve models (see experiment..TODO) when used to filter the data being input to the normal behavior model (using data points having *"Run"* as the state of operation exclusively).

### 3.3.3 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.
(TODO add graph showing an example)

## 3.4 Summary

TODO: PUSH TO THE TOP
Diagram of all methods put together: ML model + log feature + Anomaly detection + Alarms,...

CHAPTER 4

# Experiments

## Contents

## 4.1 Benchmark

### 4.1.1 Research question

The aim of this experiment is to test the ability of NBM, as defined in the literature, to early-detect failures of a faulty turbine using the ERP dataset. The final model architecture that we pick after running this experiment will serve as a baseline model used as a benchmark in the later experiments.

### 4.1.2 Setup

The following elements were used in this experiment:

- **Machine learning models:** Linear regression (baseline) and feed-forward neural network
- **Target wind turbine:** T09

- **Dataset:** Training/healthy period: 01/01/2016 - 15/02/2016, Testing/faulty period: 16/02/2016 - 18/10/2016
- **Input features:** Nac_Temp_Avg, Amb_Temp_Avg, Gen_RPM_Avg, Prod_LatestAvg_TotActPwr (or use verbose names of signals: the average temperature in the nacelle, average ambient temperature, average generator rpm, total active power)
- **Target feature:** Gen_Bear_Temp_Avg (Average temperature in generator bearing 1 (Non-Drive End))
- **Recorded failure:** *"Generator bearings replaced on October 17, 2016, 9:19 AM"*
- **Logs used:** None

### 4.1.3 Results

According to the results documented in 4.2, we conclude that both NBMs are capable of predicting the failure in the monitored part. We will, however, use only the feed-forward network model as a benchmark since it outperformed the linear regression model.

| **Comparison metric** | **Measure for linear regression** | **Measure for feed-forward network** |
|---|---|---|
| RMSE | | |
| First-detected anomaly timestamp | | |
| Number of anomalies detected | | |

Table 4.1: Experiment I results: Metrics used to compare between the benchmark models

## 4.2 Effect of incorporating log embeddings into NBM for condition monitoring when applied on a healthy turbine (T01)

### 4.2.1 Research question

The aim of this experiment is to quantitatively (RMSE) and qualitatively (number of false alarms) measure the effect of incorporating SCADA-log-based features into the benchmark NBM.

### 4.2.2 Setup

The following elements were used in this experiment:

- **Machine learning models:** Feed-forward neural network with single target features and Feed-forward neural network with multiple target features

- **Target wind turbine:** T01

- **Dataset:** Training/healthy period: 01/09/2016 - 31/12/2016, Testing period: 01/01/2017 - 31/12/2017

- **Input features (SCADA signals):** Nac_Temp_Avg, Amb_Temp_Avg, Gen_RPM_Avg, Prod_LatestAvg_TotActPwr (or use verbose names of signals: the average temperature in the nacelle, average ambient temperature, average generator rpm, total active power)

- **SCADA-log-based input features:** Operation and System log messages containing the word "vent", which resulted in four different features extracted from the following components: Generator external vent, Generator internal vent, High-voltage transformer vent, and Nacelle vent

- **Target feature for single-output model:** Gen_Bear_Temp_Avg (Average temperature in generator bearing 1 (Non-Drive End))

- **Target features for multiple-output model:** All signals whose names contain the keywords "Gen" and "Temp": 'Gen_Bear_Temp_Avg', 'Gen_Phase1_Temp_Avg', 'Gen_Phase2_Temp_Avg', 'Gen_Phase3_Temp_Avg', 'Gen_SlipRing_Temp_Avg', 'Gen_Bear2_Temp_Avg'

- **Recorded failure:** No generator-related recorded failures (hence the assumption that the turbine is healthy)

### 4.2.3 Results

According to the results documented in 4.2, we conclude that both NBMs are capable of predicting the failure in the monitored part. We will, however, use only the feed-forward network model as a benchmark since it outperformed the linear regression model.

| **Comparison metric** | **Measure for linear regression** | **Measure for feed-forward network** |
|---|---|---|
| RMSE | | |
| First-detected anomaly timestamp | | |
| Number of anomalies detected | | |

Table 4.2: Experiment I results: Metrics used to compare between the benchmark models

## 4.3 Effect of incorporating log embeddings into NBM for condition monitoring when applied on a faulty turbine (T09)

## 4.4 Effect of incorporating log embeddings into NBM for condition monitoring when applied on a faulty turbine (T06)

## 4.5 Effect of log-based data filtering on NBM for power curve modeling when applied on T01

Chapter 5

# Conclusions and Future Works

## 5.1 Conclusions

## 5.2 Future Works

Appendix A

# Appendix I

# Bibliography

[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 13.)

[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 13.)

[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 page 13.)

[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 12.)

[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 12.)

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

[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 8.)

[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 11.)

[Galton 1894] Sir Galton Francis. Natural inheritance. New York, Macmillan and co, 1894. `https://www.biodiversitylibrary.org/bibliography/46339`. (Cited on page 9.)

[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 10.)

[Goodfellow 2016] Ian Goodfellow, Yoshua Bengio and Aaron Courville. Deep learning. MIT Press, 2016. `http://www.deeplearningbook.org`. (Cited on page 10.)

[Haykin 1999] S. Haykin and S.S. Haykin. Neural networks: A comprehensive foundation. International edition. Prentice Hall, 1999. (Cited on page 10.)

[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 17.)

[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 page 17.)

[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 17.)

[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 9.)

[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 14.)

[Letzgus 2020] Simon Letzgus. *Finding meaningful representations of SCADA-log information for data-driven condition monitoring applications.* 12 2020. (Cited on page 2.)

[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 page 17.)

[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 10.)

[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 10 and 12.)

[Schmidhuber 2015] Jürgen Schmidhuber. *Deep learning in neural networks: An overview.* Neural Networks, vol. 61, pages 85–117, 2015. (Cited on page 10.)

[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 18.)

[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 18.)

[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 1, 8 and 10.)

[Vestas 2016] Vestas. *VestasOnline Enterprise User Guide.* `https://voe.vestas.com/public/UserManual.pdf`, 02 2016. Accessed: 2023-02-06. (Cited on page 6.)

[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 10.)

[Zadeh 1965] L.A. Zadeh. *Fuzzy sets.* Information and Control, vol. 8, no. 3, pages 338–353, 1965. (Cited on page 9.)

[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 page 12.)

[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 17.)